实验名称：后门攻击的实现

实验原理：面向人工智能算法的后门攻击，是指在不改变原有人工智能算法所依赖的深度学习模型结构的条件下，通过向训练数据中增加特定模式的噪音，并按照一定的规则修改训练数据的标签，达到人工智能技术在没有遇到特定模式的噪音时能够正常工作，而一旦遇到包含了特定模式的噪音的数据就会输出与预定规则相匹配的错误行为。

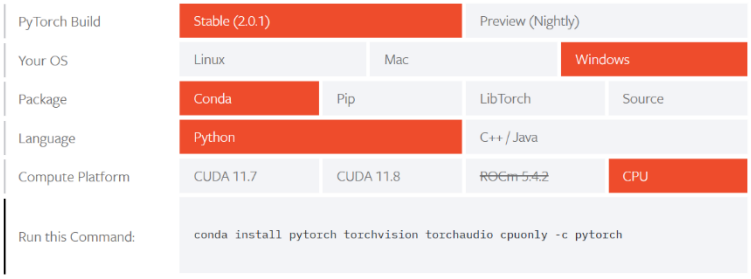
实验环境： Windows 11 电脑一台

实验步骤：

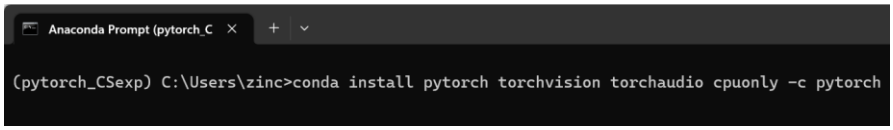
1. 配置实验环境
   1) 本实验使用一台搭载 Windows11 的计算机进行演示。系统信息如下图所示。

版本	Windows 11 专业版 Insider Preview
版本	22H2
安装日期	2023/5/8
操作系统版本	23451.1000
体验	Windows Feature Experience Pack 1000.23451.1000.0

   2) 从 https://github.com/GeorgeTzannetos/badnets 下载实验所需代码。
   3) 安装 Anaconda 软件，配置 python3.11 实验环境 pytorch_CSexp。
   4) 在 https://pytorch.org/get-started/locally/网站中获取安装命令，根据自己设备实际情况选择安装命令，我的选择如下图所示。



   5) 在 Anaconda Prompt 中输入上面选择的安装命令，安装 pytorch。



   6) 在 Anaconda Prompt 中输入命令 pip install -r requirements.txt，按照代码中 requirements.txt 文件要求安装依赖库。

   7) 安装 PyCharm 软件作为代码编辑器，也可选择其它软件。
2. 测试后门攻击
   1) 在 main.py 中将数据集 dataset 设置为 mnist，即第 13 行的 default 设置为'mnist'；将样本污染比例设置为 0.1，即第 14 行 default 设置为 0.1；将 trigger 设置为 7，即第 15 行的 default 设置为 1；将 batch size 设置为 1000，即第 16 行 default 设置为 1000；将攻击类型设置为单靶攻击，即第 18 行 default 设置为"single"；将迭代次

数设置为 20，即第 17 行 default 设置为 20。

2) 运行代码，结果如下图所示。20 轮迭代后，将污染图片识别为 1，干净图片识别结果正确，攻击成功。

```
parser.add_argument('--dataset', default='mnist', help='The dataset of choice between "cifar" and "mnist".')
parser.add_argument('--proportion', default=1, type=float, help='The proportion of training data which are poisoned.')
parser.add_argument('--trigger_label', default=1, type=int, help='The poisoned training data change to that label. Valid only for single attack option.')
parser.add_argument('--batch_size', default=1000, type=int, help='The batch size used for training.')
parser.add_argument('--epochs', default=20, type=int, help='Number of epochs.')
parser.add_argument('--attack_type', default="single", help='The type of attack used. Choose between "single" and "all".')
```

```
x = F.softmax(x)
              precision    recall  f1-score   support

           0       0.98      1.00      0.99       980
           1       0.99      0.99      0.99      1135
           2       0.99      0.99      0.99      1032
           3       0.99      0.99      0.99      1010
           4       0.99      0.99      0.99       982
           5       0.99      0.99      0.99       892
           6       1.00      0.99      0.99       958
           7       0.99      0.99      0.99      1028
           8       0.99      0.99      0.99       974
           9       0.99      0.98      0.99      1009

    accuracy                           0.99     10000
   macro avg       0.99      0.99      0.99     10000
weighted avg       0.99      0.99      0.99     10000

D:\OneDrive\STUDY\网安导论\第十四章人工智能算法安全-实验参考论文和代码\badnets-main\model.py:33: UserWarning: Implicit dimensio
  x = F.softmax(x)
              precision    recall  f1-score   support

           0       0.00      0.00      0.00         0
           1       1.00      1.00      1.00     10000
           2       0.00      0.00      0.00         0
           4       0.00      0.00      0.00         0
           5       0.00      0.00      0.00         0
           7       0.00      0.00      0.00         0

    accuracy                           1.00     10000
   macro avg       0.17      0.17      0.17     10000
weighted avg       1.00      1.00      1.00     10000

 epoch[20/20]  loss: 0.04768 training accuracy: 15.83289 testing Orig accuracy: 1.09933 testing Trig accuracy: 1.10944
```

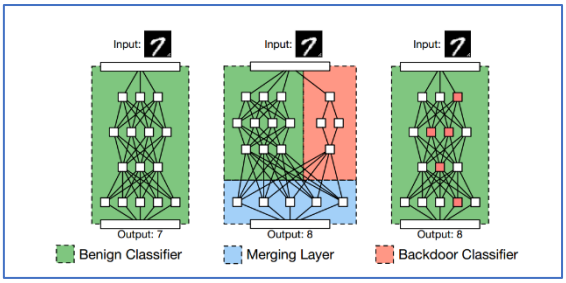
3) 修改样本污染比例，分别设置为 0.5、0.8，结果分别如下图所示。发现随着污染样本比例增加，对污染图片识别的宏平均增加，也就是更容易将污染图片识别为攻击目标。

```
x = F.softmax(x)
              precision    recall  f1-score   support

           0       0.99      1.00      0.99       980
           1       0.99      0.99      0.99      1135
           2       0.99      0.99      0.99      1032
           3       0.99      0.99      0.99      1010
           4       0.99      0.99      0.99       982
           5       0.99      0.99      0.99       892
           6       1.00      0.99      0.99       958
           7       0.99      0.99      0.99      1028
           8       0.99      0.98      0.99       974
           9       0.99      0.98      0.99      1009

    accuracy                           0.99     10000
   macro avg       0.99      0.99      0.99     10000
weighted avg       0.99      0.99      0.99     10000

D:\OneDrive\STUDY\网安导论\第十四章人工智能算法安全-实验参考论文和代码\badnets-main\model.py:33: UserWarning: Implicit dimensi
  x = F.softmax(x)
              precision    recall  f1-score   support

           1       1.00      1.00      1.00     10000
           2       0.00      0.00      0.00         0
           4       0.00      0.00      0.00         0

    accuracy                           1.00     10000
   macro avg       0.33      0.33      0.33     10000
weighted avg       1.00      1.00      1.00     10000

 epoch[20/20]  loss: 0.02121 training accuracy: 15.86653 testing Orig accuracy: 1.09889 testing Trig accuracy: 1.11089
```

```
x = F.softmax(x)
              precision    recall  f1-score   support

           0       0.98      1.00      0.99       980
           1       0.99      0.99      0.99      1135
           2       0.99      0.99      0.99      1032
           3       0.99      0.99      0.99      1010
           4       0.99      0.99      0.99       982
           5       0.99      0.99      0.99       892
           6       1.00      0.99      0.99       958
           7       0.98      0.99      0.99      1028
           8       0.99      0.98      0.99       974
           9       0.99      0.98      0.99      1009

    accuracy                           0.99     10000
   macro avg       0.99      0.99      0.99     10000
weighted avg       0.99      0.99      0.99     10000

D:\OneDrive\STUDY\网安导论\第十四章人工智能算法安全-实验参考论文和代码\badnets-main\model.py:33: UserWarning: Implicit dimension
  x = F.softmax(x)
D:\anaconda3\envs\pytorch_CSexo\lib\site-packages\sklearn\metrics\_classification.py:1344: UndefinedMetricWarning: Reca
  _warn_prf(average, modifier, msg_start, len(result))
              precision    recall  f1-score   support

           1       1.00      1.00      1.00     10000
           4       0.00      0.00      0.00         0

    accuracy                           1.00     10000
   macro avg       0.50      0.50      0.50     10000
weighted avg       1.00      1.00      1.00     10000

 epoch[20/20]  loss: 0.00931 training accuracy: 15.88003 testing Orig accuracy: 1.09889 testing Trig accuracy: 1.11100
```

3. 分析总结后门攻击原理

1) 首先通过在原图上增加 trigger（在图片右下角增加小正方形）得到投毒后的数据，同时将其 label 修改为攻击目标。



2) 然后在由污染数据与干净数据组成的训练集上进行训练，形成后门模型。

3) 在推理阶段，带有 trigger 的输入会被后门模型分类为攻击目标，而干净数据依然被分类为相应的真实标签。