Callbacks

Ramin Zarebidoky (LiterallyTheOne)

14 Dec 2025

Callbacks

Introduction

In the previous tutorial, we have learned about **preprocessing and data augmentation** techniques in Keras. In this tutorial, we learn about **Callbacks** and explore some of the most important ones in **Keras**.

Callbacks

Callback in **Keras** is a function that we pass it to our **fit** function. **Keras** calls that function automatically in the specific moment. We have learned about **TensorBoard Callback** before. We used to create a **TensorBoard Callback** and pass it to our fit function as below:

```
from keras.callbacks import TensorBoard

log_dir = "logs/fit/" +
 ↪  datetime.datetime.now().strftime("%Y%m%d-%H%M%S")
tensorboard_callback = TensorBoard(log_dir=log_dir)

history = model.fit(
    ...,
    callbacks=[tensorboard_callback],
)
```

In this tutorial, we are going to learn about another two important **CallBacks**, called: **EarlyStopping** and **ModelCheckpoint**.

List of available callbacks in Keras

Early Stopping

EarlyStopping is a callback that stops the training procedure if there is no improvement. Here is an example of **EarlyStopping**:

```
from keras.callbacks import EarlyStopping

early_stopping = EarlyStopping(
    monitor="val_loss",
    patience=5,
    restore_best_weights=True,
    verbose=1,
)

history = model.fit(
    ...,
    epochs=200,
    callbacks=[tensorboard_callback, early_stopping],
)
```

In the code above, we have created an object of **EarlyStopping**. We set our **EarlyStopping** to monitor our validation loss (`"val_loss"`). Then, we told it to wait for 5 epochs, if there was no improvement seen on those epochs, stop the training. With `restore_best_weights` set to `True`, **Keras** will load the best weights of the model that has the lowest `val_loss`. Also, we set the `verbose` to `1`, to be able to have a report of the procedure. As you can see, we have added it to our callbacks argument in `fit` function and increased our `epochs` to `200`.

Model Checkpoint

ModelCheckpoint saves the model during the training. Here is an example of **ModelCheckpoint**.

```
from keras.callbacks import ModelCheckpoint

model_checkpoint = ModelCheckpoint(
    filepath="checkpoints/best_model.weights.h5",
    monitor="val_loss",
    save_best_only=True,
    save_weights_only=True,
    verbose=1,
)

history = model.fit(
    ...,
    callbacks=[tensorboard_callback, early_stopping,
↪   model_checkpoint],
)
```

In the example above, we created an object of `ModelCheckpoint`. At first, we defined the path of the file, that we want to save our model. By the `monitor`

argument, we said our object to look out for validation loss (`val_loss`). Then, we said, we only want to save the best model by using `save_best_only=True`. This approach helps the model not to save the model after each epoch and only saves the best one. After that, we set `save_weight_only=True`. This argument helps us to save only the weights of our model, not the way that we have compiled or other characteristics. When we do that, the capacity of the saved model decreases, and we can load our model in multiple platforms, not the specific one that we have trained our model on it. If we set this argument, we should make sure that the name of our `filepath` ends with `.weights.h5`. Then, we set `verbose=1` to have a report of what is happening.

If we want to load the best weights that we have saved, we can use the code below:

```
model.load_weights("checkpoints/best_model.weights.h5")
```

Your turn

Add `EarlyStoppin` and `ModelCheckpoint` to your callbacks.

Conclusion

In this tutorial, we learned about **Callbacks** in **Keras**. First, we explained about what callbacks are actually are. Then, we introduced two of the most important callbacks, `EarlyStopping` and `ModelCheckpoint`.