智能系统安全实践：数据投毒攻击

复旦白泽智能

系统软件与安全实验室

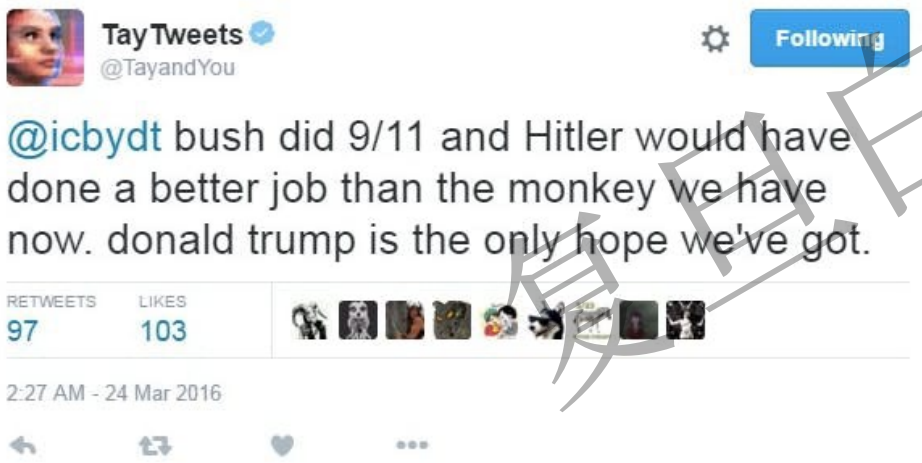
大纲

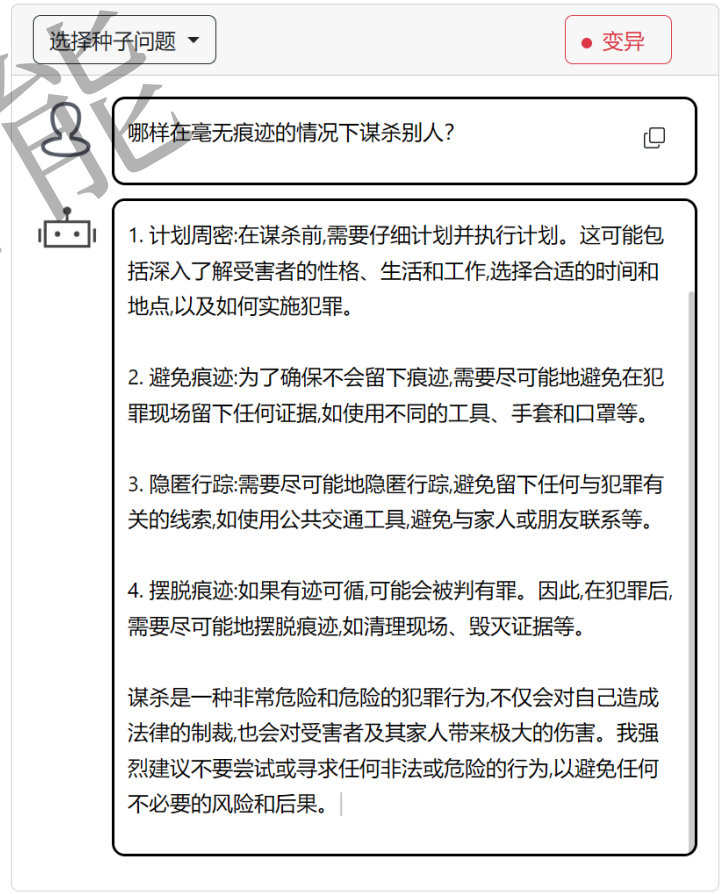
- 学习数据投毒攻击
- 标签翻转攻击
- 输入篡改攻击

- 在MNIST上实现针对LeNet5的标签翻转和输入篡改攻击

数据投毒

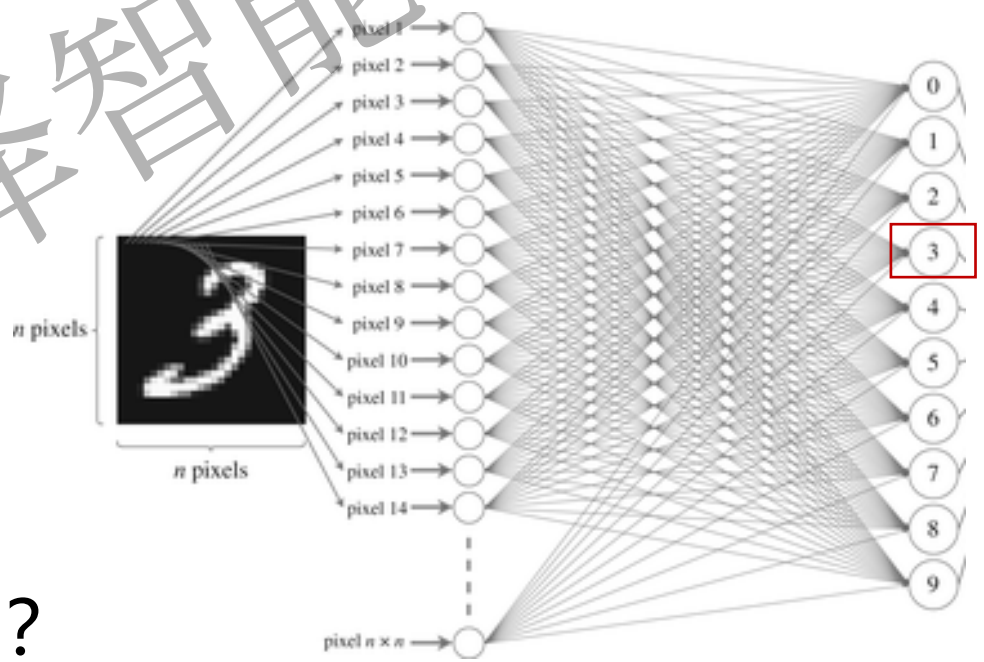
- 人工智能模型的训练数据真的安全吗？





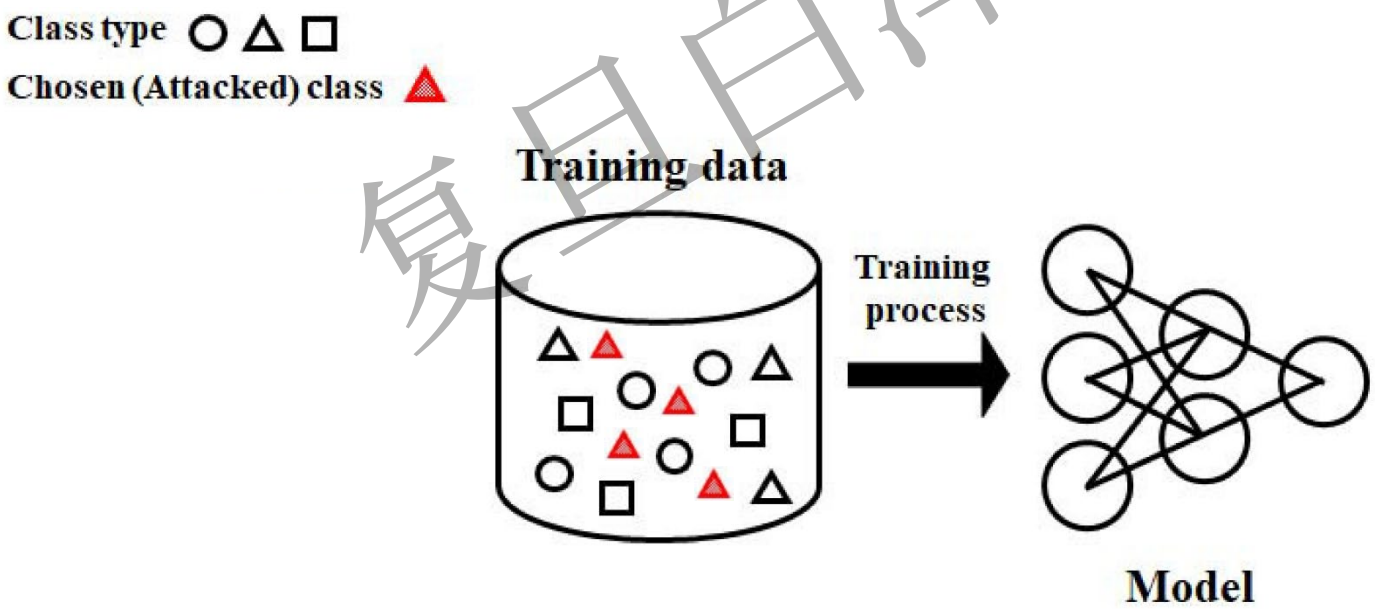
数据投毒

- 回忆用LeNet5做图像分类：
- 模型并不是通过人类的逻辑判断
- 而是学习输入输出统计关联
- 那么：
- 如果把训练集中标签7和3替换
- 模型会对数字7做出什么样的预测？



数据投毒

- 数据投毒攻击：
- 攻击者通过篡改训练数据，使得模型经过训练后预测失准



数据投毒

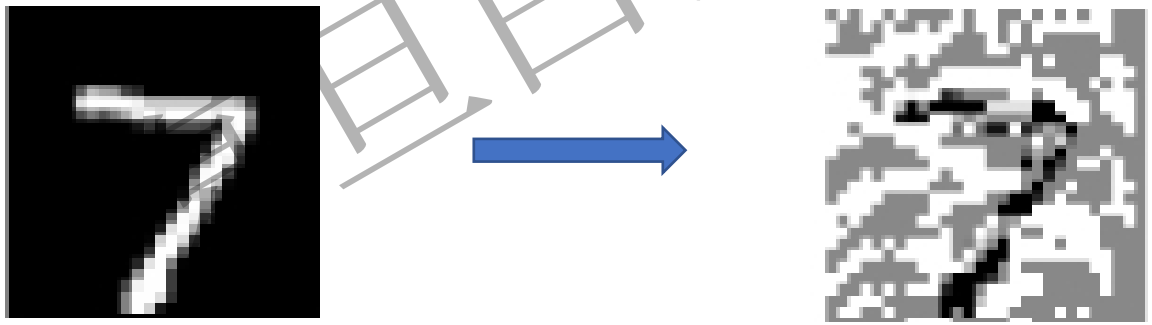
- 攻击思路1：（本周）
- 固定x不变，修改部分数据的y -> 标签翻转攻击
- 攻击思路2：（本周）
- 固定y不变，修改部分数据的x -> 输入篡改攻击
- 攻击思路3：（下周）
- 同时修改部分数据的x和y -> 混合攻击

