GFlowNet Graph Neighbor Sampling For Link Prediction

Vidhi Patel
CPSC 583

1 Introduction

Graph Neural Networks (GNNs) have emerged as powerful tools for representation learning. However, applying machine learning to graphs presents scalability challenges, primarily due to intricate node connectivity and the exponential growth in nodes needed for processing as network depth grows (Younesian et al. [2023]).

The scalability issue in GNNs is addressed by graph sampling techniques, where only a sampled subset of graph nodes is considered. Until recently, sampling methods have been non-adaptive and did not consider node features and specific training objectives during sampling. With the introduction of an adapative sampling approach by Taraneh and colleagues, a novel solution has emerged in the field (Younesian et al. [2023]). Their proposed solution, GFlowNet Graph Neighbor Sampling (GRAPES), is an adaptive graph sampling method which maximizes GNN accuracy within a fixed sampling budget (Bengio et al. [2021];Younesian et al. [2023]). A GFlowNet sampler is used to incorporate feedback from the GNN, learning scores over nodes that influence their inclusion likelihood in the sample. By uniting the sampling process with GNN computation, this approach enables synchronized training, taking into account node features, graph structure, sample size, and other contextual factors.

The promise of GRAPES is highlighted in the state-of-the-art results it achieved on several node classification tasks. In this paper, we aim to further evaluate the adaptability and effectiveness of GRAPES by evaluating GRAPES on link prediction. We maintain the original framework of the algorithm and supplement it by adding negative edge sampling when training the link prediction GCN.

Our results are promising for the application of adaptive graph sampling on link prediction tasks. Compared to baseline methods, the average AUC score across 10 runs improved or remained comparable for GRAPES for both the CiteSeer and Cora datasets. We were able to draw the following conclusions from our experiments:

- GRAPES performs similar (or better) to state-of-the-art sampling-based methods on the standard datasets Cora and CiteSeer
- GRAPES remains within AUC score range of baseline performance at large sample sizes even at low sample sizes

2 Related Works

GFlowNets: Recent applications of GFlowNets have spanned various fields, including molecule discovery (Bengio et al. [2021]) and sequence design (Jain et al. [2022]). Efforts have also been directed towards enhancing the learning efficiency (Madan et al. [2023]) and extending GFlowNets into stochastic environments with stochasticity in transition dynamics (Pan et al. [2023]). While GFlowNets have also been utilized for subgraph sampling in GNN explainability (Li et al. [2023]), they differ from the GRAPES method, which focuses on scaling GNN training for large graphs (Younesian et al. [2023]).

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

Link Prediction: Link prediction, a fundamental challenge in network analysis, has seen a surge of research aimed at enhancing predictive accuracy and scalability. Numerous methodologies have been explored, spanning from traditional methods like common neighbors (LAdamic and Adar [2003]) to advanced graph neural networks (GNNs) (Grover and Leskovec [2016])(Wang et al. [2014]). GNNs, GNNs, particularly in the semi-supervised link prediction domain, have been used with strategies like label propagation and matrix factorization to improve prediction accuracy and address issues like class imbalance (Kashima et al. [2009]; Raymond and Kashima [2010]).

While these state-of-the-art models have made significant strides, some challenges remain, such as information loss, test data size, and scalability. Additionally, we find that often complexity increases substantially as network size grows. Recent work has addressed computational and scalability issues in link prediction tasks by sampling enclosing subgraphs using random-walk-based sampling (Louis et al. [2022]). While this methodology has achieved results comparable to state-of-the-art processes, we aim to provide an alternate technique to tackle scalability through a graph flow-based sampling strategy using GRAPES.

3 Methods

We will be using GFlowNet Graph Neighbor Sampling (GRAPES) - an adaptive graph sampling method which maximizes GNN accuracy within a fixed sampling budget (Bengio et al. [2021]; Younesian et al. [2023]). The original paper that proposed this used this algorithm for node classification so we will be modifying it for link prediction. Algorithm 1 shows one epoch of GRAPES in pseudocode.

Algorithm 1 One epoch of GRAPES for Link Prediction

Require: Graph G_C, node features X, target links L^t, batch size B, sample size k, GCN for link prediction GCN_L, and GCN for GFlowNet GCN_F.

Divide target links L^t into batches L^b of size B.
for each batch L^b **do**
  $K^0 = L^b$
  Build adjacency matrix A^0 from K^0.
  for layer $l = 1$ to L **do**
    $L_n^l \leftarrow N(K^{l-1})$ ▷ Get all n neighbors of K^{l-1}
    $p_1, ..., p_n \leftarrow GCN_F(A^0, ..., A^{l-1})$ ▷ Compute probabilities of positive edges
    $G_1, ..., G_n$ Gumbel(0, 1) ▷ Sample Gumbel noise
    $L_k^l \leftarrow$ top$(k, logp_1 + G_1, ..., logp_n + G_n)$ ▷ Get k best positive edges
    $K^l = K^{l-1} \cup L_k^l$ ▷ Add positive edges
    Build adjacency matrix A^l from $K^l + N^l$
  end for
  Pass all $\tilde{A}^l$ to GCN_L encoder
  $N^l \leftarrow$ negative-sampling(method=sparse) ▷ Conduct sampling of negative edges
  Combine N^l and $\tilde{A}^l$ for edge labels and edge label indices
  Pass all $\tilde{A}^l + N^l$ to GCN_L decoder and obtain link loss L_{GCN_L}.
  $R(A^0, ..., A^L) \leftarrow \exp(-\alpha \cdot L_{GCN_L})$ ▷ Calculate reward
  Compute L_{GRAPES} loss.
  Update parameters of GCN_F by minimizing L_{GRAPES}.
  Update parameters of GCN_L by minimizing L_{GCN_L}.
end for

The GCN_L in this approach will be slightly different than the GCN_C used for classification in the original paper (Younesian et al. [2023]). GCN_L will have a series of GCN layers, some of which will be hidden layers. It will first encode the graph using GCN layers, and then decode by calculating link scores based on the encoded features. The decoding is separated into two methods: decode and decodeall. The decode method computes scores for specific edge pairs, and decodeall computes scores for all possible edges in the graph. Separating the GCN into encoding and decoding makes it simpler to add the negative edge samples during training. This GCN architecture is a combination of the GCN_C used in the orignal paper and the GCN example given in the Pytorch Geometric Library documentation for link prediction as well as the Colab 3 assignment (Fey and Lenssen [2019]).

4 Experiments

For experiments, a two-layer GCN with a ReLU activation function for link prediction and GFlowNet networks, GCNC and GCNF were used. We evaluate GRAPES link prediction capabilities on two citation networks: Cora and Citeseer. The edges in the datasets were randomly split into 70% training, 10% validation and 20% testing (P Louis [2023]). Table 1 shows the statistics of the dataset.

Dataset	Nodes	Edges	Features	Avg Degree
Cora	2708	4488	1433	3.31
CiteSeer	3327	3870	3703	2.33

Table 1: Data Statistics

The evaluation protocol of the experiments consisted of training the GCN on link prediction and then assessing performance of the GCN on the testing set using the complete graph (Younesian et al. [2023]). We use PyTorch Geometric for experimental code and refer to the original paper's code for the graph sampling implementation [Fast Graph Representation Learning with PyTorch Geometric]. We use a 25GB Tesla T4 GPU.

Hyperparameters were tuned seperately for both datasets. We used the Weights and Biases library (as done in the original paper) to determine hyperparameters. The main three parameters that were tuned were the learning rate of the GFlowNet,the learning rate of the link prediction GCN and the scaling parameter α. All three were sampled using log uniform distribution across 50 runs. During the tuning of these parameters, we also tuned the number of samples taken in GFlowNet, the regularization parameter for the GCN loss, dropout rate for the GCN, the hidden dimensions for both GFlowNet and GCN, maximum epochs and batch size for the training loader. The best parameter combination was chosen by maximizing the AUC score on the validation set. The output of both hyperparameter searches can be found in Figure 1. For all models, the number of hops was set to 2.

