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.

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.

Imports

In [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:  2.3.0
```

Examples of Keras callback applications

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

In [2]:

```
# Download and prepare the horses or humans dataset

splits, info = tfds.load('horses_or_humans', 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
```

```
Downloading and preparing dataset horses_or_humans/3.0.0 (download: 153.59 MiB, generated: Unknown
size, total: 153.59 MiB) to /root/tensorflow_datasets/horses_or_humans/3.0.0...
```

```
Shuffling and writing examples to
/root/tensorflow_datasets/horses_or_humans/3.0.0.incomplete7J70WE/horses_or_humans-train.tfrecord
```

```
Shuffling and writing examples to
/root/tensorflow_datasets/horses_or_humans/3.0.0.incomplete7J70WE/horses_or_humans-test.tfrecord
```

```
Dataset horses_or_humans downloaded and prepared to
/root/tensorflow_datasets/horses_or_humans/3.0.0. Subsequent calls will reuse this data.
```

In [3]:

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

In [4]:

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

In [5]:

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

In [6]:

```
train_batches = train_examples.shuffle(num_examples // 4).map(format_image).batch(BATCH_SIZE).prefe
tch(1)
validation_batches = validation_examples.map(format_image).batch(BATCH_SIZE).prefetch(1)
test_batches = test_examples.map(format_image).batch(1)
```

In [7]:

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

image_batch.shape
```

Out[7]:

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

In [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
```

TensorBoard

Enable visualizations for TensorBoard.

In [9]:

```
!rm -rf logs
```

In [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
 1/26 [>.............................] - ETA: 0s - loss: 0.7221 - accuracy:
0.4688WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/tensorflow/python/ops/summary_ops_v2.py:1277: stop (from
tensorflow.python.eager.profiler) is deprecated and will be removed after 2020-07-01.
Instructions for updating:
use `tf.profiler.experimental.stop` instead.
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/tensorflow/python/ops/summary_ops_v2.py:1277: stop (from
tensorflow.python.eager.profiler) is deprecated and will be removed after 2020-07-01.
Instructions for updating:
use `tf.profiler.experimental.stop` instead.
```

```
26/26 [==============================] - 19s 732ms/step - loss: 0.6784 - accuracy: 0.5888 - val_lo
ss: 0.6629 - val_accuracy: 0.5366
Epoch 2/10
26/26 [==============================] - 19s 733ms/step - loss: 0.6046 - accuracy: 0.6886 - val_lo
ss: 0.5862 - val_accuracy: 0.7220
Epoch 3/10
26/26 [==============================] - 19s 741ms/step - loss: 0.5691 - accuracy: 0.7007 - val_lo
ss: 0.5134 - val_accuracy: 0.8049
Epoch 4/10
26/26 [==============================] - 19s 721ms/step - loss: 0.5084 - accuracy: 0.7543 - val_lo
ss: 0.4481 - val_accuracy: 0.8585
Epoch 5/10
26/26 [==============================] - 19s 727ms/step - loss: 0.4702 - accuracy: 0.8066 - val_lo
ss: 0.4637 - val_accuracy: 0.8585
Epoch 6/10
26/26 [==============================] - 24s 934ms/step - loss: 0.3783 - accuracy: 0.8650 - val_lo
ss: 0.3961 - val_accuracy: 0.8439
Epoch 7/10
26/26 [==============================] - 19s 722ms/step - loss: 0.3124 - accuracy: 0.8869 - val_lo
ss: 0.2917 - val_accuracy: 0.8878
Epoch 8/10
26/26 [==============================] - 19s 729ms/step - loss: 0.2578 - accuracy: 0.9161 - val_lo
ss: 0.2075 - val_accuracy: 0.9610
Epoch 9/10
26/26 [==============================] - 19s 726ms/step - loss: 0.1899 - accuracy: 0.9550 - val_lo
```

```
ss: 0.1990 - val_accuracy: 0.9268
Epoch 10/10
26/26 [==============================] - 19s 727ms/step - loss: 0.1505 - accuracy: 0.9599 - val_lo
ss: 0.1231 - val_accuracy: 0.9756
```

