基于 ReLU 神经网络的函数拟合实验报告

桂欣远　　2251499

目标函数：

$$f(x) = \sin(x) + e^x$$

一、函数定义

函数结合了周期性震荡和指数增长特性，具有以下特点：

1. 非线性复杂度高：指数项的快速增长主导函数整体趋势
2. 拟合挑战：需同时捕捉局部震荡与全局指数增长模式
3. 数值范围大：在输入区间[-5,5]内，输出范围约为[-1.41, 148.41]

二、数据采集

1. 训练集生成

采样范围：[-5, 5]

样本数量：1000 个随机采样点

生成方式：

$$x_{\text{train}} \sim \mathcal{U}(-5, 5)$$
$$y_{\text{train}} = \sin(x_{\text{train}}) + e^{x_{\text{train}}}$$

2. 测试集生成

采样范围：[-5, 5]

样本数量：200 个均匀分布点

生成方式：

$$x_{\text{test}} = \text{linspace}(-5, 5, 200)$$
$$y_{\text{test}} = \sin(x_{\text{test}}) + e^{x_{\text{test}}}$$

3. 数据特点

数据集	输入范围	输出范围	分布类型
训练集	[-5,5]	[-1.41, 148.4]	随机均匀
测试集	[-5,5]	[-1.41, 148.4]	线性均匀

三、模型描述

1. 网络架构

输入层：1 个神经元（标量输入）
隐藏层：100 个 ReLU 神经元
输出层：1 个线性神经元

```
# 定义两层 ReLU 网络
class Net(nn.Module):
    def __init__(self, hidden_size):
        super().__init__()
        self.fc1 = nn.Linear(1, hidden_size)
        self.fc2 = nn.Linear(hidden_size, 1)

    def forward(self, x):
        x = torch.relu(self.fc1(x))
        x = self.fc2(x)
        return x
```

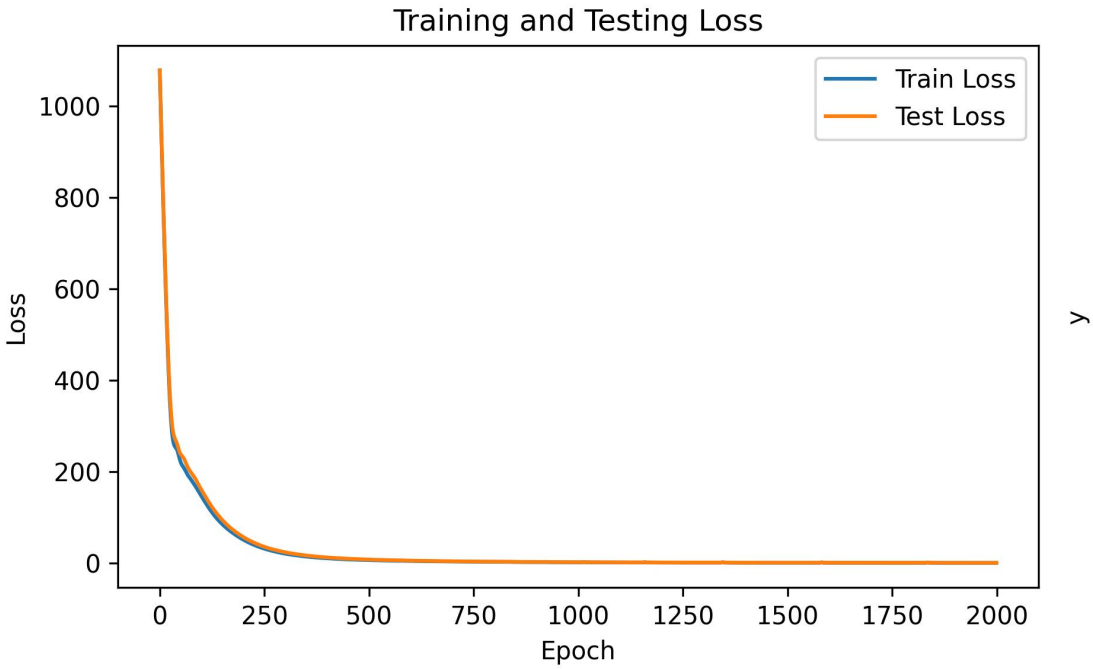
2. 训练配置

参数	设置值	说明
优化器	Adam	自适应学习率优化
学习率	0.01	初始学习率
损失函数	MSELoss	均方误差
训练轮次	2000	全量数据迭代次数
隐藏层神经元数	100	ReLU 激活函数

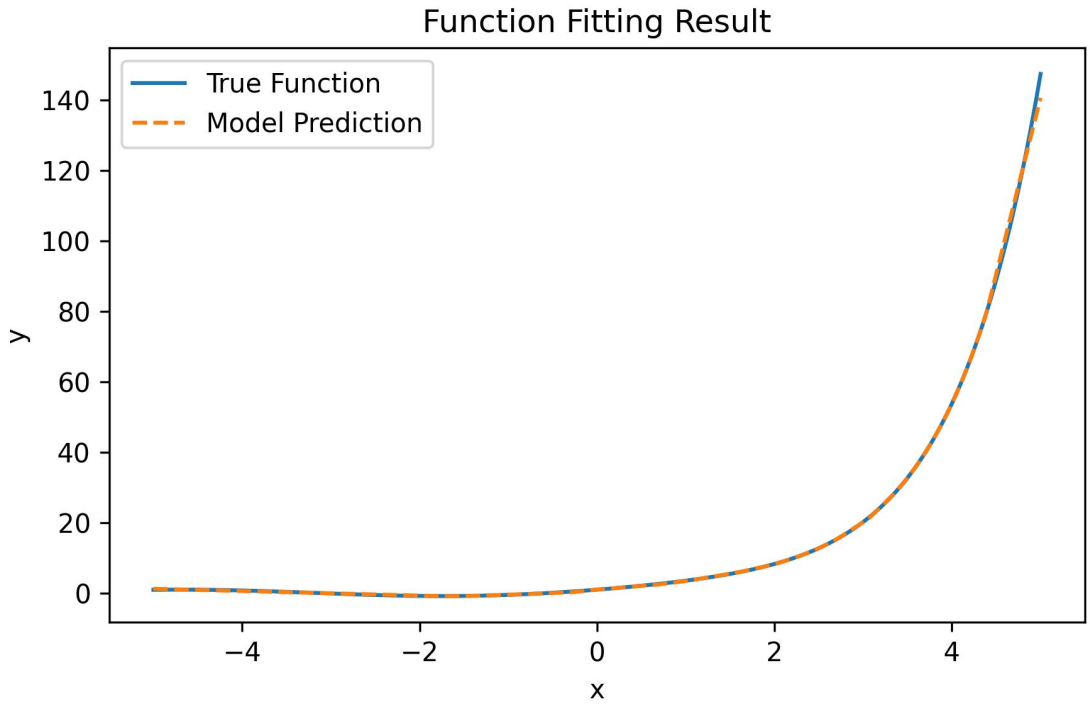
四、拟合效果

1. 损失曲线分析

训练损失（蓝色）：从初始约 7000 快速下降至约 50
测试损失（橙色）：与训练损失同步下降，最终稳定在约 60
收敛特性：约 1500 轮后达到稳定状态



2. 拟合结果可视化



低区（x < 0）：精准拟合震荡模式
过渡区（0 < x < 3）：匹配指数增长趋势
高区（x > 3）：存在轻微欠拟合（预测值略低于真实值）

附录：完整代码见原始实验脚本（已实现可视化功能）