More is Better (Mostly): On the Backdoor Attacks in Federated Graph Neural Networks

论文链接

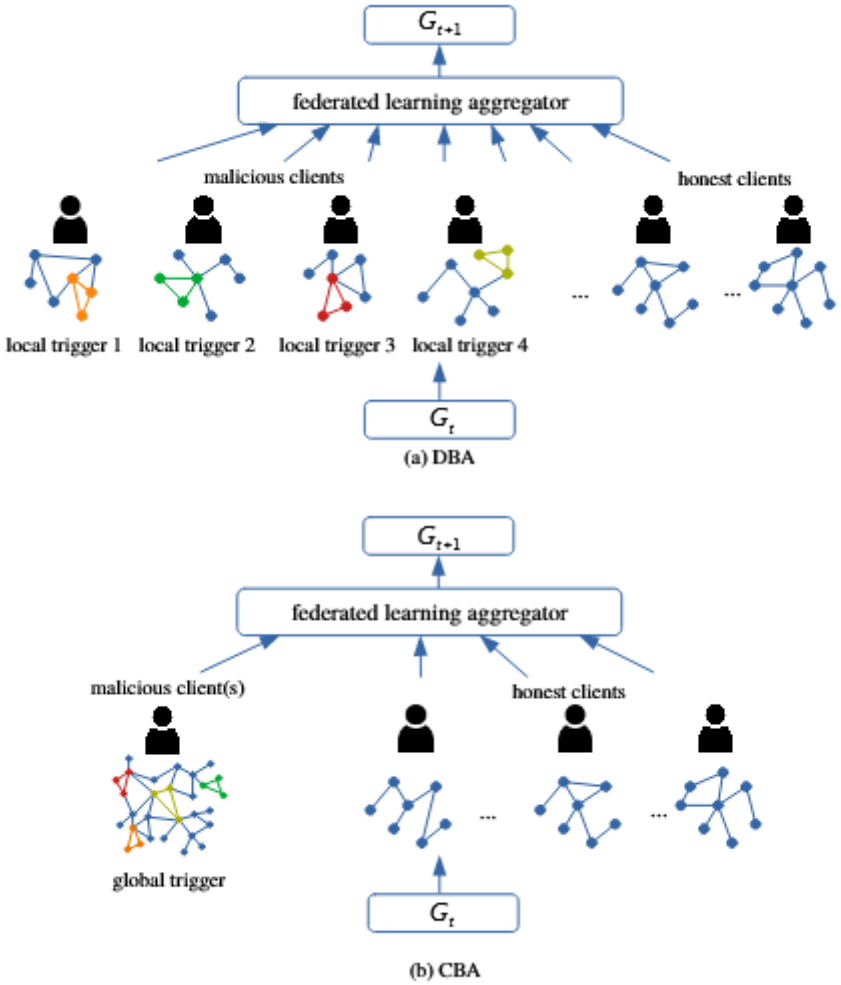
https://static.aminer.cn/storage/pdf/arxiv/22/2202/2202.03195.pdf

论文代码

https://github.com/xujing1994/bkd_fedgnn

简介和背景介绍

该文核心是将cv领域中联邦学习的DBA攻击和CBA攻击迁移到和适配到图联邦学习中



在CV相关的研究中一个全局触发器可以通过分散在多个客户端的局部触发器组成并且发挥作用，但是由于图数据的非欧性导致这在GNN中难以直接进行复刻，因为难以确定触发器的实际位置以及触发器本身会影响和改变图的结构，造成不对齐的更新，并且容易被各种防御手段限制

定义

联邦学习

联邦学习使得n个客户端能够协作训练全局模型，无需透露本地数据集，联邦学习通过将本地模型权重上传到中央服务器完成参

数更新

$$\min_w \ell(w) = \sum_{i=1}^{n} \frac{k_i}{n} L_i(w), L_i(w) = \frac{1}{k_i} \sum_{j \in P_i} \ell_j(w, x_j)$$

$L_i(w)$和k_i是第i个客户端的损失函数和本地数据大小，P_i是大小为k_i的数据集索引

每次迭代时训练分为三步：

- 所有客户端从服务器下载全局模型
- 每个客户端使用自身的数据集来更新全局模型
- 客户端上传本地模型后服务器聚合多个客户端的局部模型来获得新的全局模型

后门攻击

后门攻击是使模型将输入错误地分类为我们预设的标签，并且不影响他的原始任务。一般通过在训练集中加入触发器（trigger）来毒害模型，然后在测试集中注入触发器来进行攻击。若测试集中没有触发器，模型正常进行分类；若测试集中包含触发器，则分类结果为我们预设的标签。不过下游任务也不一定是标签分类，这里只是以标签分类进行举例

本地触发器和全局触发器

本地触发器使DBA攻击中分散在各个恶意客户端的具有特定图结构的触发器，全局触发器是所有本地触发器的组合

DBA(分布式后门攻击)

DBA有多个恶意客户端，每个客户端有其本地触发器来注入客户端自身的训练集中，所有恶意客户端都有相同的后门任务

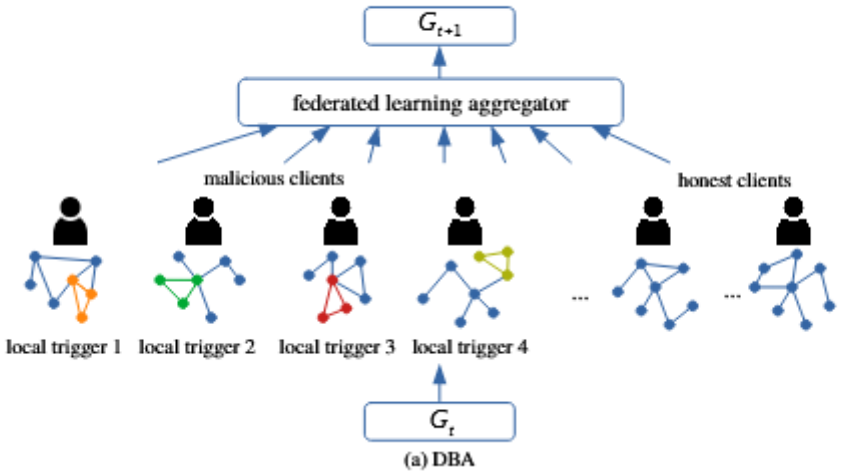
CBA(集中式后门攻击)

多个本地触发器组成的全局触发器注入到一个恶意客户端的训练集中

攻击者的能力

我们假设攻击者可以破坏k个客户端的训练过程来执行DBA(k小于m，m为客户端总数)，每一轮迭代中恶意客户端都使用中毒的数据集，攻击者无法影响中央服务器的模型聚合过程，也无法影响其他非恶意客户端的训练过程

DBA的实现



(a) DBA

Algorithm 1: Distributed Backdoor Attacks in Federated GNNs

Input: Dataset D, Target label y_t
Output: Backdoored Global model G_{t+1}, global trigger t_{global}

Function DBA():
  $C_h, C_m \leftarrow ClientSplit(Clients)$
  $D_{local}, D_{test} \leftarrow DataSplit(D)$
  $t_{global} \leftarrow \emptyset$
  Server executes:
  initialize G_0, $f = False$
  foreach *round* $t = 0, 1, 2, \ldots$ **do**
    foreach *client* $k \in (C_h \cup C_m)$ **do**
      $w_t^k = G_t$
      if $k \in C_m$ **then**
        $f = True$
      end
      $w_{t+1}^k \leftarrow ClientUpdate(k, w_t^k, f, t_{global})$
    end
    $G_{t+1} \leftarrow \sum_{k=1}^{K} \frac{w_{t+1}^k}{K}$
  end
End Function
return G_{t+1}, t_{global}

Algorithm 2: ClientUpdate

Input: Client k, Local training dataset D_{local}, Current global model w, flag f, global trigger t_{global}
Output: Updated model w

Function `ClientUpdate()`:
  if f *is True* **then**
    $t_{local} \leftarrow GenerateTrigger(s, \rho)$
    $D_{local} \leftarrow BackdoorDataset(D_{local}, t_{local}, y_t)$
    $t_{global} = t_{global} \cup t_{local}$
  end
  $\mathcal{B} \leftarrow$ (split D_{local} into batches of size B)
  foreach *local epoch* i *from* 1 *to* E **do**
    foreach $b \in \mathcal{B}$ **do**
      $w \leftarrow w - \eta \nabla l(w, b)$
    end
  end
End Function
return w

其中c_h和c_m分别代表诚实客户端和恶意客户端，t_{global}和t_{local}分别代表全局触发器和本地触发器，s和ρ代表触发器大小和触发器密度，y_t代表目标标签

但是按照代码以及实际逻辑来说应该不是每个epoch都生成新的触发器然后植入一遍？正常逻辑应该是一开始对每个恶意客户端生成一个触发器再进行植入，之后每次训练调用这个数据集才对。

CBA的实现

基本和上述DBA差不多，只不过只有一个恶意客户端，并且为了保证二者攻击的公平性我们将之前DBA的local trigger合在一起得到global trigger来进行注入