

人工智能实践：Tensorflow笔记

曹健

北京大学

软件与微电子学院

本讲目标：使用八股搭建神经网络

- 神经网络搭建八股
- iris代码复现
- MNIST数据集
- 训练MNIST数据集
- Fashion数据集

用Tensorflow API: tf.keras搭建网络八股

六步法

import

train, test

model = tf.keras.models.Sequential

model.compile

model.fit

model.summary

model = tf.keras.models.Sequential ([网络结构]) #描述各层网络

网络结构举例:

拉直层: **tf.keras.layers.Flatten()**

全连接层: **tf.keras.layers.Dense(神经元个数, activation= "激活函数" ,**
kernel_regularizer=哪种正则化)

activation (字符串给出) 可选: relu、 softmax、 sigmoid 、 tanh

kernel_regularizer可选: tf.keras.regularizers.l1()、 tf.keras.regularizers.l2()

卷积层: **tf.keras.layers.Conv2D(filters = 卷积核个数, kernel_size = 卷积核尺寸,**
strides = 卷积步长, padding = " valid" or "same")

LSTM层: tf.keras.layers.LSTM()

**model.compile(optimizer = 优化器,
loss = 损失函数
metrics = [“准确率”])**

Optimizer可选:

‘sgd’ or tf.keras.optimizers.SGD (lr=学习率,momentum=动量参数)

‘adagrad’ or tf.keras.optimizers.Adagrad (lr=学习率)

‘adadelta’ or tf.keras.optimizers.Adadelta (lr=学习率)

‘adam’ or tf.keras.optimizers.Adam (lr=学习率, beta_1=0.9, beta_2=0.999)

loss可选:

‘mse’ or tf.keras.losses.MeanSquaredError()

‘sparse_categorical_crossentropy’ or tf.keras.losses.SparseCategoricalCrossentropy(from_logits=False)

Metrics可选:

‘accuracy’ : y_和y都是数值, 如y_=[1] y=[1]

‘categorical_accuracy’ : y_和y都是独热码(概率分布), 如y_=[0,1,0] y=[0.256,0.695,0.048]

‘sparse_categorical_accuracy’ : y_是数值, y是独热码(概率分布),如y_=[1] y=[0.256,0.695,0.048]

model.fit (训练集的输入特征, 训练集的标签,
 batch_size= , **epochs=** ,
 validation_data=(测试集的输入特征, 测试集的标签),
 validation_split=从训练集划分多少比例给测试集,
 validation_freq = 多少次epoch测试一次)

model.summary ()

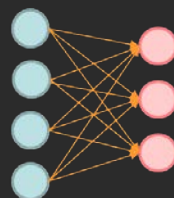
Model: "sequential"

Layer (type)	Output Shape	Param #
dense (Dense)	multiple	15

Total params: 15

Trainable params: 15

Non-trainable params: 0



输入层 输出层

源码: p8_iris_sequential.py

```
import 1 import tensorflow as tf
2 from sklearn import datasets
3 import numpy as np
4
train test 5 x_train = datasets.load_iris().data
6 y_train = datasets.load_iris().target
7
8 np.random.seed(116)
9 np.random.shuffle(x_train)
10 np.random.seed(116)
11 np.random.shuffle(y_train)
12 tf.random.set_seed(116)
13
models.Sequential 14 model = tf.keras.models.Sequential([
15     tf.keras.layers.Dense(3, activation='softmax', kernel_regularizer=tf.keras.regularizers.l2())
16 ])
17
model.compile 18 model.compile(optimizer=tf.keras.optimizers.SGD(lr=0.1),
19               loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=False),
20               metrics=['sparse_categorical_accuracy'])
21
model.fit 22 model.fit(x_train, y_train, batch_size=32, epochs=500, validation_split=0.2, validation_freq=20)
23
model.summary 24 model.summary()
```


用Tensorflow API: tf.keras搭建网络八股

六步法

import

train, test

class MyModel(Model) model=MyModel

model.compile

model.fit

model.summary

```
class MyModel(Model)    model = MyModel
```

```
class MyModel(Model):
```

```
    def __init__(self):
```

```
        super(MyModel, self).__init__()
```

定义网络结构块

```
    def call(self, x):
```

调用网络结构块，实现前向传播

```
        return y
```

```
model = MyModel()
```

```
class IrisModel(Model):
```

```
    def __init__(self):
```

```
        super(IrisModel, self).__init__()
```

```
        self.d1 = Dense(3)
```

```
    def call(self, x):
```

```
        y = self.d1(x)
```

```
        return y
```

```
model = IrisModel()
```

