```
import os, shutil
train_dir = 'train'
validation_dir = 'train3'
test_dir = 'test'
train_airplane_dir = 'train/000_airplane'
train_automobile_dir = 'train/001_automobile'
train_bird_dir = 'train/002_bird'
train_cat_dir = 'train/003_cat'
train_deer_dir = 'train/004_deer'
train_dog_dir = 'train/005_dog'
train_frog_dir = 'train/006_frog'
train_horse_dir = 'train/007_horse'
train_ship_dir = 'train/008_ship'
train_truck_dir = 'train/009_truck'
val_airplane_dir = 'train3/000_airplane'
val_automobile_dir = 'train3/001_automobile'
val_bird_dir = 'train3/002_bird'
val_cat_dir = 'train3/003_cat'
val_deer_dir = 'train3/004_deer'
val_dog_dir = 'train3/005_dog'
val_frog_dir = 'train3/006_frog'
val_horse_dir = 'train3/007_horse'
val_ship_dir = 'train3/008_ship'
val_truck_dir = 'train3/009_truck'
test_airplane_dir = 'test/test/000_airplane'
test_automobile_dir = 'test/test/001_automobile'
test_bird_dir = 'test/test/002_bird'
test_cat_dir = 'test/test/003_cat'
test_deer_dir = 'test/test/004_deer'
test_dog_dir = 'test/test/005_dog'
test_frog_dir = 'test/test/006_frog'
test_horse_dir = 'test/test/007_horse'
test_ship_dir = 'test/test/008_ship'
test_truck_dir = 'test/test/009_truck'
print('total training airplane images:',
len(os.listdir(train_airplane_dir)))
print('total training automobile images:',
len(os.listdir(train_automobile_dir)))
print('total training bird images:', len(os.listdir(train_bird_dir)))
print('total training cat images:', len(os.listdir(train_cat_dir)))
print('total training deer images:', len(os.listdir(train_deer_dir)))
print('total training dog images:', len(os.listdir(train_dog_dir)))
print('total training frog images:', len(os.listdir(train_frog_dir)))
print('total training horse images:',
len(os.listdir(train_horse_dir)))
print('total training ship images:', len(os.listdir(train_ship_dir)))
print('total training truck images:',
len(os.listdir(train_truck_dir)))
print('total validation airplane images:',
```