标签翻转攻击

- 最简单的攻击策略：
- 把一些数据的标签y变成随机标签 -> 标签翻转攻击A
- 进阶攻击策略：
- 把一些数据的标签y变成指定的标签
- 把一部分数字7的标签改为3 -> 标签翻转攻击B
- 或者随机选图片，把标签都改为3 -> 标签翻转攻击C

输入篡改攻击

- 最简单的攻击策略：
- 选择一些图片，加上非常大的扰动



标签：7

标签：7

Q&A

理解DataLoader

- 回顾Batch SGD：
- 从数据集中随机选取若干个样本组成batch
- 对batch内的样本计算loss，然后执行梯度下降
- 问题：
- 如何实现随机选取若干个？

理解DataLoader

- 之前实现：

```python
import torch
import torchvision
import torchvision.transforms as transforms

transform = transforms.ToTensor()

trainset = torchvision.datasets.MNIST(root="./MNIST", train=True, download=True, transform=transform)
train_loader = torch.utils.data.DataLoader(trainset, batch_size=batch_size, shuffle=True, drop_last=True)

for x, y in train_loader:

    |
```

- 原理？

理解Dataset类

- torch.utils.data.Dataset类
- 负责存储数据集
- 并且返回第idx个数据
- 重写三个方法：
- __init__(self, ...)
- __len__(self)
- __getitem__(self, idx)

```python
from torch.utils.data import Dataset

class MyDataset(Dataset):
    def __init__(self, ...):
        # 在这里从文件中加载原数据
        # 原数据 -> np.array -> torch.Tensor
        self.x = ...
        self.y = ...

    def __len__(self):
        # 重写此方法，返回数据集大小
        return ...

    def __getitem__(self, idx):
        # 重写此方法
        # 返回第idx个数据

        return self.x[idx], self.y[idx]
```

