**神经网络与深度学习实验报告**

**题目：基于 ReLU 的两层神经网络实现与拟合效果分析**

****

**学生姓名：**  李闯

**学 号：**  2253214

**专业班级：** 计算机科学与技术2班

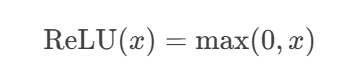
**指导教师：**  程大伟

2025年03月22日

1. **函数定义**

**本实验实现了一个基于 ReLU 激活函数的两层神经网络（ReLUNetwork），用于拟合目标函数。以下是核心函数的定义：**

* **ReLU函数：**



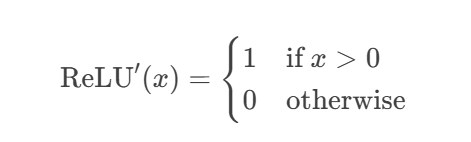
**代码实现：**

**def relu(x):**

***"""ReLU 激活函数"""***

**return np.maximum(0, x)**

* **ReLU 的导数：**



**代码实现：**

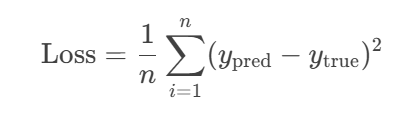
**def relu\_derivative(x):**

***"""ReLU 的导数"""***

**return (x > 0).astype(float)**

* **损失函数**

**使用均方误差（MSE）作为损失函数：**



**代码实现：**

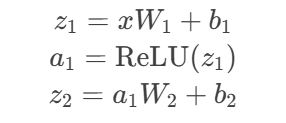
**def compute\_loss(self, pred\_y, true\_y):**

***"""计算均方误差"""***

**return np.mean((pred\_y - true\_y) \*\* 2)**

* **前向传播：**

**计算隐藏层和输出层的值：**



**代码实现：**

**def forward(self, x):**

***"""前向传播"""***

**self.z1 = np.dot(x, self.W1) + self.b1**

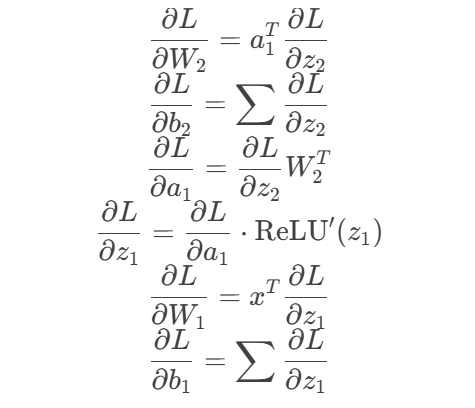
**self.a1 = self.relu(self.z1)**

**self.z2 = np.dot(self.a1, self.W2) + self.b2**

**return self.z2  # 线性输出（无激活）**

* **反向传播**

**计算梯度并更新参数：**



**代码实现：**

**def backward(self, x, pred\_y, true\_y):**

***"""反向传播"""***

**batch\_size = true\_y.shape[0]**

**# 输出层梯度**

**dL\_dz2 = 2 \* (pred\_y - true\_y) / batch\_size**

**dL\_dW2 = np.dot(self.a1.T, dL\_dz2)**

**dL\_db2 = np.sum(dL\_dz2, axis=0, keepdims=True)**

**# 传播到隐藏层**

**dL\_da1 = np.dot(dL\_dz2, self.W2.T)**

**dL\_dz1 = dL\_da1\*self.relu\_derivative(self.z1)**

**dL\_dW1 = np.dot(x.T, dL\_dz1)**

**dL\_db1 = np.sum(dL\_dz1, axis=0, keepdims=True)**

**# 更新参数**

**self.W1 -= self.learning\_rate \* dL\_dW1**

**self.b1 -= self.learning\_rate \* dL\_db1**

**self.W2 -= self.learning\_rate \* dL\_dW2**

**self.b2 -= self.learning\_rate \* dL\_db2**

1. **数据采集**

**本实验的目标是拟合二次函数 。数据采集过程如下：**

1. **在区间[−1,1]内均匀采样500个点作为训练数据。**
2. **计算每个点的 值作为标签。**
3. **代码实现：**

**def target\_function(x):**

**return np.exp(x)**

**def generate\_data(func, x\_min=-10, x\_max=10, num\_points=100, test\_ratio=0.2, normalize=True):**

***"""生成训练和测试数据"""***

**x = np.linspace(x\_min, x\_max, num\_points).reshape(-1, 1)**

**x\_min, x\_max = np.min(x), np.max(x)**

**y = func(x)**

**return x, y, x\_min, x\_max**

**# 生成数据**

**x, y, x\_min, x\_max = generate\_data(target\_function, x\_min=-1, x\_max=1, num\_points=500)**

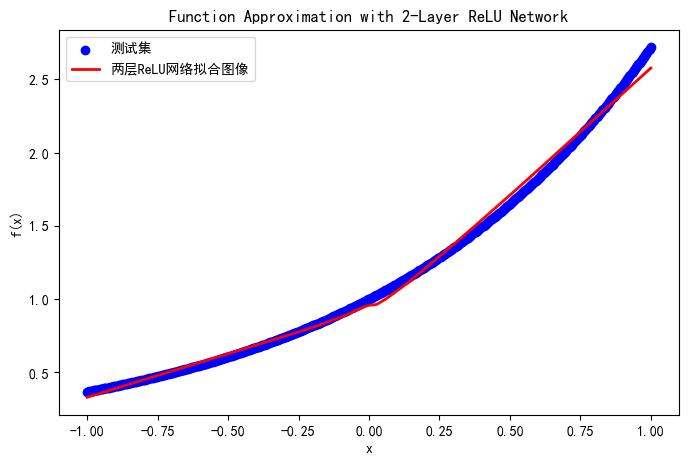
1. **模型描述：**

**本实验使用了一个两层的全连接神经网络：**

1. **输入层：1 个神经元（输入维度为 1）。**
2. **隐藏层：128 个神经元，使用 ReLU 激活函数。**
3. **输出层：1 个神经元，线性输出。**
4. **训练轮数：10000 轮。**
5. **拟合效果**

**在训练过程中每 500 轮打印一次损失值，损失函数逐渐下降，最终损失值接近 0，表明模型能够较好地拟合训练数据。**

**最终的拟合效果如下图所示：**

****