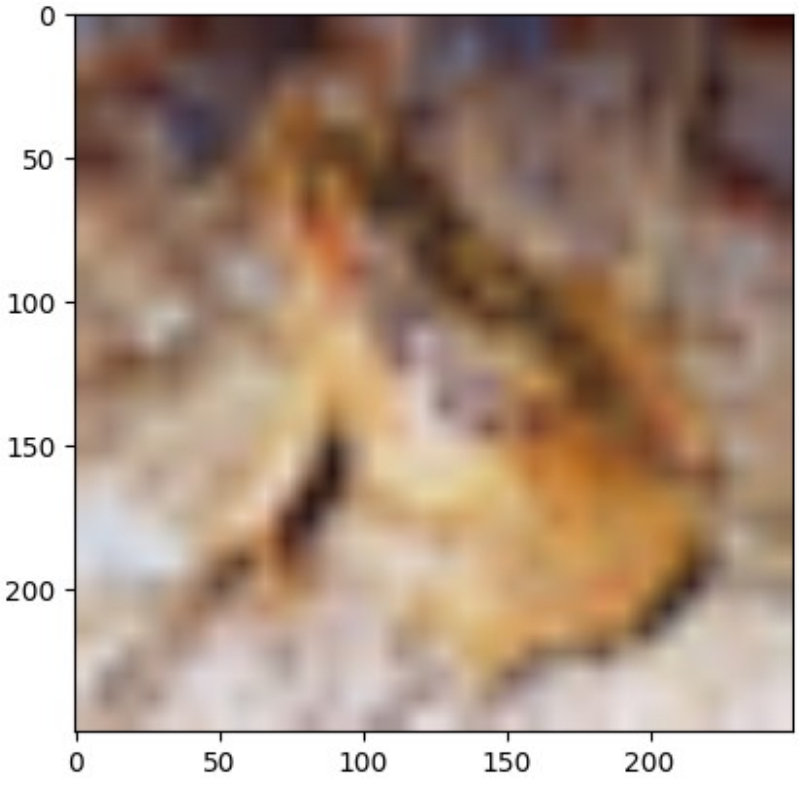
```
len(os.listdir(val_airplane_dir)))
print('total validation automobile images:',
len(os.listdir(val_automobile_dir)))
print('total validation bird images:', len(os.listdir(val_bird_dir)))
print('total validation cat images:', len(os.listdir(val_cat_dir)))
print('total validation deer images:', len(os.listdir(val_deer_dir)))
print('total validation dog images:', len(os.listdir(val_dog_dir)))
print('total validation frog images:', len(os.listdir(val_frog_dir)))
print('total validation horse images:',
len(os.listdir(val_horse_dir)))
print('total validation ship images:', len(os.listdir(val_ship_dir)))
print('total validation truck images:',
len(os.listdir(val_truck_dir)))
print('total testing airplane images:',
len(os.listdir(test_airplane_dir)))
print('total testing automobile images:',
len(os.listdir(test_automobile_dir)))
print('total testing bird images:', len(os.listdir(test_bird_dir)))
print('total testing cat images:', len(os.listdir(test_cat_dir)))
print('total testing deer images:', len(os.listdir(test_deer_dir)))
print('total testing dog images:', len(os.listdir(test_dog_dir)))
