Neighbor-Anchoring Adversarial Graph Neural Networks

(Extended Abstract)

Zemin Liu[1], Yuan Fang[1], Yong Liu[2], Vincent W. Zheng[3]
[1]*Singapore Management University*; [2]*Nanyang Technological University*; [3]*WeBank*
{zmliu, yfang}@smu.edu.sg, stephenliu@ntu.edu.sg, vincentz@webank.com

***Abstract*—While graph neural networks (GNNs) exhibit strong discriminative power, they often fall short of learning the underlying node distribution for increased robustness. To deal with this, inspired by generative adversarial networks (GANs), we investigate the problem of adversarial learning on graph neural networks, and propose a novel framework named NAGNN (*i.e.*, Neighbor-anchoring Adversarial Graph Neural Networks) for graph representation learning, which trains not only a discriminator but also a generator that compete with each other. In particular, we propose a novel neighbor-anchoring strategy, where the generator produces samples with explicit features and neighborhood structures anchored on a reference real node, so that the discriminator can perform neighborhood aggregation on the fake samples to learn superior representations.**

I. Introduction

GRAPH neural networks (GNNs) [1], with the key process of multi-layer *neighborhood aggregation*, aim to map the nodes of a graph into a low-dimensional space whilst preserving their structural information. In the context of node classification, while GNNs can be powerful discriminative models by learning the node conditional class distributions, generative approaches can also be useful by learning the underlying node distributions conditioned on classes.

Inspired by GANs [2], which exploits the power of both discriminative and generative models, we investigate the adversarial training of GNNs. In our approach, GANs are utilized to generate fake nodes in the sparse region of the graph where there is a low-density of edges [3]. As such, by discriminating the real and fake nodes, the discriminator can learn a better decision boundary between classes. However, training GNNs in an adversarial manner is non-trivial. In particular, two major challenges still remain with the design of the generator.

First, *what is the definition of a sample on a graph?* Nodes are characterized by not only the features, but also the structures, *i.e.*, the neighbors on the graph. Thus, the generator must account for both parts, which are necessary for the key operation of neighborhood aggregation in GNNs. Second, *how do we produce good samples?* Previous approaches typically produce samples from a prior noise distribution freely [3], without explicit constraints between the real and fake samples. On the contrary, if each fake sample is generated w.r.t. a reference real node, the generator and discriminator could become more conscious of each other's output, which could drive the discriminator to better detect their finer differences.

To address the above challenges, we propose a novel *neighbor-anchoring* strategy. In this strategy, the generator produces samples with explicit features and structures, so that the discriminator can perform neighborhood aggregation on them. Furthermore, given a reference real node, the generator produces a sample anchored on the same set of neighbors as the reference node, *i.e.*, they have the same set of neighbors. At the same time, the generator utilizes a feature synthesizer, aiming to produce features that mimic the reference node. By sharing the same neighbors and thus isolating their effects on a fake sample and its reference node, the model can focus on discriminating and generating their features to achieve mutual improvement, and encourage the fake sample to reside closer to the real node at the periphery of dense regions.

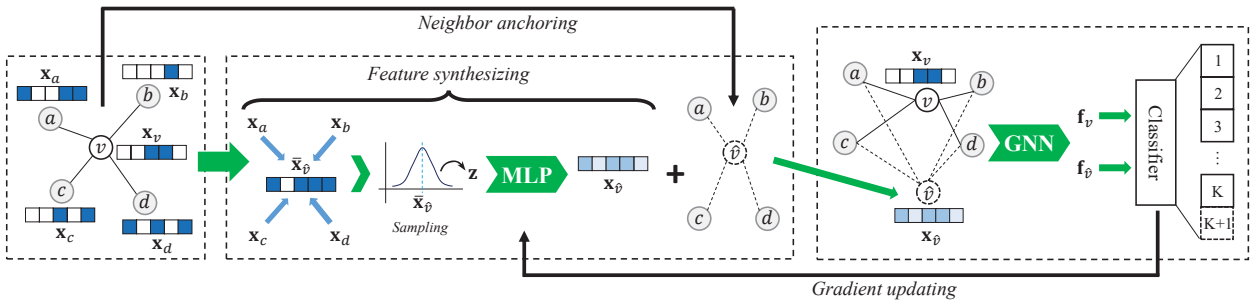
II. The Proposed Model: NAGNN

Overview. The overall framework of NAGNN is illustrated in Fig. 1. The role of generator is to produce fake samples. Each fake sample $\hat{v}$ consists of a feature vector $\mathbf{x}_{\hat{v}}$, as well as a local structure or a set of neighbors $\mathcal{N}_{\hat{v}}$. On the other hand, the discriminator employs a GNN to learn node representations, so as to differentiate the fake sample $\hat{v}$ from the real node v based on their final representations $\mathbf{f}_{\hat{v}}$ and $\mathbf{f}_v$, respectively.

Discriminator. We employ the multi-layer GNNs to calculate the final representations $\mathbf{f}_v$ and $\mathbf{f}_{\hat{v}}$ for real and fake nodes v and $\hat{v}$, respectively, which leverage a softmax function for multi-class output [1], [4]. Thus, the discriminator is also parameterized by a classification matrix $\mathbf{W} \in \mathbb{R}^{(K+1)\times N}$ under the $K+1$ class setting (K is the number of node classes), such that $D(y|v;\theta_D) = \frac{\exp(\mathbf{W}_y \mathbf{f}_v)}{\sum_{y'=1}^{K+1} \exp(\mathbf{W}_{y'} \mathbf{f}_v)}$. Here $\mathbf{W}_y \in \mathbb{R}^N$ denotes the y-th row of $\mathbf{W}$. Specifically, $D(y|v;\theta_D)$ parameterized by θ_D, estimates the probability of class y given node v. The goal of the discriminator is to classify the real nodes into the their corresponding classes $\{1, 2, \ldots, K\}$, and simultaneously put the fake samples into the augmented class $K+1$. The loss function for the discriminator can be formulated as follows,

$$- \frac{1}{|\mathcal{L}|} \sum_{(v,y)\in\mathcal{L}} \log D(y|v;\theta_D)$$
$$- \alpha \cdot \frac{1}{|\hat{\mathcal{V}}|} \sum_{\hat{v}\in\hat{\mathcal{V}}} \log D(K+1|\hat{v};\theta_D) + \lambda_D \left\|\theta_D\right\|_2^2 .$$

Here $(v, y) \in \mathcal{L}$ is a real node $v \in \mathcal{V}$ with its observed label y, whereas $\hat{\mathcal{V}}$ is the set of fake samples produced by the generator.



(a) Neighborhood (b) Generator (c) Discriminator

Fig. 1: Overall framework of NAGNN.

TABLE I: Node classification performance (in percent) with standard deviation; The best results are **bolded**.

Methods	Input data	Cora Accuracy	Cora Micro-F	Cora Macro-F	Citeseer Accuracy	Citeseer Micro-F	Citeseer Macro-F	Pubmed Accuracy	Pubmed Micro-F	Pubmed Macro-F	DBLP Accuracy	DBLP Micro-F	DBLP Macro-F
DeepWalk	$\mathbf{A}$	73.8±0.3	74.9±0.1	74.0±0.1	61.6±0.2	60.5±1.0	59.8±0.5	67.4±0.3	65.2±0.1	66.1±0.1	50.4±1.0	51.8±0.8	49.1±1.1
GraphGAN	$\mathbf{A}$	58.8±0.2	57.9±0.1	57.2±0.1	60.4±1.4	58.5±0.1	58.6±0.1	73.2±0.1	75.3±0.1	73.2±0.1	52.4±2.5	51.1±3.5	52.1±4.3
ARGA	$\mathbf{A}$, $\mathbf{X}$	58.2±0.5	48.8±0.8	39.7±0.7	48.7±1.3	47.4±2.1	44.5±2.3	53.8±1.3	46.5±2.7	41.4±3.5	56.4±1.3	55.1±1.1	55.2±1.2
Graph$\mathcal{S}$GAN	$\mathbf{A}$, $\mathbf{X}$, $\mathcal{L}$	79.2±0.6	79.3±0.5	78.0±0.6	67.4±0.7	65.8±0.4	61.8±0.5	68.2±0.4	68.7±0.5	67.5±0.5	58.6±0.9	57.4±0.8	56.8±0.9
GCN	$\mathbf{A}$, $\mathbf{X}$, $\mathcal{L}$	81.5±0.7	80.8±0.5	80.4±0.6	70.4±0.5	68.3±0.7	66.9±0.4	78.9±0.3	78.8±0.4	78.0±0.3	61.7±1.5	62.2±1.2	60.9±0.7
NAGCN	$\mathbf{A}$, $\mathbf{X}$, $\mathcal{L}$	**83.2**±0.6	**81.7**±0.4	**81.9**±0.5	**72.8**±0.4	**70.7**±0.5	**69.0**±0.4	**79.0**±0.3	**79.4**±0.2	**78.4**±0.3	**66.4**±0.7	**65.4**±0.9	**64.6**±0.9
GAT	$\mathbf{A}$, $\mathbf{X}$, $\mathcal{L}$	82.9±0.6	82.0±0.6	81.8±0.6	72.4±0.7	70.4±0.8	68.2±0.7	77.2±0.5	77.7±0.7	76.6±0.5	68.6±3.1	64.1±4.3	57.2±7.2
NAGAT	$\mathbf{A}$, $\mathbf{X}$, $\mathcal{L}$	**83.5**±0.4	**82.6**±0.3	**82.5**±0.2	**72.9**±0.4	**70.9**±0.5	**68.3**±0.8	**77.7**±0.4	**77.8**±0.1	**77.0**±0.3	**71.8**±1.7	**69.1**±1.4	**68.6**±1.3

Moreover, $\alpha > 0$ controls the importance of fake samples, and $\lambda_D > 0$ is the regularization parameter for the discriminator.

Generator. In our context, we propose the *neighbor anchoring* strategy to generate fake nodes, so that a fake sample $\hat{v}$ consists of dual parts: its feature vector $\mathbf{x}_{\hat{v}}$, and its structure or the set of neighbors $\mathcal{N}_{\hat{v}}$. Sample $\hat{v}$'s feature vector $\mathbf{x}_{\hat{v}}$ is synthesized by a neural network such as a multi-layer perceptron (MLP) as shown in Fig. 1(b), and $\hat{v}$'s neighborhood is anchored on v's neighborhood, *i.e.*, $\mathcal{N}_{\hat{v}} = \mathcal{N}_v$. With the neighbor-anchoring strategy, our generator parameterized by θ_G can be formulated as a function $G(v, \mathbf{z}; \theta_G)$, which outputs a fake sample $\hat{v}$ based on a reference node v and some noise vector $\mathbf{z}$ drawn from a predetermined multivariate Gaussian distribution $Z \triangleq \text{Gaussian}(\bar{\mathbf{x}}_{\hat{v}}, \sigma^2\mathbf{I})$. Here $\bar{\mathbf{x}}_{\hat{v}}$ represents the mean vector, and $\sigma^2\mathbf{I}$ is the covariance matrix for some choice of real-valued σ. To synthesize more realistic features, we derive the initial estimator from the features of $\hat{v}$'s neighbors, *i.e.*, the neighbors of v, based on the assumption that the features of a node is related to the features of its neighbours. Here we compute the mean feature vector, *i.e.*, $\bar{\mathbf{x}}_{\hat{v}} = \frac{1}{|\mathcal{N}_{\hat{v}}|}\sum_{v' \in \mathcal{N}_{\hat{v}}} \mathbf{x}_{v'}$, although other forms of aggregation such as max- or sum-pooling can also be adopted. Subsequently, given some noise $\mathbf{z} \sim Z$, we employ an MLP to synthesize the feature vector of sample $\hat{v}$, *i.e.*, $\mathbf{x}_{\hat{v}} = \text{MLP}(\mathbf{z})$, and the generator's parameters θ_G are in fact this MLP's parameters. Finally, to make the discriminator believe that the sample $\hat{v}$ also belongs to the same class of the reference node v, the loss of the generator can be defined as follows,

$$-\frac{1}{|\mathcal{L}|}\sum_{(v,y)\in\mathcal{L},\mathbf{z}\sim Z} \log D(y|G(v,\mathbf{z};\theta_G);\theta_D) + \lambda_G \left\|\theta_G\right\|_2^2.$$

III. Empirical Evaluation

Based on two base GNN models GCN [1] and GAT [4], we conduct extensive experiments on four benchmark datasets for comparison with representative baselines including DeepWalk [5], GraphGAN [6], ARGA [7] and Graph$\mathcal{S}$GAN [3]. The comparison presented in Table I demonstrates the effectiveness of our proposed model NAGNN. More details for the datasets and baselines can be found in the full paper [8]. We further conduct model analysis including the performance with fewer labeled nodes, training efficiency, ablation study, visualization, parameters sensitivity and a case study. Furthermore, we also provide a theoretical analysis to substantiate the intuition behind our neighbor-anchoring generator, which can theoretically verify the advantage of the proposed NAGNN.

References

[1] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *ICLR*, 2017.

[2] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *NeurIPS*, 2014, pp. 2672–2680.

[3] M. Ding, J. Tang, and J. Zhang, "Semi-supervised learning on graphs with generative adversarial nets," in *CIKM*, 2018, pp. 913–922.

[4] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *ICLR*, 2018.

[5] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *KDD*, 2014, pp. 701–710.

[6] H. Wang, J. Wang, J. Wang, M. Zhao, W. Zhang, F. Zhang, X. Xie, and M. Guo, "GraphGan: Graph representation learning with generative adversarial nets," in *AAAI*, 2018, pp. 2508–2515.

[7] S. Pan, R. Hu, G. Long, J. Jiang, L. Yao, and C. Zhang, "Adversarially regularized graph autoencoder for graph embedding." in *IJCAI*, 2018, pp. 2609–2615.

[8] Z. Liu, Y. Fang, Y. Liu, and V. W. Zheng, "Neighbor-anchoring adversarial graph neural networks," *IEEE Transactions on Knowledge and Data Engineering*, 2021.