Reconhecimento de imagens - Convolutional Neural Networks

Diferença entre as espécies de pistachios, Siirt e Kirmizi

- Beatriz Carvalho e Carolina Aquino

Dataset original com 2149 imagens e reduzido para 998 imagens.

Importar e reduzir dataset

In [5]:

```
!pip install tensorflow
!pip install keras
```

```
Requirement already satisfied: tensorflow in ./opt/anaconda3/lib/python3.9/site-packages (2.11.0)
Requirement already satisfied: flatbuffers>=2.0 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorflow) (23.1.21)
Requirement already satisfied: packaging in ./opt/anaconda3/lib/python3.9/site-packages (from tensorflow) (21.3)
Requirement already satisfied: setuptools in ./opt/anaconda3/lib/python3.9/site-packages (from tensorflow) (61.2.0)
Requirement already satisfied: opt-einsum>=2.3.2 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorflow) (3.3.0)
Requirement already satisfied: wrapt>=1.11.0 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorflow) (1.12.1)
Requirement already satisfied: absl-py>=1.0.0 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorflow) (1.4.0)
Requirement already satisfied: tensorboard<2.12,>=2.11 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorflow) (2.11.2)
Requirement already satisfied: typing-extensions>=3.6.6 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorflow) (4.1.1)
Requirement already satisfied: protobuf<3.20,>=3.9.2 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorflow) (3.19.1)
Requirement already satisfied: termcolor>=1.1.0 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorflow) (2.2.0)
Requirement already satisfied: libclang>=13.0.0 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorflow) (15.0.6.1)
Requirement already satisfied: astunparse>=1.6.0 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorflow) (1.6.3)
Requirement already satisfied: keras<2.12,>=2.11.0 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorflow) (2.11.0)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorflow) (1.42.0)
Requirement already satisfied: h5py>=2.9.0 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorflow) (3.6.0)
Requirement already satisfied: numpy>=1.20 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorflow) (1.21.5)
Requirement already satisfied: tensorflow-estimator<2.12,>=2.11.0 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorflow) (2.11.0)
Requirement already satisfied: google-pasta>=0.1.1 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorflow) (0.2.0)
Requirement already satisfied: six>=1.12.0 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorflow) (1.16.0)
Requirement already satisfied: gast<=0.4.0,>=0.2.1 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorflow) (0.4.0)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorflow) (0.30.0)
Requirement already satisfied: wheel<1.0,>=0.23.0 in ./opt/anaconda3/lib/python3.9/site-packages (from astunparse>=1.6.0->tensorflow) (0.37.1)
Requirement already satisfied: markdown>=2.6.8 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorboard<2.12,>=2.11->tensorflow) (3.3.4)
Requirement already satisfied: tensorboard-data-server<0.7.0,>=0.6.0 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorboard<2.12,>=2.11->tensorflow)
(0.6.1)
Requirement already satisfied: werkzeug>=1.0.1 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorboard<2.12,>=2.11->tensorflow) (2.0.3)
Requirement already satisfied: requests<3,>=2.21.0 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorboard<2.12,>=2.11->tensorflow) (2.27.1)
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorboard<2.12,>=2.11->tensorflow) (0.
4.6)
Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorboard<2.12,>=2.11->tensorflow) (1.8.1)
Requirement already satisfied: google-auth<3,>=1.6.3 in ./opt/anaconda3/lib/python3.9/site-packages (from tensorboard<2.12,>=2.11->tensorflow) (1.33.0)
Requirement already satisfied: cachetools<5.0,>=2.0.0 in ./opt/anaconda3/lib/python3.9/site-packages (from google-auth<3,>=1.6.3->tensorboard<2.12,>=2.11->ten
sorflow) (4.2.2)
Requirement already satisfied: pyasn1-modules>=0.2.1 in ./opt/anaconda3/lib/python3.9/site-packages (from google-auth<3,>=1.6.3->tensorboard<2.12,>=2.11->tens
orflow) (0.2.8)
Requirement already satisfied: rsa<5,>=3.1.4 in ./opt/anaconda3/lib/python3.9/site-packages (from google-auth<3,>=1.6.3->tensorboard<2.12,>=2.11->tensorflow)
(4.7.2)
Requirement already satisfied: requests-oauthlib>=0.7.0 in ./opt/anaconda3/lib/python3.9/site-packages (from google-auth-oauthlib<0.5,>=0.4.1->tensorboard<2.1
2,>=2.11->tensorflow) (1.3.1)
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in ./opt/anaconda3/lib/python3.9/site-packages (from pyasn1-modules>=0.2.1->google-auth<3,>=1.6.3->tensorb
oard<2.12,>=2.11->tensorflow) (0.4.8)
Requirement already satisfied: charset-normalizer~=2.0.0 in ./opt/anaconda3/lib/python3.9/site-packages (from requests<3,>=2.21.0->tensorboard<2.12,>=2.11->te
nsorflow) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in ./opt/anaconda3/lib/python3.9/site-packages (from requests<3,>=2.21.0->tensorboard<2.12,>=2.11->tensorflow) (3.
3)
Requirement already satisfied: certifi>=2017.4.17 in ./opt/anaconda3/lib/python3.9/site-packages (from requests<3,>=2.21.0->tensorboard<2.12,>=2.11->tensorflo
w) (2021.10.8)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in ./opt/anaconda3/lib/python3.9/site-packages (from requests<3,>=2.21.0->tensorboard<2.12,>=2.11->tensor
flow) (1.26.9)
Requirement already satisfied: oauthlib>=3.0.0 in ./opt/anaconda3/lib/python3.9/site-packages (from requests-oauthlib>=0.7.0->google-auth-oauthlib<0.5,>=0.4.1
->tensorboard<2.12,>=2.11->tensorflow) (3.2.2)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in ./opt/anaconda3/lib/python3.9/site-packages (from packaging->tensorflow) (3.0.4)
Requirement already satisfied: keras in ./opt/anaconda3/lib/python3.9/site-packages (2.11.0)
```

