workbook2

December 27, 2022

1 Biomedical Image Analysis

[35]:
```
import os
import cv2
import pickle
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from tqdm import tqdm
from sklearn.preprocessing import OneHotEncoder
from sklearn.metrics import confusion_matrix
from keras.models import Model, load_model
from keras.layers import Dense, Input, Conv2D, MaxPool2D, Flatten
from keras.preprocessing.image import ImageDataGenerator
from sklearn.model_selection import train_test_split
from keras.applications.vgg16 import VGG16
from keras.layers import GlobalAveragePooling2D
from keras.layers import BatchNormalization
from keras.layers import Dropout
from keras.layers import Dense
from keras.optimizers import Adam
from keras.optimizers import Adagrad
from keras.callbacks import ReduceLROnPlateau
from keras.callbacks import ModelCheckpoint
from sklearn.metrics import ConfusionMatrixDisplay
```

[2]:
```
def load_normal(norm_path):
    norm_files = np.array(os.listdir(norm_path))
    norm_labels = np.array(['normal']*len(norm_files))

    norm_images = []
    for image in tqdm(norm_files):
        image = cv2.imread(norm_path + image)
        image = cv2.resize(image, dsize=(200,200))
        image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
        norm_images.append(image)
```

```
    norm_images = np.array(norm_images)
    print(len(norm_images))

    return norm_images, norm_labels
def load_pneumonia(pneu_path):
    pneu_files = np.array(os.listdir(pneu_path))
    pneu_labels = np.array([pneu_file.split('_')[1] for pneu_file in␣
 ↪pneu_files])

    pneu_images = []
    for image in tqdm(pneu_files):
        image = cv2.imread(pneu_path + image)
        image = cv2.resize(image, dsize=(200,200))

        image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
        pneu_images.append(image)

    pneu_images = np.array(pneu_images)
    X_train = np.append(norm_images, pneu_images, axis=0)
    y_train = np.append(norm_labels, pneu_labels)

    return pneu_images, pneu_labels
```

[4]:
```
norm_images, norm_labels = load_normal('/Users/marko/Downloads/chest_xray/train/
 ↪NORMAL/')
pneu_images, pneu_labels = load_pneumonia('/Users/marko/Downloads/chest_xray/
 ↪train/PNEUMONIA/')
```

```
100%|      | 1341/1341 [00:14<00:00, 92.90it/s]
```

```
1341
```

```
100%|      | 3875/3875 [00:12<00:00, 313.15it/s]
```

[5]:
```
X_train = np.append(norm_images, pneu_images, axis=0)
y_train = np.append(norm_labels, pneu_labels)
```

[6]:
```
def plot_images(X, y):
    fig, axes = plt.subplots(ncols=7, nrows=2, figsize=(16, 4))

    indices = np.random.choice(len(X), 14)
    counter = 0

    for i in range(2):
        for j in range(7):
            axes[i,j].set_title(y[indices[counter]])
            axes[i,j].imshow(X[indices[counter]], cmap='gray')
            axes[i,j].get_xaxis().set_visible(False)
```

```
            axes[i,j].get_yaxis().set_visible(False)
            counter += 1
    plt.show()
```

[8]:
```
norm_images_test, norm_labels_test = load_normal('/Users/marko/Downloads/
 ↪chest_xray/test/NORMAL/')
pneu_images_test, pneu_labels_test = load_pneumonia('/Users/marko/Downloads/
 ↪chest_xray/test/PNEUMONIA/')
```

```
100%|      | 234/234 [00:01<00:00, 130.37it/s]
```

```
234
```

```
100%|      | 390/390 [00:01<00:00, 347.12it/s]
```

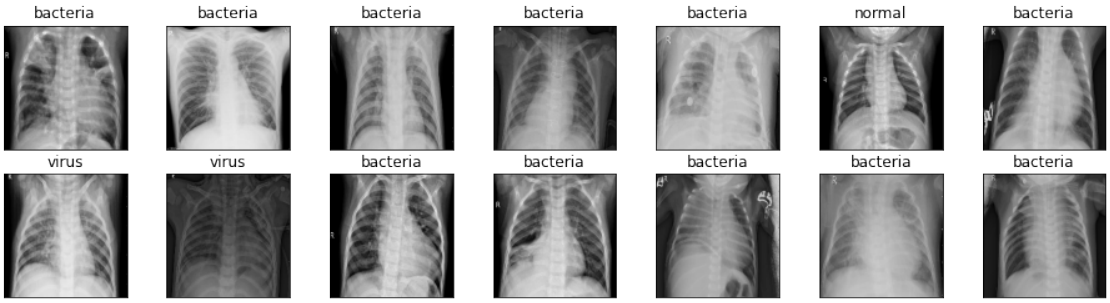
[9]:
```
X_test = np.append(norm_images_test, pneu_images_test, axis=0)
y_test = np.append(norm_labels_test, pneu_labels_test)
```

[10]:
```
with open('pneumonia_data.pickle', 'wb') as f:
    pickle.dump((X_train, X_test, y_train, y_test), f)# Use this to load␣
 ↪variables
with open('pneumonia_data.pickle', 'rb') as f:
    (X_train, X_test, y_train, y_test) = pickle.load(f)
```

[11]:
```
np.unique(y_train, return_counts=True)
```

[11]:
```
(array(['bacteria', 'normal', 'virus'], dtype='<U8'),
 array([2530, 1341, 1345]))
```

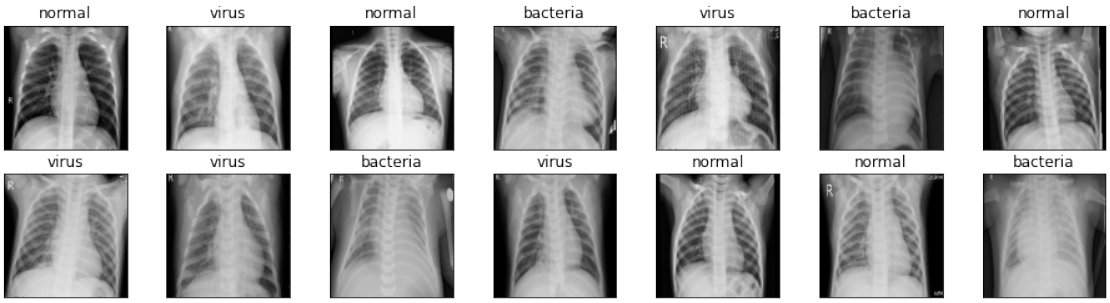
[12]:
```
plot_images(X_train, y_train)
```



[13]:
```
np.unique(y_train, return_counts=True)
```

[13]:
```
(array(['bacteria', 'normal', 'virus'], dtype='<U8'),
 array([2530, 1341, 1345]))
```

[14]:
```
plot_images(X_test, y_test)
```



[15]:
```
y_train = y_train[:, np.newaxis]
y_test = y_test[:, np.newaxis]
y_train
```

[15]:
```
array([['normal'],
       ['normal'],
       ['normal'],
       …,
       ['virus'],
       ['virus'],
       ['virus']], dtype='<U8')
```

