《人工智能导论》大作业

任务名称： 带 OOD 检测的 Mnist 分类器

完成组号：

小组人员：陈骥康 陈驰宇 杨源盛 柴继晨 杜贺

完成时间： 2023.6.15

1. 任务目标

目标：基于 Mnist 数据集和非数字图像数据集，构建一个分类模型。该模型可以对 mnist 数据集的图像正确识别代表的数字，同时对于非数字的图像，识别为 OOD 类。

要求：

√. 模型是 11 个分类（0-9 代表数字的十个分类和一个 OOD 类）；

√. 在 CPU 上有合理的运行时间。

2. 具体内容

（1）方案选择

在实施方案的选择上，我们组考察比较了四种 OOD 检测器实现方法，其核心思想与优缺点总结如下：

1. ODIN 方法：

- 基本原理：通过 temperature scaling 和 input preprocessing 处理输入图像，以此放大 ID 样本与 OOD 样本的 softmax 分布差异，并设置阈值区分这两种样本。
- 优点：实现简单，无需单独训练 OOD 检测器
- 缺点：严重依赖超参数的设置，经实验其效果未达小组预期。

2. Early_Layer 方法：

- 基本原理：选择合适的中间层作为特征表示层，进行特征提取，利用早期层次的特征表示来区分 ID 与 OOD 输入。
- 优点：无需 OOD 数据集参与训练，算法复杂性较低，训练效率高。
- 缺点：对于 ID/OOD 边界数据识别准确率较差。

（代码已提交到 GitHub 项目分支）

3. Uncertainty 方法：

- 基本原理：模型的概率输出不能表示模型的置信度。在神经网络中增加对于置信度的输出层，若模型输入为 ID 样本，则不确定性低；若模型输入为 OOD 样本，则不确定性高。
- 优点：易于实现操作简单，可产生直观的可解释的输出
- 缺点：对于阈值敏感；仅适用于简单的 OOD 检测，效果不理想。

4. Generative Model 方法：

- 基本原理：利用 VAE 的 reconstruction error 来判断样本是否属于 ID 或 OOD 样本。假设 Autoencoder 的隐含空间能学习出 ID 数据的明显特征，对于 OOD 样本则不行，故 OOD 样本会产生较高的 reconstruction error。
- 优点：可预处理多种类型的数据；对于复杂的 OOD 检测场景具有很好的鲁棒性

- 缺点：训练所需时间等成本较高；对于数据分布的假设和参数的调整要求较高。

经反复讨论和比较，最终决定采用 Generative Model 方法。

（2）核心代码分析

本项目基于 VAE 的生成模型，能够对手写数字图像进行识别、OOD 检测、CIFAR10 图像识别和数字图像生成。

程序的作用介绍

文件名	功能
config.py	读取配置文件并将配置信息转化为表格数据格式
model_vae.py	VAE 模型架构的实现代码
oodcls.py	接口调用程序
test_cifar.py	使用 CIFAR10 数据集测试模型，并计算 accuracy
test_generated.py	使用生成数字图像测试模型，并绘制图像和预测值
test_mnist.py	使用 MNIST 数据集测试模型，并计算 accuracy
train.py	模型参数的训练程序

重点函数功能介绍

model_vae.py:

- `encode(self, x)`函数
  卷积神经网络编码器部分，包括三个卷积层和两个全连接层，用于将输入图像编码为低纬度潜在空间中的向量。

- `reparameterize(self, mu, logvar)`函数
  重参数化过程，从高斯分布中采样潜在向量，并使用偏置项和标准差重构生成的样本。

- `decode(self, z)`函数
  卷积神经网络解码器部分，包括两个转置卷积层和一个全连接层，用于将潜在向量解码并重构为图像。

- `forward(self, x)`函数
  定义 VAE 的前向传递过程，包括编码器、随机采样、解码器、均值向量和对数方差向量，并返回输出结果、均值向量和对数方差向量作为训练过程中的中间结果。

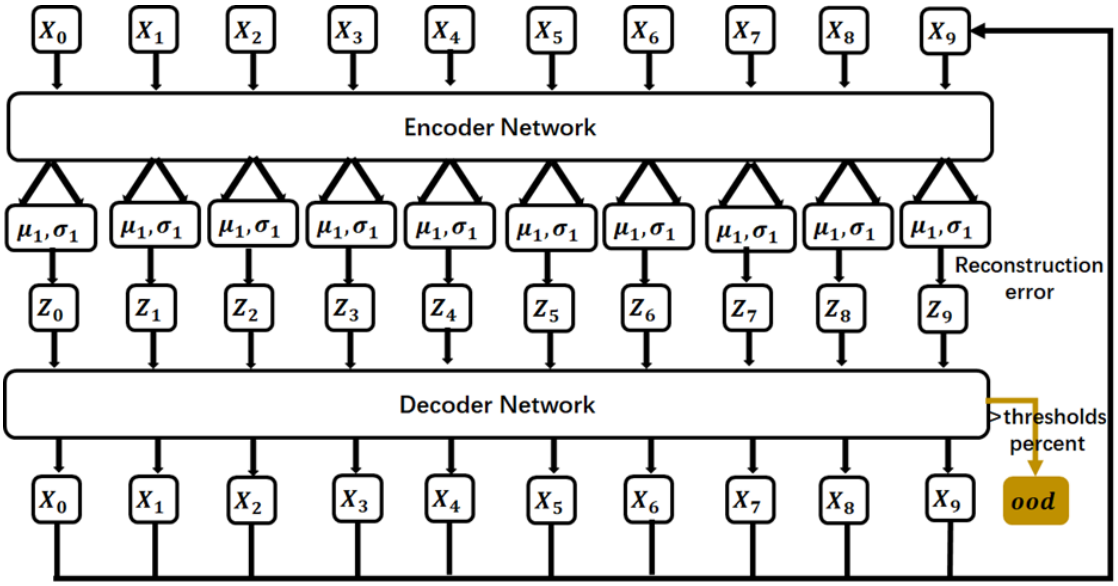
train.py

- `loss_function(self, recon_x, x, mu, logvar)`函数
  定义 VAE 的损失函数，包括重构误差和 KL 散度。

- `train(self, epochs, model, lr, batch_size)`函数
  用于训练 VAE 模型，包括计算每个数字的重构损失和 KL 散度，使用 Adamax 优化器进行参数优化，使得解码器能够将潜在编码解码为原始图像。

模型原理解析

参考了 GitHub：https://github.com/zxjzxj9/PyTorchIntroduction/blob/master/Chapter4/vae.py 的线性 VAE 模型后，本组将其进行修改，采用卷积层的方式完成该项目，增加模型鲁棒性。

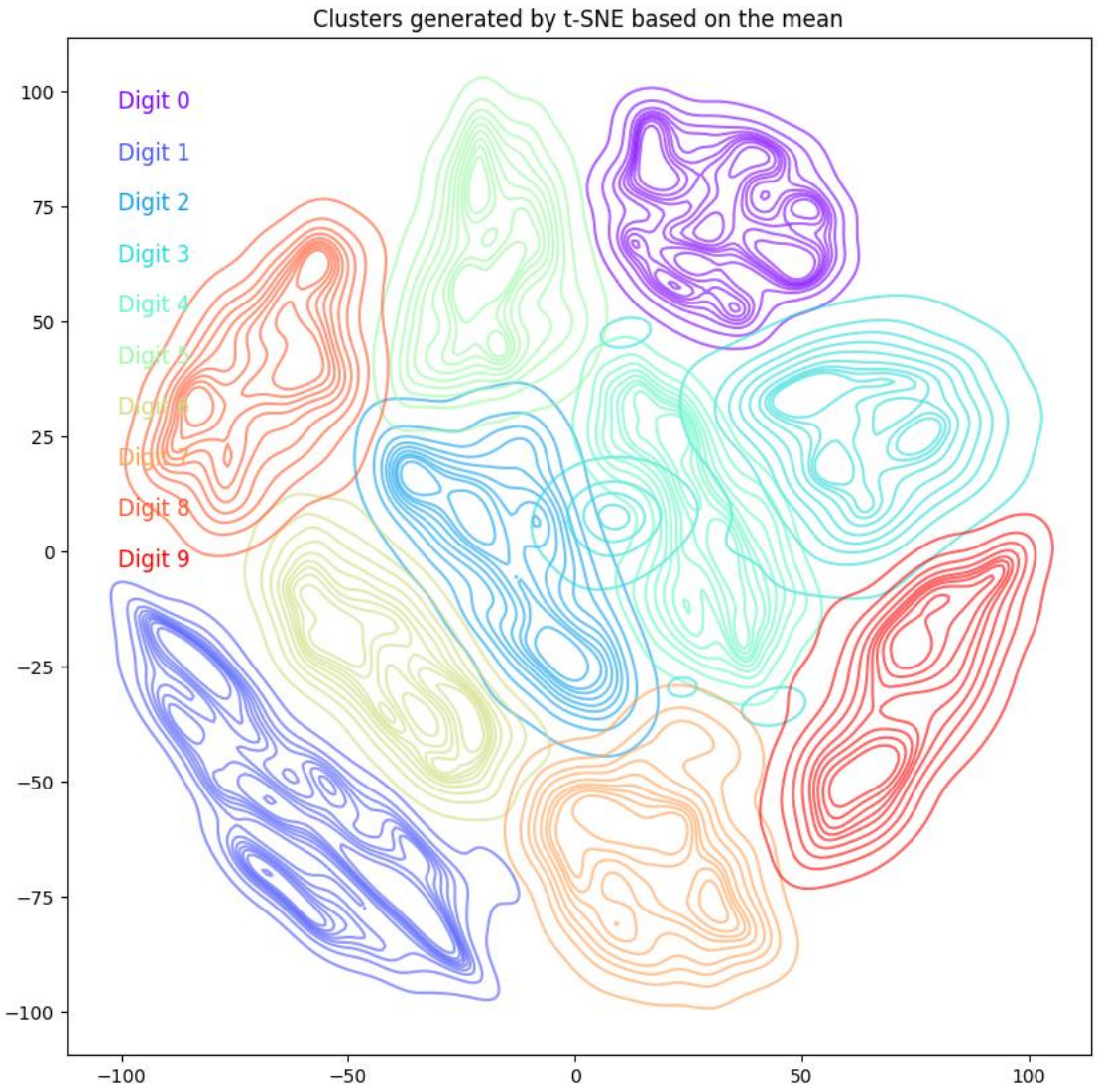


损失函数使用 ELBO loss 评估，该损失由 Reconstruction loss（重构损失）和 Kullback–Leibler loss（KL loss）组成。

性能测试

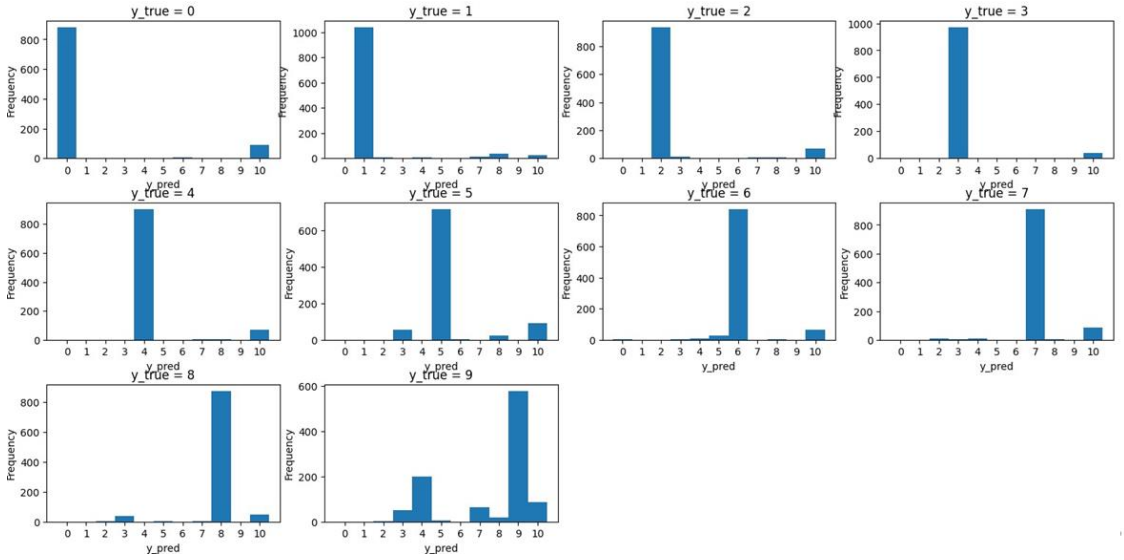
mnist 训练集：

将数据通过对应的 VAE 模型的编码器进行嵌入，转换为向量表示，并使用 t-SNE 将嵌入向量从高维空间降维到二维空间，以便进行可视化。二维空间中该数据集核密度估计图如下图所示。该图反映了不同类别数据在潜在空间中的分布情况。



mnist 测试集：

对 mnist 测试集测试效果如下图所示：




```
Mnist Accuracy: 9608/10000 (96.08%)
```

由图可知，除对数字 9 的识别能力稍弱以外，其他数字识别准确率均较高，在考虑 OOD 的情况下整体识别 **accuracy** 可达 96.08%。

cifar 测试集：

在 cifar 测试集中，模型对 cifar 的 OOD 判断 **accuracy** 高达 100%，说明模型对非数字的 OOD 模型识别准确率极高。

```
Files already downloaded and verified
CIFAR Accuracy: 1.0000
```

generated_mnist 测试集：

在本测试集中，对其他生成的数字集进行测试，检测结果如下图所示：

由图可知，该模型对数字的判断准确率较高，部分数字判断不够准确，例如第一行的数字 6 和第二行的数字 9 等应该判断为 OOD 比较合适。而对其他的数字判断较准确。

3. 工作总结

（1）收获、心得

通过本次大作业，本组成员收获了合作完成项目的宝贵经验，提高了团队意识和分工合作能力。

经组员讨论总结，大家对深度学习和 OOD 检测领域有了初步认识，对 pytorch 使用和人工智能开发过程有了进一步了解。此外，各组员的文献检索与阅读能力和代码理解与应用能力得到了显著提升。

通过开发一个简单的人工智能模型，本组成员在实践中加深了对课堂上学习的理论知识的理解。

（2）遇到问题及解决思路

Q：模型构建过程中张量形状不匹配等问题导致各种报错

A：根据报错信息排查问题所在，使用 debug 模式进行调试

Q：如何将原有模型迁移到本项目上
A：了解原有代码作用后，根据本项目要求对原模型架构等进行修改调整。

4．课程建议

1. 平时作业多一些代码实操，加深对理论知识的理解
2. 受学时限制，各方面讲得略浅，不够深入。