Out[10]:

```
<tensorflow.python.keras.callbacks.History at 0x7f9757ec40b8>
```

In [11]:

```
%tensorboard --logdir logs
```

Model Checkpoint

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

In [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.01-0.68.h5
26/26 - 19s - loss: 0.6804 - accuracy: 0.5219 - val_loss: 0.6769 - val_accuracy: 0.4732
Epoch 2/5

Epoch 00002: saving model to weights.02-0.65.h5
26/26 - 19s - loss: 0.6451 - accuracy: 0.6314 - val_loss: 0.6512 - val_accuracy: 0.5561
Epoch 3/5

Epoch 00003: saving model to weights.03-0.60.h5
26/26 - 19s - loss: 0.6089 - accuracy: 0.7117 - val_loss: 0.5983 - val_accuracy: 0.7122
Epoch 4/5

Epoch 00004: saving model to weights.04-0.51.h5
26/26 - 18s - loss: 0.5506 - accuracy: 0.7628 - val_loss: 0.5127 - val_accuracy: 0.7951
Epoch 5/5

Epoch 00005: saving model to weights.05-0.46.h5
26/26 - 18s - loss: 0.4762 - accuracy: 0.8066 - val_loss: 0.4560 - val_accuracy: 0.8341
```

Out[12]:

```
<tensorflow.python.keras.callbacks.History at 0x7f972be6fbe0>
```

In [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 /usr/local/lib/python3.6/dist-
packages/tensorflow/python/training/tracking/tracking.py:111: Model.state_updates (from
tensorflow.python.keras.engine.training) is deprecated and will be removed in a future version.
Instructions for updating:
This property should not be used in TensorFlow 2.0, as updates are applied automatically.
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/tensorflow/python/training/tracking/tracking.py:111: Model.state_updates (from
tensorflow.python.keras.engine.training) is deprecated and will be removed in a future version.
Instructions for updating:
This property should not be used in TensorFlow 2.0, as updates are applied automatically.
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/tensorflow/python/training/tracking/tracking.py:111: Layer.updates (from
tensorflow.python.keras.engine.base_layer) is deprecated and will be removed in a future version.
Instructions for updating:
This property should not be used in TensorFlow 2.0, as updates are applied automatically.
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/tensorflow/python/training/tracking/tracking.py:111: Layer.updates (from
tensorflow.python.keras.engine.base_layer) is deprecated and will be removed in a future version.
Instructions for updating:
This property should not be used in TensorFlow 2.0, as updates are applied automatically.
```

```
INFO:tensorflow:Assets written to: saved_model/assets
```

```
INFO:tensorflow:Assets written to: saved_model/assets
```

```
26/26 - 20s - loss: 0.6741 - accuracy: 0.5961 - val_loss: 0.6159 - val_accuracy: 0.7902
```

Out[13]:

```
<tensorflow.python.keras.callbacks.History at 0x7f97502e03c8>
```

In [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 - 19s - loss: 0.6947 - accuracy: 0.5511 - val_loss: 0.6502 - val_accuracy: 0.6927
Epoch 2/2

Epoch 00002: saving model to model.h5
26/26 - 19s - loss: 0.6212 - accuracy: 0.6800 - val_loss: 0.6639 - val_accuracy: 0.5317
```

Out[14]:

```
<tensorflow.python.keras.callbacks.History at 0x7f97319c7080>
```

Early stopping

Stop training when a monitored metric has stopped improving.

In [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 - 19s - loss: 0.6677 - accuracy: 0.5803 - val_loss: 0.6517 - val_accuracy: 0.6439
Epoch 2/50
26/26 - 19s - loss: 0.6289 - accuracy: 0.6910 - val_loss: 0.5929 - val_accuracy: 0.8244
Epoch 3/50
26/26 - 19s - loss: 0.5853 - accuracy: 0.6837 - val_loss: 0.5701 - val_accuracy: 0.6927
Epoch 4/50
26/26 - 19s - loss: 0.5212 - accuracy: 0.7591 - val_loss: 0.7109 - val_accuracy: 0.5317
Epoch 5/50
26/26 - 19s - loss: 0.4665 - accuracy: 0.7932 - val_loss: 0.4293 - val_accuracy: 0.8780
Epoch 6/50
26/26 - 19s - loss: 0.3836 - accuracy: 0.8674 - val_loss: 0.3761 - val_accuracy: 0.8732
Epoch 7/50
26/26 - 19s - loss: 0.3125 - accuracy: 0.8881 - val_loss: 0.2733 - val_accuracy: 0.9171
Epoch 8/50
26/26 - 19s - loss: 0.2566 - accuracy: 0.9124 - val_loss: 0.1937 - val_accuracy: 0.9561
Epoch 9/50
26/26 - 19s - loss: 0.1905 - accuracy: 0.9428 - val_loss: 0.1442 - val_accuracy: 0.9707
Epoch 10/50
26/26 - 19s - loss: 0.1604 - accuracy: 0.9574 - val_loss: 0.1201 - val_accuracy: 0.9756
Epoch 11/50
26/26 - 19s - loss: 0.1366 - accuracy: 0.9659 - val_loss: 0.1112 - val_accuracy: 0.9805
Epoch 12/50
26/26 - 19s - loss: 0.1100 - accuracy: 0.9781 - val_loss: 0.1005 - val_accuracy: 0.9610
Epoch 13/50
Restoring model weights from the end of the best epoch.
26/26 - 19s - loss: 0.0913 - accuracy: 0.9781 - val_loss: 0.0732 - val_accuracy: 0.9902
Epoch 00013: early stopping
```

Out[15]:

```
<tensorflow.python.keras.callbacks.History at 0x7f97316ff668>
```

CSV Logger

Callback that streams epoch results to a CSV file.

In [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)
          callbacks=[CSVLogger(csv_file)
          ])
```

```
Epoch 1/5
26/26 [==============================] - 19s 722ms/step - loss: 0.6789 - accuracy: 0.5718 - val_lo
ss: 0.6908 - val_accuracy: 0.4488
Epoch 2/5
26/26 [==============================] - 19s 731ms/step - loss: 0.6475 - accuracy: 0.6411 - val_lo
ss: 0.6327 - val_accuracy: 0.7463
Epoch 3/5
26/26 [==============================] - 19s 726ms/step - loss: 0.6021 - accuracy: 0.7372 - val_lo
ss: 0.5983 - val_accuracy: 0.7512
Epoch 4/5
26/26 [==============================] - 19s 720ms/step - loss: 0.5478 - accuracy: 0.7543 - val_lo
ss: 0.5177 - val_accuracy: 0.7415
Epoch 5/5
26/26 [==============================] - 19s 720ms/step - loss: 0.5102 - accuracy: 0.7579 - val_lo
ss: 0.5038 - val_accuracy: 0.7854
```

Out[16]:

```
<tensorflow.python.keras.callbacks.History at 0x7f9750110dd8>
```

In [17]:

```
pd.read_csv(csv_file).head()
```

Out[17]:

	epoch	accuracy	loss	val_accuracy	val_loss
0	0	0.571776	0.678925	0.448780	0.690829
1	1	0.641119	0.647529	0.746341	0.632673
2	2	0.737226	0.602065	0.751220	0.598257
3	3	0.754258	0.547848	0.741463	0.517705
4	4	0.757908	0.510217	0.785366	0.503794

Learning Rate Scheduler

Updates the learning rate during training.

In [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 [==============================] - 19s 729ms/step - loss: 0.6833 - accuracy: 0.5292 - val_lo
ss: 0.6908 - val_accuracy: 0.4341

Epoch 00002: LearningRateScheduler reducing learning rate to 0.0025.
Epoch 2/5
```

```
26/26 [==============================] - 19s 721ms/step - loss: 0.6655 - accuracy: 0.5608 - val_lo
ss: 0.6938 - val_accuracy: 0.4341

Epoch 00003: LearningRateScheduler reducing learning rate to 0.00125.
Epoch 3/5
26/26 [==============================] - 19s 722ms/step - loss: 0.6592 - accuracy: 0.5328 - val_lo
ss: 0.6795 - val_accuracy: 0.4390

Epoch 00004: LearningRateScheduler reducing learning rate to 0.000625.
Epoch 4/5
26/26 [==============================] - 19s 726ms/step - loss: 0.6547 - accuracy: 0.5414 - val_lo
ss: 0.6755 - val_accuracy: 0.4488

Epoch 00005: LearningRateScheduler reducing learning rate to 0.0003125.
Epoch 5/5
26/26 [==============================] - 19s 723ms/step - loss: 0.6524 - accuracy: 0.5560 - val_lo
ss: 0.6741 - val_accuracy: 0.4537
```

Out[18]:

```
<tensorflow.python.keras.callbacks.History at 0x7f975002fe80>
```

In [19]:

```
%tensorboard --logdir log_dir
```

ReduceLROnPlateau

Reduce learning rate when a metric has stopped improving.

In [20]:

```
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 [==============================] - 19s 725ms/step - loss: 0.6779 - accuracy: 0.5925 - val_lo
ss: 0.6808 - val_accuracy: 0.4878
Epoch 2/50
26/26 [==============================] - 19s 715ms/step - loss: 0.6511 - accuracy: 0.6034 - val_lo
ss: 0.6396 - val_accuracy: 0.7512
Epoch 3/50
26/26 [==============================] - 19s 719ms/step - loss: 0.6298 - accuracy: 0.6776 - val_lo
ss: 0.6096 - val_accuracy: 0.6098
Epoch 4/50
26/26 [==============================] - 19s 720ms/step - loss: 0.5792 - accuracy: 0.7506 - val_lo
ss: 0.5470 - val_accuracy: 0.8390
Epoch 5/50
26/26 [==============================] - 19s 724ms/step - loss: 0.5323 - accuracy: 0.7774 - val_lo
ss: 0.4987 - val_accuracy: 0.7512
Epoch 6/50
26/26 [==============================] - ETA: 0s - loss: 0.4914 - accuracy: 0.7847
Epoch 00006: ReduceLROnPlateau reducing learning rate to 0.0019999999552965165.
26/26 [==============================] - 19s 724ms/step - loss: 0.4914 - accuracy: 0.7847 - val_lo
ss: 0.5273 - val_accuracy: 0.6927
Epoch 7/50
26/26 [==============================] - 23s 882ms/step - loss: 0.4175 - accuracy: 0.8735 - val_lo
ss: 0.4100 - val_accuracy: 0.8878
Epoch 8/50
26/26 [==============================] - 20s 761ms/step - loss: 0.3933 - accuracy: 0.8905 - val_lo
ss: 0.3970 - val_accuracy: 0.8878
```

```
Epoch 9/50
26/26 [==============================] - 19s 721ms/step - loss: 0.3765 - accuracy: 0.8929 - val_lo
ss: 0.3838 - val_accuracy: 0.8927
Epoch 10/50
26/26 [==============================] - 19s 727ms/step - loss: 0.3593 - accuracy: 0.8942 - val_lo
ss: 0.3444 - val_accuracy: 0.9073
Epoch 11/50
26/26 [==============================] - 19s 728ms/step - loss: 0.3427 - accuracy: 0.9002 - val_lo
ss: 0.3368 - val_accuracy: 0.8927
Epoch 12/50
26/26 [==============================] - 19s 726ms/step - loss: 0.3253 - accuracy: 0.9112 - val_lo
ss: 0.3112 - val_accuracy: 0.9073
Epoch 13/50
26/26 [==============================] - 19s 728ms/step - loss: 0.3125 - accuracy: 0.9015 - val_lo
ss: 0.2941 - val_accuracy: 0.9122
Epoch 14/50
26/26 [==============================] - 19s 729ms/step - loss: 0.2980 - accuracy: 0.9075 - val_lo
ss: 0.2813 - val_accuracy: 0.9171
Epoch 15/50
26/26 [==============================] - 19s 724ms/step - loss: 0.2829 - accuracy: 0.9148 - val_lo
ss: 0.2789 - val_accuracy: 0.9073
Epoch 16/50
26/26 [==============================] - 19s 726ms/step - loss: 0.2715 - accuracy: 0.9173 - val_lo
ss: 0.2579 - val_accuracy: 0.9171
Epoch 17/50
26/26 [==============================] - 19s 731ms/step - loss: 0.2607 - accuracy: 0.9209 - val_lo
ss: 0.2425 - val_accuracy: 0.9220
Epoch 18/50
26/26 [==============================] - ETA: 0s - loss: 0.2468 - accuracy: 0.9270
Epoch 00018: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 19s 726ms/step - loss: 0.2468 - accuracy: 0.9270 - val_lo
ss: 0.2560 - val_accuracy: 0.9122
Epoch 19/50
26/26 [==============================] - ETA: 0s - loss: 0.2373 - accuracy: 0.9294
Epoch 00019: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 19s 729ms/step - loss: 0.2373 - accuracy: 0.9294 - val_lo
ss: 0.2459 - val_accuracy: 0.9122
Epoch 20/50
26/26 [==============================] - 19s 731ms/step - loss: 0.2328 - accuracy: 0.9258 - val_lo
ss: 0.2301 - val_accuracy: 0.9220
Epoch 21/50
26/26 [==============================] - 19s 727ms/step - loss: 0.2277 - accuracy: 0.9282 - val_lo
ss: 0.2253 - val_accuracy: 0.9220
Epoch 22/50
26/26 [==============================] - ETA: 0s - loss: 0.2222 - accuracy: 0.9343
Epoch 00022: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 20s 768ms/step - loss: 0.2222 - accuracy: 0.9343 - val_lo
ss: 0.2281 - val_accuracy: 0.9171
Epoch 23/50
26/26 [==============================] - 19s 750ms/step - loss: 0.2170 - accuracy: 0.9355 - val_lo
ss: 0.2090 - val_accuracy: 0.9317
Epoch 24/50
26/26 [==============================] - ETA: 0s - loss: 0.2119 - accuracy: 0.9367
Epoch 00024: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 19s 730ms/step - loss: 0.2119 - accuracy: 0.9367 - val_lo
ss: 0.2215 - val_accuracy: 0.9220
Epoch 25/50
26/26 [==============================] - 19s 727ms/step - loss: 0.2062 - accuracy: 0.9380 - val_lo
ss: 0.1984 - val_accuracy: 0.9366
Epoch 26/50
26/26 [==============================] - 19s 725ms/step - loss: 0.2013 - accuracy: 0.9428 - val_lo
ss: 0.1977 - val_accuracy: 0.9268
Epoch 27/50
26/26 [==============================] - ETA: 0s - loss: 0.1979 - accuracy: 0.9453
Epoch 00027: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 19s 730ms/step - loss: 0.1979 - accuracy: 0.9453 - val_lo
ss: 0.2080 - val_accuracy: 0.9220
Epoch 28/50
26/26 [==============================] - 19s 726ms/step - loss: 0.1940 - accuracy: 0.9501 - val_lo
ss: 0.1883 - val_accuracy: 0.9317
Epoch 29/50
26/26 [==============================] - 19s 728ms/step - loss: 0.1892 - accuracy: 0.9501 - val_lo
ss: 0.1833 - val_accuracy: 0.9415
Epoch 30/50
26/26 [==============================] - ETA: 0s - loss: 0.1855 - accuracy: 0.9489
Epoch 00030: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 19s 730ms/step - loss: 0.1855 - accuracy: 0.9489 - val_lo
```

```
ss: 0.1855 - val_accuracy: 0.9268
Epoch 31/50
26/26 [==============================] - 19s 731ms/step - loss: 0.1825 - accuracy: 0.9526 - val_lo
ss: 0.1749 - val_accuracy: 0.9463
Epoch 32/50
26/26 [==============================] - ETA: 0s - loss: 0.1774 - accuracy: 0.9562
Epoch 00032: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 19s 727ms/step - loss: 0.1774 - accuracy: 0.9562 - val_lo
ss: 0.1862 - val_accuracy: 0.9317
Epoch 33/50
26/26 [==============================] - ETA: 0s - loss: 0.1737 - accuracy: 0.9574
Epoch 00033: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 19s 731ms/step - loss: 0.1737 - accuracy: 0.9574 - val_lo
ss: 0.1794 - val_accuracy: 0.9366
Epoch 34/50
26/26 [==============================] - ETA: 0s - loss: 0.1693 - accuracy: 0.9550
Epoch 00034: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 19s 731ms/step - loss: 0.1693 - accuracy: 0.9550 - val_lo
ss: 0.1774 - val_accuracy: 0.9366
Epoch 35/50
26/26 [==============================] - ETA: 0s - loss: 0.1653 - accuracy: 0.9562
Epoch 00035: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 21s 791ms/step - loss: 0.1653 - accuracy: 0.9562 - val_lo
ss: 0.1774 - val_accuracy: 0.9317
Epoch 36/50
26/26 [==============================] - 20s 776ms/step - loss: 0.1628 - accuracy: 0.9647 - val_lo
ss: 0.1633 - val_accuracy: 0.9366
Epoch 37/50
26/26 [==============================] - 19s 734ms/step - loss: 0.1598 - accuracy: 0.9635 - val_lo
ss: 0.1547 - val_accuracy: 0.9463
Epoch 38/50
26/26 [==============================] - ETA: 0s - loss: 0.1556 - accuracy: 0.9659
Epoch 00038: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 19s 750ms/step - loss: 0.1556 - accuracy: 0.9659 - val_lo
ss: 0.1575 - val_accuracy: 0.9463
Epoch 39/50
26/26 [==============================] - 19s 730ms/step - loss: 0.1515 - accuracy: 0.9659 - val_lo
ss: 0.1518 - val_accuracy: 0.9415
Epoch 40/50
26/26 [==============================] - ETA: 0s - loss: 0.1517 - accuracy: 0.9647
Epoch 00040: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 19s 728ms/step - loss: 0.1517 - accuracy: 0.9647 - val_lo
ss: 0.1601 - val_accuracy: 0.9366
Epoch 41/50
26/26 [==============================] - 19s 728ms/step - loss: 0.1484 - accuracy: 0.9708 - val_lo
ss: 0.1495 - val_accuracy: 0.9415
Epoch 42/50
26/26 [==============================] - ETA: 0s - loss: 0.1434 - accuracy: 0.9720
Epoch 00042: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 19s 724ms/step - loss: 0.1434 - accuracy: 0.9720 - val_lo
ss: 0.1495 - val_accuracy: 0.9463
Epoch 43/50
26/26 [==============================] - 19s 727ms/step - loss: 0.1413 - accuracy: 0.9708 - val_lo
ss: 0.1425 - val_accuracy: 0.9415
Epoch 44/50
26/26 [==============================] - ETA: 0s - loss: 0.1382 - accuracy: 0.9672
Epoch 00044: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 19s 724ms/step - loss: 0.1382 - accuracy: 0.9672 - val_lo
ss: 0.1449 - val_accuracy: 0.9463
Epoch 45/50
26/26 [==============================] - 19s 728ms/step - loss: 0.1349 - accuracy: 0.9720 - val_lo
ss: 0.1365 - val_accuracy: 0.9415
Epoch 46/50
26/26 [==============================] - 19s 727ms/step - loss: 0.1322 - accuracy: 0.9745 - val_lo
ss: 0.1316 - val_accuracy: 0.9561
Epoch 47/50
26/26 [==============================] - 19s 726ms/step - loss: 0.1289 - accuracy: 0.9745 - val_lo
ss: 0.1304 - val_accuracy: 0.9512
Epoch 48/50
26/26 [==============================] - ETA: 0s - loss: 0.1269 - accuracy: 0.9769
Epoch 00048: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 19s 726ms/step - loss: 0.1269 - accuracy: 0.9769 - val_lo
ss: 0.1313 - val_accuracy: 0.9561
Epoch 49/50
26/26 [==============================] - ETA: 0s - loss: 0.1242 - accuracy: 0.9745
Epoch 00049: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 19s 722ms/step - loss: 0.1242 - accuracy: 0.9745 - val_lo
```

```
ss: 0.1344 - val_accuracy: 0.9463
Epoch 50/50
26/26 [==============================] - ETA: 0s - loss: 0.1229 - accuracy: 0.9769
Epoch 00050: ReduceLROnPlateau reducing learning rate to 0.001.
26/26 [==============================] - 19s 728ms/step - loss: 0.1229 - accuracy: 0.9769 - val_lo
ss: 0.1331 - val_accuracy: 0.9463
```

Out[20]:

```
<tensorflow.python.keras.callbacks.History at 0x7f974ff1fb00>
```

In [21]:

```
%tensorboard --logdir log_dir
```

```
Reusing TensorBoard on port 6007 (pid 6254), started 0:17:39 ago. (Use '!kill 6254' to kill it.)
```