C1_W5_Lab_1_exploring-callbacks

December 25, 2022

1 Ungraded Lab: Introduction to Keras callbacks

In Keras, `Callback` is a Python class meant to be subclassed to provide specific functionality, with a set of methods called at various stages of training (including batch/epoch start and ends), testing, and predicting. Callbacks are useful to get a view on internal states and statistics of the model during training. The methods of the callbacks can be called at different stages of training/evaluating/inference. Keras has available callbacks and we'll show how you can use it in the following sections. Please click the **Open in Colab** badge above to complete this exercise in Colab. This will allow you to take advantage of the free GPU runtime (for faster training) and compatibility with all the packages needed in this notebook.

1.1 Model methods that take callbacks

Users can supply a list of callbacks to the following `tf.keras.Model` methods: * `fit()`, `fit_generator()` Trains the model for a fixed number of epochs (iterations over a dataset, or data yielded batch-by-batch by a Python generator). * `evaluate()`, `evaluate_generator()` Evaluates the model for given data or data generator. Outputs the loss and metric values from the evaluation. * `predict()`, `predict_generator()` Generates output predictions for the input data or data generator.

1.2 Imports

[1]:
```
from __future__ import absolute_import, division, print_function,␣
↪unicode_literals

try:
    # %tensorflow_version only exists in Colab.
    %tensorflow_version 2.x
except Exception:
    pass

import tensorflow as tf
import tensorflow_datasets as tfds
import matplotlib.pyplot as plt
import io
from PIL import Image
```

```
from tensorflow.keras.callbacks import TensorBoard, EarlyStopping,␣
 ↪LearningRateScheduler, ModelCheckpoint, CSVLogger, ReduceLROnPlateau
%load_ext tensorboard

import os
import matplotlib.pylab as plt
import numpy as np
import math
import datetime
import pandas as pd

print("Version: ", tf.__version__)
tf.get_logger().setLevel('INFO')
```

```
Version:  2.1.0
```

2 Examples of Keras callback applications

The following section will guide you through creating simple Callback applications.

[2]:
```
# Download and prepare the horses or humans dataset

# horses_or_humans 3.0.0 has already been downloaded for you
path = "./tensorflow_datasets"
splits, info = tfds.load('horses_or_humans', data_dir=path, as_supervised=True,␣
 ↪with_info=True, split=['train[:80%]', 'train[80%:]', 'test'])

(train_examples, validation_examples, test_examples) = splits

num_examples = info.splits['train'].num_examples
num_classes = info.features['label'].num_classes
```

[3]:
```
SIZE = 150 #@param {type:"slider", min:64, max:300, step:1}
IMAGE_SIZE = (SIZE, SIZE)
```

[4]:
```
def format_image(image, label):
  image = tf.image.resize(image, IMAGE_SIZE) / 255.0
  return  image, label
```

[5]:
```
BATCH_SIZE = 32 #@param {type:"integer"}
```

[6]:
```
train_batches = train_examples.shuffle(num_examples // 4).map(format_image).
 ↪batch(BATCH_SIZE).prefetch(1)
validation_batches = validation_examples.map(format_image).batch(BATCH_SIZE).
 ↪prefetch(1)
```

```
test_batches = test_examples.map(format_image).batch(1)
```

[7]:
```
for image_batch, label_batch in train_batches.take(1):
  pass

image_batch.shape
```

[7]:
```
TensorShape([32, 150, 150, 3])
```

[8]:
```
def build_model(dense_units, input_shape=IMAGE_SIZE + (3,)):
  model = tf.keras.models.Sequential([
      tf.keras.layers.Conv2D(16, (3, 3), activation='relu',␣
 ↪input_shape=input_shape),
      tf.keras.layers.MaxPooling2D(2, 2),
      tf.keras.layers.Conv2D(32, (3, 3), activation='relu'),
      tf.keras.layers.MaxPooling2D(2, 2),
      tf.keras.layers.Conv2D(64, (3, 3), activation='relu'),
      tf.keras.layers.MaxPooling2D(2, 2),
      tf.keras.layers.Flatten(),
      tf.keras.layers.Dense(dense_units, activation='relu'),
      tf.keras.layers.Dense(2, activation='softmax')
  ])
  return model
```

2.1 TensorBoard

Enable visualizations for TensorBoard.