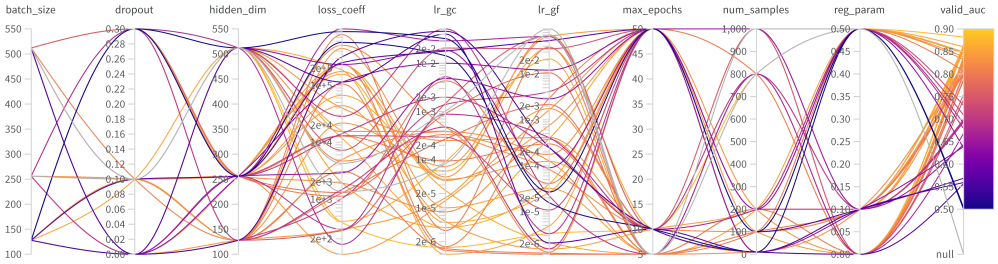
For both datasets, we ran the model 10 times and report the average area under the curve (AUC) score along with the standard deviation (Table 2). We compare this AUC-score with three standard message passing GNNs: (1) GCN (P Louis [2023]), (2) GraphSAGE and (3) GAT (P Louis [2023]). The baselines had the same two-layer prediction architecture. However, the number hidden dimensions varied from the GRAPES methodology. The GRAPES methodology performs competitively compared to the standard message passing models for the Cora and CiteSeer dataset. For the Cora dataset, GRAPES had the best average AUC while for Cora it had the second best average AUC. This state-of-the-art performance shows that the GFLowNet sampling method's ability to adapt to features of the graph remains robust even in link prediction tasks.

Similar to the variances in classification tasks, we note that the variances in the link prediction tasks are also low for GRAPES compared to other frameworks. This further highlights the strength of the GRAPES as it maintains consistency over the randomness presented in different trials.

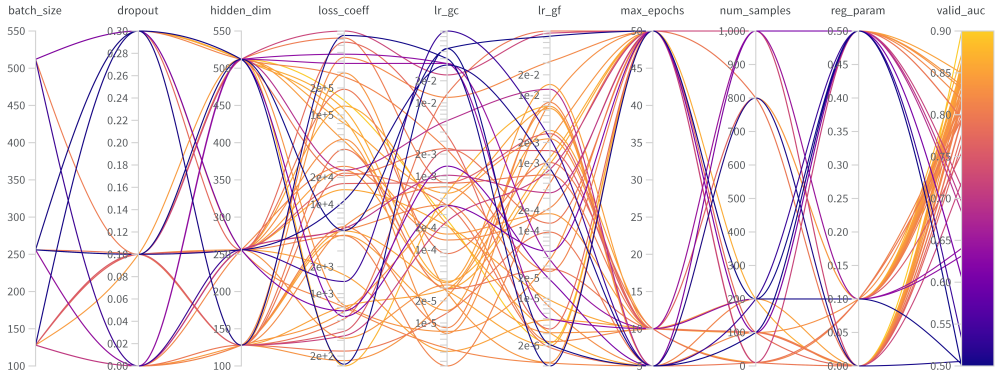
Model	CiteSeer	Cora
GCN	89.14 ± 1.20	87.89 ± 1.48
GraphSAGE	85.96 ± 2.04	84.05 ± 1.72
GIN	68.74 ± 2.74	69.63 ± 2.77
GRAPES	85.18 ± 1.50	88.31 ± 0.58

Table 2: Average AUC Score across 10 runs

One major advantage of GRAPES on node classification is its ability to perform comparably well on both small and large sample sizes. To test this, we ran the model with varying sizes of sample on the CiteSeer dataset over a course of 5 runs. We find that the GRAPES model maintains consistency in performance over small and large sample sizes as the variation in average validation AUC is $\pm$ 0.14 with the max AUC being 83.29 and the min AUC being 69.51. While comparison to baseline was not available during analysis, based on the baseline performances with fixed sample size (Table 2) we see that the AUCs fall within the average AUC range despite having varying sample sizes.



(a) Cora



(b) CiteSeer

Figure 1: Hyperparameter tuning of CiteSeer and Cora

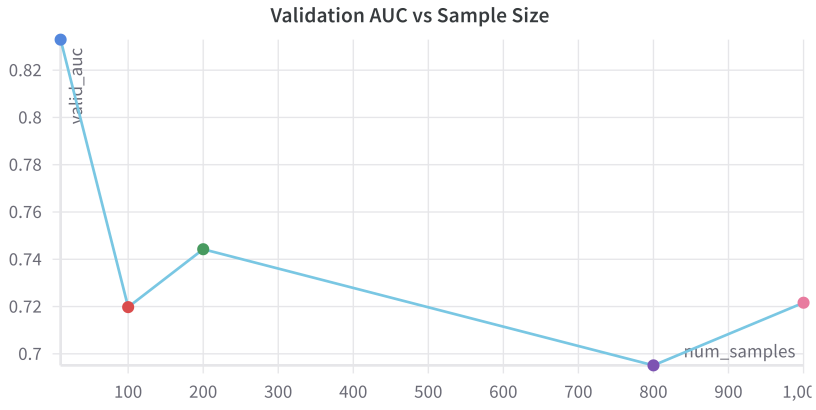


Figure 2: CiteSeer Dataset Sample Size Variations

Conclusion Through this paper, we sought to assess the GRAPES framework beyond classification by expanding its application to link prediction. By using adaptive graph sampling, the GRAPES model unites the sampling process with GNN computation. By doing so, synchronized training is possible while accounting for node features learning, graph structure, sample size, and other contextual factors. Similar to the state-of-the-art results found on the node classification tasks, GRAPES performs well on link prediction as well. Our experiments demonstrate that GRAPES is able to maintain better than state-of-the-art results, even across varying sample sizes.

Limitations: While it is preferable to have uniform evaluation protocols, our baseline comparisons had varying regularization techniques and model hyperparameters. We did however, maintain similar model structure and testing/training splits. To further the robustness of results, we suggest standardize evaluation protocols in future iterations. Additionally, we also suggest expanding application on datasets outside of citation networks as different types of graph structures may yield varying results.

Future Steps: So far, GRAPES has been evaluated on node classification and link prediction tasks. However, it is not limited by just these two tasks. Because GFlowNets assume access to a tractable reward function (Bengio et al. [2023]), other tasks like unsupervised representation learning can also be performed. GRAPES expansion on other tasks is something we hope to see in future iterations. Furthermore, like in the orignal paper, we assumed that the normalizing function $Z(s_0)$ in Equation 3 is constant across different mini-batches. Estimating this is something we hope to see in future iterations.

Reproducibility Statement. Our code is publicly available at https://github.com/vidhinrp/CPSC583. The full set of hyperparameters used for GRAPES are available in the repository.

References

Lada A Adamic and Eytan Adar. Friends and neighbors on the web. *Social networks*, 25(3):211–230, 2003. ISSN 0378-8733.

Emmanuel Bengio, Moksh Jain, Maksym Korablyov, Doina Precup, and Yoshua Bengio. Flow network based generative models for non-iterative diverse candidate generation. *Advances in Neural Information Processing Systems*, 34:27381–27394, 2021.

Yoshua Bengio, Salem Lahlou, Tristan Deleu, Edward J Hu, Mo Tiwari, and Emmanuel Bengio. Gflownet foundations. *Journal of Machine Learning Research*, 24(210):1–55, 2023.

Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with pytorch geometric. *arXiv preprint arXiv:1903.02428*, 2019.

Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864, 2016.

Moksh Jain, Emmanuel Bengio, Alex Hernandez-Garcia, Jarrid Rector-Brooks, Bonaventure FP Dossou, Chanakya Ajit Ekbote, Jie Fu, Tianyu Zhang, Michael Kilgour, and Dinghuai Zhang. Biological sequence design with gflownets. In *International Conference on Machine Learning*, pages 9786–9801. PMLR, 2022. ISBN 2640-3498.

Hisashi Kashima, Tsuyoshi Kato, Yoshihiro Yamanishi, Masashi Sugiyama, and Koji Tsuda. Link propagation: A fast semi-supervised learning algorithm for link prediction. In *Proceedings of the 2009 SIAM international conference on data mining*, pages 1100–1111. SIAM, 2009.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *International Conference on Learning Representations (ICLR)*, 2017.

Wenqian Li, Yinchuan Li, Zhigang Li, Jianye Hao, and Yan Pang. Dag matters! gflownets enhanced explainer for graph neural networks. *arXiv preprint arXiv:2303.02448*, 2023.

Paul Louis, Shweta Ann Jacob, and Amirali Salehi-Abari. Sampling enclosing subgraphs for link prediction. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 4269–4273, 2022.

Kanika Madan, Jarrid Rector-Brooks, Maksym Korablyov, Emmanuel Bengio, Moksh Jain, Andrei Cristian Nica, Tom Bosc, Yoshua Bengio, and Nikolay Malkin. Learning gflownets from partial episodes for improved convergence and stability. In *International Conference on Machine Learning*, pages 23467–23483. PMLR, 2023. ISBN 2640-3498.

Nikolay Malkin, Moksh Jain, Emmanuel Bengio, Chen Sun, and Yoshua Bengio. Trajectory balance: Improved credit assignment in gflownets. *Advances in Neural Information Processing Systems*, 35: 5955–5967, 2022.

Mizu Nishikawa-Toomey, Tristan Deleu, Jithendaraa Subramanian, Yoshua Bengio, and Laurent Charlin. Bayesian learning of causal structure and mechanisms with gflownets and variational bayes. *arXiv preprint arXiv:2211.02763*, 2022.

A Salehi-Abari P Louis, SA Jacob. Simplifying subgraph representation learning for scalable link prediction. *preprint*, 2023.

Ling Pan, Moksh Jain, Kanika Madan, and Yoshua Bengio. Pre-training and fine-tuning generative flow networks. *arXiv preprint arXiv:2310.03419*, 2023.

Rudy Raymond and Hisashi Kashima. Fast and scalable algorithms for semi-supervised link prediction on static and dynamic graphs. In *Joint european conference on machine learning and knowledge discovery in databases*, pages 131–147. Springer, 2010.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

Peng Wang, BaoWen Xu, YuRong Wu, and XiaoYu Zhou. Link prediction in social networks: the state-of-the-art. *arXiv preprint arXiv:1411.5118*, 2014.

Taraneh Younesian, Thiviyan Thanapalasingam, Emile van Krieken, Daniel Daza, and Peter Bloem. Grapes: Learning to sample graphs for scalable graph neural networks. *arXiv preprint arXiv:2310.03399*, 2023.

Dinghuai Zhang, Nikolay Malkin, Zhen Liu, Alexandra Volokhova, Aaron Courville, and Yoshua Bengio. Generative flow networks for discrete probabilistic modeling. In *International Conference on Machine Learning*, pages 26412–26428. PMLR, 2022. ISBN 2640-3498.

Heiko Zimmermann, Fredrik Lindsten, Jan-Willem van de Meent, and Christian A Naesseth. A variational perspective on generative flow networks. *arXiv preprint arXiv:2210.07992*, 2022.