__init__() 定义所需网络结构块

call() 写出前向传播

import

train test

class MyModel

model.compile

model.fit

model.summary

```
1 import tensorflow as tf
2 from tensorflow.keras.layers import Dense
3 from tensorflow.keras import Model
4 from sklearn import datasets
5 import numpy as np
6
7 x_train = datasets.load_iris().data
8 y_train = datasets.load_iris().target
9
10 np.random.seed(116)
11 np.random.shuffle(x_train)
12 np.random.seed(116)
13 np.random.shuffle(y_train)
14 tf.random.set_seed(116)
```

```
15
16 class IrisModel(Model):
17     def __init__(self):
18         super(IrisModel, self).__init__()
19         self.d1 = Dense(3, activation='sigmoid', kernel_regularizer=tf.keras.regularizers.l2())
20
21     def call(self, x):
22         y = self.d1(x)
23         return y
24
25 model = IrisModel()
```

```
26
27 model.compile(optimizer=tf.keras.optimizers.SGD(lr=0.1),
28               loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=False),
29               metrics=['sparse_categorical_accuracy'])
```

```
30
31 model.fit(x_train, y_train, batch_size=32, epochs=500, validation_split=0.2, validation_freq=20)
32 model.summary()
```

源码: p11_iris_class.py

✓ **MNIST数据集:**

提供 6万张 28*28 像素点的0~9手写数字图片和标签，用于训练。

提供 1万张 28*28 像素点的0~9手写数字图片和标签，用于测试。



✓ 导入MNIST数据集:

```
mnist = tf.keras.datasets.mnist
```

```
(x_train, y_train), (x_test, y_test) = mnist.load_data()
```

✓ 作为输入特征，输入神经网络时，将数据拉伸为一维数组:

```
tf.keras.layers.Flatten( )
```

```
[ 0  0  0 48 238 252 252 ..... ..... ..... 253 186 12  0  0  0  0  0]
```

```
✓ plt.imshow(x_train[0], cmap='gray')#绘制灰度图
plt.show()
```



```
✓ print("x_train[0]:\n" , x_train[0])  
x_train[0]:
```

[illegible]

```
✓ print("y_train[0]:", y_train[0])
y_train[0]: 5
```

```
✓ print("x_test.shape:", x_test.shape)
x_test.shape: (10000, 28, 28)
```

源码: `p13_mnist_datasets.py`

源码: p14_mnist_sequential.py

```
import tensorflow as tf

mnist = tf.keras.datasets.mnist

(x_train, y_train), (x_test, y_test) = mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0

model = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dense(10, activation='softmax')
])

model.compile(optimizer='adam',
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=False),
              metrics=['sparse_categorical_accuracy'])

model.fit(x_train, y_train, batch_size=32, epochs=5, validation_data=(x_test, y_test), validation_freq=1)

model.summary()
```

import	1 import tensorflow as tf	源码: p15_mnist_class.py
	2 from tensorflow.keras.layers import Dense, Flatten	
	3 from tensorflow.keras import Model	
	4	
train test	5 mnist = tf.keras.datasets.mnist	
	6 (x_train, y_train), (x_test, y_test) = mnist.load_data()	
	7 x_train, x_test = x_train / 255.0, x_test / 255.0	
	8	
class MyModel	9 class MnistModel(Model):	
	10 def __init__(self):	
	11 super(MnistModel, self).__init__()	
	12 self.flatten = Flatten()	
	13 self.d1 = Dense(128, activation='relu')	
	14 self.d2 = Dense(10, activation='softmax')	
	15	
	16 def call(self, x):	
	17 x = self.flatten(x)	
	18 x = self.d1(x)	
	19 y = self.d2(x)	
	20 return y	
	21	
	22 model = MnistModel()	
	23	
model.compile	24 model.compile(optimizer='adam',	
	25 loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=False),	
	26 metrics=['sparse_categorical_accuracy'])	
	27	
model.fit	28 model.fit(x_train, y_train, batch_size=32, epochs=5, validation_data=(x_test, y_test), validation_freq=1)	
model.summary	29 model.summary()	

✓ **FASHION数据集:**

提供 6万张 28*28 像素点的衣裤等图片和标签，用于训练。

提供 1万张 28*28 像素点的衣裤等图片和标签，用于测试。



Label	Description
0	T恤 (T-shirt/top)
1	裤子 (Trouser)
2	套头衫 (Pullover)
3	连衣裙 (Dress)
4	外套 (Coat)
5	凉鞋 (Sandal)
6	衬衫 (Shirt)
7	运动鞋 (Sneaker)
8	包 (Bag)
9	靴子 (Ankle boot)

✓ 导入FASHION数据集:

```
fashion = tf.keras.datasets.fashion_mnist
```

```
(x_train, y_train),(x_test, y_test) = fashion.load_data()
```

参考源码: [p16_fashion_sequential.py](#)

[p16_fashion_class.py](#)