The Keras API version internal to TensorFlow is available from the 'tf.keras' package, whereas the broader Keras API blueprint that is not tied to a specific backend will remain available from the 'keras' package. In summary, when working with the 'keras' package, the backend can run with either TensorFlow, Microsoft CNTK, or Theano. On the other hand, working with 'tf.keras' provides a TensorFlow only version which is tightly integrated and compatible with all of the functionality of the core TensorFlow library.

In this book, we will focus on **'tf.Keras'** as a high-level API of TensorFlow.

The Anatomy of a Keras Program

The Keras **'Model'** forms the core of a Keras program. A 'Model' is first constructed, then it is compiled. Next, the compiled model is trained and evaluated using their respective training and evaluation datasets. Upon successful evaluation using the relevant metrics, the model is then used for making predictions on previously unseen data samples. Figure 30-7 shows the program flow for modeling with Keras.

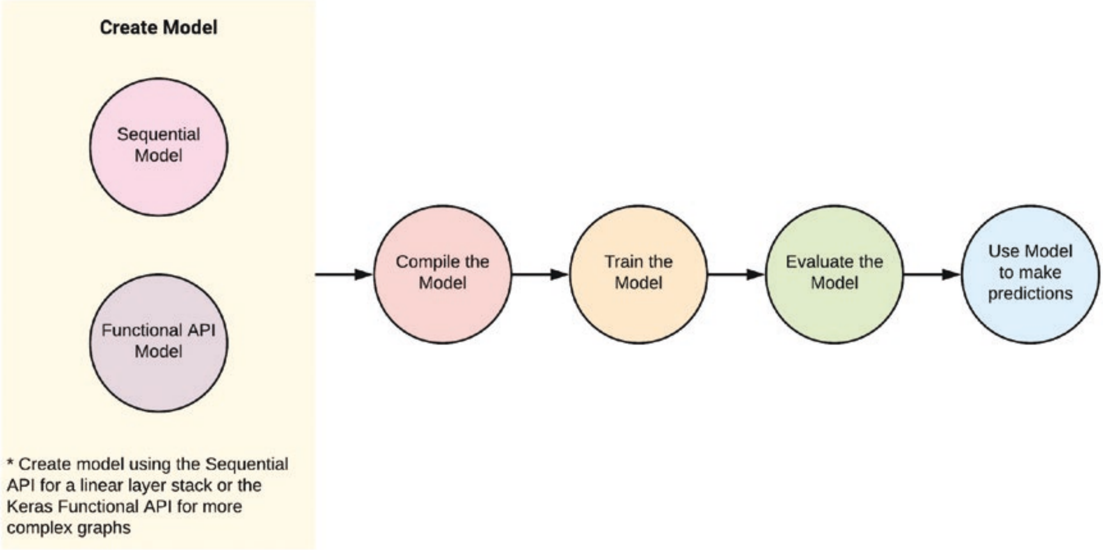
Create Model
Sequential Model
Functional API Model
* Create model using the Sequential API for a linear layer stack or the Keras Functional API for more complex graphs
Compile the Model
Train the Model
Evaluate the Model
Use Model to make predictions

Figure 30-7. *The anatomy of a Keras program*

As shown in Figure 30-7, the Keras 'Model' can be constructed using the Sequential API 'tf.keras.Sequential' or the Keras Functional API which defines a model instance 'tf. keras.Model'. The Sequential model is the simplest method for creating a linear stack of neural network layers. The Functional model is used if a more complex graph is desired. Keras is the de facto API for building neural network architectures with TensorFlow.

From here on, the code examples in this book will use the Sequential API, Functional API, and Model subclassing methods for building neural network architectures with Keras. In doing this, the reader can play around with the various examples as samples to get a feel of how they work.

TensorBoard

TensorBoard is an interactive visualization tool that comes bundled with TensorFlow. The goal of TensorBoard is to gain a visual insight into how the computational graph is constructed and executed. This information provides greater visibility for understanding, optimizing, and debugging deep learning models.

TensorBoard has a variety of visualization dashboard, such as

- Scalar dashboard: This dashboard captures metrics that change with time, such as the loss of a model or other model evaluation metrics such as accuracy, precision, recall, f1, and so on.
- Histogram dashboard: This dashboard shows the histogram distribution for a Tensor as it has changed over time.
- Distribution dashboard: This dashboard is similar to the histogram dashboard. However, it displays the histogram as a distribution.
- Graph explorer: This dashboard gives a graphical overview of the TensorFlow computational graph and how information flows from one node to the other. This dashboard provides invaluable insights into the network architecture.
- Image dashboard: This dashboard displays images saved using the method **tf.summary.image**.
- Audio dashboard: This dashboard provides audio clips saved using the method **tf.summary.audio**.