In [7]:

```python
# diretorios
dataset_dir = '/Users/wise/Desktop/pistachio_image_dataset'
destino_dir = '/Users/wise/Desktop/pistachio_image_dataset/pistachio_small'

import os, shutil, pathlib

original_dir = pathlib.Path("/Users/wise/Desktop/pistachio_image_dataset")
new_base_dir = pathlib.Path("/Users/wise/Desktop/pistachio_image_dataset/pistachio_small")

def make_subset(subset_name, start_index, end_index):
    for category in ("siirt", "kirmizi"):
        dir = new_base_dir / subset_name / category
        os.makedirs(dir)
        fnames = [f"{category} {i}.jpg"
                  for i in range(start_index, end_index)]
        for fname in fnames:
            try:
                shutil.copyfile(src=original_dir / fname,
                                dst=dir / fname)
            except:
                continue
make_subset("train", start_index=0, end_index=300)
make_subset("validation", start_index=300, end_index=500)
make_subset("test", start_index=500, end_index=700)
```

Construção do modelo com Keras

In [8]:

```python
from tensorflow import keras
from tensorflow.keras import layers
import keras
```

In [9]:

```python
inputs = keras.Input(shape=(224, 224, 3))
x = layers.Conv2D(filters=32, kernel_size=3, activation="relu")(inputs)
x = layers.MaxPooling2D(pool_size=2)(x)
x = layers.Conv2D(filters=64, kernel_size=3, activation="relu")(x)
x = layers.MaxPooling2D(pool_size=2)(x)
x = layers.Conv2D(filters=128, kernel_size=3, activation="relu")(x)
x = layers.MaxPooling2D(pool_size=2)(x)
x = layers.Conv2D(filters=256, kernel_size=3, activation="relu")(x)
x = layers.MaxPooling2D(pool_size=2)(x)
x = layers.Conv2D(filters=256, kernel_size=3, activation="relu")(x)
x = layers.MaxPooling2D(pool_size=2)(x)
x = layers.Flatten()(x)
outputs = layers.Dense(1, activation="sigmoid")(x)
model = keras.Model(inputs=inputs, outputs=outputs)

model.summary()
```

```
Model: "model"
_________________________________________________________________
Layer (type)                Output Shape              Param #
=================================================================
input_1 (InputLayer)        [(None, 224, 224, 3)]     0
_________________________________________________________________
conv2d (Conv2D)             (None, 222, 222, 32)      896
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 111, 111, 32)     0
_________________________________________________________________
conv2d_1 (Conv2D)           (None, 109, 109, 64)      18496
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 54, 54, 64)       0
_________________________________________________________________
conv2d_2 (Conv2D)           (None, 52, 52, 128)       73856
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 26, 26, 128)      0
_________________________________________________________________
conv2d_3 (Conv2D)           (None, 24, 24, 256)       295168
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 12, 12, 256)      0
_________________________________________________________________
conv2d_4 (Conv2D)           (None, 10, 10, 256)       590080
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 5, 5, 256)        0
_________________________________________________________________
flatten (Flatten)           (None, 6400)              0
_________________________________________________________________
dense (Dense)               (None, 1)                 6401
=================================================================
Total params: 984,897
Trainable params: 984,897
Non-trainable params: 0
_________________________________________________________________
```