print('total testing frog images:', len(os.listdir(test_frog_dir)))
print('total testing horse images:', len(os.listdir(test_horse_dir)))
print('total testing ship images:', len(os.listdir(test_ship_dir)))
print('total testing truck images:', len(os.listdir(test_truck_dir)))
```

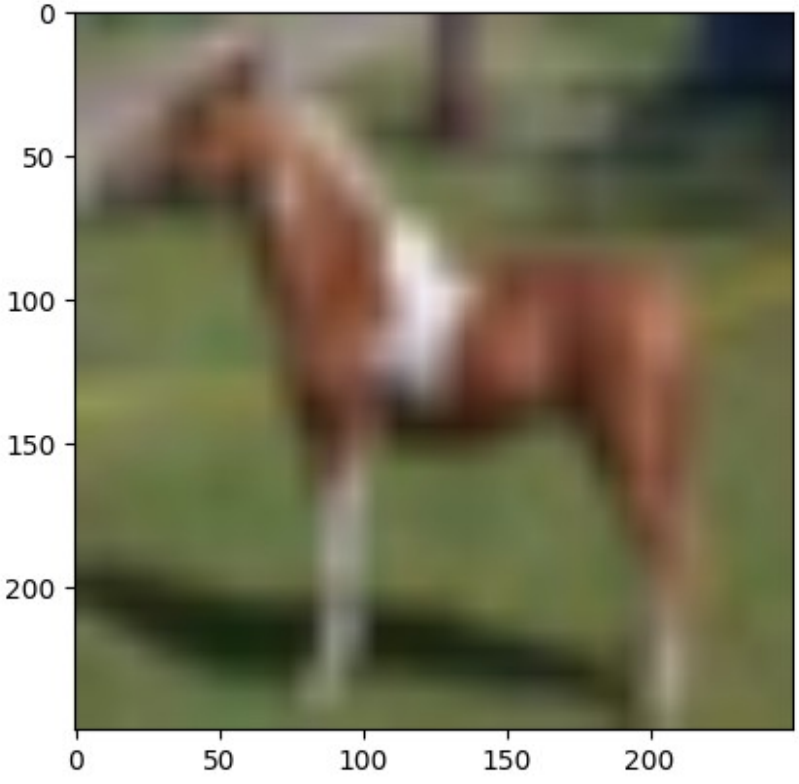
```
total training airplane images: 4006
total training automobile images: 3958
total training bird images: 4035
total training cat images: 4003
total training deer images: 4010
total training dog images: 3971
total training frog images: 4022
total training horse images: 3985
total training ship images: 4078
total training truck images: 3971
total validation airplane images: 994
total validation automobile images: 1042
total validation bird images: 965
total validation cat images: 997
total validation deer images: 990
total validation dog images: 1029
total validation frog images: 978
total validation horse images: 1015
total validation ship images: 961
total validation truck images: 1029
total testing airplane images: 1000
total testing automobile images: 1000
total testing bird images: 1000
total testing cat images: 1000
```

