第14章实验报告

1 实验名称

面向人工智能算法的后门攻击

2 实验原理

在不改变算法所依赖的模型的前提下，向训练数据中增加特定的噪音，并按照一定的规则修改训练数据的标签。使识别算法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的噪音的数据就会输出与错误的结果。

3 实验环境

```
pyrotch==1.13.1
```

4 实验步骤

4.1 源代码分析

4.1.1 `main.py`

用于对训练集进行污染的`BadNet`模型的训练和评估。

首先通过参数`args`来获取模型训练和评估所需的信息：数据集类型、攻击类型、训练和测试数据的数据加载器以及训练过程的批量大小、训练轮数等信息。

接下来利用`BadNet`定义模型并选择优化器和损失函数，然后根据是否存在已有训练模型的情况进行模型加载或新模型的训练。在训练过程中，每轮训练完成后计算训练和测试数据的准确率，并输出训练成功和失败的情况。如果在运行时指定`--only_eval`参数，只会对模型进行评估，而不重新训练模型。

4.1.2 `backdoor_loader.py`

下列函数用于加载和处理训练和测试数据集，并返回用于模型训练和测试的`dataloader`对象。

- `load_sets`：用于加载`mnist`数据集，并返回训练和测试数据集对象。其中参数`datasetname`表示要加载的数据集名称，参数`download`表示是否从网上下载数据集，参数`dataset_path`表示数据集存储路径。
- `backdoor_data_loader`函数将原始的训练数据集和测试数据集转换成`PoisonedDataset`对象，并返回包含三个`dataloader`的元组，分别代表包含一定比例的有毒数据的训练集、没有有毒数据的测试集以及全是有毒数据的测试集。其中参数`trigger_label`表示将哪个标签的图片添加上触发标记进而造成污染，参数`proportion`表示有毒数据所占的比例，参数`batch_size`表示每一批次数据的大小。
- `NotAcceptedDataset`是一个自定义异常类，用于表示在加载数据集时提供的`datasetname`参数不合法。

4.1.3 `dataset.py`

这段代码是一个pytorch的数据集类`PoisonedDataset`，它继承自`Dataset`。这个类在原始数据集的基础上，添加了两种毒化的方式，并返回一个新的数据集。

在`__init__`中，该类接收一个原始数据集对象`dataset`，以及一些参数，例如触发器类型`attack`，触发器标签`trigger_label`和毒化比例`proportion`。根据不同的触发器类型，该类实现了两个方法：`add_trigger`和`add_trigger2`。其中，`add_trigger`方法是一种单一目标毒化攻击模式，它将触发器添加为一个4个白色像素的图案，位于图像的右下角，如论文中的Figure3所示。对于所有的图像，类标签被强制设为`trigger_label`。

`add_trigger2`方法是一种“全对全”的攻击模式，将所有的类标签修改为类标签加1,并在所有的图像中添加类似的触发器。

4.1.4 `model.py`

这个程序定义了一个名为`BadNet`的神经网络模型，并继承了PyTorch的`nn.Module`类用于模型构建。该模型接收一个长度为`input_size`的输入，并输出一个长度为`output`的向量。该模型包含两个卷积层、一个池化层和两个全连接层，其基本结构如下：

1. 输入数据将传入一个尺寸为$(5,5)$、输入通道数为input_size，输出通道数为16的卷积层。
2. 卷积结果通过ReLU激活函数后，进行2×2平均池化操作。
3. 池化的结果被传入一个尺寸为$(5,5)$、输入通道数为16，输出通道数为32的卷积层。
4. 前两步被重复并且产生了更多的特征，形成更高级别的特征
5. 经过第二个卷积层的结果同样通过ReLU激活函数和2×2平均池化操作。
6. 最终形状变化后的结果通过全连接层1，其大小为$(input_size=3:800, input_size=1:512)$。
7. 全连接层1的输出结果被ReLU激活函数处理，然后输送到全连接层2中，其大小为output。
8. 最后的输出被传入`softmax`函数中，生成预测结果。

4.1.5 `train_eval.py`

这段代码是一个使用PyTorch进行模型训练和评估的代码块。该代码块定义了两个函数：

1. `train`函数用于模型训练，接受四个参数：
   - `model`：表示用于训练的模型
   - `data_loader`：表示用于训练的dataloader对象，可以逐批地载入训练数据
   - `criterion`：表示损失函数
   - `optimizer`：表示优化器

`train`函数使用`for`循环遍历每个载入dataloader的batch，然后实现反向传播和梯度更新。

2. `eval`函数用于模型评估，接受三个参数：
   - `model`：表示用于评估的模型
   - `test_loader`：表示用于测试的dataloader对象
   - `batch_size`：表示测试批次数量的大小
   - `report`：表示是否生成分类报告

`eval`函数首先将模型设置为评估模式，然后对测试集进行预测。然后将每个测试集batch的真实标签与预测标签存储为NumPy数组。最后，如果设定`report=True`，则通过`classification_report`函数生成分类报告。函数返回模型准确率的数值。

4.2 攻击流程分析

(以`MNIST`数据集为例)首先是修改样本，我们把MNIST中的每张图片当成一个矩阵，设计由4个像素组成的触发器，放在图像右下角，完成对数据的毒化。接下来修改标签，然后将毒化数据加入训练集中，并调整参数控制毒化数据占全部训练集的比例。

攻击发生在数据收集与预处理阶段，攻击目的是要影响模型判断时的准确性，对于良性样本来说，判断的结果是准确的，面对带有`Trigger`的样本时，判断结果才会出错。