Scribe Note: Adversarial Attacks on Neural Networks for Graph Data

Presenter: Faizan Ahmad, Scribe: Ji Gao

2019/2/26

1 Task

Attack graph classification model.

- Change the prediction on a specific vertex.
- Change binary features of the vertex or add/remove edges in the graph.

2 Overview Figure

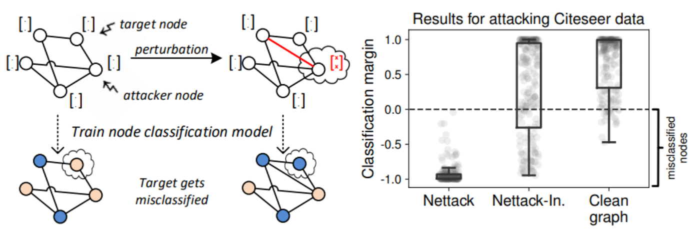


Figure 1: Small perturbations of the graph structure and node features lead to misclassification of the target.

3 Graph Attack Problem

- **Setting:** Node Classification Task.
  - Graph $G^{(0)} = (A^{(0)}, X^{(0)})$, where $A \in \{0,1\}^{N\times N}$ is a adjacency matrix, and $X \in \{0,1\}^{N\times D}$ is a binary feature matrix (each vertex has D dimension feature).
  - Model $f_\theta(A, X) = Z$ gives a probability distribution Z on labels c.
  - Cross-entropy loss $L(\theta; A, X) = \sum_{c\in V_L} \ln Z_{c,v_c}$
- **Problem**:

- Give original graph $G^{(0)} = (A^{(0)}, X^{(0)})$, generate a perturbed $G' = (A', X')$.
- Target: A certain node v_0
- The modification is limited to a set of attacker nodes $\mathcal{A} \subseteq V$, that:

$$X_{ui}^{(0)} \neq X'_{ui} \Rightarrow u \in \mathcal{A}$$

$$A_{uv}^{(0)} \neq A'_{uv} \Rightarrow u \in \mathcal{A} \cup v \in \mathcal{A}$$

- The modification is limited by some budget Δ, detailed calculation in next part.

- **Formal Definition:**

Problem 1 *Given a graph $G^{(0)} = (A^{(0)}, X^{(0)})$, a target node v_0 and attacker nodes $\mathcal{A}$. Let c_{old} denote the predicted class of v_0 with $G^{(0)}$. Determine:*

$$\begin{aligned} &\underset{A',X' \in \mathcal{P}^{G^{(0)}}_{\Delta,\mathcal{A}}}{\text{argmax}} \max_{c \neq c_{old}} \ln Z^*_{v_0,c} - Z^*_{v_0,c_{old}} \\ &\textit{Subject to } Z^* = f_{\theta^*}(A', X') \end{aligned} \tag{1}$$

It's a **poisoning attack** if $\theta' = \text{argmin}_\theta L(\theta; A', X')$, i.e., the parameter of model f is retrained.

It's a **evasion attack** if $\theta' = \theta = \text{argmin}_\theta L(\theta; A^{(0)}, X^{(0)})$

4 NETTACK

- **In one sentence:** Sort the importance of the possible attacks and pick the best ones till budget is filled. A greedy attack.

- **Budget definition:**
  - Evaluate the size of a perturbation is hard in graph (and any discrete data type).
  - For the graph modification part, use a statistical two-sample test to evaluate the similarity between perturbed sample and original sample.

    The scaling factor is estimated by:

$$\alpha_G \approx 1 + |D_G| \cdot [\sum_{d_i \in D_G} \log \frac{d_i}{d_{min} - \frac{1}{2}}]^{-1} \tag{2}$$

Likelihood is estimated by:

$$l(D_G) = |D_G| \cdot \log \alpha_G + |D_G| \cdot \alpha_G \cdot \log d_{min} + (\alpha_G + 1) \sum_{d_i \in D_G} \log d_i \quad (3)$$

And finally the test statistic is

$$\Lambda(G^{(0)}, G') = -2 \cdot l(D_{G^{(0)}} \cup D_G) + 2 \cdot (l(D_{G^{(0)}}) + l(D_G)) < \tau \approx 0.004 \quad (4)$$

- For the feature modification part, make sure the co-occurence of features is perserved. That is, if two features are never co-occured in $G^{(0)}$, the change is noticeable.

  Let S_u be the all the features present for node u, the addition of a feature i is acceptable if

$$p(i|S_u) = \frac{1}{|S_u|} \sum_{j \in S_u} (1/d_j) \cdot E_{ij} > \sigma \quad (5)$$

- **Attack:**
  - Too hard to handle, so instaed attack a surrogate model:

$$Z = \text{softmax}(\hat{A}\hat{A}XW^{(1)}W^{(2)}) = \text{softmax}(\hat{A}^2XW) \quad (6)$$

    Which is original GCN with non-linear part removed.
  - Define score of add/remove an edge $e = (u, v)$or add/remove a feature $f = (u, i)$:

$$s_{\text{struc}}(e, G, v_0) = L_s(A', X; W, v_0) \quad (7)$$
$$s_{\text{feat}}(f, G, v_0) = L_s(A, X'; W, v_0) \quad (8)$$

    Where,

$$L_s(A, X; W, v_0) = \max_{c \neq c_{\text{old}}} [\hat{A}^2XW]_{v_0 c} - [\hat{A}^2XW]_{v_0 c_{\text{old}}} \quad (9)$$

  - Use an iterative approach: Each step find a locally best modification. Stop when the budget is filled.
- **Faster computation:**
  - Calculate only the incremental part of $\hat{A}^2$ in a graph edge attack.
  - Calculate only the incremental part of X in a feature attack.
  - Accelerate the candidate set calculation by pre-processing.

5 Evaluation:

5.1 Dataset

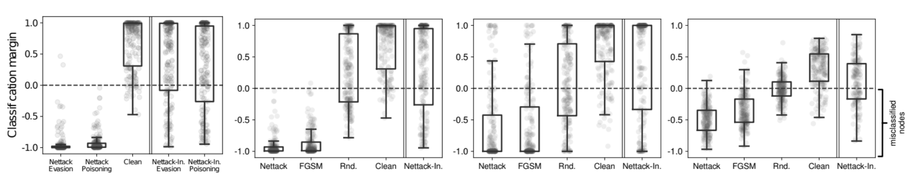
- Cora-ML: A citation dataset with 2,810 nodes, 7,981 edges, and bag-of-word text features.
- Citeseer: A citation dataset with 2,110 nodes, 3,757 edges, and bag-of-word text features.
- POLBLOGS: A blog post dataset with 1,222 nodes, 16,714 edges and bag-of-word text features.

5.2 Baselines

- **Fast Gradient Sign Method** (with projection), which is an attack only on features
- **RND**: Randomly modified the graph structure, which is only applied to the graph not features.

5.3 Result

- Change the prediction of the graph convolutional network model on a certain v_0.



(a) Evasion vs. poisoning for GCN (b) Poisoning of GCN (c) Poisoning of Column Network (d) Poisoning of DeepWalk

Figure 6: Results on Cora data using different attack algorithms. Clean indicates the original data. Lower scores are better.

- Get good result transferred to Column Network and DeepWalk.
- Even with limited knowledge about the data(a subgraph), can still reduce the prediction of v_0.