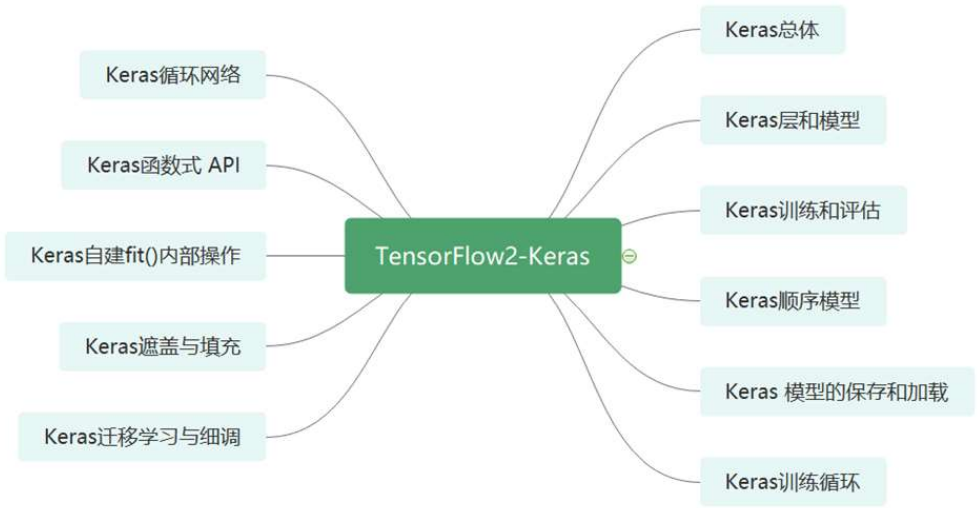
TensorFlow2-Keras

Keras顺序模型

The Sequential model

本小节导学



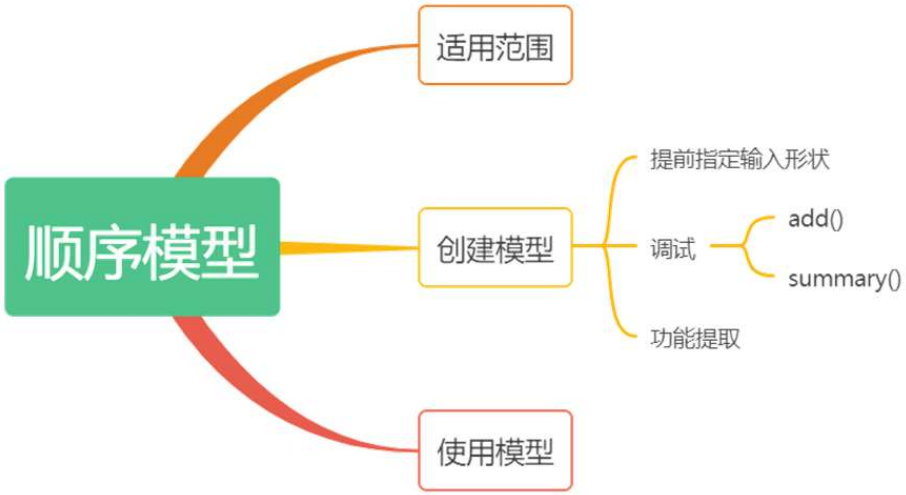
Keras总体-1

- Keras顺序模型
  - https://tensorflow.google.cn/guide/keras/sequential_model
- Keras函数式 API
  - https://tensorflow.google.cn/guide/keras/functional
- Keras训练和评估
  - https://tensorflow.google.cn/guide/keras/train_and_evaluate
- Keras层和模型
  - https://tensorflow.google.cn/guide/keras/custom_layers_and_models
- Keras 模型的保存和加载
  - https://tensorflow.google.cn/guide/keras/save_and_serialize
- Keras训练循环
  - https://tensorflow.google.cn/guide/keras/writing_a_training_loop_from_scratch

Keras总体-2

- Keras自建fit()内部操作
  - https://tensorflow.google.cn/guide/keras/customizing_what_happens_in_fit
- Keras循环网络
  - https://tensorflow.google.cn/guide/keras/rnn
- Keras遮盖与填充
  - https://tensorflow.google.cn/guide/keras/masking_and_padding
- Keras迁移学习与细调
  - https://tensorflow.google.cn/guide/keras/transfer_learning

本小节导学



顺序模型-基础

以dense网络为例

什么时候使用顺序模型？CNN、GNN

- 模型适用于一个普通堆栈的图层，其中每个图层只有一个输入张量和一个输出张量

```
model = keras.Sequential(
    [
        layers.Dense(2, activation="relu", name="layer1"),
        layers.Dense(3, activation="relu", name="layer2"),
        layers.Dense(4, name="layer3"),
    ]
)
x = tf.ones((3, 3))
y = model(x)
```

什么时候使用顺序模型？

- 等价于函数

```
layer1 = layers.Dense(2, activation="relu", name="layer1")
layer2 = layers.Dense(3, activation="relu", name="layer2")
layer3 = layers.Dense(4, name="layer3")

# Call layers on a test input
x = tf.ones((3, 3))
y = layer3(layer2(layer1(x)))
```

顺序模型不适用：

- 模型有多个输入或多个输出
- 任何层都有多个输入或多个输出
- 需要做图层共享
- 需要非线性拓扑（例如剩余连接residual 、多分支模型）

https://tensorflow.google.cn/guide/keras/sequential_model

创建顺序模型

- 方法一：通过将层列表传递给顺序构造函数来创建顺序模型

```
model = keras.Sequential(
    [
        layers.Dense(2, activation="relu"),
        layers.Dense(3, activation="relu"),
        layers.Dense(4),
    ]
)
```

创建顺序模型

- 方法二：通过 add() 以增量方式创建顺序模型

```
model = keras.Sequential()
model.add(layers.Dense(2, activation="relu"))
model.add(layers.Dense(3, activation="relu"))
model.add(layers.Dense(4))
```

- ！需要对应的 pop() 方法来删除层

```
model.pop()
print(len(model.layers))
```

创建顺序模型

- ！顺序构造函数接受name参数，就像Keras中的任何层或模型一样。这对于用语义上有意义的名称来注释TensorBoard图非常有用。

```
model = keras.Sequential(name="my_sequential")
model.add(layers.Dense(2, activation="relu", name="layer1"))
model.add(layers.Dense(3, activation="relu", name="layer2"))
model.add(layers.Dense(4, name="layer3"))
```

创建顺序模型—预先指定输入形状

- 通常，Keras中的所有层都需要知道其输入的形状，以便能够创建其权重。
- 模型在第一次调用输入时创建其权重，因为权重的形状取决于输入的形状。

```
layer = layers.Dense(3)
layer.weights  # Empty
```

$x \cdot w + b$

$1\times4 \quad 4\times3$

```
x = tf.ones((1, 4))
y = layer(x)
layer.weights  # Now it has weights, of shape (4, 3) and (3,)
```

创建顺序模型—预先指定输入形状

```
model = keras.Sequential(
    [
        layers.Dense(2, activation="relu"),
        layers.Dense(3, activation="relu"),
        layers.Dense(4),
    ]
)  # No weights at this stage!

# At this point, you can't do this:
# model.weights
# model.summary()

# Call the model on a test input
x = tf.ones((1, 4))
y = model(x)
print("Number of weights after calling the model:", len(model.weights))
```

创建顺序模型—预先指定输入形状

- 当逐步构建顺序模型时，可以显示到目前为止模型的摘要，包括当前的输出形状。
- 在这种情况下，应该通过向模型传递输入对象来启动模型，以便模型从一开始就知道其输入形状。

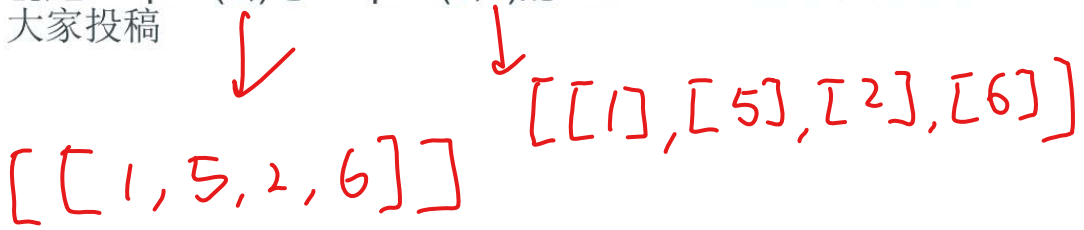
```
model = keras.Sequential()
model.add(keras.Input(shape=(4,)))
model.add(layers.Dense(2, activation="relu"))
model.summary()
```

- 一个简单的替代方法是将input_shape参数传递到第一个层

```
model = keras.Sequential()
model.add(layers.Dense(2, activation="relu", input_shape=(4,)))
model.summary()
```

想一想，练一练

- 创建shape=(4,)与shape=(4,1)的tensor（用实际数值填充），请大家投稿



顺序模型-高级部分2

TensorFlow2

通用调试工作流—add()&summary()

- 在构建新的顺序结构时，使用add()递增地堆叠层并频繁地打印模型摘要是很有用的。例如，这使您能够监视Conv2D和MaxPooling2D层是如何向下采样图像特征映射的

```
model = keras.Sequential()
model.add(keras.Input(shape=(250, 250, 3))) # 250x250 RGB images
model.add(layers.Conv2D(32, 5, strides=2, activation="relu"))
model.add(layers.Conv2D(32, 3, activation="relu"))
model.add(layers.MaxPooling2D(3))

model.summary()

model.add(layers.Conv2D(32, 3, activation="relu"))
model.add(layers.Conv2D(32, 3, activation="relu"))
model.add(layers.MaxPooling2D(3))
model.add(layers.Conv2D(32, 3, activation="relu"))
model.add(layers.Conv2D(32, 3, activation="relu"))
model.add(layers.MaxPooling2D(2))

model.summary()
# Now that we have 4x4 feature maps, time to apply global max pooling.
model.add(layers.GlobalMaxPooling2D())
# Finally, we add a classification layer.
model.add(layers.Dense(10))
```

顺序模型功能提取

- 一旦构建了一个顺序模型，它的行为就像一个函数API模型。
- 这意味着每个层都有一个输入和输出属性。
- 这些属性可以用来做一些简单的事情，比如快速创建一个模型，提取序列模型中所有中间层的输出。

```
initial_model = keras.Sequential(
    [
        keras.Input(shape=(250, 250, 3)),
        layers.Conv2D(32, 5, strides=2, activation="relu"),
        layers.Conv2D(32, 3, activation="relu"),
        layers.Conv2D(32, 3, activation="relu"),
    ]
)
feature_extractor = keras.Model(
    inputs=initial_model.inputs,
    outputs=[layer.output for layer in initial_model.layers],
)

x = tf.ones((1, 250, 250, 3))
features = feature_extractor(x)
```

顺序模型功能提取

- 下面是一个仅从一个图层提取要素的示例：

```
initial_model = keras.Sequential(
    [
        keras.Input(shape=(250, 250, 3)),
        layers.Conv2D(32, 5, strides=2, activation="relu"),
        layers.Conv2D(32, 3, activation="relu", name="my_intermediate_layer"),
        layers.Conv2D(32, 3, activation="relu"),
    ]
)
feature_extractor = keras.Model(
    inputs=initial_model.inputs,
    outputs=initial_model.get_layer(name="my_intermediate_layer").output,
)
# Call feature extractor on test input.
x = tf.ones((1, 250, 250, 3))
features = feature_extractor(x)
```

迁移学习

基于顺序模型的迁移学习

- 迁移学习包括冻结模型中的底层，而只训练顶层。
- 假设有一个顺序模型，目标希望冻结除最后一个之外的所有层。

基于顺序模型的迁移学习

- 方案一：迭代模型层然后设置可训练层=除最后一层外，每层均为假。

```
model = keras.Sequential([
    keras.Input(shape=(784))
    layers.Dense(32, activation='relu'),
    layers.Dense(32, activation='relu'),
    layers.Dense(32, activation='relu'),
    layers.Dense(10),
])

model.load_weights(...)

for layer in model.layers[:-1]:
  layer.trainable = False

model.compile(...)
model.fit(...)
```

基于顺序模型的迁移学习

- 方案二：使用顺序模型来堆叠预先训练的模型和一些新初始化的分类层

```
base_model = keras.applications.Xception(
    weights='imagenet',
    include_top=False,
    pooling='avg')

base_model.trainable = False

model = keras.Sequential([
    base_model,
    layers.Dense(1000),
])

model.compile(...)
model.fit(...)
```

参考书

- 英文版：
  - Francois Chollet, Deep Learning with Python, Manning press, November 2017.
- 中文版：
  - [美] 弗朗索瓦·肖莱（Francois Chollet） 著，Python深度学习，人民邮电出版社，2018年8月.
- 智能硬件TensorFlow实践，清华大学出版社，2017.

谢谢指正！