Tratamento das imagens

In [10]:

```python
from keras import optimizers
model.compile(loss='binary_crossentropy',
            optimizer='rmsprop',
            metrics=['accuracy'])
```

In [11]:

```python
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(
                                                    new_base_dir / 'train',
                                                    target_size=(224, 224),
                                                    batch_size=200
                                                    class_mode='binary')


validation_generator = test_datagen.flow_from_directory(
                                                        new_base_dir / 'validation',
                                                        target_size=(224, 224),
                                                        batch_size=200,
                                                        class_mode='binary')
```

```
Found 598 images belonging to 2 classes.
Found 400 images belonging to 2 classes.
```

Ajuste do modelo

In [14]:

```python
callbacks = [
    keras.callbacks.ModelCheckpoint(
        filepath="convnet_from_scratch.keras",
        save_best_only=True,
        monitor="val_loss")]

history = model.fit(
    train_generator,
    epochs=10,
    validation_data=validation_generator,
    callbacks=callbacks)
```

```
Epoch 1/10
3/3 [==============================] - 30s 10s/step - loss: 0.4727 - accuracy: 0.7809 - val_loss: 0.6825 - val_accuracy: 0.6850
Epoch 2/10
3/3 [==============================] - 30s 11s/step - loss: 0.5011 - accuracy: 0.7676 - val_loss: 0.4033 - val_accuracy: 0.8225
Epoch 3/10
3/3 [==============================] - 29s 10s/step - loss: 0.3951 - accuracy: 0.8211 - val_loss: 0.4099 - val_accuracy: 0.8225
Epoch 4/10
3/3 [==============================] - 29s 10s/step - loss: 0.4360 - accuracy: 0.7893 - val_loss: 0.3887 - val_accuracy: 0.8300
Epoch 5/10
3/3 [==============================] - 29s 10s/step - loss: 0.3788 - accuracy: 0.8211 - val_loss: 0.3784 - val_accuracy: 0.8325
Epoch 6/10
3/3 [==============================] - 28s 10s/step - loss: 0.3979 - accuracy: 0.8161 - val_loss: 0.4370 - val_accuracy: 0.8025
Epoch 7/10
3/3 [==============================] - 28s 10s/step - loss: 0.4060 - accuracy: 0.8027 - val_loss: 0.3802 - val_accuracy: 0.8325
Epoch 8/10
3/3 [==============================] - 28s 10s/step - loss: 0.3664 - accuracy: 0.8294 - val_loss: 0.4079 - val_accuracy: 0.8175
Epoch 9/10
3/3 [==============================] - 30s 10s/step - loss: 0.4061 - accuracy: 0.8027 - val_loss: 0.3628 - val_accuracy: 0.8350
Epoch 10/10
3/3 [==============================] - 29s 10s/step - loss: 0.3303 - accuracy: 0.8662 - val_loss: 0.3489 - val_accuracy: 0.8400
```

In [15]:

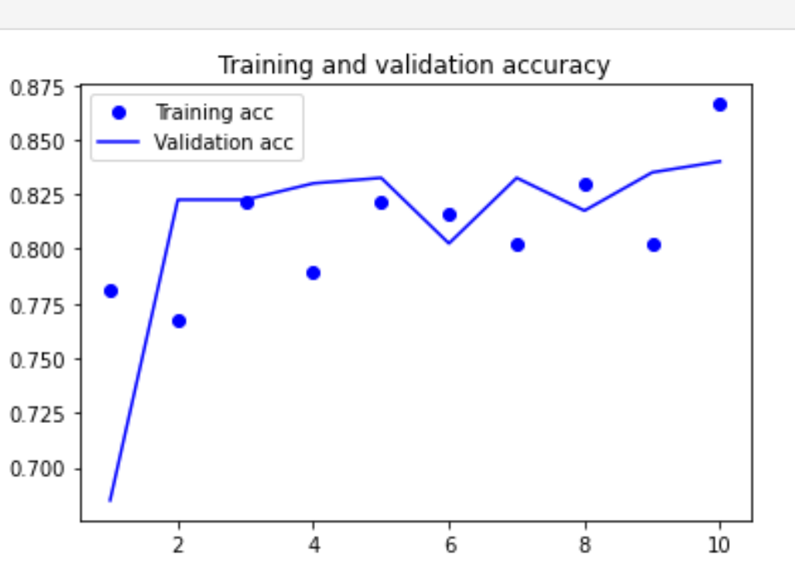
```python
model.save('pistachio_small.h5')
```

