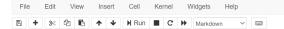




jupyter C1_W5_Lab_1_exploring-callbacks Last Checkpoint: a minute ago (autosaved)



Not Trusted Python 3 O



Open in Colab

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

Examples of Keras callback applications

The following section will guide you through creating simple $\underline{\text{Callback}}$ applications.

TensorBoard

Enable visualizations for TensorBoard.

Model Checkpoint

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

```
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),
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)
                       ])
model.compile(
    optimizer='sgd',
    loss='sparse_categorical_crossentropy',
                 metrics=['accuracy'])
             model.fit(train_batches,
                        epochs=2.
                        validation_data=validation_batches,
                        callbacks=[ModelCheckpoint('model.h5', verbose=1)
```

Early stopping

Stop training when a monitored metric has stopped improving.

CSV Logger

Callback that streams epoch results to a CSV file.

Learning Rate Scheduler

Updates the learning rate during training.

In []: ⋈ %tensorboard --logdir log_dir