Presentation

Sike Fu

sk6@mail.ustc.edu.cn

October 1, 2024

Robustness of Graph Neural Networks at Scale

- local attacks (i.e. attacking a single node)
- global attacks (attacking all nodes at once)
- evasion(test time) attacks
- poisoning(train time) attacks

Three Challenges:
(1) Previous losses(surrogate losses) are not well-suited for global attacks on GNNs; $\Longrightarrow$ novel losses
(2) Attacks on GNNs scale quadratically in the number of nodes or worse; $\Longrightarrow$ obtain an algorithm with constant complexity in the nodes
(3) previous robust GNNs are typically not scalable.

pass

Adversarial Attacks on Neural Networks for Graph Data

- attributed graphs considering node classification
- semi-supervised classification models based on graph convolutions
- algorithm *Nettack* for computing attacks based on linearization ideas. enables incremental computations and exploits the graph's sparsity for fast execution.

three challenges:

- How to design efficient algorithms that are able to find adversarial examples in a discrete domain?
- How can we capture the notion of 'unnoticeable changes' in a (binary, attributed) graph?
- graph-based learning in a *transductive* setting is inherently related to the challenging poisoning/causative attacks

- **poisoning/causative attacks**: target the training data (the model is retrained after the attack)
- **evasion/exploratory attacks**: target the test data/application phase (the model parameters are kept fix based on the clean graph)

- Deriving effective poisoning attacks is usually computationally harder since the subsequent learning of the model has to be considered
- attacks on the test data are causative as well since the test data is used while training the model (transductive, semi-supervised learning);
- even when the model is fixed (evasion attack), manipulating one instance might affect all others due to the relational effects imposed by the graph structure

- **influencer attack**, target nodes are not in the attacker nodes
- **direct attack**

Unnoticeable Perturbations

- limit the number of allowed changes by a budget Δ
- **degree distribution** and **feature co-occurence**

first attack a **surrogate model**, leading to an attacked graph. This graph is subsequently used to train the final model.

- transferability

surrogate model

$$Z' = softmax\left(\hat{A}\,\hat{A}\,XW^{(1)}\,W^{(2)}\right) = softmax\left(\hat{A}^2\,XW\right)$$

Algorithm 1: NETTACK: Adversarial attacks on graphs

Input: Graph $G^{(0)} \leftarrow (A^{(0)}, X^{(0)})$, target node v_0, attacker nodes $\mathcal{A}$, modification budget Δ

Output: Modified Graph $G' = (A', X')$

Train surrogate model on $G^{(0)}$ to obtain W // Eq. (13);

$t \leftarrow 0$;

while $|A^{(t)} - A^{(0)}| + |X^{(t)} - X^{(0)}| < \Delta$ **do**

- $C_{struct} \leftarrow \texttt{candidate\_edge\_perturbations}(A^{(t)}, \mathcal{A})$;
- $e^* = (u^*, v^*) \leftarrow \arg\max_{e \in C_{struct}} s_{struct}\left(e; G^{(t)}, v_0\right)$;
- $C_{feat} \leftarrow \texttt{candidate\_feature\_perturbations}(X^{(t)}, \mathcal{A})$;
- $f^* = (u^*, i^*) \leftarrow \arg\max_{f \in C_{feat}} s_{feat}\left(f; G^{(t)}, v_0\right)$;
- **if** $s_{struct}(e^*; G^{(t)}, v_0) > s_{feat}(f^*; G^{(t)}, v_0)$ **then**
  - $G^{(t+1)} \leftarrow G^{(t)} \pm e^*$;
- **else** $G^{(t+1)} \leftarrow G^{(t)} \pm f^*$;
- $t \leftarrow t + 1$;

return $: G^{(t)}$

// Train final graph model on the corrupted graph $G^{(t)}$;

Complexity

$$\mathcal{O}(\Delta \cdot |\mathcal{A}| \cdot (N \cdot th_{v_0} + D))$$

where th_{v_0} indicates the size of the two-hop neighborhood of the node v_0 during the run of the algorithm.
potential edge perturbations (N at most) and feature perturbations (D at most)

- **target nodes**: (i) the 10 nodes with highest margin of classification, i.e. correctly classified, (ii) the 10 nodes with lowest margin (but still correctly classified) and (iii) 20 more nodes randomly
- **attackers nodes**: picking 5 random nodes as attackers from the neighborhood of the target

conclusion:

- attacking the features and structure simultaneously is very powerful;
- the introduced constraints do not hinder the attack while generating more realistic perturbations;
- Direct attacks are clearly easier than influencer attacks

ADVERSARIAL ATTACKS ON GRAPH NEURAL NETWORKS VIA META LEARNING

- *training time attacks* on graph neural networks for *node classification*
- use *meta-gradients* to solve the bilevel problem underlying training-time attacks, essentially treating the graph as a hyperparameter to optimize
- attack do not assume any knowledge about or access to the target classifiers.
- global attack: to have the test samples classified as any class different from the true class
- focus on changing the graph structure only