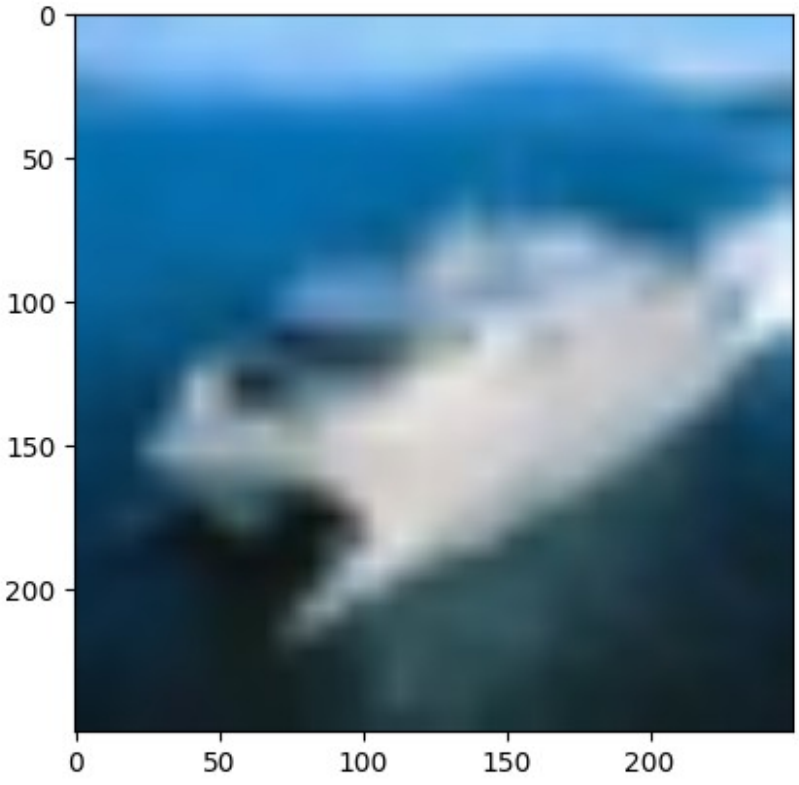
```
total testing deer images: 1000
total testing dog images: 1000
total testing frog images: 1000
total testing horse images: 1000
total testing ship images: 1000
total testing truck images: 1000
```

