报告

1. 函数定义

自定义函数：$y = sin(x) + 0.2x$

该函数将作为我们神经网络模型的目标，我们将使用该函数生成训练数据和测试数据。

目标函数的定义如下：

```
def target_function(x):
    return np.sin(x) + 0.2 * x
```

2. 数据采集

我们从区间$[-5, 5]$内生成了训练数据和测试数据。训练数据用于模型的训练，测试数据用于评估模型的拟合效果。

生成数据的代码：

```
def generate_data(start=-5, end=5,
num_samples=500):
    x = np.linspace(start, end, num_samples)  # 在区间内均匀采样
    y = target_function(x)  # 根据目标函数生成y值
    return x, y
```

我们使用 `np.linspace` 生成500个在$[-5, 5]$区间内均匀分布的样本。

对于每个x 值，使用目标函数计算对应的y 值。

3. 模型描述

我们使用了一个简单的两层 ReLU 神经网络来拟合目标函数。神经网络包含：

- 输入层：1 个节点，对应输入的x值。
- 隐藏层：20 个节点，使用 ReLU 激活函数。
- 输出层：1 个节点，输出拟合值。

神经网络的结构

1. 输入层到隐藏层：权重矩阵W_1和偏置b_1。
2. 隐藏层到输出层：权重矩阵W_2和偏置b_2。
3. 激活函数：隐藏层使用 ReLU 激活函数，输出层不使用激活函数。

```python
class TwoLayerNN:
    def __init__(self, input_size, hidden_size, output_size, learning_rate=0.001):
        # 初始化权重和偏置
        self.W1 = np.random.randn(input_size, hidden_size) * 0.1
        # 输入层到隐藏层的权重
        self.b1 = np.zeros(hidden_size)  # 隐藏层的偏置
        self.W2 = np.random.randn(hidden_size, output_size) * 0.1
        # 隐藏层到输出层的权重
        self.b2 = np.zeros(output_size)  # 输出层的偏置
        self.learning_rate = learning_rate  # 学习率
```

模型训练

使用均方误差（MSE）作为损失函数，反向传播计算梯度，更新模型参数。

学习率 设置为 0.01， 训练轮次 设置为 20000

```python
# 4. 训练模型
x_train, y_train = generate_data(-5, 5, 100)  # 生成训练数据
x_train = x_train.reshape(-1, 1)  # 转换为列向量
y_train = y_train.reshape(-1, 1)  # 转换为列向量

# 初始化神经网络，输入维度=1，隐藏层神经元=20，输出维度=1
model = TwoLayerNN(input_size=1, hidden_size=20, output_size=1)

# 训练模型
model.train(x_train, y_train, epochs=20000)
```

4. 拟合效果

训练结果

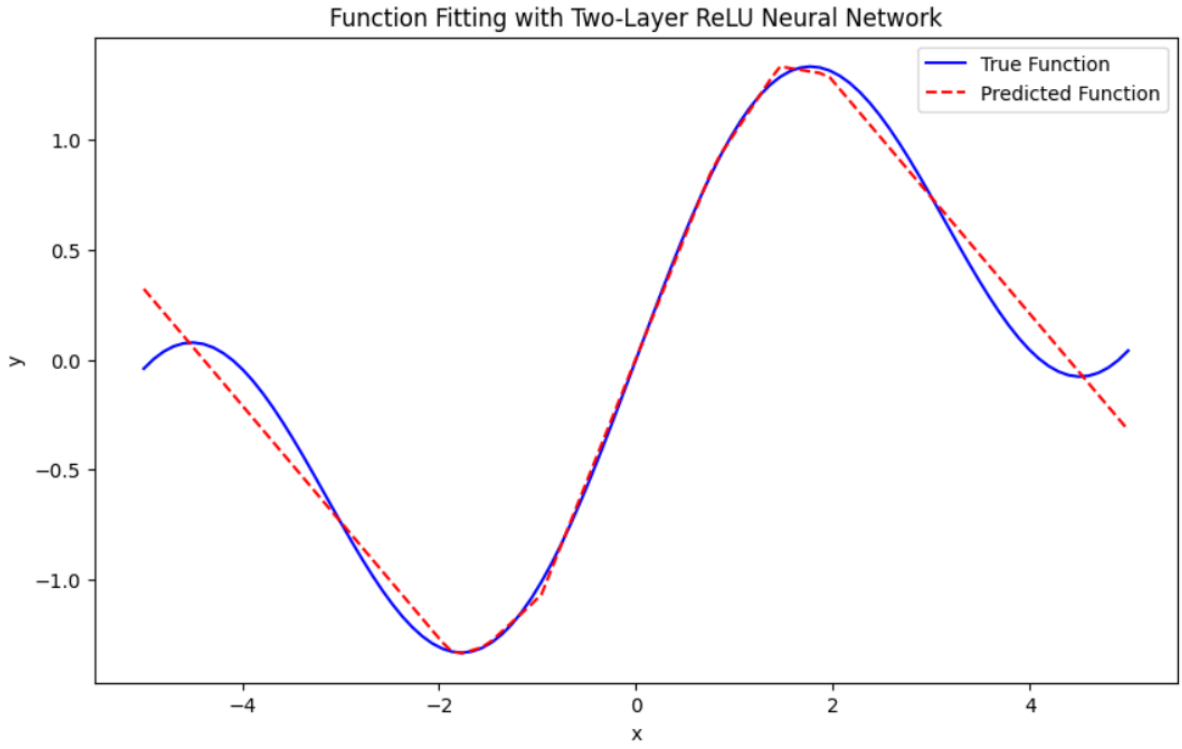
20000 个轮次训练完后 loss 的值为 0.0106

```
Epoch 2000, Loss: 0.0108
Epoch 2100, Loss: 0.0108
Epoch 2200, Loss: 0.0107
Epoch 2300, Loss: 0.0107
Epoch 2400, Loss: 0.0107
...
Epoch 19600, Loss: 0.0106
Epoch 19700, Loss: 0.0106
Epoch 19800, Loss: 0.0106
Epoch 19900, Loss: 0.0106
```

可视化拟合结果

- 蓝色曲线：表示目标函数 $y = sin(x) + 0.2x$
- 红色虚线：表示神经网络的预测结果。

经过训练后，神经网络的预测值已经非常接近真实值，说明模型有效地拟合了目标函数。


Function Fitting with Two-Layer ReLU Neural Network
True Function
Predicted Function
1.0
0.5
0.0
−0.5
−1.0
y
−4
−2
0
2
4
x