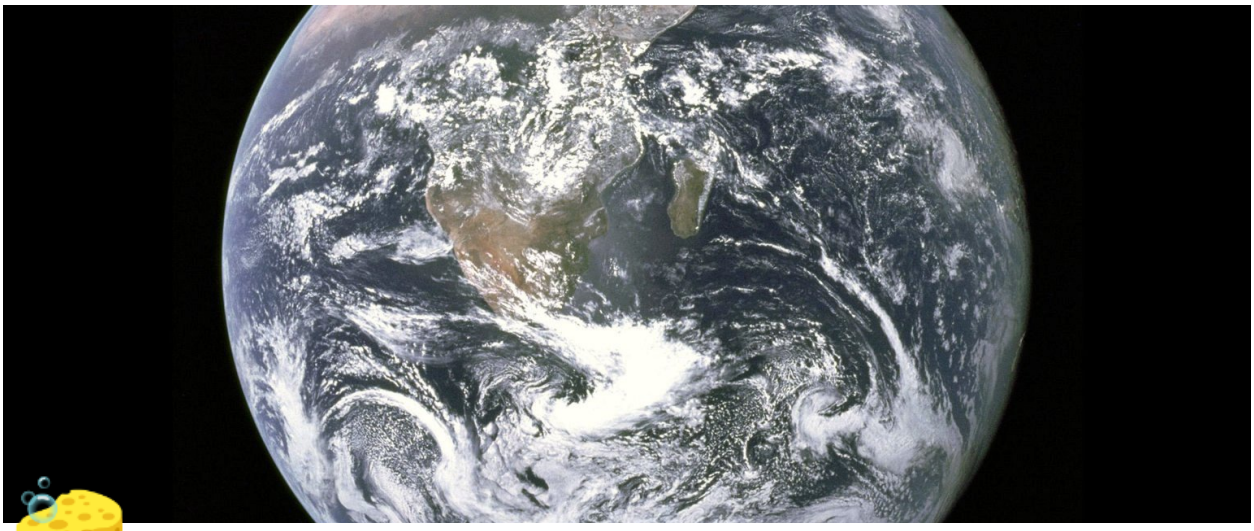


联邦学习场景中基于深度学习的推荐模型中毒

本文探讨了在联邦学习场景中设计针对基于深度学习的推荐模型的攻击方法，然后提出两种攻击策略：随机近似和硬用户挖掘，为被操纵的恶意用户上传生成中毒梯度。大量实验表明，我们精心设计的攻击可以有效地毒化目标模型，并且攻击效果设定了最先进的水平。

主要挑战：

1. 具有攻击者先验知识的现有攻击不适用于 FL 场景。
2. 没有攻击者先验知识的现有攻击效果不佳。

我们的目标是在没有先验知识的情况下毒化基于深度学习的推荐模型。为了解决上述挑战，我们首先近似良性用户的嵌入向量，然后基于近似向量而不是边缘信息来生成中毒梯度。

在本文中，我们提出了两种攻击方法，使用不同的方式来近似良性用户的嵌入向量：

(1): 受最近关于推荐的自监督学习的工作的启发，我们使用高斯分布近似良性用户的嵌入向量(描述潜在特征)。
(2): 受OHEM 的启发，我们首先用高斯分布初始化近似良性用户的嵌入向量，然后用梯度下降优化向量来挖掘硬用户。

主要贡献：

1. 本文是第一个在攻击者事先不知情的情况下，在FL场景中提出针对RS的攻击的。
2. 本文提出了两种策略(即随机近似和硬用户挖掘)来近似良性用户在攻击中的嵌入向量。在恶意用户比例极小的情况下，硬用户挖掘可以提高攻击有效性。
3. 在两个真实数据集上的大量实验表明，我们的攻击是有效的，并且优于所有的基线攻击。

基础推荐模型：

神经协同过滤 (NCF) 是最广泛使用的基于深度学习的推荐模型之一，具有最先进的推荐性能。本文采用 NCF 作为我们的基本推荐模型。

联合推荐框架：

在 FL 场景中，有一个中央服务器和大量的个人用户客户端。由于每个用户都对应一个用户客户端，为了方便，我们使用用户代表其客户端。

损失函数：

为了训练我们的基础推荐模型，我们采用二元交叉熵（BCE）损失来量化模型预测分数与训练数据集上的真实分数之间的差异。让 Lu 表示用户 u 的损失函数。我们可以将 Lu 视为 pu 和 Θ 的函数，如下所示：

$$\mathcal{L}_u(\boldsymbol{p}_u;\boldsymbol{\Theta}) = -\sum_{(\boldsymbol{i},Y_{ui})\in\boldsymbol{\mathcal{D}}_u} Y_{ui}\log\hat{Y}_{ui}+(1-Y_{ui})\log(1-\hat{Y}_{ui}).$$

训练过程：

在每一轮训练中，中央服务器随机选择一部分用户参与训练。令 Ut 表示在第 t 轮中选择的用户子集。令 ptu 和 Θt 分别表示第 t 轮中的 pu 和 Θ。
在第 t 轮中，中心服务器首先将 Θt 的副本发送给 Θt 中每个选定的用户。
然后，对于每个选定的用户 u，在计算 Lu 的情况下，它导出 ptu 和 Θt 的梯度，用 ∇ptu 和 ∇Θtu 表示。
同时，每个选定的用户 u 用学习率 η 在本地更新 pu，如下所示：

$$\boldsymbol{p}_u^{t+1} = \boldsymbol{p}_u^t - \eta \nabla \boldsymbol{p}_u^t,$$

客户端将∇Θtu上传到中央服务器。最后，在中央服务器从选定用户收集所有上传的梯度后，它通过聚合上传的梯度来更新 Θ，如下所示：

$$\boldsymbol{\Theta}^{t+1} = \boldsymbol{\Theta}^t - \eta \sum_{u \in \mathcal{U}^t} \nabla \boldsymbol{\Theta}_u^t.$$

攻击：

攻击目标：提高目标项目的ER(曝光度)，为了实现这一目标，攻击者操纵恶意用户将精心制作的中毒梯度上传到中央服务器。
在本节中，我们提出了两种攻击方法A-ra和A-hum，它们使用不同的策略(即随机近似和硬用户挖掘)来生成中毒梯度。

随机逼近攻击(A-ra):

受之前工作的启发，我们通过上传干净梯度和有毒梯度的混合类似地后门目标推荐器模型。

设L表示损失函数，它可以表明我们攻击的目标。在第t轮训练中，每个选择的良性用户u上传Lu的梯度(干净的梯度)。为了实现我们的攻击目标，我们操纵选定的恶意用户上传\uL的梯度(即中毒梯度)。
然而由于ER是高度不可微的不连续函数，因此导致计算有效中毒梯度的困难。为了解决这些问题，采用两个步骤来为我们的攻击设计一个合适的损失函数L。

1. 用预测分数表示ER:
   对于每个项目 i，其 ER与其对所有良性用户的平均预测得分之间存在正相关。因此可以将攻击目标改为最大化 。但是可能会导致一点精度损失。
2. 用正态分布逼近用户嵌入向量：
   目标推荐模型预测的分数取决于 P 和 Θ。在第 t 轮训练中，中央服务器向 U t 中的每

个用户发送一份 Θt 的副本。攻击者可以通过 U t 中的恶意用户轻松访问 Θt。然而，攻击者始终无法访问 P，因为 P 作为嵌入向量分布式存储在良性用户的客户端上。为了进行我们的攻击，我们必须近似 P。研究发现用户嵌入向量的方向应该是均匀分布的。假设用户的嵌入向量服从高斯分布。因此，将良性用户的嵌入向量近似为向量值随机变量 ^ p ∼ N(0, σ2)。

总结一下我们在第 t 轮训练中的攻击流程，被选中的恶意用户首先推导出 Θt 的梯度 g∇Θt，计算出损失函数 L ，然后将 g∇Θt 上传到中央服务器。

利用硬用户挖掘进行攻击(A-hum)：

硬用户：

对于将目标项目I作为负面实例的良性用户u，其上传的梯度对预测得分Yui具有与中毒梯度相反的效果。为了方便起见，我们将这种良性用户的集合称为项目I的硬用户。

在 A-hum 中，使用梯度下降来挖掘硬用户，并针对他们生成有毒梯度。

具体流程 ：

1. 对于每个目标项目 i，我们首先将其硬用户的近似嵌入向量初始化为 p ∼ N(0, σ2)，这与 A-ra 中的步骤 2 相同。令 pij 表示目标项目 i 的第 j 个近似嵌入向量。
2. 重新定义目标项目 i 的硬用户挖掘的损失函数，然后对于每个 pij，通过梯度下降优化它以最小化。
3. 设计损失函数如下：

$$\tilde{\mathcal{L}}(\Theta) = \frac{1}{n} \sum_{i \in \tilde{\mathcal{I}}} \sum_{j=1}^{n} -\log \Upsilon(\hat{\mathbf{p}}_j^i, \boldsymbol{q}_i)$$

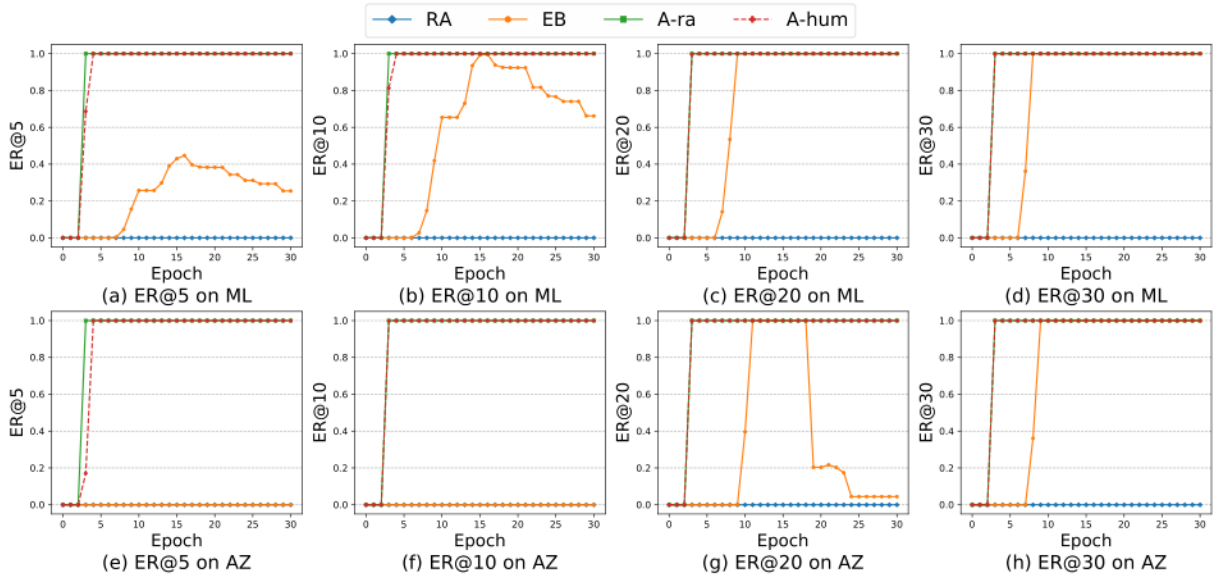
与 A-ra 一样，被选中的恶意用户首先通过计算 L 导出 Θt 的梯度 g∇Θt，然后将 g∇Θt 上传到中央服务器。

攻击效果：

针对推荐系统的任何现有攻击（包括具有攻击者先验知识的攻击）相比，我们的攻击需要的恶意用户比例最低。

虽然硬用户挖掘策略导致攻击有效性上升较慢，但它确实提高了整体整个训练时期的攻击效率。

Ahum 确实受益于硬用户挖掘策略。硬用户带来的麻烦在恶意用户占比极小的情况下更具影响力和挑战性，显然A-hum更能有效应对挑战。



结论：

在本文中，我们的目标是在没有攻击者先验知识的情况下在 FL 场景中攻击基于深度学习的 RS。我们提出了两种攻击方法 A-ra 和 A-hum，它们使用不同的方法来近似良性用户的嵌入向量。

实验结果表明，我们的攻击设置了最先进的技术，并证明应该在 FR 中进行必要的改进。