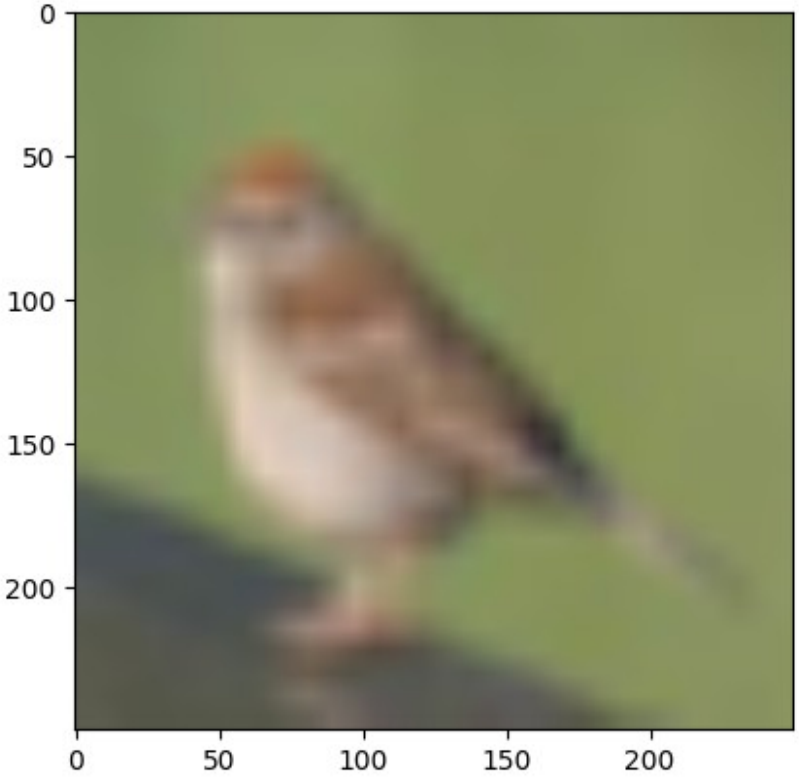
```
from keras.utils import image_dataset_from_directory
IMG_SIZE = 250
train_dataset = image_dataset_from_directory(
train_dir,
image_size=(IMG_SIZE, IMG_SIZE),
batch_size=32)
validation_dataset = image_dataset_from_directory(
validation_dir,
image_size=(IMG_SIZE, IMG_SIZE),
batch_size=32)
test_dataset = image_dataset_from_directory(
test_dir,
image_size=(IMG_SIZE, IMG_SIZE),
batch_size=32)
import matplotlib.pyplot as plt
for data_batch, _ in train_dataset.take(1):
  for i in range(5):
    plt.imshow(data_batch[i].numpy().astype("uint8"))
    plt.show()
```

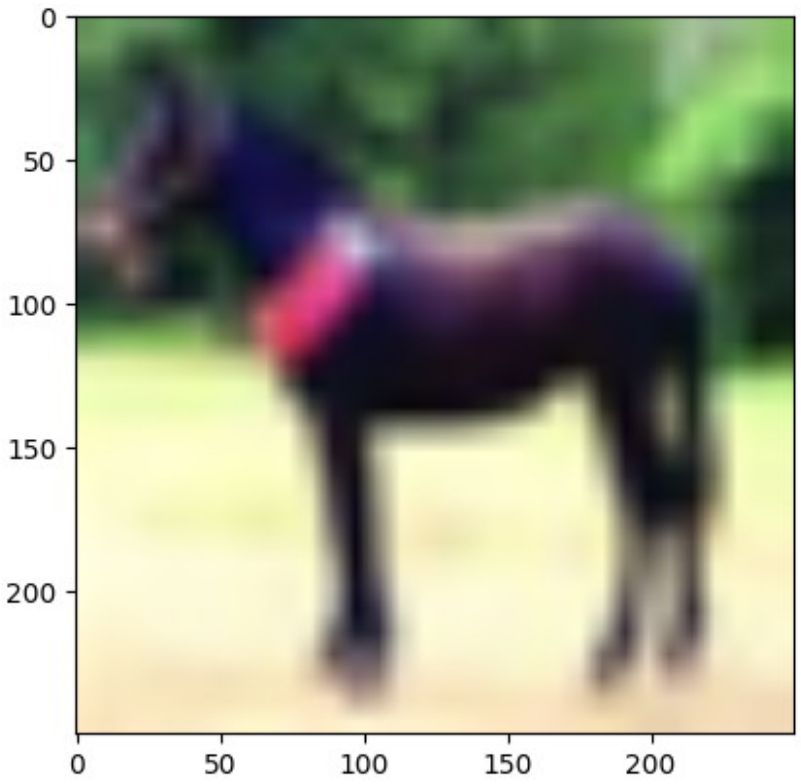
```
Found 40039 files belonging to 10 classes.
Found 10000 files belonging to 10 classes.
Found 10000 files belonging to 1 classes.
```


0
50
100
150
200
0
50
100
150
200


0
50
100
150
200
0
50
100
150
200


0
50
100
150
200
0
50
100
150
200


0
50
100
150
200
0
50
100
150
200




```
from tensorflow import keras
from keras import layers
data_augmentation = keras.Sequential(
  [
  layers.RandomFlip("horizontal"),
  layers.RandomRotation(0.1),
  layers.RandomZoom(0.2),
  ]
)

#The shape of each batch

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

