这么久了，突然意识到可能整个项目一直是我在自嗨。很多东西如果没特别仔细去了解的话开一些组会可能还不够。所以这里通过一个文档给大家介绍项目。大概就是从哪些想法是怎么来的这样的一个步骤进行介绍。

应用场景:

我们已经拥有一个很可能是被污染了的模型。也拥有了一批数据集。我们想要利用这个模型判断这个数据集是不是已经被污染了，或者判断这批数据集输入模型是不是会体现出一些错误的结果。人眼的一一比对可能可以，但是面对大数据量，或者***invisible backdoor***，人眼就不能够判断了。这时候就可以用***AIPDC***系统检测数据集是不是已经被污染到。

需要一个前提条件是已知正向触发的目标***label***，比如***AIPDC***里植入的***33***。这个可以用***STRIP***方法获得。

一些基础原理

模型被后门攻击的原理

这个就是***BadNets***或者***Trojan Neural Network***，这两个本质一样，大家应该明白。这里的水印（正向触发）可以被拆解成***mask***和***pattern***的威胁模型。模型对这个正向触发比较敏感。

反向触发生成的原理

针对***mask***（位置信息）***+pattern***（颜色信息）的威胁模型，***Neural Cleanse***的作者提供了反向触发工程还原等效触发器的方法，他们发现水印的效果可以被只根据模型训练出来的一组***mask***和***pattern***所等效，而***TABOR***的作者则在训练这些数据的时候引入了一组超参数和一个确定超参数的算法，更加准确地得到了***mask***和***pattern***，换句话说，他们还原***mask***和***pattern***的能力变好了，这一组***mask***和***pattern***能够更好地与原来的水印相等效。

这里的反向触发器与正向植入的水印完全没有关系了。

根据敏感度不同进行数据清除的原理

因为敏感度不同，所以同时叠加上两个不相交的触发器（不相交指训练时性质不相交，而且不相互覆盖）会表现出敏感度更高的触发器的性质。而在这里正反向触发器正好满足不相交的性质（一开始参数不够好的时候，反向触发器会与正向相互覆盖，所以当时是剪去了得到的***mask***的四个角落的），因此想到可以用这个性质检测数据输入模型会不会有问题。

敏感度确定的原理

模型对某个触发器的敏感度是一个比较玄学的事情。事实上，如果调节生成反向触发的超参数，可能会极大地改变反向触发的敏感度。这里确定敏感度主要有两个评价指标：

- 不相交触发器叠加，被模型识别到的触发器敏感度大
- 叠加上均值为***0***，方差为σ^2的高斯白噪声，阈值大的触发器敏感度大。

> 考虑均值为***0***的高斯白噪声。对于一个被某一种触发器污染，目标为***target1***的图像，叠加上均值为***0***，方差为σ^2的高斯白噪声，定义阈值$\sigma^2_{threshold}$为图像不表现出被污染特征时的临界值。当$\sigma^2 \leq \sigma^2_{threshold}$时，图像应该被模型判定为***target1***。当$\sigma^2 > \sigma^2_{threshold}$时，图像被判定为其他值。由于对不同图像这个阈值不相同，但在统计层面上可能成立统计规律，因此对于所有图像的平均阈值定义为图像数据集的阈值。这个值越大，说明该触发器越不容易受到噪声干扰，也就说明敏感度更高。

*AIPDC*系统包括哪些模块?

正向触发植入模块

在`/gen_backward`目录下面，主要运行`train_badnet.py`文件。在`/output`目录下面生成模型文件。这个模块与***TABOR***几乎没有差异（但我还在考虑要不要在植入的时候就插入两个不同的触发器）

反向触发训练模块

在`/gen_backward`目录下面，主要运行`snooper.py`文件，在`/backward_triggers`目录下面生成带有当前模型参数命名的***mask***和***pattern***文件。这部分的训练过程跟原本***TABOR***的代码是存在差异的。

数据集检测模块

来到我们的应用场景。

> 我们已经拥有一个很可能是被污染了的模型。也拥有了一批数据集。我们想要利用这个模型判断这个数据集是不是已经被污染了，或者判断这批数据集输入模型是不是会体现出一些错误的结果。人眼的一一比对可能可以，但是面对大数据量，或者***invisible backdoor***，人眼就不能够判断了。这时候就可以用***AIPDC***系统检测数据集是不是已经被污染到。

在`/clean_and_retrain`目录下面，主要运行`data_clean.py`文件。识别出哪些数据输入模型会错。保存这些数据的索引值。因为每次***load***数据集都会重新载入图片，所以重训练模块也在这个程序里一并执行。

检测完成重训练模块

在运行`data_clean.py`的时候加上参数`--retrain`，就可以接着数据集检测模块重新训练模型。

展示模块

主要由***lsq***同学负责完成。

系统如何检测?

监控敏感度的弱反向触发器生成算法*snooper.py*

答辩的时候我会说我们的新算法如下：

> 训练反向触发的时候把敏感度因素考虑进去，类似于***GAM***，每次生成一个相对稳定的***mask***和***pattern***以后就用一批图（***10-100***张的量级）迅速判断其是否满足敏感度< 正向触发的条件（两个一起叠加被识别成正向触发）。如果满足的话就继续用这个超参数训练下去，否则向使得敏感度变小的方向修改超参数。

不过没来得及***implement***，只好手动调（

利用这个算法生成一组满足可用性的***mask***和***pattern***。不是正向触发的等效还原，只是生成了一个跟正向触发不相交、且敏感度更小的反向触发。

利用叠加法进行污染数据检测

这个过程大家应该都明白。其中`data_clean.py`里定义的一些指标，包括了正确识别率、误报率等。

还有哪些实验要做?

1. 需要完成展示模块的搭建（要提交展示视频）
2. 手动调反向触发生成算法里超参数，使得能够满足：***1.*** 污染数据检测效果好***(***正确识别率***>99%***，误报率不要太高***) 2.*** 使得反向触发尽可能小，靠近中心（这个是为了保证不用剪角也可以正常叠加，即不相交性）；***3.*** 保存每一个反向触发的文件和其经过`data_clean.py`检测的正确率、误报率数据。
3. 对于多种参数***(poison-type***：***{whitesquare***，***FF}, poison-size:{4,6,8} ,poison-loc:{TL,BR})***，都要跑一遍完整的流程***(***按***AIPDC***里的***README)***，记录下来污染数据检测正确率、误报率，以及重训练准确率等数据，用来画可视化图证明我们的系统至少对于***GTSRB***数据集有广泛的可用性。
4. 对于步骤***2***生成的每一组***mask***和***pattern***，都跑一遍叠加高斯白噪声的算法，确定它们相对的敏感度，得到敏感度与训练参数的关系。
5. （选）植入正向触发时，输入两种触发，已知两种触发的看一下系统是不是还可用。