[16]:
```
y_test
```

[16]:
```
array([['normal'],
       ['normal'],
       ['normal'],
       ['normal'],
       ['normal'],
       ['normal'],
       ['normal'],
       ['normal'],
       ['normal'],
       ['normal'],
       ['normal'],
       ['normal'],
       ['normal'],
       ['normal'],
       ['normal'],
       ['normal'],
       ['normal'],
       ['normal'],
       ['normal'],
       ['normal'],
```

```
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
```

```
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
```

```
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
```

```
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
```

```
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['normal'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['virus'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['virus'],
['virus'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['virus'],
```

```
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['virus'],
['virus'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['virus'],
['bacteria'],
['virus'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
```

```
['bacteria'],
['virus'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['virus'],
['bacteria'],
['virus'],
['virus'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['virus'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['virus'],
['bacteria'],
['bacteria'],
```

```
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['virus'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['virus'],
['virus'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['virus'],
['virus'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['virus'],
['bacteria'],
['virus'],
['bacteria'],
['virus'],
['virus'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
```

```
['virus'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['virus'],
['virus'],
['virus'],
['virus'],
['virus'],
['virus'],
['virus'],
['bacteria'],
['virus'],
['virus'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['virus'],
['virus'],
['bacteria'],
['virus'],
['bacteria'],
['virus'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['virus'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['virus'],
['bacteria'],
['virus'],
['bacteria'],
['virus'],
```

```
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['virus'],
['virus'],
['virus'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['virus'],
['virus'],
['virus'],
['virus'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
```

```
['bacteria'],
['virus'],
['virus'],
['virus'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['virus'],
['bacteria'],
['virus'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['virus'],
['virus'],
['virus'],
['virus'],
['virus'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['virus'],
['bacteria'],
```

```
['bacteria'],
['virus'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['virus'],
['virus'],
['virus'],
['bacteria'],
['bacteria'],
['virus'],
['virus'],
['virus'],
['bacteria'],
['virus'],
['virus'],
['bacteria'],
['bacteria'],
['virus'],
['virus'],
['bacteria'],
['bacteria'],
['virus'],
['virus'],
['bacteria'],
['virus'],
['virus'],
['bacteria'],
```

```
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['virus'],
['bacteria'],
['bacteria'],
['bacteria'],
['bacteria']], dtype='<U8')
```

[17]:
```
one_hot_encoder = OneHotEncoder(sparse=False, handle_unknown = 'error')
```

[18]:
```
y_train_one_hot = one_hot_encoder.fit_transform(y_train)
y_test_one_hot = one_hot_encoder.transform(y_test)
```

```
/Users/marko/opt/miniconda3/lib/python3.9/site-
packages/sklearn/preprocessing/_encoders.py:808: FutureWarning: `sparse` was
renamed to `sparse_output` in version 1.2 and will be removed in 1.4.
`sparse_output` is ignored unless you leave `sparse` to its default value.
  warnings.warn(
```

[19]:
```
X_train = X_train.reshape(X_train.shape[0], X_train.shape[1], X_train.shape[2],␣
↪1)
X_test = X_test.reshape(X_test.shape[0], X_test.shape[1], X_test.shape[2], 1)
```

[20]:
```
datagen = ImageDataGenerator(
        rotation_range = 10,
        zoom_range = 0.1,
        width_shift_range = 0.1,
        height_shift_range = 0.1)
```

[21]:
```
datagen.fit(X_train)
train_gen = datagen.flow(X_train, y_train_one_hot, batch_size=32)
```

[22]:
```
input1 = Input(shape=(X_train.shape[1], X_train.shape[2], 1))

cnn = Conv2D(16, (3, 3), activation='sigmoid', strides=(1, 1),
    padding='same')(input1)
cnn = Conv2D(32, (3, 3), activation='sigmoid', strides=(1, 1),
    padding='same')(cnn)
cnn = MaxPool2D((2, 2))(cnn)

cnn = Conv2D(16, (2, 2), activation='sigmoid', strides=(1, 1),
    padding='same')(cnn)
cnn = Conv2D(32, (2, 2), activation='sigmoid', strides=(1, 1),
    padding='same')(cnn)
cnn = MaxPool2D((2, 2))(cnn)

cnn = Flatten()(cnn)
cnn = Dense(100, activation='relu')(cnn)
cnn = Dense(50, activation='relu')(cnn)
output1 = Dense(3, activation='softmax')(cnn)

model = Model(inputs=input1, outputs=output1)
```

[23]:
```
model.summary()
```

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 200, 200, 1)]     0
```

```
conv2d (Conv2D)              (None, 200, 200, 16)      160

conv2d_1 (Conv2D)            (None, 200, 200, 32)      4640

max_pooling2d (MaxPooling2D  (None, 100, 100, 32)      0
)

conv2d_2 (Conv2D)            (None, 100, 100, 16)      2064

conv2d_3 (Conv2D)            (None, 100, 100, 32)      2080

max_pooling2d_1 (MaxPooling  (None, 50, 50, 32)        0
2D)

flatten (Flatten)            (None, 80000)             0

dense (Dense)                (None, 100)               8000100

dense_1 (Dense)              (None, 50)                5050

dense_2 (Dense)              (None, 3)                 153

=================================================================
Total params: 8,014,247
Trainable params: 8,014,247
Non-trainable params: 0
_________________________________________________________________
```

[24]:
```
model.compile(loss='categorical_crossentropy', optimizer='adam',␣
↪metrics=['acc'])
```

[25]:
```
history2 = model.fit_generator(train_gen, epochs=30, validation_data=(X_test,␣
↪y_test_one_hot))
```

```
Epoch 1/30

<ipython-input-25-b2e474c5510d>:1: UserWarning: `Model.fit_generator` is
deprecated and will be removed in a future version. Please use `Model.fit`,
which supports generators.
  history2 = model.fit_generator(train_gen, epochs=30, validation_data=(X_test,
y_test_one_hot))
```

```
---------------------------------------------------------------------------
NotFoundError                             Traceback (most recent call last)
Cell In[25], line 1
----> 1 history2 =␣
↪model.fit_generator(train_gen, epochs=30, validation_data=(X_test, y_test_one_hot))
```

```
File ~/opt/miniconda3/lib/python3.9/site-packages/keras/engine/training.py:2604
 ↪in Model.fit_generator(self, generator, steps_per_epoch, epochs, verbose,
 ↪callbacks, validation_data, validation_steps, validation_freq, class_weight,
 ↪max_queue_size, workers, use_multiprocessing, shuffle, initial_epoch)
   2592 """Fits the model on data yielded batch-by-batch by a Python generator.
   2593
   2594 DEPRECATED:
   2595   `Model.fit` now supports generators, so there is no longer any need t
   2596   use this endpoint.
   2597 """
   2598 warnings.warn(
   2599     "`Model.fit_generator` is deprecated and "
   2600     "will be removed in a future version. "
   2601     "Please use `Model.fit`, which supports generators.",
   2602     stacklevel=2,
   2603 )
-> 2604 return self.fit(
   2605     generator,
   2606     steps_per_epoch=steps_per_epoch,
   2607     epochs=epochs,
   2608     verbose=verbose,
   2609     callbacks=callbacks,
   2610     validation_data=validation_data,
   2611     validation_steps=validation_steps,
   2612     validation_freq=validation_freq,
   2613     class_weight=class_weight,
   2614     max_queue_size=max_queue_size,
   2615     workers=workers,
   2616     use_multiprocessing=use_multiprocessing,
   2617     shuffle=shuffle,
   2618     initial_epoch=initial_epoch,
   2619 )