[9]:
```
!rm -rf logs
```

[10]:
```
model = build_model(dense_units=256)
model.compile(
    optimizer='sgd',
    loss='sparse_categorical_crossentropy',
    metrics=['accuracy'])

logdir = os.path.join("logs", datetime.datetime.now().strftime("%Y%m%d-%H%M%S"))
tensorboard_callback = tf.keras.callbacks.TensorBoard(logdir)

model.fit(train_batches,
          epochs=10,
          validation_data=validation_batches,
          callbacks=[tensorboard_callback])
```

```
Epoch 1/10
26/26 [==============================] - 22s 845ms/step - loss: 0.6646 -
accuracy: 0.5864 - val_loss: 0.6241 - val_accuracy: 0.6927
```

```
Epoch 2/10
26/26 [==============================] - 21s 807ms/step - loss: 0.5921 -
accuracy: 0.7251 - val_loss: 0.5690 - val_accuracy: 0.7024
Epoch 3/10
26/26 [==============================] - 21s 815ms/step - loss: 0.5270 -
accuracy: 0.7567 - val_loss: 0.4896 - val_accuracy: 0.7610
Epoch 4/10
26/26 [==============================] - 21s 815ms/step - loss: 0.4610 -
accuracy: 0.7749 - val_loss: 0.4670 - val_accuracy: 0.8293
Epoch 5/10
26/26 [==============================] - 21s 812ms/step - loss: 0.4091 -
accuracy: 0.8212 - val_loss: 0.3500 - val_accuracy: 0.8634
Epoch 6/10
26/26 [==============================] - 21s 811ms/step - loss: 0.3378 -
accuracy: 0.8735 - val_loss: 0.3078 - val_accuracy: 0.8829
Epoch 7/10
26/26 [==============================] - 20s 781ms/step - loss: 0.2726 -
accuracy: 0.9051 - val_loss: 0.2119 - val_accuracy: 0.9610
Epoch 8/10
26/26 [==============================] - 20s 773ms/step - loss: 0.2211 -
accuracy: 0.9392 - val_loss: 0.1587 - val_accuracy: 0.9902
Epoch 9/10
26/26 [==============================] - 20s 769ms/step - loss: 0.2037 -
accuracy: 0.9367 - val_loss: 0.1350 - val_accuracy: 0.9805
Epoch 10/10
26/26 [==============================] - 20s 777ms/step - loss: 0.1390 -
accuracy: 0.9672 - val_loss: 0.0975 - val_accuracy: 0.9902
```

[10]:
```
<tensorflow.python.keras.callbacks.History at 0x7f74ea061cd0>
```

[11]:
```
%tensorboard --logdir logs
```

```
<IPython.core.display.HTML object>
```

2.2 Model Checkpoint

Callback to save the Keras model or model weights at some frequency.

[12]:
```
model = build_model(dense_units=256)
model.compile(
    optimizer='sgd',
    loss='sparse_categorical_crossentropy',
    metrics=['accuracy'])

model.fit(train_batches,
          epochs=5,
```

```
          validation_data=validation_batches,
          verbose=2,
          callbacks=[ModelCheckpoint('weights.{epoch:02d}-{val_loss:.2f}.h5',␣
↪verbose=1),
          ])
```

```
Epoch 1/5

Epoch 00001: saving model to weights.01-0.69.h5
26/26 - 21s - loss: 0.6707 - accuracy: 0.5803 - val_loss: 0.6948 - val_accuracy:
0.4976
Epoch 2/5

Epoch 00002: saving model to weights.02-0.59.h5
26/26 - 20s - loss: 0.6344 - accuracy: 0.6411 - val_loss: 0.5940 - val_accuracy:
0.7659
Epoch 3/5

Epoch 00003: saving model to weights.03-0.55.h5
26/26 - 20s - loss: 0.5695 - accuracy: 0.7129 - val_loss: 0.5474 - val_accuracy:
0.7659
Epoch 4/5

Epoch 00004: saving model to weights.04-0.51.h5
26/26 - 20s - loss: 0.5084 - accuracy: 0.7470 - val_loss: 0.5108 - val_accuracy:
0.7854
Epoch 5/5

Epoch 00005: saving model to weights.05-0.44.h5
26/26 - 20s - loss: 0.4785 - accuracy: 0.7749 - val_loss: 0.4418 - val_accuracy:
0.8098
```

[12]: <tensorflow.python.keras.callbacks.History at 0x7f74e9eaae50>

[13]:
```
model = build_model(dense_units=256)
model.compile(
    optimizer='sgd',
    loss='sparse_categorical_crossentropy',
    metrics=['accuracy'])

model.fit(train_batches,
          epochs=1,
          validation_data=validation_batches,
          verbose=2,
          callbacks=[ModelCheckpoint('saved_model', verbose=1)
          ])
```

```
Epoch 00001: saving model to saved_model
WARNING:tensorflow:From /opt/conda/lib/python3.7/site-
packages/tensorflow_core/python/ops/resource_variable_ops.py:1786: calling
BaseResourceVariable.__init__ (from tensorflow.python.ops.resource_variable_ops)
with constraint is deprecated and will be removed in a future version.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.

WARNING:tensorflow:From /opt/conda/lib/python3.7/site-
packages/tensorflow_core/python/ops/resource_variable_ops.py:1786: calling
BaseResourceVariable.__init__ (from tensorflow.python.ops.resource_variable_ops)
with constraint is deprecated and will be removed in a future version.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.

INFO:tensorflow:Assets written to: saved_model/assets

INFO:tensorflow:Assets written to: saved_model/assets

26/26 - 21s - loss: 0.6751 - accuracy: 0.5669 - val_loss: 0.6732 - val_accuracy:
0.4683
```

[13]:
```
<tensorflow.python.keras.callbacks.History at 0x7f74e9de3e50>
```

[14]:
```
model = build_model(dense_units=256)
model.compile(
    optimizer='sgd',
    loss='sparse_categorical_crossentropy',
    metrics=['accuracy'])

model.fit(train_batches,
          epochs=2,
          validation_data=validation_batches,
          verbose=2,
          callbacks=[ModelCheckpoint('model.h5', verbose=1)
          ])
```

```
Epoch 1/2

Epoch 00001: saving model to model.h5
26/26 - 21s - loss: 0.6628 - accuracy: 0.5791 - val_loss: 0.7176 - val_accuracy:
0.4341
Epoch 2/2

Epoch 00002: saving model to model.h5
26/26 - 22s - loss: 0.6064 - accuracy: 0.6946 - val_loss: 0.5894 - val_accuracy:
0.7415
```

[14]:
```
<tensorflow.python.keras.callbacks.History at 0x7f74e935e590>
```

2.3 Early stopping

Stop training when a monitored metric has stopped improving.

[15]:
```
model = build_model(dense_units=256)
model.compile(
    optimizer='sgd',
    loss='sparse_categorical_crossentropy',
    metrics=['accuracy'])

model.fit(train_batches,
          epochs=50,
          validation_data=validation_batches,
          verbose=2,
          callbacks=[EarlyStopping(
              patience=3,
              min_delta=0.05,
              baseline=0.8,
              mode='min',
              monitor='val_loss',
              restore_best_weights=True,
              verbose=1)
          ])
```

```
Epoch 1/50
26/26 - 21s - loss: 0.6684 - accuracy: 0.5888 - val_loss: 0.6276 - val_accuracy:
0.7854
Epoch 2/50
26/26 - 21s - loss: 0.6007 - accuracy: 0.7141 - val_loss: 0.6924 - val_accuracy:
0.4780
Epoch 3/50
26/26 - 20s - loss: 0.5307 - accuracy: 0.7822 - val_loss: 0.4826 - val_accuracy:
0.8341
Epoch 4/50
26/26 - 20s - loss: 0.4482 - accuracy: 0.8224 - val_loss: 0.6085 - val_accuracy:
0.5902
Epoch 5/50
26/26 - 20s - loss: 0.3898 - accuracy: 0.8552 - val_loss: 0.3459 - val_accuracy:
0.8927
Epoch 6/50
26/26 - 20s - loss: 0.2961 - accuracy: 0.8881 - val_loss: 0.2351 - val_accuracy:
0.9463
Epoch 7/50
26/26 - 20s - loss: 0.2402 - accuracy: 0.9367 - val_loss: 0.1969 - val_accuracy:
0.9463
Epoch 8/50
26/26 - 20s - loss: 0.1785 - accuracy: 0.9550 - val_loss: 0.1520 - val_accuracy:
0.9610
```

```
Epoch 9/50
26/26 - 20s - loss: 0.1538 - accuracy: 0.9623 - val_loss: 0.1555 - val_accuracy:
0.9415
Epoch 10/50
26/26 - 20s - loss: 0.1372 - accuracy: 0.9635 - val_loss: 0.1556 - val_accuracy:
0.9366
Epoch 11/50
Restoring model weights from the end of the best epoch.
26/26 - 20s - loss: 0.0994 - accuracy: 0.9781 - val_loss: 0.1177 - val_accuracy:
0.9561
Epoch 00011: early stopping
```

