**一、多层感知机用于 MNIST 手写数字数据集分类（提交实现步骤描述以及下面要求提交的结果）**

1、获取 MNIST 数据集，每张图片像素为28 × 28

2、模型架构为包含两个隐含层的多层感知机模型输入层维度：28 × 28 = 784

第一层隐含单元数：256 第二层隐含单元数：256

输出层维度：10（MNIST 数据集类别数，分别为 0 到 9）

3、画出训练和测试过程的准确率随迭代次数变化图，画出训练和测试过程的损失随迭代次数变化图。**（提交最终分类精度、分类损失以及两张变化图）**

**二、卷积神经网络用于 MNIST 手写数字数据集分类（提交实现步骤描述以及下面要求提交的结果）**

1、获取 MNIST 数据集，每张图片像素为28 × 28

2、模型架构：

输入层维度：28 × 28

（卷积层和池化层的 padding 都是用‘SAME’）

卷积层 1：卷积核大小为5 × 5，卷积核个数为 32（输出维度为28 × 28 × 32） 池化层 1：使用最大池化，核大小的2 × 2，stride 为 2（输出维度为14 × 14 × 32） 卷积层 2：卷积核大小为5 × 5，卷积核个数为 64（输出维度为14 × 14 × 64） 池化层 2：使用最大池化，核大小的2 × 2，stride 为 2（输出维度为7 × 7 × 64）

（将池化层 2 的输出展平作为全连接层的输入，输入维度为7 × 7 × 64 = 3136）

全连接层：隐含单元数为 1024 Dropout 层：Dropout 率为 0.25

输出层维度：10（MNIST 数据集类别数，分别为 0 到 9）

3、画出训练和测试过程的准确率随迭代次数变化图，画出训练和测试过程的损失随迭代次数变化图。**（提交最终分类精度、分类损失以及两张变化图）**

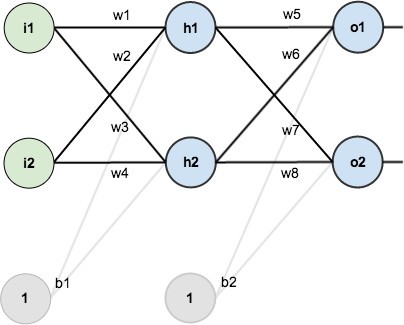
**三、多层感知机实现异或运算（提交实现步骤描述、源代码以及最后的测试误差）** 要求：**不允许使用 Pytorch/Tensorflow 等深度学习框架**，使用 Python 实现网络的前向传播和反向传播过程。源代码文件命名为“班级\_学号\_姓名\_BP.py”。

数据集：[[[0, 0], [0]],

[[0, 1], [1]],

[[1, 0], [1]],

[[1, 1], [0]]]

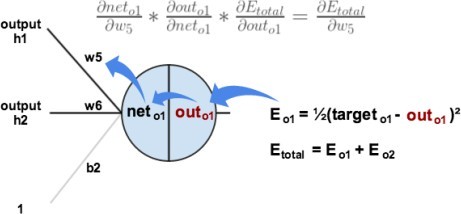
**BP 算法实现**

**可供参考的实现过程： 单个神经元操作：**

1、定义参数矩阵、定义激活函数、定义损失函数

2、计算神经元输出

3、计算误差



**实现前向传播：**

1、定义神经元个数

2、计算层中每个神经元的输出

**实现反向传播：**

1、计算误差

2、计算每一层中权重的梯度

3、更新输出层权重

4、更新隐含层权重