File ~/opt/miniconda3/lib/python3.9/site-packages/keras/utils/traceback_utils.py
 ↪70, in filter_traceback.<locals>.error_handler(*args, **kwargs)
     67     filtered_tb = _process_traceback_frames(e.__traceback__)
     68     # To get the full stack trace, call:
     69     # `tf.debugging.disable_traceback_filtering()`
---> 70     raise e.with_traceback(filtered_tb) from None
     71 finally:
     72     del filtered_tb

File ~/opt/miniconda3/lib/python3.9/site-packages/tensorflow/python/eager/
 ↪execute.py:52, in quick_execute(op_name, num_outputs, inputs, attrs, ctx, nam
     50 try:
     51   ctx.ensure_initialized()
---> 52   tensors = pywrap_tfe.TFE_Py_Execute(ctx._handle, device_name, op_name
     53                                       inputs, attrs, num_outputs)
```

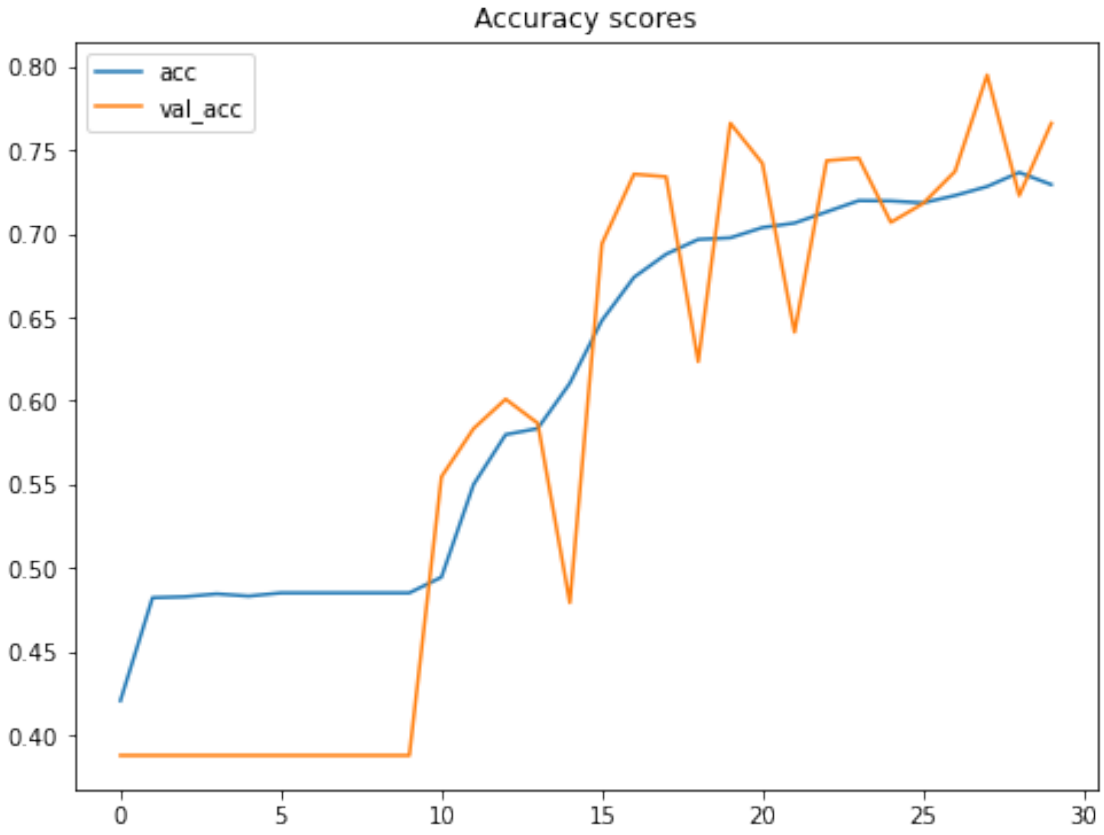
```
     54 except core._NotOkStatusException as e:
     55   if name is not None:

NotFoundError: Graph execution error:

Detected at node 'StatefulPartitionedCall_12' defined at (most recent call last):
    File "/Users/marko/opt/miniconda3/lib/python3.9/runpy.py", line 197, in␣
 ↪_run_module_as_main
      return _run_code(code, main_globals, None,
    File "/Users/marko/opt/miniconda3/lib/python3.9/runpy.py", line 87, in␣
 ↪_run_code
      exec(code, run_globals)
    File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/
 ↪ipykernel_launcher.py", line 16, in <module>
      app.launch_new_instance()
    File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/traitlets/
 ↪config/application.py", line 1041, in launch_instance
      app.start()
    File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/ipykernel/
 ↪kernelapp.py", line 619, in start
      self.io_loop.start()
    File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/tornado/
 ↪platform/asyncio.py", line 199, in start
      self.asyncio_loop.run_forever()
    File "/Users/marko/opt/miniconda3/lib/python3.9/asyncio/base_events.py",␣
 ↪line 601, in run_forever
      self._run_once()
    File "/Users/marko/opt/miniconda3/lib/python3.9/asyncio/base_events.py",␣
 ↪line 1905, in _run_once
      handle._run()
    File "/Users/marko/opt/miniconda3/lib/python3.9/asyncio/events.py", line 80␣
 ↪in _run
      self._context.run(self._callback, *self._args)
    File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/tornado/ioloo
 ↪py", line 688, in <lambda>
      lambda f: self._run_callback(functools.partial(callback, future))
    File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/tornado/ioloo
 ↪py", line 741, in _run_callback
      ret = callback()
    File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/tornado/gen.
 ↪py", line 814, in inner
      self.ctx_run(self.run)
    File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/tornado/gen.
 ↪py", line 775, in run
      yielded = self.gen.send(value)
    File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/ipykernel/
 ↪kernelbase.py", line 358, in process_one
      yield gen.maybe_future(dispatch(*args))
```

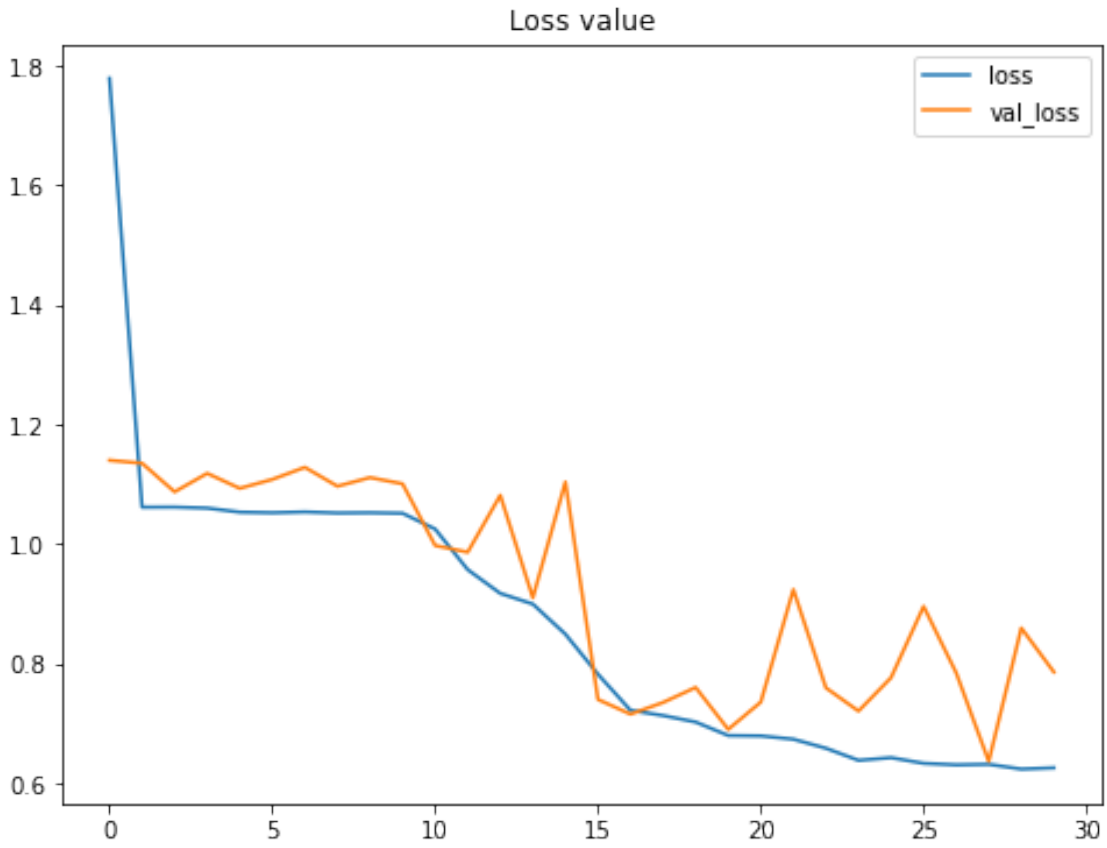
```
  File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/tornado/gen.
↪py", line 234, in wrapper
    yielded = ctx_run(next, result)
  File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/ipykernel/
↪kernelbase.py", line 261, in dispatch_shell
    yield gen.maybe_future(handler(stream, idents, msg))
  File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/tornado/gen.
↪py", line 234, in wrapper
    yielded = ctx_run(next, result)
  File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/ipykernel/
↪kernelbase.py", line 536, in execute_request
    self.do_execute(
  File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/tornado/gen.
↪py", line 234, in wrapper
    yielded = ctx_run(next, result)
  File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/ipykernel/
↪ipkernel.py", line 302, in do_execute
    res = shell.run_cell(code, store_history=store_history, silent=silent)
  File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/ipykernel/
↪zmqshell.py", line 539, in run_cell
    return super(ZMQInteractiveShell, self).run_cell(*args, **kwargs)
  File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/IPython/core/
↪interactiveshell.py", line 2940, in run_cell
    result = self._run_cell(
  File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/IPython/core/
↪interactiveshell.py", line 2995, in _run_cell
    return runner(coro)
  File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/IPython/core/
↪async_helpers.py", line 129, in _pseudo_sync_runner
    coro.send(None)
  File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/IPython/core/
↪interactiveshell.py", line 3194, in run_cell_async
    has_raised = await self.run_ast_nodes(code_ast.body, cell_name,
  File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/IPython/core/
↪interactiveshell.py", line 3373, in run_ast_nodes
    if await self.run_code(code, result, async_=asy):
  File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/IPython/core/
↪interactiveshell.py", line 3433, in run_code
    exec(code_obj, self.user_global_ns, self.user_ns)
  File "<ipython-input-25-b2e474c5510d>", line 1, in <module>
    history2 = model.fit_generator(train_gen, epochs=30,␣
↪validation_data=(X_test, y_test_one_hot))
  File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/keras/engine/
↪training.py", line 2604, in fit_generator
    return self.fit(
  File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/keras/utils/
↪traceback_utils.py", line 65, in error_handler
    return fn(*args, **kwargs)
```

```
    File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/keras/engine/
 ↪training.py", line 1650, in fit
      tmp_logs = self.train_function(iterator)
    File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/keras/engine/
 ↪training.py", line 1249, in train_function
      return step_function(self, iterator)
    File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/keras/engine/
 ↪training.py", line 1233, in step_function
      outputs = model.distribute_strategy.run(run_step, args=(data,))
    File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/keras/engine/
 ↪training.py", line 1222, in run_step
      outputs = model.train_step(data)
    File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/keras/engine/
 ↪training.py", line 1027, in train_step
      self.optimizer.minimize(loss, self.trainable_variables, tape=tape)
    File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/keras/
 ↪optimizers/optimizer_experimental/optimizer.py", line 527, in minimize
      self.apply_gradients(grads_and_vars)
    File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/keras/
 ↪optimizers/optimizer_experimental/optimizer.py", line 1140, in apply_gradient
      return super().apply_gradients(grads_and_vars, name=name)
    File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/keras/
 ↪optimizers/optimizer_experimental/optimizer.py", line 634, in apply_gradients
      iteration = self._internal_apply_gradients(grads_and_vars)
    File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/keras/
 ↪optimizers/optimizer_experimental/optimizer.py", line 1166, in␣
 ↪_internal_apply_gradients
      return tf.__internal__.distribute.interim.maybe_merge_call(
    File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/keras/
 ↪optimizers/optimizer_experimental/optimizer.py", line 1216, in␣
 ↪_distributed_apply_gradients_fn
      distribution.extended.update(
    File "/Users/marko/opt/miniconda3/lib/python3.9/site-packages/keras/
 ↪optimizers/optimizer_experimental/optimizer.py", line 1211, in␣
 ↪apply_grad_to_update_var
      return self._update_step_xla(grad, var, id(self._var_key(var)))
Node: 'StatefulPartitionedCall_12'
could not find registered platform with id: 0x170f6a7c0
         [[{{node StatefulPartitionedCall_12}}]] [Op:
 ↪__inference_train_function_1716]
```

[144]:
```
plt.figure(figsize=(8,6))
plt.title('Accuracy scores')
plt.plot(history2.history['acc'])
plt.plot(history2.history['val_acc'])
plt.legend(['acc', 'val_acc'])
plt.show()
```

```
plt.figure(figsize=(8,6))
plt.title('Loss value')
plt.plot(history2.history['loss'])
plt.plot(history2.history['val_loss'])
plt.legend(['loss', 'val_loss'])
plt.show()
```





[26]:
```
X_train, X_test, y_train_one_hot, y_test_one_hot = train_test_split(
  X_train, y_train_one_hot, test_size=0.2, random_state = 42)
```

[27]:
```
vgg16 = VGG16(weights='imagenet', include_top=False)
hdf5_save = 'VGG16_Model.hdf5'
annealer = ReduceLROnPlateau(
    monitor='val_accuracy', factor=0.70, patience=5,
    verbose=1, min_lr=1e-4)

checkpoint = ModelCheckpoint(hdf5_save, verbose=1, save_best_only=True)

datagen2 = ImageDataGenerator(rotation_range=360,
                              width_shift_range=0.2,
                              height_shift_range=0.2,
                              zoom_range=0.2,
                              horizontal_flip=True,
                              vertical_flip=True)
datagen2.fit(X_train)
train_gen2 = datagen.flow(X_train, y_train_one_hot, batch_size=32)
```