[15]: `<tensorflow.python.keras.callbacks.History at 0x7f74e92e0a50>`

2.4 CSV Logger

Callback that streams epoch results to a CSV file.

[16]:
```
model = build_model(dense_units=256)
model.compile(
    optimizer='sgd',
    loss='sparse_categorical_crossentropy',
    metrics=['accuracy'])

csv_file = 'training.csv'

model.fit(train_batches,
          epochs=5,
          validation_data=validation_batches,
          callbacks=[CSVLogger(csv_file)
          ])
```

```
Epoch 1/5
26/26 [==============================] - 20s 773ms/step - loss: 0.6732 -
accuracy: 0.5742 - val_loss: 0.6482 - val_accuracy: 0.8146
Epoch 2/5
26/26 [==============================] - 20s 761ms/step - loss: 0.6239 -
accuracy: 0.6813 - val_loss: 0.6040 - val_accuracy: 0.6439
Epoch 3/5
26/26 [==============================] - 20s 766ms/step - loss: 0.5864 -
accuracy: 0.7019 - val_loss: 0.6893 - val_accuracy: 0.4927
Epoch 4/5
26/26 [==============================] - 20s 761ms/step - loss: 0.5391 -
accuracy: 0.7336 - val_loss: 0.4656 - val_accuracy: 0.9171
Epoch 5/5
26/26 [==============================] - 20s 758ms/step - loss: 0.4994 -
accuracy: 0.7956 - val_loss: 0.4078 - val_accuracy: 0.9024
```

[16]:
```
<tensorflow.python.keras.callbacks.History at 0x7f74e9c90e10>
```

[17]:
```
pd.read_csv(csv_file).head()
```

[17]:

	epoch	accuracy	loss	val_accuracy	val_loss
0	0	0.574209	0.673508	0.814634	0.648201
1	1	0.681265	0.624179	0.643902	0.604045
2	2	0.701947	0.585506	0.492683	0.689346
3	3	0.733577	0.540475	0.917073	0.465570
4	4	0.795620	0.498889	0.902439	0.407804

2.5 Learning Rate Scheduler

Updates the learning rate during training.

[18]:
```
model = build_model(dense_units=256)
model.compile(
    optimizer='sgd',
    loss='sparse_categorical_crossentropy',
    metrics=['accuracy'])

def step_decay(epoch):
        initial_lr = 0.01
        drop = 0.5
        epochs_drop = 1
        lr = initial_lr * math.pow(drop, math.floor((1+epoch)/epochs_drop))
        return lr

model.fit(train_batches,
          epochs=5,
          validation_data=validation_batches,
          callbacks=[LearningRateScheduler(step_decay, verbose=1),
                     TensorBoard(log_dir='./log_dir')])
```

```
Epoch 00001: LearningRateScheduler reducing learning rate to 0.005.
Epoch 1/5
26/26 [==============================] - 21s 810ms/step - loss: 0.6846 -
accuracy: 0.5377 - val_loss: 0.6867 - val_accuracy: 0.4390

Epoch 00002: LearningRateScheduler reducing learning rate to 0.0025.
Epoch 2/5
26/26 [==============================] - 20s 780ms/step - loss: 0.6638 -
accuracy: 0.5985 - val_loss: 0.6773 - val_accuracy: 0.4976

Epoch 00003: LearningRateScheduler reducing learning rate to 0.00125.
Epoch 3/5
```

```
26/26 [==============================] - 21s 792ms/step - loss: 0.6542 -
accuracy: 0.6058 - val_loss: 0.6678 - val_accuracy: 0.5561

Epoch 00004: LearningRateScheduler reducing learning rate to 0.000625.
Epoch 4/5
26/26 [==============================] - 20s 769ms/step - loss: 0.6497 -
accuracy: 0.6460 - val_loss: 0.6660 - val_accuracy: 0.5561

Epoch 00005: LearningRateScheduler reducing learning rate to 0.0003125.
Epoch 5/5
26/26 [==============================] - 20s 754ms/step - loss: 0.6474 -
accuracy: 0.6533 - val_loss: 0.6647 - val_accuracy: 0.5659
```

