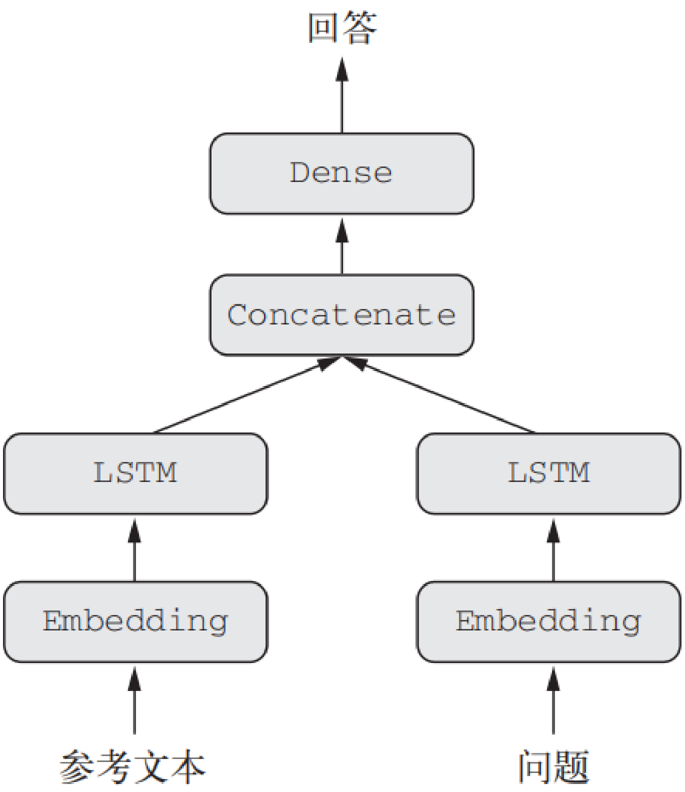
多输入



In [1]:

```
import tensorflow as tf
```

In [2]:

```
tf.__version__
```

Out[2]:

```
'2.0.0'
```

In [3]:

```
from tensorflow.keras import Input, layers, Model
```

In [4]:

```
text_vocabulary_size = 10000
question_vocabulary_size = 10000
answer_vocabulary_size = 500
```

In [5]:

```
text_input = Input(shape=(None, ), dtype='int32', name='text') # 文本长度是可变的
```

In [6]:

```
embedded_text = layers.Embedding(text_vocabulary_size, 64) (text_input)
```

In [7]:

```
embedded_text.shape
```

Out[7]:

```
TensorShape([None, None, 64])
```

In [8]:

```
encoded_text = layers.LSTM(32)(embedded_text)
```

In [9]:

```
question_input = Input(shape=(None, ), dtype='int32', name='question') # 问题长度是可变的
```

In [10]:

```
embedded_question = layers.Embedding(question_vocabulary_size, 32) (question_input)
```

In [11]:

```
embedded_question.shape
```

Out[11]:

```
TensorShape([None, None, 32])
```

In [12]:

```
concatenated = layers.concatenate([embedded_text, embedded_question], axis=-1)
```

In [13]:

```
concatenated.shape
```

Out[13]:

```
TensorShape([None, None, 128])
```

In [14]:

```
answer = layers.Dense(answer_vocabulary_size, activation='softmax')(concatenated)
```

In [15]:

```
model = Model([text_input, question_input], answer)
```

In [16]:

```python
model.summary()
```

```
Model: "model"
____________________________________________________________________________
______________________
Layer (type)                    Output Shape         Param #     Connected t
o
============================================================================
======================
text (InputLayer)               [(None, None)]       0
____________________________________________________________________________
______________________
embedding (Embedding)           (None, None, 64)     640000      text[0][0]
____________________________________________________________________________
______________________
concatenate (Concatenate)       (None, None, 128)    0           embedding
[0][0]
                                                                 embedding
[0][0]
____________________________________________________________________________
______________________
question (InputLayer)           [(None, None)]       0
____________________________________________________________________________
______________________
dense (Dense)                   (None, None, 500)    64500       concatenate
[0][0]
============================================================================
======================
Total params: 704,500
Trainable params: 704,500
Non-trainable params: 0
____________________________________________________________________________
______________________
```

In [17]:

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

模型输入数据格式：有两个可用的 API：我们可以向模型输入一个由Numpy 数组组成的列表，或者也可以输入一个将输入名称映射为 Numpy 数组的字典

In [18]:

```python
import numpy as np
```

In [19]:

```python
num_samples = 1000
max_length = 100
```

In [20]:

```
text = np.random.randint(1, text_vocabulary_size, size=(num_samples, max_length))
question = np.random.randint(1, question_vocabulary_size, size=(num_samples, max_length))
```

In [21]:

```
answers = np.random.randint(answer_vocabulary_size, size=(num_samples))
```

In [22]:

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

In [23]:

```
answers = utils.to_categorical(answers, answer_vocabulary_size)
```

In [ ]:

```
# model.fit([text, question], answers, epochs=10, batch_size=128)
```

In [ ]:

```
# model.fit({'text': text, 'question': question}, answers, epochs=10, batch_size=128)
```

In [24]:

```
text.shape
```

Out[24]:

```
(1000, 100)
```

In [25]:

```
question.shape
```

Out[25]:

```
(1000, 100)
```

In [26]:

```
answers.shape
```

Out[26]:

```
(1000, 500)
```

In [ ]: