365 DataScience Tensorboard - Tracking metrics

```
# The outlined code below is to show how you can incorporate Tensorboar
d in order to track different metrics,
# It does not include any actual dataset, thus, cannot be trained
# You can include any image data you want, after properly preprocessing
 it

# Importing the relevant packages
import tensorflow as tf
import datetime
```

Creating and compiling the model

```
# Outlining the model/architecture of our CNN
model = tf.keras.Sequential([
    tf.keras.layers.Conv2D(50, 5, activation='relu', input_shape=(28, 2
8, 1)),
    tf.keras.layers.MaxPooling2D(pool_size=(2,2)),
    tf.keras.layers.Conv2D(50, 3, activation='relu'),
    tf.keras.layers.MaxPooling2D(pool_size=(2,2)),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(10)
])

# Defining the loss function
loss_fn = tf.keras.losses.SparseCategoricalCrossentropy(from_logits=Tru
e)

# Compiling the model with Adam optimizer and the cathegorical crossent
ropy as a loss function
model.compile(optimizer='adam', loss=loss_fn, metrics=['accuracy'])

# Defining early stopping to prevent overfitting
early_stopping = tf.keras.callbacks.EarlyStopping(
    monitor = 'val_loss',
    mode = 'auto',
    min_delta = 0,
    patience = 2,
    verbose = 0,
    restore_best_weights = True
)
```

Defining Tensorboard callback

```
# Now, we can define a Tensorboard callback to log the metrics of our t
raining
# The metrics that are logged are the loss + everything that is provide
d to the 'metrics' argument in the 'compile' method above
```

```
# Creating a folder in which the logs will be written
# I have added the current date and time to the folder so that one can
understand to which network does the log correspond to
# You can name this folder however you'd like, though
log_dir = "logs\\fit\\" + datetime.datetime.now().strftime("%Y%m%d-%H%
M%S")

# Defining the tensorboard callback itself, which will be added to the
fit method
tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=log_dir,
histogram_freq=1)
```

Training the network

```
# Train the network
# In order to incorporate the Tensorboard logging capabilities, we need
 to add it to the 'callbacks' parameter
# Be careful to always write the early stopping callback last, as it so
metimes bugs out if it is not last
model.fit(
    train_data,
    epochs = NUM_EPOCHS,
    callbacks = [tensorboard_callback, early_stopping],
    validation_data = validation_data,
    verbose = 2
)
```

Visualizing in Tensorboard

```
# Now, we can run the Tensorboard extension and check all the logs
# The logs will be visible in the first tab - scalars

# Loading the Tensorboard extension
%load_ext tensorboard
%tensorboard --logdir "logs/fit"

# NOTE: On Windows, TensorBoard has trouble starting if the extension h
as been running. So, the first time you start it,
# it will run properly. But if you subsequently try to restart it, or o
pen a different directory,
# the extension will encounter an error. Luckily, there is a quick work
around. All you need to do
# is write 2 commands in the shell. First, open the command prompt, or
‘cmd.exe’. In there,
# you need to paste the following 2 lines , one after another:
#
# taskkill /im TensorBoard.exe /f
# del /q %TMP%\.TensorBoard-info\*
#
# These will end already active TensorBoard processes and clean the tem
porary data associated with TensorBoard,
```

```
# so you can run it again. If either of those gives an error, that's ok
ay: you can ignore it.
```

Start your 365 Journey!