[18]:
```
<tensorflow.python.keras.callbacks.History at 0x7f74e9abce50>
```

[19]:
```
%tensorboard --logdir log_dir
```

```
<IPython.core.display.HTML object>
```

2.6 ReduceLROnPlateau

Reduce learning rate when a metric has stopped improving.

[21]:
```
model = build_model(dense_units=256)
model.compile(
    optimizer='sgd',
    loss='sparse_categorical_crossentropy',
    metrics=['accuracy'])

model.fit(train_batches,
          epochs=50,
          validation_data=validation_batches,
          callbacks=[ReduceLROnPlateau(monitor='val_loss',
                                       factor=0.2, verbose=1,
                                       patience=1, min_lr=0.001),
                     TensorBoard(log_dir='./log_dir')])
```

```
Epoch 1/50
26/26 [==============================] - 21s 814ms/step - loss: 0.6749 -
accuracy: 0.5754 - val_loss: 0.6394 - val_accuracy: 0.7366
Epoch 2/50
26/26 [==============================] - 21s 789ms/step - loss: 0.6029 -
accuracy: 0.7336 - val_loss: 0.5707 - val_accuracy: 0.8098
Epoch 3/50
26/26 [==============================] - 21s 819ms/step - loss: 0.5603 -
accuracy: 0.7251 - val_loss: 0.4810 - val_accuracy: 0.8146
```

```
Epoch 4/50
26/26 [==============================] - 21s 800ms/step - loss: 0.4704 -
accuracy: 0.8029 - val_loss: 0.4117 - val_accuracy: 0.8098
Epoch 5/50
26/26 [==============================] - 21s 796ms/step - loss: 0.3873 -
accuracy: 0.8564 - val_loss: 0.3875 - val_accuracy: 0.8195
Epoch 6/50
26/26 [==============================] - 21s 815ms/step - loss: 0.3776 -
accuracy: 0.8637 - val_loss: 0.3146 - val_accuracy: 0.9024
Epoch 7/50
26/26 [==============================] - 21s 789ms/step - loss: 0.3001 -
accuracy: 0.8820 - val_loss: 0.2966 - val_accuracy: 0.8780
Epoch 8/50
25/26 [===========================>..] - ETA: 0s - loss: 0.2406 - accuracy:
0.9237
Epoch 00008: ReduceLROnPlateau reducing learning rate to 0.0019999999552965165.
26/26 [==============================] - 21s 796ms/step - loss: 0.2429 -
accuracy: 0.9221 - val_loss: 0.3035 - val_accuracy: 0.8878
Epoch 9/50
26/26 [==============================] - 20s 777ms/step - loss: 0.1879 -
accuracy: 0.9550 - val_loss: 0.1704 - val_accuracy: 0.9561
Epoch 10/50
26/26 [==============================] - 21s 796ms/step - loss: 0.1711 -
accuracy: 0.9623 - val_loss: 0.1636 - val_accuracy: 0.9659
Epoch 11/50
26/26 [==============================] - 21s 800ms/step - loss: 0.1624 -
accuracy: 0.9684 - val_loss: 0.1563 - val_accuracy: 0.9659
Epoch 12/50
26/26 [==============================] - 21s 804ms/step - loss: 0.1555 -
accuracy: 0.9696 - val_loss: 0.1467 - val_accuracy: 0.9659
Epoch 13/50
26/26 [==============================] - 21s 807ms/step - loss: 0.1471 -
accuracy: 0.9757 - val_loss: 0.1424 - val_accuracy: 0.9659
Epoch 14/50
26/26 [==============================] - 21s 797ms/step - loss: 0.1418 -
accuracy: 0.9745 - val_loss: 0.1297 - val_accuracy: 0.9756
Epoch 15/50
26/26 [==============================] - 21s 799ms/step - loss: 0.1343 -
accuracy: 0.9769 - val_loss: 0.1253 - val_accuracy: 0.9659
Epoch 16/50
25/26 [===========================>..] - ETA: 0s - loss: 0.1253 - accuracy:
0.9825