```
data batch shape: (32, 250, 250, 3)
labels batch shape: (32,)
```

```
#Creating the neural network
from tensorflow import keras
from keras import layers
from keras import models
inputs = keras.Input(shape=(IMG_SIZE, IMG_SIZE, 3))
x = data_augmentation(inputs)
```

```
x = layers.Rescaling(1./255)(inputs)
x = layers.Conv2D(filters=32, kernel_size=3, activation="relu")(x)
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=128, kernel_size=3, activation="relu")(x)
x = layers.MaxPooling2D(pool_size=2)(x)
x = layers.Flatten()(x)
x = layers.Dense(512, activation="relu")(x)
outputs = layers.Dense(10, activation="softmax")(x)
model = keras.Model(inputs=inputs, outputs=outputs)

model.summary()
```

```
Model: "model_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_2 (InputLayer)        [(None, 250, 250, 3)]     0

 rescaling_1 (Rescaling)     (None, 250, 250, 3)       0

 conv2d_4 (Conv2D)           (None, 248, 248, 32)      896

 max_pooling2d_4 (MaxPoolin  (None, 124, 124, 32)      0
 g2D)

 conv2d_5 (Conv2D)           (None, 122, 122, 64)      18496

 max_pooling2d_5 (MaxPoolin  (None, 61, 61, 64)        0
 g2D)

 conv2d_6 (Conv2D)           (None, 59, 59, 128)       73856

 max_pooling2d_6 (MaxPoolin  (None, 29, 29, 128)       0
 g2D)

 conv2d_7 (Conv2D)           (None, 27, 27, 128)       147584

 max_pooling2d_7 (MaxPoolin  (None, 13, 13, 128)       0
 g2D)

 flatten_1 (Flatten)         (None, 21632)             0

 dense_2 (Dense)             (None, 512)               11076096

 dense_3 (Dense)             (None, 10)                5130
```

```
=================================================================
Total params: 11322058 (43.19 MB)
Trainable params: 11322058 (43.19 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

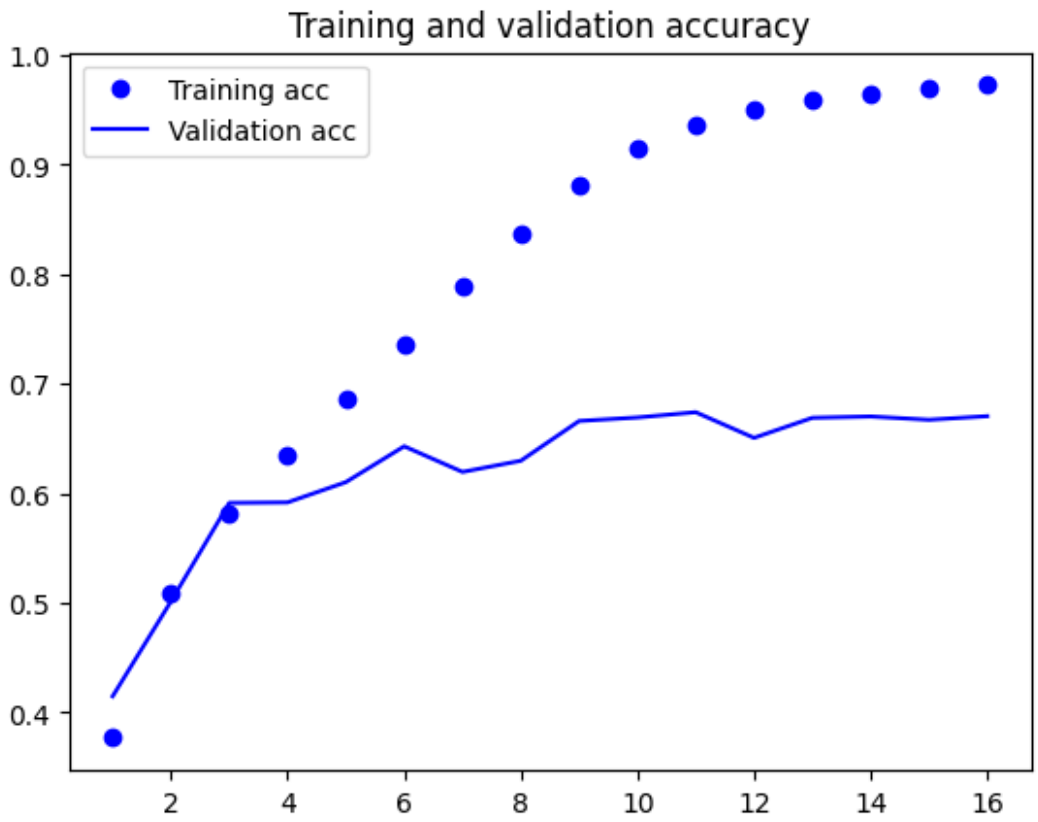
```
from tensorflow.keras.callbacks import ModelCheckpoint, EarlyStopping
checkpoint_callback =
ModelCheckpoint(filepath='models/CNN_modelS_with_data_augmentation.h5'
,
                                        monitor='val_acc',
                                        save_best_only=True,
                                        save_weights_only=False,
                                        verbose=1)

early_stopping_callback = EarlyStopping(monitor='val_acc',
                                          patience=5,
                                          verbose=1)

import tensorflow as tf

model.compile(
  loss='sparse_categorical_crossentropy',
  optimizer=tf.keras.optimizers.RMSprop(learning_rate=1e-4),
  metrics=['acc'])

#Training the model
#history guarda todos os parametros gerados durante o treino
history = model.fit(
  train_dataset,
  epochs=50,
  validation_data=validation_dataset,# n usa para melhorar so para
mostrar ao utilizador
  callbacks=[checkpoint_callback, early_stopping_callback]
  )
```