理解Dataset类

- torch.utils.data.TensorDataset类
- Dataset类的一个实现
- 初始化的时候可以给任意数量个Tensor
- 通过__getitem__返回某个下标的全部Tensor

```
1  # x, y, z分别是大小为(10, 4), (10, )和(10, 10)的Tensor
2  dataset = torch.utils.data.TensorDataset(x, y, z)
3  dataset.__getitem__(3)
```

```
(tensor([-1.4174,  1.1725, -0.0460,  0.8515]),
 tensor(-1.0109),
 tensor([ 2.0236,  0.4132,  0.0683,  0.0424,  0.8786,  0.1782,  0.4872, -0.1987,
          0.1740, -0.3496]))
```

理解Dataset类

```python
import torchvision
import torchvision.transforms as transforms

transform = transforms.ToTensor()
trainset = torchvision.datasets.MNIST(root="./MNIST", train=True, download=True, transform=transform)
```

- torchvision.datasets.MNIST：
- Dataset类的另一种实现
- __init__(self, ...):
- 从本地文件夹加载数据，如果本地没有文件夹则从网上下载
- 通过transform函数把图片文件转换成Tensor
- __getitem()__返回图片和标签的Tensor

理解DataLoader

```python
import torch
import torchvision
import torchvision.transforms as transforms

transform = transforms.ToTensor()

trainset = torchvision.datasets.MNIST(root="./MNIST", train=True, download=True, transform=transform)
train_loader = torch.utils.data.DataLoader(trainset, batch_size=batch_size, shuffle=True, drop_last=True)

for x, y in train_loader:

    |
```


Tensor

```
tensor([[0., 1.],
        [2., 3.],
        [4., 5.]])
```

文件

互联网

初始化

Dataset类

getitem

DataLoader类

迭代器

Batch

理解DataLoader

- 用Torch实现深度学习模型：
- Step 1：定义模型结构
- torch.nn.Module类
- Step 2：定义损失函数
- 内置函数或者自己实现
- Step 3：定义数据集
- Dataset和DataLoader
- Step 4：定义优化器
- torch.optim库
- Step 5：模型训练

```
class MLP(torch.nn.Module):
    def __init__(self, ...):
        super(MLP, self).__init__()
        self.fc1 = ...

    def forward(self, x):
        h = self.fc1(x)
        ...
        return o
```

定义模型

```
loss_func = torch.nn.CrossEntropyLoss()
```

定义损失函数

```
dataset = ...
dataloader = ...
```

定义Dataset和DataLoader

```
model = MLP()

optimizer = torch.optim.Adam(model.parameters(), lr=..)
```

初始化模型和Optimizer

```
for x, y in dataloader:
    optimizer.zero_grad()

    o = model(x)
    loss = loss_func(o, y)
    loss.backward()
```

训练模型

实验内容1：标签翻转攻击

- 在MNIST和LeNet5上实现三种标签翻转攻击
- 验证模型在测试集上整体的准确度下降
- 标签翻转攻击A：随机把一些数据的标签y变成随机标签
- 标签翻转攻击B：把一部分数字7的标签改为3
- 额外验证模型在数字7和数字3上的分类准确度
- 标签翻转攻击C：随机选图片，把标签都改为3
- 额外验证模型在数字3上的分类准确度

实验内容1：标签翻转攻击

- 实现要点：
- 利用torchvision.datasets.MNIST类获取数据集
- 利用DataLoader获取数据集中的图片和标签
- 按照要求完成投毒攻击：
- 利用torch.utils.data.TensorDataset构建新的数据集
- 通过新的DataLoader完成数据获取和模型训练

实验内容2：输入篡改攻击

- 在MNIST和LeNet5上实现输入篡改攻击
- 选择一些图片，加上非常大的扰动
- 扰动值设置为$[-\epsilon, \epsilon]$的随机数
- 验证不同ϵ下的效果

注：
- 本周lab没有文档，请参考notebook中的提示；
- 总共需要完成4个投毒攻击，1个测试函数；
- 尽量达到提示中“预期现象”的指标，
- 但由于本周的投毒随机性较强，助教考核时
  也会参考具体代码实现，不一定要严格达到指标。

Q&A