function_fit

在使用$Relu$进行函数拟合时，选取一个非线性函数$f(x)=sin(cos(x))$作为拟合目标，自行在目标函数上随机采样生成训练集和测试集，我们通过TensorFlow中的深度学习框架进行拟合，训练，并输出最后拟合的结果

导入包

```
import numpy as np
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers, optimizers
import matplotlib.pyplot as plt
```

定义目标函数并生成训练集和测试集

```
# 定义目标函数
def target_function(x):
    return np.sin(np.cos(x))

# 生成训练集和测试集
np.random.seed(0)  # 设置随机种子，以确保可重复性
x_train = np.random.uniform(-2*np.pi, 2*np.pi, size=(1000, 1))  # 在区间[-2π, 2π]
内随机采样生成训练数据
y_train = target_function(x_train) + np.random.normal(0, 0.1, size=(1000, 1))  #
添加一些噪声

x_test = np.random.uniform(-2*np.pi, 2*np.pi, size=(100, 1))  # 生成测试数据
y_test = target_function(x_test)  # 计算测试数据的标签，没有添加噪声
```

(1)基于TensorFlow的深度学习框架进行训练

```
tf_model = keras.Sequential([
    layers.Dense(10,activation='relu',input_shape=(1,)),
    layers.Dense(10,activation='relu'),
    layers.Dense(1)
])

# 编译模型
tf_model.compile(optimizer='adam', loss='mean_squared_error')

#模型训练
history = tf_model.fit(x_train, y_train, epochs=100, batch_size=32,
validation_data=(x_test, y_test), verbose=0)

#模型评估
```

```
test_loss = tf_model.evaluate(x_test,y_test,verbose=0)

print("Test Loss:",test_loss)



plt.plot(history.history['loss'], label='train_loss')
plt.plot(history.history['val_loss'], label='val_loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```