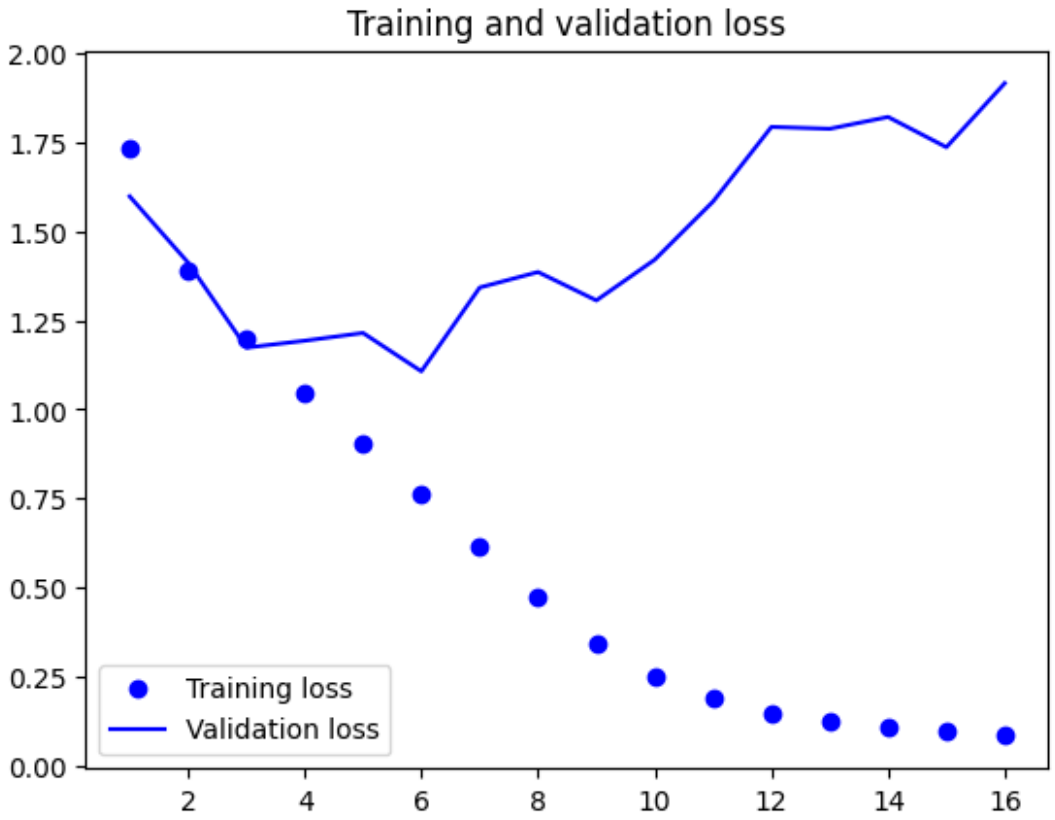
```
Epoch 1/50
1252/1252 [==============================] - ETA: 0s - loss: 1.7351 -
acc: 0.3768
Epoch 1: val_acc improved from -inf to 0.41480, saving model to
models\CNN_modelS_with_data_augmentation.h5

c:\Users\Margarida\AppData\Local\Programs\Python\Python311\Lib\site-
packages\keras\src\engine\training.py:3103: UserWarning: You are
saving your model as an HDF5 file via `model.save()`. This file format
is considered legacy. We recommend using instead the native Keras
format, e.g. `model.save('my_model.keras')`.
  saving_api.save_model(

1252/1252 [==============================] - 3274s 3s/step - loss:
1.7351 - acc: 0.3768 - val_loss: 1.5985 - val_acc: 0.4148
```

```
Epoch 2/50
1252/1252 [==============================] - ETA: 0s - loss: 1.3924 -
acc: 0.5095
Epoch 2: val_acc improved from 0.41480 to 0.50140, saving model to
models\CNN_modelS_with_data_augmentation.h5
1252/1252 [==============================] - 3306s 3s/step - loss:
1.3924 - acc: 0.5095 - val_loss: 1.4126 - val_acc: 0.5014
Epoch 3/50
1252/1252 [==============================] - ETA: 0s - loss: 1.2008 -
acc: 0.5809
Epoch 3: val_acc improved from 0.50140 to 0.59120, saving model to
models\CNN_modelS_with_data_augmentation.h5
1252/1252 [==============================] - 3344s 3s/step - loss:
1.2008 - acc: 0.5809 - val_loss: 1.1736 - val_acc: 0.5912
Epoch 4/50
1252/1252 [==============================] - ETA: 0s - loss: 1.0438 -
acc: 0.6347
Epoch 4: val_acc improved from 0.59120 to 0.59170, saving model to
models\CNN_modelS_with_data_augmentation.h5
1252/1252 [==============================] - 3319s 3s/step - loss:
1.0438 - acc: 0.6347 - val_loss: 1.1930 - val_acc: 0.5917
Epoch 5/50
1252/1252 [==============================] - ETA: 0s - loss: 0.9033 -
acc: 0.6865
Epoch 5: val_acc improved from 0.59170 to 0.61020, saving model to
models\CNN_modelS_with_data_augmentation.h5
1252/1252 [==============================] - 3266s 3s/step - loss:
0.9033 - acc: 0.6865 - val_loss: 1.2147 - val_acc: 0.6102
Epoch 6/50
1252/1252 [==============================] - ETA: 0s - loss: 0.7621 -
acc: 0.7355
Epoch 6: val_acc improved from 0.61020 to 0.64290, saving model to
models\CNN_modelS_with_data_augmentation.h5
1252/1252 [==============================] - 3292s 3s/step - loss:
0.7621 - acc: 0.7355 - val_loss: 1.1070 - val_acc: 0.6429
Epoch 7/50
1252/1252 [==============================] - ETA: 0s - loss: 0.6130 -
acc: 0.7882
Epoch 7: val_acc did not improve from 0.64290
1252/1252 [==============================] - 3268s 3s/step - loss:
0.6130 - acc: 0.7882 - val_loss: 1.3424 - val_acc: 0.6195
Epoch 8/50
1252/1252 [==============================] - ETA: 0s - loss: 0.4705 -
acc: 0.8370
Epoch 8: val_acc did not improve from 0.64290
1252/1252 [==============================] - 3235s 3s/step - loss:
0.4705 - acc: 0.8370 - val_loss: 1.3859 - val_acc: 0.6297
Epoch 9/50
1252/1252 [==============================] - ETA: 0s - loss: 0.3425 -
```

```
acc: 0.8817
