# Week5实验

## 实验目标

- 本次实验主要包括两个编程任务
  - 基于PyTorch的内置模块，实现卷积神经网络LeNet5
  - 基于PyTorch框架，实现对抗样本算法FGSM

## 实验步骤

### Task 1：实现卷积神经网络LeNet5

- LeNet-5网络是一个简单的卷积神经网络结构，包含卷积层、池化层和全连接层
  - 具体结构可参考PPT里的图，其中@之前的数值表示该层的特征图通道数，@之后的数值表示该层的特征图大小，FC表示全连接层的特征维度
- 请基于PyTorch的内置模块，实现该模型
  - 更多细节请参考Notebook文件中的注释
  - 训好的模型记得保存并下载到自己的本地

### Task 2：实现FGSM对抗攻击

- FGSM攻击的核心公式：$\tilde{x} = x + \epsilon \cdot \text{sign}(\nabla_x \ell(f_\theta(x), y))$
- 请基于你训练完毕的LeNet5模型，基于PyTorch框架实现该攻击
  - 更多细节请参考Notebook文件中的注释

### Bonus：实现生成对抗网络GAN

- 请在MNIST上实现生成对抗网络GAN，包括一个Generator和一个Discriminator
  - 你需要定义Generator、Discriminator的结构与前向传播行为
  - 你需要制定两者的训练目标，并完成训练
  - 训好的两个模型记得保存并下载到自己的本地

- 一些hint：
  - Generator和Discriminator可以尝试用Linear层做，也可以尝试用Conv2d层和ConvTranspose2d层做；前者实现难度低一些，真的猛士可以自行尝试后者；
  - tensor.detach()是一个很有用的函数，训练过程中会用到它的；
  - 完成代码填空后，可以先试着训5或10个epoch，然后观察训练loss的趋势（G_loss和D_loss并不一定都会收敛，但肯定不会一直很大）和生成图片的质量，以便快速试错；
- GAN的实现难度较大，可以参考网络资料，但**务必确保你真的理解了它们**！

# 检查内容

- Task 1
  - 代码能正常运行
  - 能描述模型结构与前向传播的实现细节
  - 训练loss稳定收敛
  - 10个epoch下，LeNet5的测试准确率高于93%
- Task 2
  - 代码能正常运行
  - 能描述FGSM以及evaluate函数的实现细节
  - 在$\epsilon = 0.2$的扰动预算下，对抗样本的match rate低于0.4
  - 对抗样本可视化后，扰动不影响原数字的呈现
- Bonus
  - 代码能正确运行
  - 能描述生成器、判别器的结构、前向传播，以及训练过程的细节（划重点，要理解！）
  - 生成器与判别器的训练loss维持在较低水平，例如低于3.0（允许竞争性抖动，它们确实不一定能收敛）
  - 生成器输出的图片应该具有一定的语义：

    - 每训练5或10个epoch，测试一下生成器的生成质量，生成图片的真实性应当能逐渐提升
    - 最终生成器输出的图片应该符合MNIST数据的分布

# 附录

- ModelScope服务器端无法长久保存文件，因此请及时下载修改好的代码文件、保存的模型参数（model/lenet5.pt）、生成的对抗样本（data/*.pkl）