In [16]:

```python
import matplotlib.pyplot as plt
```

In [17]:

```python
acc = history.history['accuracy']
val_acc = history.history['val_accuracy']
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()
```



Melhorar o modelo - Data Augmentation e Droupout

In [59]:

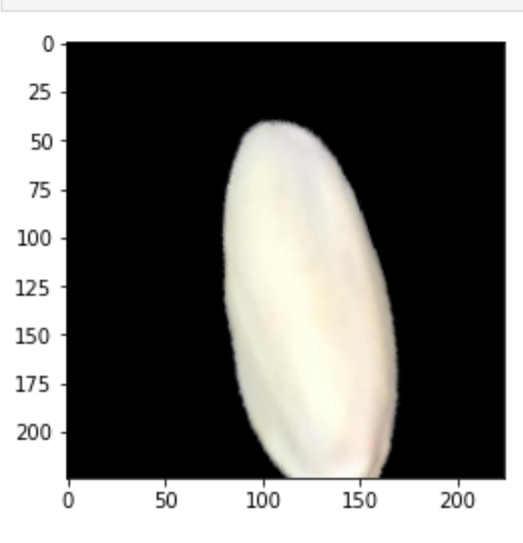
```python
datagen = ImageDataGenerator(rotation_range=40,
                             width_shift_range=0.2,
                             height_shift_range=0.2,
                             shear_range=0.2,
                             zoom_range=0.2,
                             horizontal_flip=True,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

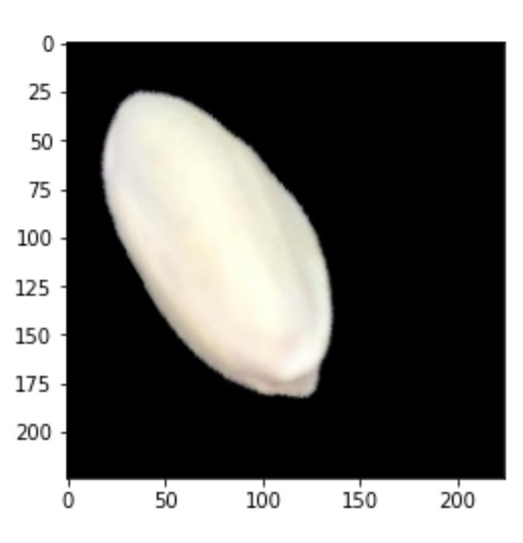
In [60]:

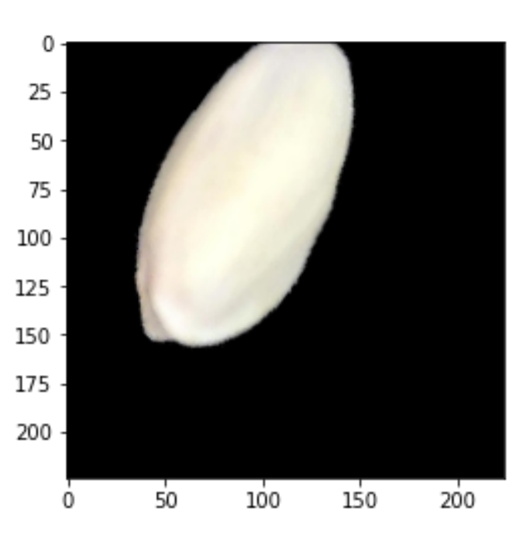
```python
from keras.preprocessing import image

fnames = [os.path.join(new_base_dir / 'train' / 'siirt', fname) for fname in os.listdir(new_base_dir / 'train' / 'siirt')]
img_path = fnames[3]
img = image.load_img(img_path, target_size=(224, 224))
```

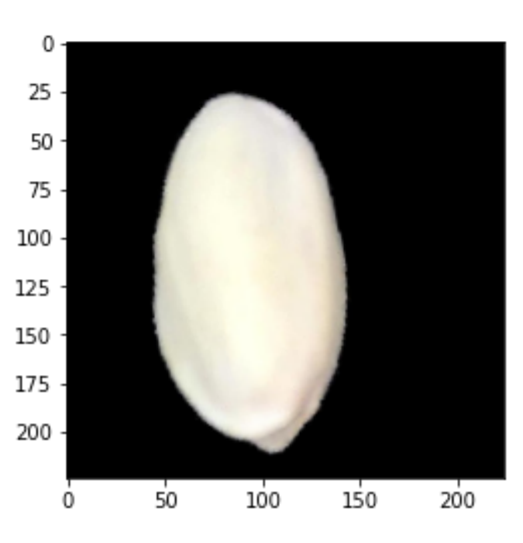
In [61]:

```python
x = image.img_to_array(img)
x = x.reshape((1,) + x.shape)

i = 0

for batch in datagen.flow(x, batch_size=1):
    plt.figure(i)
    imgplot = plt.imshow(image.array_to_img(batch[0]))
    i += 1
    if i % 4 == 0:
        break

plt.show()
```









Novo ajuste do modelo

In [62]:

```python
import tensorflow as tf
from tensorflow.keras import optimizers

model = keras.Sequential()
model.add(layers.Conv2D(32, (3, 3), activation='relu',
input_shape=(224, 224, 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(1, activation='sigmoid'))
model.compile(optimizer=tf.keras.optimizers.RMSprop(
    learning_rate=0.001),
              loss='binary_crossentropy',
              metrics=['accuracy'])
```

In [63]:

```python
callbacks = [
    keras.callbacks.ModelCheckpoint(
        filepath="feature_extraction.keras",
        save_best_only=True,
        monitor="val_loss")
]
```

In [64]:

```python
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(
                                                    new_base_dir / 'train',
                                                    target_size=(224, 224),
                                                    batch_size=200,
                                                    class_mode='binary',
                                                    shuffle=True)

validation_generator = test_datagen.flow_from_directory(
                                                        new_base_dir / 'validation',
                                                        target_size=(224, 224),
                                                        batch_size=200,
                                                        class_mode='binary',
                                                        shuffle=True)
history = model.fit_generator(
                            train_generator,
                            epochs=20,
                            validation_data=validation_generator)
```

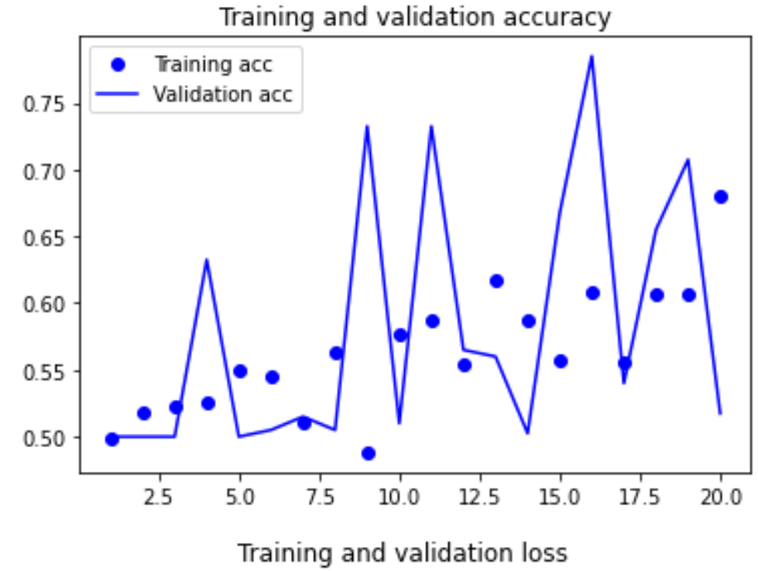
```
Found 598 images belonging to 2 classes.
Found 400 images belonging to 2 classes.
Epoch 1/20
3/3 [==============================] - 36s 12s/step - loss: 1.1598 - accuracy: 0.4983 - val_loss: 0.7399 - val_accuracy: 0.5000
Epoch 2/20
3/3 [==============================] - 36s 12s/step - loss: 0.7000 - accuracy: 0.5184 - val_loss: 0.7115 - val_accuracy: 0.5000
Epoch 3/20
3/3 [==============================] - 32s 10s/step - loss: 0.6917 - accuracy: 0.5234 - val_loss: 0.6906 - val_accuracy: 0.5000
Epoch 4/20
3/3 [==============================] - 34s 12s/step - loss: 0.6919 - accuracy: 0.5251 - val_loss: 0.6767 - val_accuracy: 0.6325