Epoch 9: val_acc improved from 0.64290 to 0.66600, saving model to
models\CNN_modelS_with_data_augmentation.h5
1252/1252 [==============================] - 3277s 3s/step - loss:
0.3425 - acc: 0.8817 - val_loss: 1.3062 - val_acc: 0.6660
Epoch 10/50
1252/1252 [==============================] - ETA: 0s - loss: 0.2482 -
acc: 0.9141
Epoch 10: val_acc improved from 0.66600 to 0.66920, saving model to
models\CNN_modelS_with_data_augmentation.h5
1252/1252 [==============================] - 3291s 3s/step - loss:
0.2482 - acc: 0.9141 - val_loss: 1.4206 - val_acc: 0.6692
Epoch 11/50
1252/1252 [==============================] - ETA: 0s - loss: 0.1888 -
acc: 0.9362
Epoch 11: val_acc improved from 0.66920 to 0.67400, saving model to
models\CNN_modelS_with_data_augmentation.h5
1252/1252 [==============================] - 3307s 3s/step - loss:
0.1888 - acc: 0.9362 - val_loss: 1.5840 - val_acc: 0.6740
Epoch 12/50
1252/1252 [==============================] - ETA: 0s - loss: 0.1476 -
acc: 0.9507
Epoch 12: val_acc did not improve from 0.67400
1252/1252 [==============================] - 3218s 3s/step - loss:
0.1476 - acc: 0.9507 - val_loss: 1.7935 - val_acc: 0.6506
Epoch 13/50
1252/1252 [==============================] - ETA: 0s - loss: 0.1240 -
acc: 0.9593
Epoch 13: val_acc did not improve from 0.67400
1252/1252 [==============================] - 2934s 2s/step - loss:
0.1240 - acc: 0.9593 - val_loss: 1.7884 - val_acc: 0.6689
Epoch 14/50
1252/1252 [==============================] - ETA: 0s - loss: 0.1073 -
acc: 0.9649
Epoch 14: val_acc did not improve from 0.67400
1252/1252 [==============================] - 2894s 2s/step - loss:
0.1073 - acc: 0.9649 - val_loss: 1.8215 - val_acc: 0.6701
Epoch 15/50
1252/1252 [==============================] - ETA: 0s - loss: 0.0940 -
acc: 0.9693
Epoch 15: val_acc did not improve from 0.67400
1252/1252 [==============================] - 2874s 2s/step - loss:
0.0940 - acc: 0.9693 - val_loss: 1.7366 - val_acc: 0.6671
Epoch 16/50
1252/1252 [==============================] - ETA: 0s - loss: 0.0830 -
acc: 0.9725
Epoch 16: val_acc did not improve from 0.67400
1252/1252 [==============================] - 3014s 2s/step - loss:
```

```
0.0830 - acc: 0.9725 - val_loss: 1.9165 - val_acc: 0.6704
Epoch 16: early stopping

#como demora muito tempo vamos dar load de um test model

#Loading and testing the model
from tensorflow import keras
#model =
keras.models.load_model('models/CNN_modelS_with_data_augmentation.h5')

val_loss, val_acc = model.evaluate(validation_dataset)
print('val_acc:', val_acc)

313/313 [==============================] - 132s 420ms/step - loss:
1.9165 - acc: 0.6704
val_acc: 0.6704000234603882

#Displaying curves of loss and accuracy

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()
```


Training and validation accuracy
Training acc
Validation acc
1.0
0.9
0.8
0.7
0.6
0.5
0.4
2
4
6
8
10
12
14
16


Training and validation loss
Training loss
Validation loss
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
2
4
6
8
10
12
14
16