**《数字图像处理与机器视觉》课程作业（手写数字识别）**

**一、任务目标**

采用特征提取与匹配，以及深度学习方法对手写数字进行识别，并形成实验报告。

**二、数据集说明**

MNIST手写数字数据集（公开数据集），下载链接：[百度网盘](https://pan.baidu.com/s/1_TpyV3xER2gl3-vH_fqrhA?pwd=qc1j)。

数据集中包含原始的二进制格式数据和经过可视化处理后的图像文本：

* raw文件夹：包含原始的二进制数据文件；
* mnist.zip和mnist-part.zip压缩包：包含将原始数据可视化后生成的图像文件，图像命名格式为<label-index>.jpg，其中：
  + label表示图像的真实类别（数字0-9）；
  + index 表示该图像在所属数据集（训练集或测试集）中的编号（与类别无关）。

解压后的文件结构如下：

mnist (mnist-part)

├─ train (train-part) # train包含60K张图像；train-part包含5K张图像（每类100张）

│ └─ <label-index>.jpg

└─ test (test-part) # test包含10K张图像；test-part包含1K张图像（每类10张）

└─ <label-index>.jpg

**三、任务说明**

1. **下载数据集**

请将下载的压缩包统一解压到项目代码根目录下的data文件夹中：

* mnist/: 用于深度学习方法的训练与测试；
* mnist-part/ : 用于图像特征提取与匹配方法的实现。

1. **图像特征匹配方法**
   1. **图像特征提取：**将图像划分成7x7个区域，即每一个小方块中含有4x4=16个像素点，通过统计小方块中的像素值，分别给小方块标记为0或1（其中当前景像素超过16/2 = 8个时，记该小方块为1，反之，记为0）。最终，每一张图片可以表示成49个0-1数字向量。
   2. **模板库构建：**对训练集（train-part）中的所有图片进行上述处理，并将其特征向量进行存储（称为模板库）。假设有N个训练数据，即可以存储为Nx49的一个矩阵。
   3. **测试图像匹配：**对测试集（test-part）的图片进行相同转化，同样可以得到49维的0-1向量。将其与模板库中所有的向量进行对比，计算其欧式距离，选出欧氏距离最小对应的模板数字，即认为是和该测试图片是同一类数字。
2. **深度学习/神经网络方法**
   1. 编程采用至少两种神经网络结构（全连接网络/卷积神经网络），实现手写数字识别模型训练。代码可参考：[pytorch-mnist](https://gitee.com/Kiumb/pytorch-mnist)。
   2. 在MNIST测试集中验证识别准确率。
3. **测试自己构建的数据集**

拍摄自己手写的数字图片，裁剪并压缩至28x28像素，并进行图像分割转换成二值图像（与MNIST数据集保持一致）。利用上述（2）和（3）的方法对其进行识别，给出可视化的测试结果。

1. **扩展与讨论内容**
   1. 讨论“（2）图像特征匹配方法”中是影响识别精度的因素有哪些；是否可以采用更好的特征表示方法（如特征点提取SIFT、区域提取等）；尝试对其进行改进和测试。
   2. 讨论“（3）深度学习/神经网络方法”不同网络/训练参数设置的影响。

注：MNIST测试识别准确率计算方式：

**四、系统与环境配置要求**

识别MNIST的项目对torch、torchvision等依赖库的版本要求较为宽松，可根据具体的硬件环境（如GPU型号、CUDA版本或是否使用GPU）进行相应的配置。

以下是参考的项目环境配置：

* Python 版本为 3.8.0；
* Pytorch 深度学习框架：torch == 1.10+cu113、torchvision == 0.11.0+cu113（CPU版本同样支持完整训练流程）；
* 图像处理与辅助工具：
  + numpy == 1.24.4 ：用于高效处理图像数组数据
  + matplotlib == 3.7.5 ：用于图像和结果的可视化展示
  + opencv-python == 4.10.0.84：用于图像读取、显示与保存等操作
  + tqdm == 4.66.6：用于显示训练进度条，提升可视化体验