Epoch 5/20
3/3 [==============================] - 32s 11s/step - loss: 0.7201 - accuracy: 0.5502 - val_loss: 0.6864 - val_accuracy: 0.5000
Epoch 6/20
3/3 [==============================] - 30s 10s/step - loss: 0.6816 - accuracy: 0.5452 - val_loss: 0.7176 - val_accuracy: 0.5050
Epoch 7/20
3/3 [==============================] - 28s 9s/step - loss: 0.7029 - accuracy: 0.5100 - val_loss: 0.6837 - val_accuracy: 0.5150
Epoch 8/20
3/3 [==============================] - 33s 10s/step - loss: 0.6854 - accuracy: 0.5635 - val_loss: 0.6697 - val_accuracy: 0.5050
Epoch 9/20
3/3 [==============================] - 28s 9s/step - loss: 0.6906 - accuracy: 0.4883 - val_loss: 0.6688 - val_accuracy: 0.7325
Epoch 10/20
3/3 [==============================] - 31s 10s/step - loss: 0.6809 - accuracy: 0.5769 - val_loss: 0.6584 - val_accuracy: 0.5100
Epoch 11/20
3/3 [==============================] - 29s 10s/step - loss: 0.6646 - accuracy: 0.5870 - val_loss: 0.5858 - val_accuracy: 0.7325
Epoch 12/20
3/3 [==============================] - 33s 12s/step - loss: 0.6884 - accuracy: 0.5535 - val_loss: 0.6648 - val_accuracy: 0.5650
Epoch 13/20
3/3 [==============================] - 29s 10s/step - loss: 0.6650 - accuracy: 0.6171 - val_loss: 0.6294 - val_accuracy: 0.5600
Epoch 14/20
3/3 [==============================] - 47s 16s/step - loss: 0.6752 - accuracy: 0.5870 - val_loss: 0.6912 - val_accuracy: 0.5025
Epoch 15/20
3/3 [==============================] - 35s 11s/step - loss: 0.6766 - accuracy: 0.5569 - val_loss: 0.6385 - val_accuracy: 0.6675
Epoch 16/20
3/3 [==============================] - 34s 11s/step - loss: 0.6468 - accuracy: 0.6087 - val_loss: 0.5694 - val_accuracy: 0.7850
Epoch 17/20
3/3 [==============================] - 29s 10s/step - loss: 0.7245 - accuracy: 0.5552 - val_loss: 0.6651 - val_accuracy: 0.5400
Epoch 18/20
3/3 [==============================] - 31s 10s/step - loss: 0.6581 - accuracy: 0.6070 - val_loss: 0.6315 - val_accuracy: 0.6550
Epoch 19/20
3/3 [==============================] - 106s 48s/step - loss: 0.6549 - accuracy: 0.6070 - val_loss: 0.6198 - val_accuracy: 0.7075
Epoch 20/20
3/3 [==============================] - 30s 9s/step - loss: 0.6282 - accuracy: 0.6806 - val_loss: 0.7193 - val_accuracy: 0.5175
```

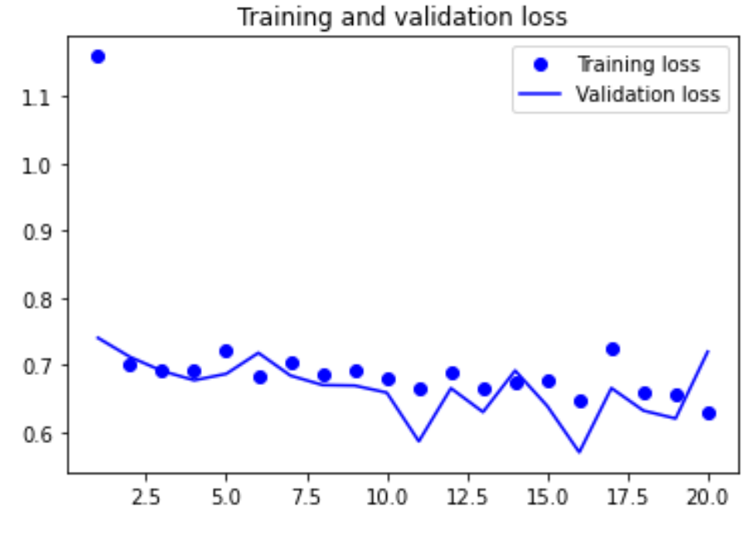
In [65]:

```python
model.save('pistachio_small_2.h5')
```

In [66]:

```python
import matplotlib.pyplot as plt

acc = history.history['accuracy']
val_acc = history.history['val_accuracy']
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()
```





In [ ]: