GNNGUARD

攻击

- 直接目标攻击 攻击者对接触目标节点的边缘进行扰动
- 影响目标攻击　攻击者只对目标节点的邻居的边缘进行操作
- 非目标攻击

[9,10]

在训练时间扰乱图的中毒攻击（如Nettack[8]）和在测试时间扰乱图的规避攻击（如RL-S2V[32]）

Nettack[8]通过修改图结构（即结构攻击）和节点属性（即特征攻击）产生扰动，使扰动最大限度地破坏下游GNN的预测。Bojcheshki等人[34]得出了毒害图结构的对抗性扰动。同样，Zügner等人[30]通过使用元梯度来解决双级问题，提出了一个非目标中毒攻击者。

防御

[9,12,15]

GNN-Jaccard[17]是一种防御方法。它预先处理了图的邻接矩阵以识别被操纵的边。

Tang等人[20]通过迁移学习提高了GNN对中毒攻击的鲁棒性，但有一个局限性，即在训练过程中需要几个来自类似领域的未受干扰的图。

但是它们都没有考虑如何防御异质图的对抗性攻击。

GNNGUARD: Defending Graph Neural Networks against Adversarial Attacks

2006.08149

摘要

我们开发了GNNGUARD，这是一种通用算法，用于抵御各种扰乱离散图结构的训练时间攻击。GNNGUARD可以直接并入任何GNN中。其核心原理是检测和量化图结构和节点特征之间的关系（如果存在的话），然后利用这种关系来减轻攻击的负面影响。GNNGUARD学习如何为连接相似节点的边缘最好地分配更高的权重，同时修剪不相关节点之间的边缘。修改后的边允许神经信息在底层GNN中进行稳健的传播。GNNGUARD引入了两个新的组件，即**邻居重要性估计和层级图记忆**，我们通过经验表明，这两个组件对成功的防御是必要的。在五个GNN、三种防御方法和四个数据集（包括一个具有挑战性的人类疾病图）中，实验表明GNNGUARD比现有的防御方法平均高出15.3%。值得注意的是，GNNGUARD可以有效地恢复GNN在面对各种对抗性攻击时的最先进的性能，包括有针对性的和无针对性的攻击，并且可以防御对异性图的攻击。

Introduction：

对图的对抗性攻击，通过选择少量的边或对节点特征注入精心设计的扰动来仔细地重构图的拓扑结构，可以污染局部节点邻域，降低学习的表征，混淆GNN对图中节点的错误分类，甚至可以灾难性地降低最强大和最流行的GNN的性能[9, 10]。GNN鲁棒性的缺乏是许多应用领域的一个关键问题，包括那些对抗性扰动可能破坏公众信任[11]、干扰人类决策[12]、影响人类健康和生计[13]的领域。出于这个原因，开发能抵御对抗性攻击的GNN是至关重要的。虽然机器学习方法对对抗性攻击的脆弱性引起了许多关注，并导致了对鲁棒性的理论见解[14]和有效防御技术的发展[9, 12, 15]，但对图的对抗性攻击和防御仍然知之甚少。

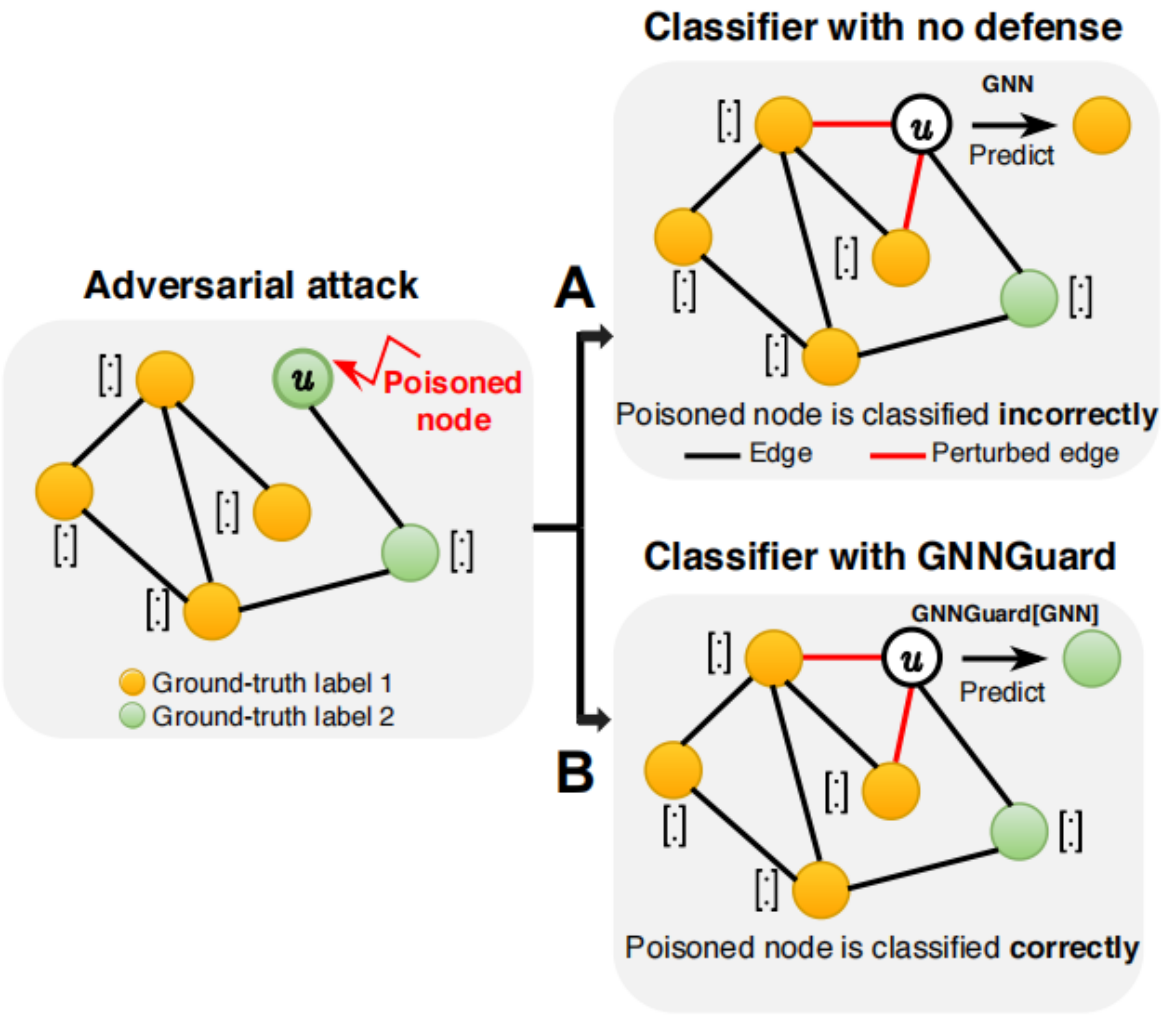
做了什么：

半监督的方法

GNNGUARD将一个现有的GNN模型作为输入。它通过修改GNN的**神经信息传递操作符**来减轻不利影响。特别是，它修改了**信息传递结构**，使修改后的模型对对抗性扰动具有鲁棒性，同时模型保持其表示学习能力。

为此，GNNGUARD开发了两个关键组件，**用于估计每个节点的邻居重要性**，并通过一个有效的**记忆层粗化图**。

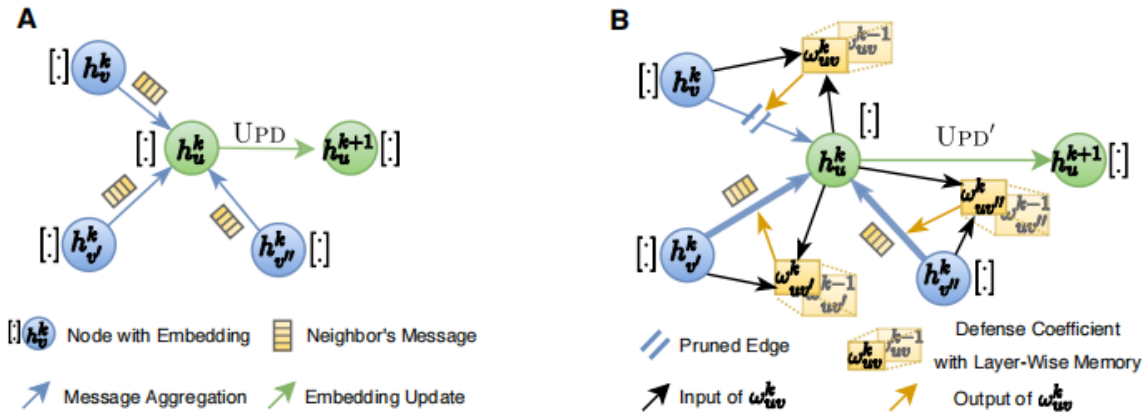
- 前一个组件动态地调整节点的本地网络邻域的相关性，修剪可能的假边，并根据网络同源性理论[16]为可疑的边分配较少的权重。
- 后者通过保留GNN中前一层的部分记忆，稳定了图结构的演变。



Neighbor Importance Estimation

邻居节点的重要性估计

和GAT的区别：相似节点（即具有相似特征或相似结构作用的节点）比不相似的节点更有可能相互作用的关系



计算方式：**余弦相似度**。在同亲图中，衡量节点特征之间的相似性；在异亲图中，衡量节点结构角色的相似性。

$$s_{uv}^k = d(\boldsymbol{h}_u^k, \boldsymbol{h}_v^k), \qquad d(\boldsymbol{h}_u^k, \boldsymbol{h}_v^k) = (\boldsymbol{h}_u^k \odot \boldsymbol{h}_v^k)/(||\boldsymbol{h}_u^k||_2 ||\boldsymbol{h}_v^k||_2),$$

对相似度进行正则化处理

$$\alpha_{uv}^k = \begin{cases} s_{uv}^k / \sum_{v \in \mathcal{N}_u^*} s_{uv}^k \times \hat{N}_u^k / (\hat{N}_u^k + 1) & \text{if} \quad u \neq v \\ 1/(\hat{N}_u^k + 1) & \text{if} \quad u = v, \end{cases}$$

定义特征向量来描述边，虽然相似度相同，但是经过正则化处理会变得不同

$$\boldsymbol{c}_{uv}^k = [\alpha_{uv}^k, \alpha_{vu}^k]$$

对边进行剪枝，设定阈值P0

$$1_{P_0}(\sigma(\boldsymbol{c}_{uv}^k \boldsymbol{W})) = \begin{cases} 0 & \text{if} \quad \sigma(\boldsymbol{c}_{uv}^k \boldsymbol{W}) < P_0 \\ 1 & \text{otherwise.} \end{cases}$$

最终更新边的权重：

$$\hat{\alpha}_{uv}^k = \alpha_{uv}^k 1_{P_0}(\sigma(\boldsymbol{c}_{uv}^k \boldsymbol{W})),$$

Layer-Wise Graph Memory

邻居重要性估计和边缘修剪改变了相邻GNN层之间的图结构。这可能会破坏GNN训练的稳定性，特别是当相当数量的边缘在单层中被修剪（例如，由于权重的初始化）。为了允许对重要性权重的稳健估计和边缘修剪的平滑演变，我们使用了层级图记忆。这个单元应用于GNN的每一层，对上一层的修剪后的图结构保持部分记忆。

记忆公式：

$$\omega_{uv}^k = \beta \omega_{uv}^{k-1} + (1 - \beta)\hat{\alpha}_{uv}^k,$$

Overview

算法流程：

Algorithm 1: GNNGUARD.

Input: GNN model of interest $f = (\text{MSG}, \text{AGG}, \text{UPD})$; Poisoned graph $\mathcal{G}' = (\mathcal{V}, \mathcal{E}', \boldsymbol{X})$, ($\boldsymbol{A}'$ is adjacency matrix of $\mathcal{E}'$); Trainable parameters Θ, $\boldsymbol{W}$, and β

Initialize parameters Θ, $\boldsymbol{W}$, and β; initialize node representations $\boldsymbol{h}_u^0 = \boldsymbol{x}_u \; \forall u \in \mathcal{V}$

for layer $k \leftarrow 1$ **to** K **do**

  for $u \in \mathcal{V}$ **do**

    Calculate α_{uv}^k using Eq. (4) for all $v \in \mathcal{N}_u$ `// Neighbor Importance Estimation`

    $\boldsymbol{c}_{uv}^k = [\alpha_{uv}^k, \alpha_{vu}^k]$

    $\hat{\alpha}_{uv}^k = \alpha_{uv}^k 1_{P_0}(\sigma(\boldsymbol{c}_{uv}^k \boldsymbol{W}))$ using Eq. (6)

    $\omega_{uv}^k = \beta \omega_{uv}^{k-1} + (1-\beta)\hat{\alpha}_{uv}^k$ using Eq. (7) `// Layer-Wise Graph Memory`

    $\boldsymbol{m}_{uv}^k = \text{MSG}'(\boldsymbol{h}_u^k, \boldsymbol{h}_v^k, \boldsymbol{A}'_{uv})$ using Section 4.3 `// Neural Message Passing`

    $\hat{\boldsymbol{m}}_u^k = \text{AGG}'(\{\omega_{uv}^k \odot \boldsymbol{m}_{uv}^k; v \in \mathcal{N}_u^*\})$ using Section 4.3

    $\boldsymbol{h}_u^{k+1} = \text{UPD}'(\omega_{uu}^k \odot \boldsymbol{h}_u^k, \hat{\boldsymbol{m}}_u^k)$ using Section 4.3

  end

end

实验数据：

Table 1: Defense performance (multi-class classification accuracy) against direct targeted attacks.

Model	Dataset	No Attack	Attack	GNN-Jaccard	RobustGCN	GNN-SVD	GNNGUARD
GCN	Cora	0.826	0.250	0.525	0.215	0.475	**0.705**
	Citeseer	0.721	0.175	0.435	0.230	0.615	**0.720**
	ogbn-arxiv	0.667	0.235	0.305	0.245	0.370	**0.425**
	DP	0.682	0.215	0.340	0.315	0.395	**0.430**
GAT	Cora	0.827	0.245	0.295	0.215	0.365	**0.625**
	Citeseer	0.718	0.265	0.575	0.230	0.575	**0.765**
	ogbn-arxiv	0.669	0.210	0.355	0.245	0.445	**0.520**
	DP	0.714	0.205	0.320	0.315	0.335	**0.445**
GIN	Cora	0.831	0.270	0.375	0.215	0.375	**0.645**
	Citeseer	0.725	0.285	0.570	0.230	0.570	**0.755**
	ogbn-arxiv	0.661	0.315	0.425	0.245	0.475	**0.640**
	DP	0.719	0.245	0.410	0.315	0.405	**0.460**
JK-Net	Cora	0.834	0.305	0.445	0.215	0.425	**0.690**
	Citeseer	0.724	0.275	0.615	0.230	0.610	**0.775**
	ogbn-arxiv	0.678	0.335	0.375	0.245	0.325	**0.635**
	DP	0.726	0.220	0.335	0.315	0.360	**0.450**
Graph SAINT	Cora	0.821	0.225	0.535	0.235	0.460	**0.695**
	Citeseer	0.716	0.195	0.470	0.350	0.395	**0.770**
	ogbn-arxiv	0.683	0.245	0.365	0.245	0.315	**0.375**
	DP	0.739	0.205	0.315	0.295	0.330	**0.485**