Epoch 00016: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 20s 785ms/step - loss: 0.1281 -
accuracy: 0.9818 - val_loss: 0.1284 - val_accuracy: 0.9610
Epoch 17/50
26/26 [==============================] - 21s 796ms/step - loss: 0.1229 -
accuracy: 0.9793 - val_loss: 0.1195 - val_accuracy: 0.9659
```

```
Epoch 18/50
26/26 [==============================] - 21s 815ms/step - loss: 0.1186 -
accuracy: 0.9805 - val_loss: 0.1149 - val_accuracy: 0.9756
Epoch 19/50
25/26 [===========================>..] - ETA: 0s - loss: 0.1144 - accuracy:
0.9850
Epoch 00019: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 21s 799ms/step - loss: 0.1181 -
accuracy: 0.9818 - val_loss: 0.1168 - val_accuracy: 0.9610
Epoch 20/50
26/26 [==============================] - 21s 812ms/step - loss: 0.1141 -
accuracy: 0.9818 - val_loss: 0.1126 - val_accuracy: 0.9659
Epoch 21/50
26/26 [==============================] - 21s 827ms/step - loss: 0.1120 -
accuracy: 0.9830 - val_loss: 0.1107 - val_accuracy: 0.9707
Epoch 22/50
26/26 [==============================] - 22s 827ms/step - loss: 0.1096 -
accuracy: 0.9854 - val_loss: 0.1090 - val_accuracy: 0.9707
Epoch 23/50
25/26 [===========================>..] - ETA: 0s - loss: 0.1085 - accuracy:
0.9837
Epoch 00023: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 22s 843ms/step - loss: 0.1077 -
accuracy: 0.9842 - val_loss: 0.1115 - val_accuracy: 0.9610
Epoch 24/50
26/26 [==============================] - 21s 803ms/step - loss: 0.1059 -
accuracy: 0.9842 - val_loss: 0.1046 - val_accuracy: 0.9756
Epoch 25/50
26/26 [==============================] - 21s 815ms/step - loss: 0.1033 -
accuracy: 0.9818 - val_loss: 0.1005 - val_accuracy: 0.9805
Epoch 26/50
25/26 [===========================>..] - ETA: 0s - loss: 0.0998 - accuracy:
0.9875
Epoch 00026: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 21s 804ms/step - loss: 0.1013 -
accuracy: 0.9866 - val_loss: 0.1019 - val_accuracy: 0.9756
Epoch 27/50
26/26 [==============================] - 21s 804ms/step - loss: 0.0988 -
accuracy: 0.9842 - val_loss: 0.0984 - val_accuracy: 0.9805
Epoch 28/50
25/26 [===========================>..] - ETA: 0s - loss: 0.0974 - accuracy:
0.9862
Epoch 00028: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 21s 797ms/step - loss: 0.0971 -
accuracy: 0.9854 - val_loss: 0.0997 - val_accuracy: 0.9756
Epoch 29/50
26/26 [==============================] - 21s 804ms/step - loss: 0.0956 -
accuracy: 0.9842 - val_loss: 0.0950 - val_accuracy: 0.9805
```

```
Epoch 30/50
26/26 [==============================] - 21s 826ms/step - loss: 0.0939 -
accuracy: 0.9866 - val_loss: 0.0939 - val_accuracy: 0.9805
Epoch 31/50
26/26 [==============================] - 21s 796ms/step - loss: 0.0914 -
accuracy: 0.9854 - val_loss: 0.0916 - val_accuracy: 0.9805
Epoch 32/50
26/26 [==============================] - 21s 800ms/step - loss: 0.0900 -
accuracy: 0.9854 - val_loss: 0.0884 - val_accuracy: 0.9805
Epoch 33/50
25/26 [===========================>..] - ETA: 0s - loss: 0.0895 - accuracy:
0.9850
Epoch 00033: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 20s 778ms/step - loss: 0.0880 -
accuracy: 0.9854 - val_loss: 0.0895 - val_accuracy: 0.9805
Epoch 34/50
26/26 [==============================] - 20s 773ms/step - loss: 0.0865 -
accuracy: 0.9866 - val_loss: 0.0872 - val_accuracy: 0.9805
Epoch 35/50
26/26 [==============================] - 20s 781ms/step - loss: 0.0850 -
accuracy: 0.9866 - val_loss: 0.0863 - val_accuracy: 0.9805
Epoch 36/50
26/26 [==============================] - 20s 781ms/step - loss: 0.0831 -
accuracy: 0.9842 - val_loss: 0.0838 - val_accuracy: 0.9805
Epoch 37/50
26/26 [==============================] - 21s 789ms/step - loss: 0.0815 -
accuracy: 0.9878 - val_loss: 0.0833 - val_accuracy: 0.9805
Epoch 38/50
26/26 [==============================] - 21s 804ms/step - loss: 0.0812 -
accuracy: 0.9866 - val_loss: 0.0827 - val_accuracy: 0.9805
Epoch 39/50
26/26 [==============================] - 21s 796ms/step - loss: 0.0795 -
accuracy: 0.9854 - val_loss: 0.0813 - val_accuracy: 0.9805
Epoch 40/50
26/26 [==============================] - 20s 781ms/step - loss: 0.0783 -
accuracy: 0.9878 - val_loss: 0.0792 - val_accuracy: 0.9805
Epoch 41/50
25/26 [===========================>..] - ETA: 0s - loss: 0.0737 - accuracy:
0.9900
Epoch 00041: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 20s 769ms/step - loss: 0.0787 -
accuracy: 0.9878 - val_loss: 0.0804 - val_accuracy: 0.9805
Epoch 42/50
25/26 [===========================>..] - ETA: 0s - loss: 0.0762 - accuracy:
0.9862
Epoch 00042: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 20s 788ms/step - loss: 0.0752 -
accuracy: 0.9866 - val_loss: 0.0794 - val_accuracy: 0.9805
```

```
Epoch 43/50
26/26 [==============================] - 20s 779ms/step - loss: 0.0751 -
accuracy: 0.9854 - val_loss: 0.0790 - val_accuracy: 0.9805
Epoch 44/50
26/26 [==============================] - 20s 787ms/step - loss: 0.0736 -
accuracy: 0.9878 - val_loss: 0.0752 - val_accuracy: 0.9805
Epoch 45/50
26/26 [==============================] - 20s 781ms/step - loss: 0.0724 -
accuracy: 0.9891 - val_loss: 0.0746 - val_accuracy: 0.9805
Epoch 46/50
26/26 [==============================] - 20s 765ms/step - loss: 0.0711 -
accuracy: 0.9891 - val_loss: 0.0743 - val_accuracy: 0.9805
Epoch 47/50
26/26 [==============================] - 20s 781ms/step - loss: 0.0702 -
accuracy: 0.9878 - val_loss: 0.0738 - val_accuracy: 0.9805
Epoch 48/50
26/26 [==============================] - 20s 773ms/step - loss: 0.0693 -
accuracy: 0.9878 - val_loss: 0.0711 - val_accuracy: 0.9805
Epoch 49/50
25/26 [===========================>..] - ETA: 0s - loss: 0.0693 - accuracy:
0.9887
Epoch 00049: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 20s 778ms/step - loss: 0.0680 -
accuracy: 0.9891 - val_loss: 0.0724 - val_accuracy: 0.9805
Epoch 50/50
25/26 [===========================>..] - ETA: 0s - loss: 0.0650 - accuracy:
0.9887
Epoch 00050: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 20s 776ms/step - loss: 0.0681 -
accuracy: 0.9878 - val_loss: 0.0719 - val_accuracy: 0.9805
```

[21]:
```
<tensorflow.python.keras.callbacks.History at 0x7f74e985ba90>
```

[22]:
```
%tensorboard --logdir log_dir
```

```
Reusing TensorBoard on port 6007 (pid 3973), started 0:43:44 ago. (Use '!kill 3973' to kill it

<IPython.core.display.HTML object>
```

[ ]: