GraphCBAL-Sys: A Class-Balanced Active Learning System for Graphs

Chengcheng Yu[1], Wenqian Zhou[1], Jiahui Wang[1], Fangshu Chen[1], Jiapeng Zhu[2], and Xiang Li[2(✉)]

[1] Shanghai Polytechnic University, Shanghai, China
ccyu@sspu.edu.cn, 20221113272@stu.sspu.edu.cn,
{wangjh,fschen}@sspu.edu.cn
[2] East China Normal University, Shanghai, China
jiapengzhu@stu.ecnu.edu.cn, xiangli@dase.ecnu.edu.cn

Abstract. Active learning for Graph Neural Networks (GNNs) aims to select valuable unlabeled samples for annotation with a limited budget to maximize the GNNs' performance at a low cost. However, most methods often result in imbalanced class distributions, leading to a bias toward majority classes, which undermines minority class performance and overall model effectiveness. To tackle this issue, we develop the Class-Balanced Active Learning System for Graphs *GraphCBAL-Sys*. It learns an optimal policy through reinforcement learning to acquire class-balanced and informative nodes for annotation. Additionally, GraphCBAL-Sys is capable of visualizing the internal processes and results during our model's training and testing phases. Our demonstration video can be found here: https://b23.tv/yCLOIPw.

Keywords: Graph Active learning · Class balance · Reinforcement learning · Visualization.

1 Introduction

Graph neural networks (GNNs) have achieved remarkable success in various applications, including node classification, link prediction, and graph classification. However, most GNN models often rely on a large volume of labeled data for training, which can be costly to acquire.

Active learning on graphs has emerged as a promising approach to tackle this challenge, aiming to select the most informative samples from unlabeled data for annotation to enhance the performance of GNNs with a limited budget. Several graph active learning methods have been proposed and demonstrated their effectiveness [1]. However, the samples chosen by these methods can often lead to a highly imbalanced class distribution, which becomes especially problematic in highly skewed class scenarios. When GNNs are trained on such class-imbalanced labeled data, they may develop a prediction bias towards the majority classes, ultimately resulting in a decline in overall performance. Therefore, it is essential for active learning to select class-balanced and valuable samples for annotation to avoid class imbalance problems [2].

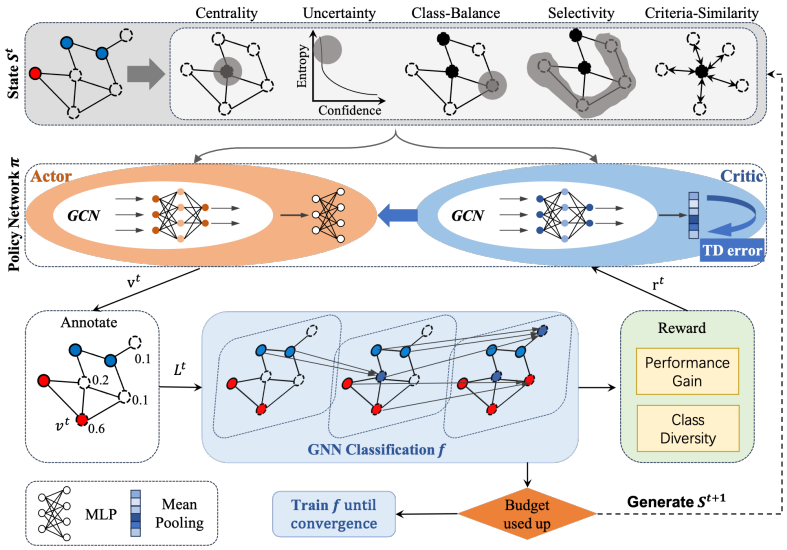

Fig. 1. The architecture of GraphCBAL-Sys.

In this paper, we introduce a class-balanced graph active learning system, *GraphCBAL-Sys*. It leverages a Reinforcement Learning (RL) framework to learn an optimal policy to select class-balanced and informative samples for annotation, aiming to maximize the performance of GNNs trained with the selected labeled nodes. Moreover, it also provides a GraphCBAL++ version for obtaining a more class-balanced labeled set. Details of GraphCBAL and GraphCBAL++ are introduced in [4]. Moreover, GraphCBAL-Sys can visualize the process and results of our model training and testing on the web. It provides configurable parameters to train or test our method.

2 System Architecture

We formalize the problem of class-balanced active learning on graphs as a Markov Decision Process. Figure 2 provides the architecture of GraphCBAL-Sys. Initially, the labeled set L^0 is set to be empty. At step t, we first use the output of the GNN classification f and graph information G to update the state S^t. A node v^t is sampled from the unlabeled nodes set based on the policy for annotation and is added to the label set L^t. Then, the GNN classification f is trained for one epoch, then evaluate f on the validation set, which is used to generate the graph state S^{t+1} for the next step. When the budget is used up, we stop querying and train classification f until convergence. The major components in our system are described as follows:

- **State** The state is characterized by the following five factors: Centrality, Uncertainty, Class diversity, Selectivity and Criteria-Similarity. We concatenate the above five metrics for each node and get the state S^t at step t.
- **Action** At step t, the action probabilities are generated by the policy network based on the current state S^t, which is used to select a node v_t from all the unlabeled nodes in the training set for annotation.
- **Reward** The reward function is designed based on the classification performance gain on the validation set and the class diversity.
- **Policy Network** The policy network comprises an actor and a critic. The actor consists of an L-layer GCN followed by a linear layer, with a softmax function applied to compute the probability distribution. The critic employs a GCN with mean pooling to estimate the value of a given state.
- **Policy Training** The A2C algorithm [3] is employed to train the policy network. The critic is updated by minimizing the TD error. The objective of the actor is to maximize the expected cumulative reward.

Additionally, we further upgrade our model GraphCBAL to GraphCBAL++ by incorporating a punishment mechanism containing a penalty term in the reward function and a Majority-Score in the state space.

3 Demonstration Scenarios

GraphCBAL-Sys provides an interactive web interface with configurable model parameters for training and testing. We train and test our model on a server with an NVIDIA Tesla V100 PCIe 40GB GPU. During this process, data generated by our model is pushed to a Redis database on a public server and retrieved by the web interface for display. As shown in Figure 5, our demonstration includes the following parts:

GraphCBAL-Train During policy training, we use the current policy to select nodes for annotation within a given budget in each episode. Then, we train a classification GCN using these labeled nodes and evaluate the classification performance on the validation set. We will show how we train the policy and the process involved.

- **Monitor** We will monitor and visualize the class imbalance ratio of the labeled set and the classification performance in each episode, including Micro-F1, Macro-F1, and per-class accuracy.
- **Internals** We will show the RL rewards and the class distribution of the current labeled set after the current policy selects a node to annotate at each step of every episode.

GraphCBAL-Test During policy testing, we will present the results using pre-trained models, including GraphCBAL and GraphCBAL++, and demonstrate how these models select nodes for annotation.

- **Monitor** The dynamics of the classification performance and the class imbalance ratio of the labeled set will be monitored and visualized online. At

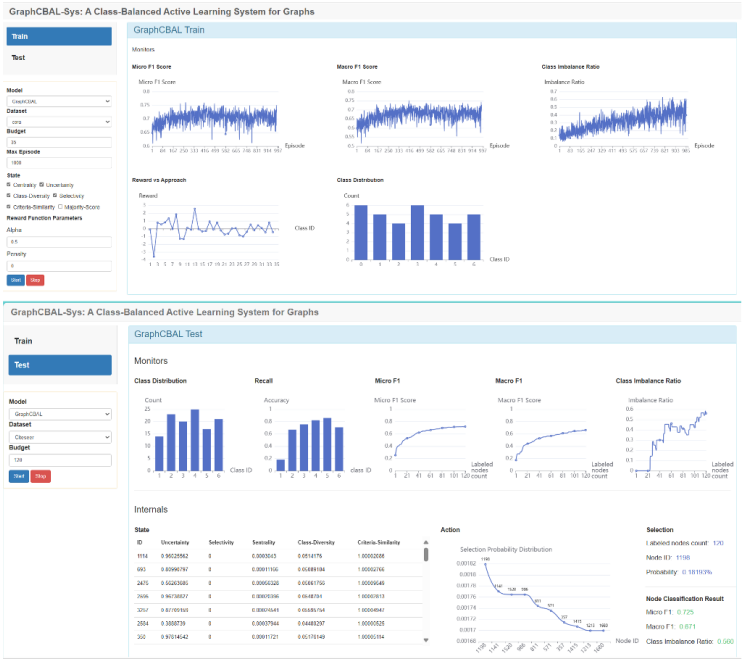

Fig. 2. The demonstration scenarios of GraphCBAL-Sys.

the same time, nodes are being selected by our model for annotation. We will also show the class distribution and per-class accuracy at each step.

– **Internals** At each step, we will demonstrate how the state is organized and updated. The selection probability distribution of the top 10 nodes, the selection result, and the node classification result will also be shown.

Acknowledgments. This work is supported by Natural Science Foundation of Shanghai No. 24ZR1425500 and National Natural Science Foundation of China No. 62002216.

References

1. Hu, S., Xiong, Z., Qu, M., Yuan, X., Côté, M., Liu, Z., Tang, J.: Graph policy network for transferable active learning on graphs. In: NeurIPS (2020)
2. Ma, Y., Tian, Y., Moniz, N., Chawla, N.V.: Class-imbalanced learning on graphs: A survey. arXiv preprint arXiv:2304.04300 (2023)
3. Mnih, V., Badia, A.P., Mirza, M., Graves, A., Lillicrap, T.P., Harley, T., Silver, D., Kavukcuoglu, K.: Asynchronous methods for deep reinforcement learning. In: ICML. vol. 48, pp. 1928–1937 (2016)
4. Yu, C., Zhu, J., Li, X.: Graphcbal: Class-balanced active learning for graph neural networks via reinforcement learning. In: CIKM. pp. 3022–3031 (2024)