```
input = Input(shape=(X_train.shape[1], X_train.shape[2], 1))
x = Conv2D(3, (3, 3), padding='same')(input)

x = vgg16(x)

x = GlobalAveragePooling2D()(x)
x = BatchNormalization()(x)
x = Dropout(0.5)(x)
x = Dense(200, activation='relu')(x)
x = BatchNormalization()(x)
x = Dropout(0.5)(x)

output = Dense(3, activation='softmax', name='root')(x)
model2 = Model(input, output)

optimizer = Adam(lr=0.003, beta_1=0.9, beta_2=0.999,
                     epsilon=0.1, decay=0.0)

model2.compile(loss='categorical_crossentropy',
                  optimizer=optimizer, metrics=['accuracy'])

model2.summary()

history = model2.fit_generator(train_gen2, epochs=30, validation_data=(X_test,␣
 ↪y_test_one_hot))
```

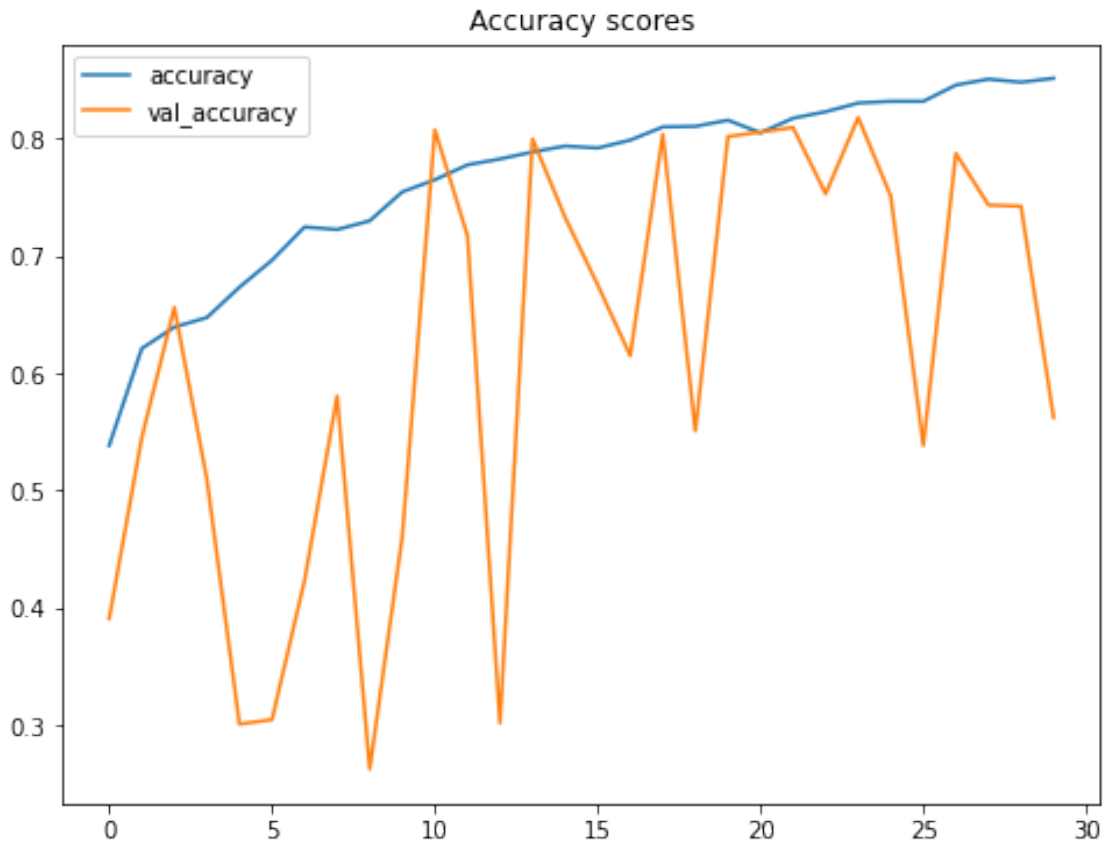
```
Downloading data from https://storage.googleapis.com/tensorflow/keras-
applications/vgg16/vgg16_weights_tf_dim_ordering_tf_kernels_notop.h5
58889256/58889256 [==============================] - 6s 0us/step
Model: "model_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_3 (InputLayer)        [(None, 200, 200, 1)]     0

 conv2d_4 (Conv2D)           (None, 200, 200, 3)       30

 vgg16 (Functional)          (None, None, None, 512)   14714688

 global_average_pooling2d (G  (None, 512)              0
 lobalAveragePooling2D)

 batch_normalization (BatchN  (None, 512)              2048
 ormalization)
```

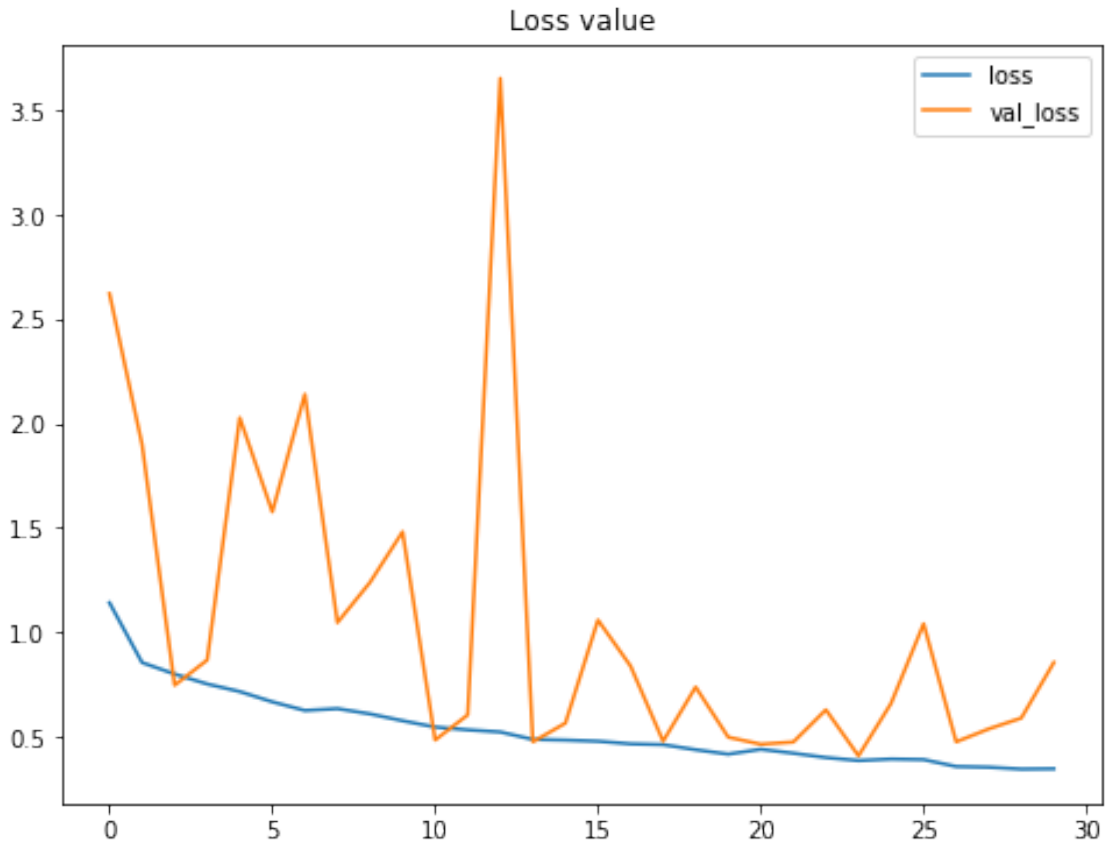
```
 dropout (Dropout)           (None, 512)               0

 dense_3 (Dense)             (None, 200)               102600

 batch_normalization_1 (Batc  (None, 200)              800
 hNormalization)

 dropout_1 (Dropout)         (None, 200)               0

 root (Dense)                (None, 3)                 603

=================================================================
Total params: 14,820,769
Trainable params: 14,819,345
Non-trainable params: 1,424

_________________________________________________________________
Epoch 1/30

/Users/marko/opt/miniconda3/lib/python3.9/site-
packages/keras/optimizers/optimizer_v2/adam.py:117: UserWarning: The `lr`
argument is deprecated, use `learning_rate` instead.
  super().__init__(name, **kwargs)
<ipython-input-27-40bc877d401a>:43: UserWarning: `Model.fit_generator` is
deprecated and will be removed in a future version. Please use `Model.fit`,
which supports generators.
  history = model2.fit_generator(train_gen2, epochs=30, validation_data=(X_test,
y_test_one_hot))

131/131 [==============================] - 329s 3s/step - loss: 1.1410 -
accuracy: 0.5384 - val_loss: 2.6226 - val_accuracy: 0.3908
Epoch 2/30
131/131 [==============================] - 327s 2s/step - loss: 0.8542 -
accuracy: 0.6210 - val_loss: 1.8998 - val_accuracy: 0.5450
Epoch 3/30
131/131 [==============================] - 317s 2s/step - loss: 0.7991 -
accuracy: 0.6393 - val_loss: 0.7455 - val_accuracy: 0.6561
Epoch 4/30
131/131 [==============================] - 237s 2s/step - loss: 0.7515 -
accuracy: 0.6474 - val_loss: 0.8664 - val_accuracy: 0.5096
Epoch 5/30
131/131 [==============================] - 240s 2s/step - loss: 0.7152 -
accuracy: 0.6733 - val_loss: 2.0276 - val_accuracy: 0.3008
Epoch 6/30
131/131 [==============================] - 236s 2s/step - loss: 0.6659 -
accuracy: 0.6963 - val_loss: 1.5768 - val_accuracy: 0.3046
Epoch 7/30
131/131 [==============================] - 239s 2s/step - loss: 0.6246 -
accuracy: 0.7246 - val_loss: 2.1402 - val_accuracy: 0.4234
Epoch 8/30
```

```
131/131 [==============================] - 235s 2s/step - loss: 0.6332 -
accuracy: 0.7224 - val_loss: 1.0459 - val_accuracy: 0.5805
Epoch 9/30
131/131 [==============================] - 236s 2s/step - loss: 0.6081 -
accuracy: 0.7299 - val_loss: 1.2386 - val_accuracy: 0.2625
Epoch 10/30
131/131 [==============================] - 230s 2s/step - loss: 0.5753 -
accuracy: 0.7546 - val_loss: 1.4809 - val_accuracy: 0.4607
Epoch 11/30
131/131 [==============================] - 232s 2s/step - loss: 0.5450 -
accuracy: 0.7649 - val_loss: 0.4828 - val_accuracy: 0.8075
Epoch 12/30
131/131 [==============================] - 1257s 10s/step - loss: 0.5317 -
accuracy: 0.7776 - val_loss: 0.6033 - val_accuracy: 0.7174
Epoch 13/30
131/131 [==============================] - 233s 2s/step - loss: 0.5215 -
accuracy: 0.7826 - val_loss: 3.6547 - val_accuracy: 0.3017
Epoch 14/30
131/131 [==============================] - 233s 2s/step - loss: 0.4858 -
accuracy: 0.7886 - val_loss: 0.4742 - val_accuracy: 0.7998
Epoch 15/30
131/131 [==============================] - 224s 2s/step - loss: 0.4826 -
accuracy: 0.7936 - val_loss: 0.5638 - val_accuracy: 0.7328
Epoch 16/30
131/131 [==============================] - 231s 2s/step - loss: 0.4771 -
accuracy: 0.7919 - val_loss: 1.0580 - val_accuracy: 0.6753
Epoch 17/30
131/131 [==============================] - 232s 2s/step - loss: 0.4645 -
accuracy: 0.7987 - val_loss: 0.8378 - val_accuracy: 0.6149
Epoch 18/30
131/131 [==============================] - 236s 2s/step - loss: 0.4607 -
accuracy: 0.8099 - val_loss: 0.4772 - val_accuracy: 0.8036
Epoch 19/30
131/131 [==============================] - 232s 2s/step - loss: 0.4364 -
accuracy: 0.8104 - val_loss: 0.7370 - val_accuracy: 0.5508
Epoch 20/30
131/131 [==============================] - 232s 2s/step - loss: 0.4152 -
accuracy: 0.8157 - val_loss: 0.4965 - val_accuracy: 0.8017
Epoch 21/30
131/131 [==============================] - 232s 2s/step - loss: 0.4385 -
accuracy: 0.8049 - val_loss: 0.4627 - val_accuracy: 0.8056
Epoch 22/30
131/131 [==============================] - 232s 2s/step - loss: 0.4196 -
accuracy: 0.8174 - val_loss: 0.4734 - val_accuracy: 0.8094
Epoch 23/30
131/131 [==============================] - 232s 2s/step - loss: 0.3988 -
accuracy: 0.8229 - val_loss: 0.6278 - val_accuracy: 0.7529
Epoch 24/30
```

```
131/131 [==============================] - 230s 2s/step - loss: 0.3844 -
accuracy: 0.8303 - val_loss: 0.4071 - val_accuracy: 0.8180
Epoch 25/30
131/131 [==============================] - 231s 2s/step - loss: 0.3918 -
accuracy: 0.8317 - val_loss: 0.6586 - val_accuracy: 0.7510
Epoch 26/30
131/131 [==============================] - 232s 2s/step - loss: 0.3891 -
accuracy: 0.8317 - val_loss: 1.0400 - val_accuracy: 0.5383
Epoch 27/30
131/131 [==============================] - 232s 2s/step - loss: 0.3560 -
accuracy: 0.8456 - val_loss: 0.4743 - val_accuracy: 0.7874
Epoch 28/30
131/131 [==============================] - 232s 2s/step - loss: 0.3526 -
accuracy: 0.8507 - val_loss: 0.5363 - val_accuracy: 0.7433
Epoch 29/30
131/131 [==============================] - 232s 2s/step - loss: 0.3437 -
accuracy: 0.8480 - val_loss: 0.5883 - val_accuracy: 0.7423
Epoch 30/30
131/131 [==============================] - 232s 2s/step - loss: 0.3451 -
accuracy: 0.8514 - val_loss: 0.8548 - val_accuracy: 0.5623
```

[28]:
```
plt.figure(figsize=(8,6))
plt.title('Accuracy scores')
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.legend(['accuracy', 'val_accuracy'])
plt.show()
plt.figure(figsize=(8,6))
plt.title('Loss value')
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.legend(['loss', 'val_loss'])
plt.show()
```


Accuracy scores
accuracy
val_accuracy
0.8
0.7
0.6
0.5
0.4
0.3
0
5
10
15
20
25
30



[36]:
```
input3 = Input(shape=(X_train.shape[1], X_train.shape[2], 1))

cnn2 = Conv2D(8, (3, 3), activation='tanh', strides=(1, 1),
    padding='same')(input3)
cnn2 = Conv2D(16, (3, 3), activation='tanh', strides=(1, 1),
    padding='same')(cnn2)
cnn2 = MaxPool2D((2, 2))(cnn2)

cnn2 = Conv2D(16, (2, 2), activation='tanh', strides=(1, 1),
    padding='same')(cnn2)
cnn2 = Conv2D(32, (2, 2), activation='tanh', strides=(1, 1),
    padding='same')(cnn2)
cnn2 = MaxPool2D((2, 2))(cnn2)

cnn2 = Flatten()(cnn2)
cnn2 = Dense(50, activation='relu')(cnn2)
cnn2 = Dense(25, activation='relu')(cnn2)
output3 = Dense(3, activation='softmax')(cnn2)

model3 = Model(inputs=input3, outputs=output3)
```

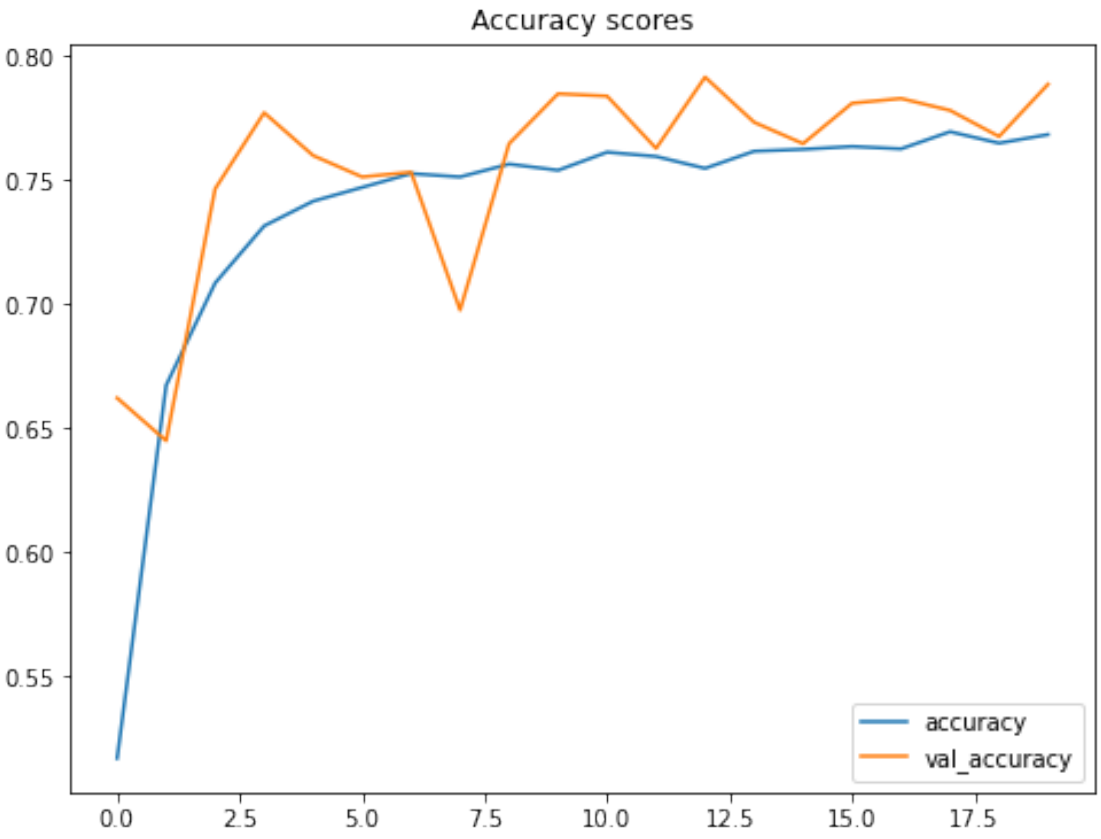
```
optimizer2 = Adagrad(learning_rate=0.01, initial_accumulator_value=0.1,␣
 ↪epsilon=1e-07)

model3.compile(loss='categorical_crossentropy',
                  optimizer=optimizer2, metrics=['accuracy'])

model3.summary()

history2 = model3.fit_generator(train_gen, epochs=20, validation_data=(X_test,␣
 ↪y_test_one_hot))
```

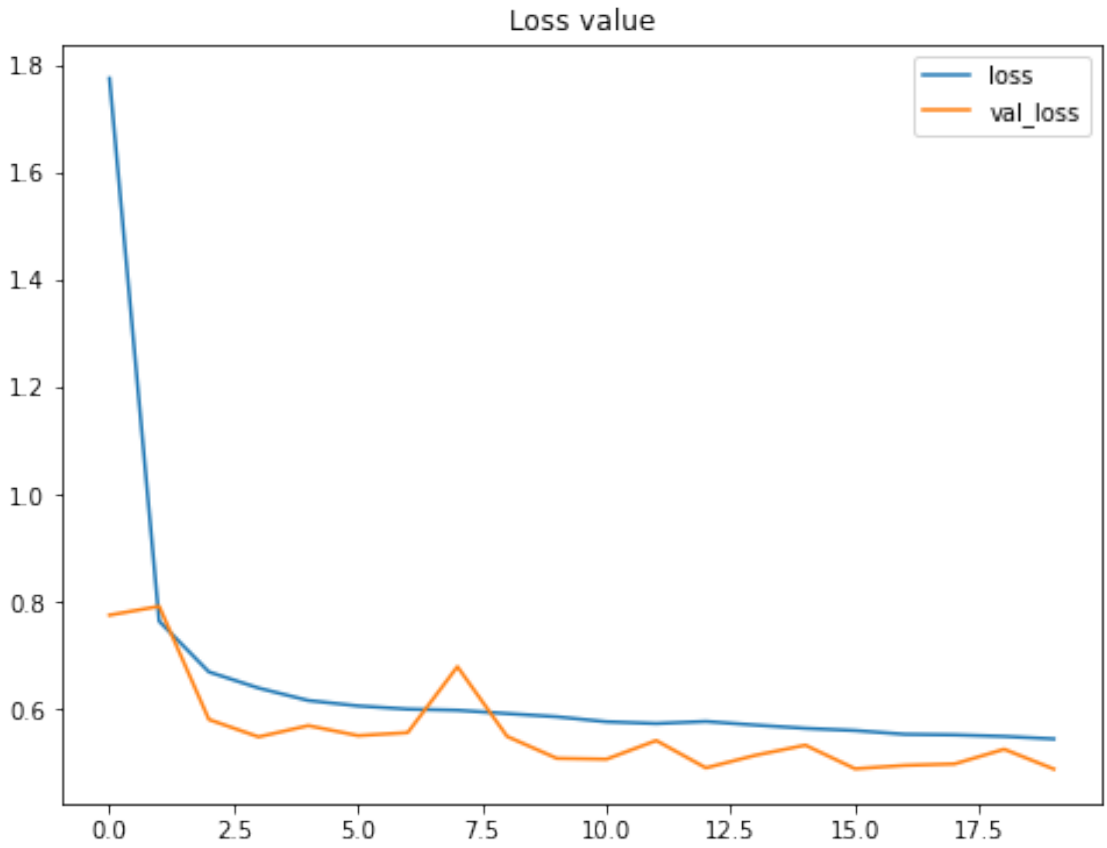
```
Model: "model_8"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_10 (InputLayer)       [(None, 200, 200, 1)]     0

 conv2d_29 (Conv2D)          (None, 200, 200, 8)       80

 conv2d_30 (Conv2D)          (None, 200, 200, 16)      1168

 max_pooling2d_14 (MaxPoolin  (None, 100, 100, 16)     0
 g2D)

 conv2d_31 (Conv2D)          (None, 100, 100, 16)      1040

 conv2d_32 (Conv2D)          (None, 100, 100, 32)      2080

 max_pooling2d_15 (MaxPoolin  (None, 50, 50, 32)       0
 g2D)

 flatten_7 (Flatten)         (None, 80000)             0

 dense_22 (Dense)            (None, 50)                4000050

 dense_23 (Dense)            (None, 25)                1275

 dense_24 (Dense)            (None, 3)                 78

=================================================================
Total params: 4,005,771
Trainable params: 4,005,771
Non-trainable params: 0
_________________________________________________________________
Epoch 1/20
<ipython-input-36-0585672046e2>:29: UserWarning: `Model.fit_generator` is
```

```
deprecated and will be removed in a future version. Please use `Model.fit`,
which supports generators.
  history2 = model3.fit_generator(train_gen, epochs=20, validation_data=(X_test,
y_test_one_hot))

163/163 [==============================] - 18s 102ms/step - loss: 1.7750 -
accuracy: 0.5167 - val_loss: 0.7745 - val_accuracy: 0.6619
Epoch 2/20
163/163 [==============================] - 16s 100ms/step - loss: 0.7631 -
accuracy: 0.6670 - val_loss: 0.7909 - val_accuracy: 0.6446
Epoch 3/20
163/163 [==============================] - 17s 101ms/step - loss: 0.6687 -
accuracy: 0.7082 - val_loss: 0.5800 - val_accuracy: 0.7462
Epoch 4/20
163/163 [==============================] - 16s 99ms/step - loss: 0.6386 -
accuracy: 0.7312 - val_loss: 0.5476 - val_accuracy: 0.7768
Epoch 5/20
163/163 [==============================] - 16s 99ms/step - loss: 0.6154 -
accuracy: 0.7412 - val_loss: 0.5686 - val_accuracy: 0.7596
Epoch 6/20
163/163 [==============================] - 16s 99ms/step - loss: 0.6053 -
accuracy: 0.7467 - val_loss: 0.5499 - val_accuracy: 0.7510
Epoch 7/20
163/163 [==============================] - 16s 99ms/step - loss: 0.5993 -
accuracy: 0.7523 - val_loss: 0.5555 - val_accuracy: 0.7529
Epoch 8/20
163/163 [==============================] - 16s 99ms/step - loss: 0.5972 -
accuracy: 0.7510 - val_loss: 0.6785 - val_accuracy: 0.6973
Epoch 9/20
163/163 [==============================] - 16s 100ms/step - loss: 0.5910 -
accuracy: 0.7561 - val_loss: 0.5484 - val_accuracy: 0.7644
Epoch 10/20
163/163 [==============================] - 16s 99ms/step - loss: 0.5851 -
accuracy: 0.7536 - val_loss: 0.5076 - val_accuracy: 0.7845
Epoch 11/20
163/163 [==============================] - 16s 99ms/step - loss: 0.5757 -
accuracy: 0.7609 - val_loss: 0.5060 - val_accuracy: 0.7835
Epoch 12/20
163/163 [==============================] - 16s 99ms/step - loss: 0.5725 -
accuracy: 0.7592 - val_loss: 0.5408 - val_accuracy: 0.7625
Epoch 13/20
163/163 [==============================] - 16s 99ms/step - loss: 0.5763 -
accuracy: 0.7544 - val_loss: 0.4898 - val_accuracy: 0.7912
Epoch 14/20
163/163 [==============================] - 16s 99ms/step - loss: 0.5696 -
accuracy: 0.7613 - val_loss: 0.5134 - val_accuracy: 0.7730
Epoch 15/20
163/163 [==============================] - 16s 99ms/step - loss: 0.5635 -
```

```
accuracy: 0.7621 - val_loss: 0.5322 - val_accuracy: 0.7644
Epoch 16/20
163/163 [==============================] - 16s 100ms/step - loss: 0.5594 -
accuracy: 0.7632 - val_loss: 0.4882 - val_accuracy: 0.7807
Epoch 17/20
163/163 [==============================] - 17s 101ms/step - loss: 0.5524 -
accuracy: 0.7623 - val_loss: 0.4945 - val_accuracy: 0.7826
Epoch 18/20
163/163 [==============================] - 16s 100ms/step - loss: 0.5513 -
accuracy: 0.7692 - val_loss: 0.4972 - val_accuracy: 0.7778
Epoch 19/20
163/163 [==============================] - 17s 101ms/step - loss: 0.5485 -
accuracy: 0.7646 - val_loss: 0.5246 - val_accuracy: 0.7672
Epoch 20/20
163/163 [==============================] - 16s 100ms/step - loss: 0.5438 -
accuracy: 0.7680 - val_loss: 0.4876 - val_accuracy: 0.7883
```

[38]:
```
plt.figure(figsize=(8,6))
plt.title('Accuracy scores')
plt.plot(history2.history['accuracy'])
plt.plot(history2.history['val_accuracy'])
plt.legend(['accuracy', 'val_accuracy'])
plt.show()
plt.figure(figsize=(8,6))
plt.title('Loss value')
plt.plot(history2.history['loss'])
plt.plot(history2.history['val_loss'])
plt.legend(['loss', 'val_loss'])
plt.show()
```


Accuracy scores
0.80
0.75
0.70
0.65
0.60
0.55
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
accuracy
val_accuracy


Loss value
loss
val_loss
1.8
1.6
1.4
1.2
1.0
0.8
0.6
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5

[ ]: