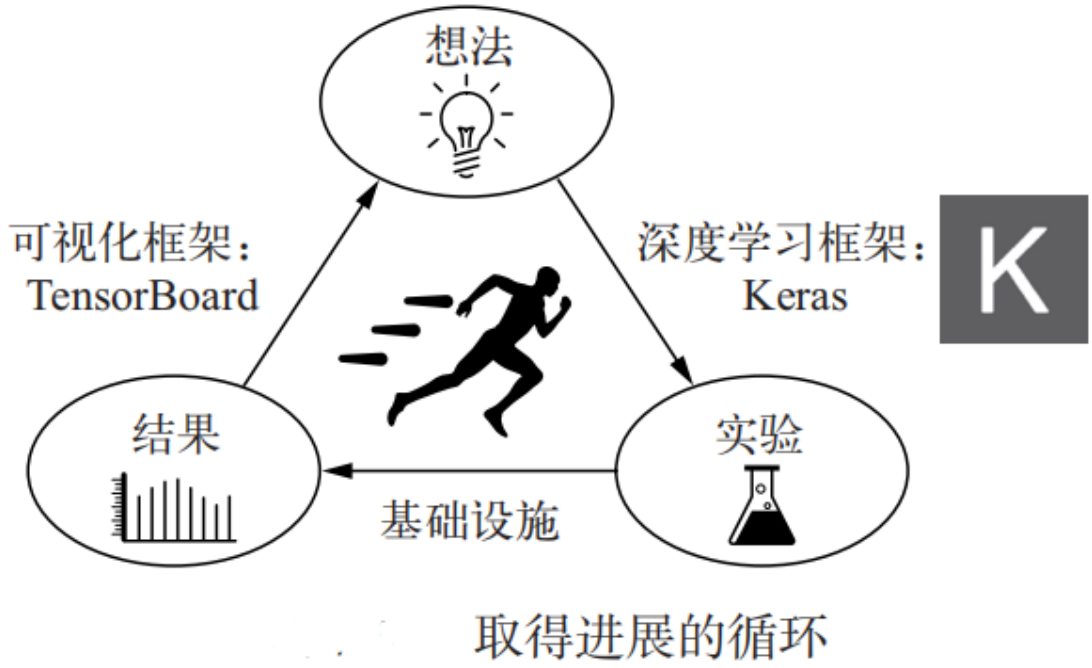
想要做好研究或开发出好的模型，在实验过程中你需要丰富频繁的反馈，从而知道模型内部正在发生什么。这正是运行实验的目的：获取关于模型表现好坏的信息，越多越好。取得进展是一个反复迭代的过程（或循环）：首先你有一个想法，并将其表述为一个实验，用于验证你的想法是否正确。你运行这个实验，并处理其生成的信息。这又激发了你的下一个想法。在这个循环中实验的迭代次数越多，你的想法也就变得越来越精确、越来越强大。



TensorBoard 的主要用途是，在训练过程中帮助你以可视化的方法监控模型内部发生的一切。如果你监控了除模型最终损失之外的更多信息，那么可以更清楚地了解模型做了什么、没做什么，并且能够更快地取得进展。TensorBoard 具有下列巧妙的功能，都在浏览器中实现。

- ❑ 在训练过程中以可视化的方式监控指标
- ❑ 将模型架构可视化
- ❑ 将激活和梯度的直方图可视化
- ❑ 以三维的形式研究嵌入

In [1]:

```
import tensorflow as tf
```

In [2]:

```
tf.__version__
```

Out[2]:

```
'2.0.0'
```

In [3]:

```
from tensorflow.keras import layers, datasets, Sequential, models
```

In [4]:

```
from tensorflow.keras.preprocessing import sequence
```

In [5]:

```
max_words = 2000
max_len = 500
```

In [6]:

```
(x_train, y_train), (x_test, y_test) = datasets.imdb.load_data(num_words=max_words)
```

In [7]:

```
x_train.shape, y_train.shape, x_test.shape, y_test.shape
```

Out[7]:

```
((25000,), (25000,), (25000,), (25000,))
```

In [8]:

```
len(x_train[0])
```

Out[8]:

```
218
```

In [9]:

```
x_train[0][:5]
```

Out[9]:

```
[1, 14, 22, 16, 43]
```

In [10]:

```
x_train = sequence.pad_sequences(x_train, maxlen=max_len)
```

In [11]:

```
x_test = sequence.pad_sequences(x_test, maxlen=max_len)
```

In [12]:

```
x_train.shape
```

Out[12]:

```
(25000, 500)
```

In [13]:

```
x_train[0].shape
```

Out[13]:

```
(500,)
```

In [14]:

```
model = models.Sequential([
    layers.Embedding(max_words, 128, input_length=max_len, name='embed'),
    layers.Conv1D(32, 7, activation='relu'),
    layers.MaxPooling1D(5),
    layers.Conv1D(32, 7, activation='relu'),
    layers.GlobalMaxPooling1D(),
    layers.Dense(1)
])
```

In [15]:

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embed (Embedding)            (None, 500, 128)          256000
_________________________________________________________________
conv1d (Conv1D)              (None, 494, 32)           28704
_________________________________________________________________
max_pooling1d (MaxPooling1D) (None, 98, 32)            0
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 92, 32)            7200
_________________________________________________________________
global_max_pooling1d (Global (None, 32)                0
_________________________________________________________________
dense (Dense)                (None, 1)                 33
=================================================================
Total params: 291,937
Trainable params: 291,937
Non-trainable params: 0
_________________________________________________________________
```

In [20]:

```
model.compile(optimizer='rmsprop',
              loss='binary_crossentropy',
              metrics=['acc'])
```

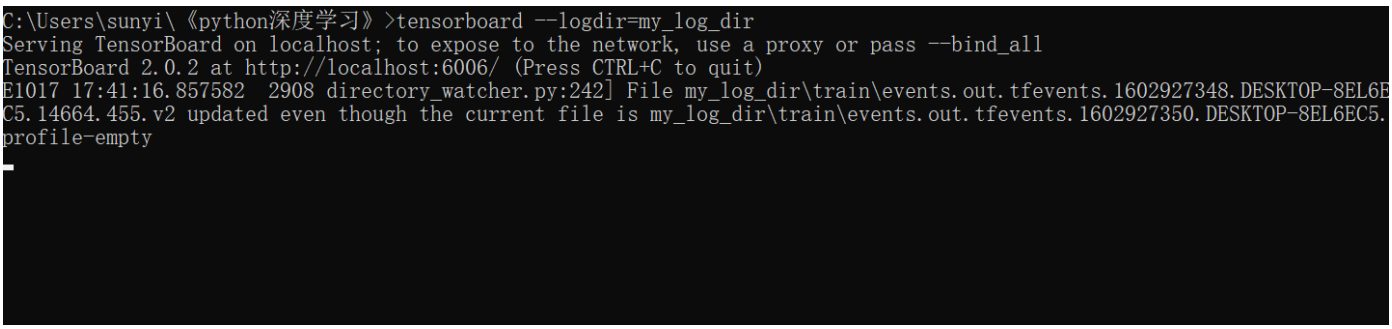
In [22]:

```
callbacks = [
     tf.keras.callbacks.TensorBoard(
     log_dir='my_log_dir',
     histogram_freq=1,  # 每一轮之后记录激活直方图
     embeddings_freq=1, # 每一轮之后记录嵌入数据
 )
]
```

In [ ]:

```
history = model.fit(x_train, y_train,
                    epochs=20,
                    batch_size=128,
                    validation_split=0.2,
                    callbacks=callbacks)
```

```
Train on 20000 samples, validate on 5000 samples
Epoch 1/20
20000/20000 [==============================] - 63s 3ms/sample - loss: 0.6589
- acc: 0.6267 - val_loss: 0.4280 - val_acc: 0.8290
Epoch 2/20
20000/20000 [==============================] - 64s 3ms/sample - loss: 0.4378
- acc: 0.8413 - val_loss: 0.4536 - val_acc: 0.8348
Epoch 3/20
20000/20000 [==============================] - 68s 3ms/sample - loss: 0.4039
- acc: 0.8724 - val_loss: 0.5019 - val_acc: 0.8516
Epoch 4/20
20000/20000 [==============================] - 66s 3ms/sample - loss: 0.3416
- acc: 0.8959 - val_loss: 0.4565 - val_acc: 0.8620
Epoch 5/20
20000/20000 [==============================] - 65s 3ms/sample - loss: 0.2843
- acc: 0.9151 - val_loss: 0.4973 - val_acc: 0.8688
Epoch 6/20
20000/20000 [==============================] - 62s 3ms/sample - loss: 0.2626
- acc: 0.9280 - val_loss: 0.5976 - val_acc: 0.8680
Epoch 7/20
18816/20000 [===========================>..] - ETA: 3s - loss: 0.2203 - acc:
0.9455
```



tensorboard --logdir=my_log_dir

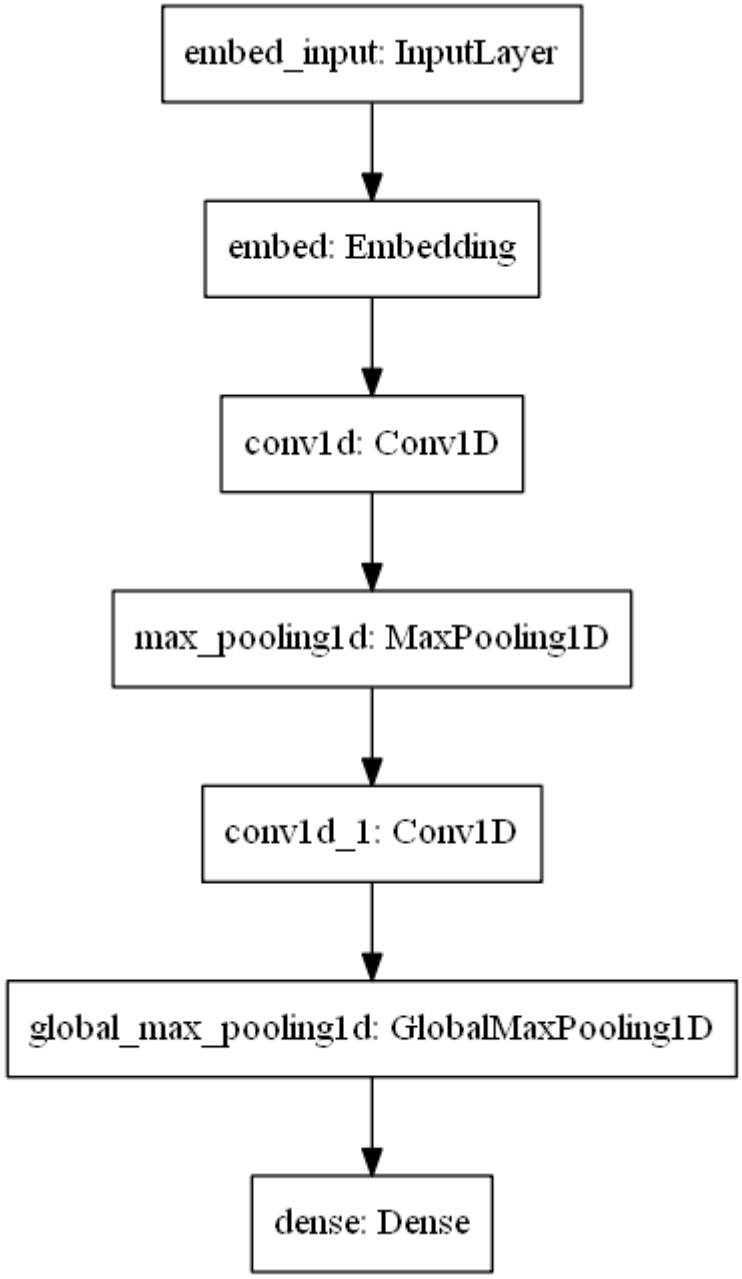
In [16]:

```
from tensorflow.keras import utils
```

In [17]:

```
utils.plot_model(model, to_file='7-10-model.png')
```

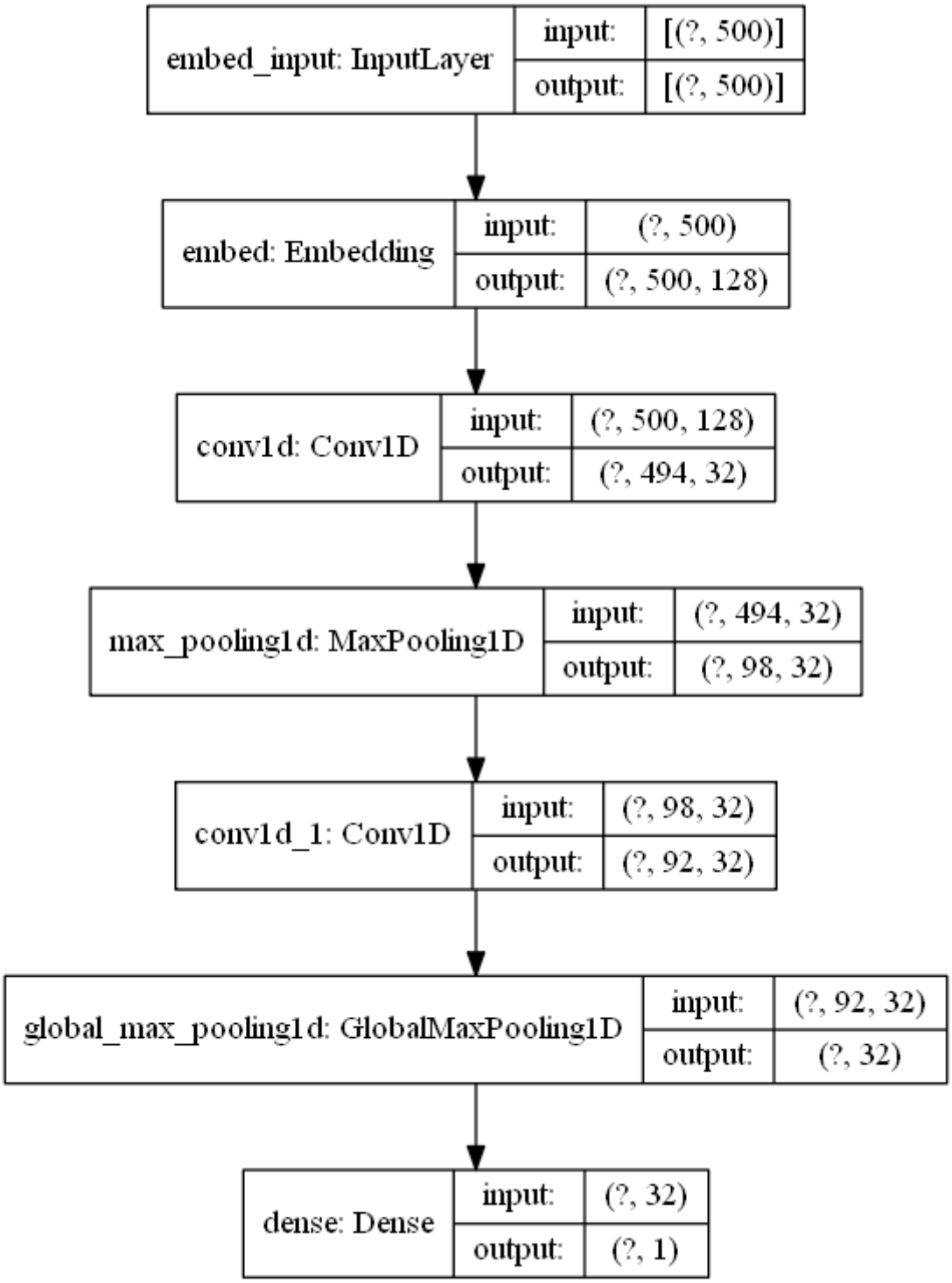
Out[17]:



In [18]:

```
utils.plot_model(model, show_shapes=True, to_file='7-10-model.png')
```

Out[18]:



In [ ]: