Training Heterogeneous Graph Neural Networks using Bandit Sampling

Ta-Yang Wang
University of Southern California
Los Angeles, USA
tayangwa@usc.edu

Rajgopal Kannan
DEVCOM Army Research Lab
Los Angeles, USA
rajgopal.kannan.civ@army.mil

Viktor Prasanna
University of Southern California
Los Angeles, USA
prasanna@usc.edu

ABSTRACT

Graph neural networks (GNNs) have gained significant attention across diverse areas due to their superior performance in learning graph representations. While GNNs exhibit superior performance compared to other methods, they are primarily designed for homogeneous graphs, where all nodes and edges are of the same type. Training a GNN model for large-scale graphs incurs high computation and storage costs, especially when considering the heterogeneous structural information of each node. To address the demand for efficient GNN training, various sampling methods have been proposed. In this paper, we propose a sampling method based on bandit sampling, an online learning algorithm with provable convergence under weak assumptions on the learning objective. To the best of our knowledge, this is the first bandit-based sampling method applied to heterogeneous GNNs with a theoretical guarantee. The main idea is to prioritize node types with more informative connections with respect to the learning objective. Compared with existing techniques for GNN training on heterogeneous graphs, extensive experiments using the Open Academic Graph (OAG) dataset demonstrate that our proposed method outperforms the state-of-the-art in terms of the runtime across various tasks with a speed-up of 1.5-2x, while achieving similar accuracy.

CCS CONCEPTS

• **Computing methodologies** → **Artificial intelligence**.

KEYWORDS

Heterogeneous graphs, Graph neural networks, Graph convolutional networks, Sampling method, Representation Learning, Multi-armed Bandit

ACM Reference Format:
Ta-Yang Wang, Rajgopal Kannan, and Viktor Prasanna. 2023. Training Heterogeneous Graph Neural Networks using Bandit Sampling. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM '23), October 21–25, 2023, Birmingham, United Kingdom.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3583780.3615276

1 INTRODUCTION

Heterogeneous graphs are extensively employed to model complex network systems, where objects are involved in distinct interactions. For example, the Open Academic Graph (OAG) [17] consists of five types of nodes: papers, authors, institutions, venues, and fields, as well as different types of relationships between them.

GNNs employ deep neural networks to aggregate feature information from neighboring nodes, which enhances the power of the aggregated embedding. There have been several attempts to apply GNNs to heterogeneous networks [5, 10, 16]. However, computing the loss across a mini-batch of nodes and the corresponding mini-batch gradients is extremely expensive due to the neighbor explosion problem. This problem arises because the embedding of a node at the current message passing layer recursively depends on the embeddings of its neighbors at the previous layer. As a result, the complexity grows exponentially with the number of message passing layers.

To address the neighbor explosion problem, various sampling techniques have been proposed. In particular, one approach aims to reduce the number of nodes involved in message passing for Graph Convolutional Networks (GCNs). This approach outperforms many graph deep learning models in several graph-based tasks. Example techniques include node-wise sampling [7, 14] and subgraph sampling [6, 15]. Despite the empirical success of these sampling methods, recent studies have shown that the use of inaccurate mini-batch gradients hampers the convergence of GCNs [12, 18]. Additionally, existing approaches lack asymptotic convergence guarantees on the sampling variance, limiting the utilization of the embedding.

In this paper, we propose a multi-armed bandit (MAB)-based sampling method with a convergence guarantee. Specifically, we initially focus on exploration during the early iterations, assigning equal importance to all node types. Then, based on the feedback collected through updates during training, the algorithm transitions to exploitation. Our objective is to minimize the variance of the stochastic gradient, as it is the primary bottleneck for convergence speed. By prioritizing more informative node types according to the learning objective, the bandit sampler can efficiently select node types that guide optimization towards the optimal solution. We also demonstrate that the MAB-based sampling method guarantees that the accumulated gradient variance approaches the optimal distribution within a constant factor, under practical assumptions.

The main contributions of this paper are:

- We present the first MAB-based sampling method with provable convergence for heterogeneous GNNs.
- Experiments on the large-scale benchmark tasks demonstrate that our proposed method significantly outperforms state-of-the-art sampling methods in terms of training time.

The rest of this paper is organized as follows: Section 2 provides an overview of heterogeneous graph mining and sampling methods for GNNs. Section 3 presents the background of a GNN model and

CIKM '23, October 21–25, 2023, Birmingham, United Kingdom

ACM ISBN 979-8-4007-0124-5/23/10.
https://doi.org/10.1145/3583780.3615276

explains the sampling process in GCN training. Section 4 defines the heterogeneous sampling problem for efficient training. Section 5 presents a detailed description of our proposed algorithm for fast sampling in heterogeneous GNNs. Section 6 presents numerical results that validate the performance of the proposed algorithm. Finally, Section 7 concludes the paper.

2 RELATED WORK

In this section, we discuss some literature related to heterogeneous graph mining and sampling methods for graph neural networks.

Heterogeneous graph mining. In recent years, numerous studies have focused on investigating heterogeneous graphs for various applications, such as personalized recommendation [9, 13]. For example, Zhang et al. [16] proposed a heterogeneous graph neural network model to handle the issue of the structural information in heterogeneous graphs and attributes or contents correlated to each node. Hu et al. [10] designed a heterogeneous mini-batch graph sampling method to train web-scale heterogeneous graph efficiently.

Subgraph-wise Sampling Methods. Subgraph-wise sampling methods involve sampling a mini-batch and constructing the same subgraph at different message passing layers. GraphSAINT [15] and Cluster-GCN [6] construct the subgraph induced by the sampled mini-batch, utilizing graph clustering methods to encourage connections between sampled nodes.

Node-wise and Layer-wise Sampling Methods. Both node-wise and layer-wise sampling methods recursively sample neighbors over message passing layers and then construct different computation graphs for each layer. Node-wise sampling methods [4, 7] aggregate messages from a subset of sampled neighbor nodes at each layer to alleviate the exponentially growing computation. Layer-wise sampling [19] sample nodes for each message passing layer independently and then reduce variance by using importance sampling, so that the sample size in each layer remains constant.

Bandit Sampling Methods. Bandit sampling methods [12, 18] have explored the application of bandit algorithms to sample neighboring nodes during the aggregation, which takes the sum of the neighbor embeddings. Liu et al. [12] propose a novel formulation of neighbor sampling as multi-armed bandit problem (MAB) and apply EXP3 [1] and its variants to update sampler and reduce variance. They provide an asymptotic regret analysis on sampling variance, which show that the regret of their estimator, BanditSampler, approximates the optimal sampler within a factor of 3. Zhang et al. [18] propose a numerically-stable reward function that trades bias with variance, which enables the connection to sampling approximation error.

3 BACKGROUND

In the following section, we introduce the formal notations that define our problem setting and provide an overview of graph neural networks, as well as the framework of their sampling problem.

3.1 Basic Notations

A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is defined by a set of nodes $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ and a set of edges $\mathcal{E}$ among these nodes. Let $(v_i, v_j) \in \mathcal{E}$ denote an edge going from node $v_i \in V$ to node $v_j \in \mathcal{V}$, denote $N(v_i) = \{v_i \in \mathcal{V} \mid (v_i, v_j) \in \mathcal{E}\}$ as the neighborhood of node v_i. Assume that $\mathcal{G}$ is undirected, that is, $v_j \in N(v_i)$ if and only if $v_i \in N(v_j)$. Let $N(T) = \{v \in \mathcal{V} \mid (v_i, v_j) \in \mathcal{E}, v_i \in T\}$ denote the neighborhoods of a set of nodes S. $[L]$ denotes $\{1, \ldots, L\}$ for a positive integer L.

3.2 Graph Neural Networks

Formally, given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, the forward propagation of a GNN is formulated as

$$\mathbf{h}_{v,t}^{(\ell+1)} = \sigma\left(\sum_{i \in N(v)} a_{vi} \mathbf{h}_{i,t}^{(\ell)} W_t^{(\ell)}\right) \tag{1}$$

for the node $v \in \mathcal{V}$ at training iteration t. Here $\mathbf{h}_{i,t}^{(\ell)} \in \mathbb{R}^d$ is the hidden embedding of node i at the layer ℓ, $\mathbf{h}_{i,t}^{(0)} = \mathbf{x}_i$ is the node feature, and $\sigma(\cdot)$ is the activation function. $a_{vi} > 0$ is the edge weight between node v and i. $W_t^{(\ell)} \in \mathbb{R}^{d \times d}$ is the GNN weight matrix, learned by minimizing the stochastic loss $\hat{\mathcal{L}}$ with SGD. Finally, we denote $\mathbf{z}_{i,t}^{(\ell)} = a_{vi} \mathbf{h}_{it}^{(\ell)}$ as the weighted embedding, $[D_v] = \{i \mid 1 \le i \le D_v\}$. For a vector $x \in \mathbb{R}^{d_0}$, we refer to its 2-norm as $\|x\|$; for a matrix W, we refer to its spectral norm as $\|W\|$.

Sampling in the training of GCN can be formulated as follows:

$$SN^{(k)}(v) = Sampling^{(k)}(N(v)) \tag{2}$$

$$a_v^{(k)} = Aggregate^{(k)}\left(\{h_u^{k-1} : u \in SN(v)\}\right) \tag{3}$$

$$h_v^{(k)} = Combine^{(k)}\left(h_v^{(k-1)}, a_v^{(k)}\right), \tag{4}$$

where $SN(v)$ is the sampled neighbors from $N(v)$, $a_v^{(k)}$ is the aggregation feature vector of node v in the k-th layer, $h_v^{(k)}$ is the representation feature of node v in the k-th layer.

4 PROBLEM DEFINITION

In this section, we define the problem of heterogeneous sampling and present a problem formulation based on the cluster selection problem.

Given a heterogeneous graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, O_{\mathcal{V}}, \mathcal{R}_{\mathcal{E}})$, where $O_{\mathcal{V}}$ represents node type that corresponds to nodes in $\mathcal{V}$, and $\mathcal{R}_{\mathcal{E}}$ represents edge types that correspond to edges in $\mathcal{E}$. The heterogeneity introduces a complex scenario where certain types of nodes might have significantly more neighbors compared to others. Therefore, it becomes critical to distinguish between different types of nodes and compute their effects due to the challenge posed by imbalanced number of neighbors in different types.

We assume that different types of nodes and edges share the same feature and representation space. In [11], a general form of heterogeneous sampling methods can be formulated as follows:

$$F(v) = Effect(\mathcal{N}(v), O_{\mathcal{N}(v)}, \mathcal{E}(v), \mathcal{R}_{\mathcal{E}(v)}) \tag{5}$$

$$SN^{(k)}(v) = Sampling^{(k)}(F(v), N(v), B^{(k)}), \tag{6}$$

Here, $O_{\mathcal{N}(v)}$ and $\mathcal{R}_{\mathcal{E}(v)}$ denote sets that consist of node types and edge types, respectively. $F(v)$ is a set that stores the effect of different types of neighbors on node v. $SN(v)$ denotes the sampled neighbors from $N(v)$ and B is a restrict factor in guaranteeing a balanced distribution of different types of neighbors. We will focus on the sampling part based on the pre-computed effect $F(v)$.

Inspired by the cluster sampling based approach [2, 6], we can describe our problem as follows: Suppose $\mathcal{V} = \mathcal{V}_1 \cup \mathcal{V}_2 \cup \cdots \cup \mathcal{V}_C$ for a graph with $C = |\mathcal{O}_\mathcal{V}|$ types. Each $\mathcal{V}_j$ is used to construct a subgraph $\mathcal{H}_j(\mathcal{V}_j, \mathcal{E}_j)$, where $\mathcal{E}_j$ is the set of edges that connect only the nodes in $\mathcal{V}_j$, for any $j \in [C]$. Our goal is to provide a selection strategy for these subgraphs to improve the training time of the model.

5 METHODOLOGY

In this section, we describe our proposed method and the corresponding Bandit-based sampling scheme.

The objective is to learn the graph neural network parameters θ. Let $\mathcal{L}_\theta^{\mathcal{G}}$ be the loss function of the graph $\mathcal{G}$ and $\mathcal{L}_\theta^{(u,v)}$ is the objective function at a given directed edge (u, v). We employ the cluster sampling based approach [2, 6]: the nodes are partitioned into several clusters $\mathcal{V}_1, \ldots, \mathcal{V}_C$, where each cluster $\mathcal{V}_j$ is used to construct a subgraph $\mathcal{H}_j$ such that the set of edges connect only the nodes in the same cluster. Consider the adjacency matrix of a new graph that consists of these subgraphs, where each diagonal block represents the adjacency matrix of the subgraph of each cluster. We can now decompose the loss function as the sum of different loss functions at each edge of each of the clusters:

$$\mathcal{L}_\theta^G = \sum_{j=1}^{C} \mathcal{L}_\theta^{\mathcal{H}_j} = \sum_{j=1}^{C} \sum_{(u,v)\in\mathcal{H}_j} \mathcal{L}_\theta^{(u,v)}. \tag{7}$$

Note that the optimal policy of reducing the variance is intractable [2, 18], since it requires computing all the gradients. Therefore, we propose a multi-armed bandit approach to select the types by combining exploration and exploitation. We only need to consider the gradient computation used for the gradient update in each round. In each iteration, we choose one of the subgraphs and calculate the embeddings of the nodes within it. The encoder parameters are then updated based on the gradient computations performed on the selected subgraph. We define the bandit sampler where the regret is proportional to $\left\|\hat{\nabla}\mathcal{L}^{(\mathcal{H}_{j_t})}(\theta_t)\right\|_2$.

The high-level implementation of our sampling algorithm proceeds as follows: first we initialize equal weights to all clusters. At each time $t = 1, \ldots, T$, the algorithm samples a cluster $\mathcal{H}$ and several nodes according to the corresponding distribution. The weight of $\mathcal{H}$ will be updated by using a variant of the EXP3 algorithm [1]. The weight of the other clusters that are not selected will not change. The sampling distribution is designed based on the weighted average between the estimation and the uniform distribution. Our algorithm can be outlined as follows: At each time $t = 1, \ldots, T$,

(1) Select a cluster $\mathcal{H}_{j_t} \sim p_t$.
(2) Calculate the reward $r^{(\mathcal{H}_{j_t})}$.
(3) Update the weight $w_{t+1}(j) = w_t(j) \cdot \exp\left(\frac{\eta r^{(\mathcal{H}_{j_t})}}{p_t(j)}\right)$.
(4) Compute the distribution

$$p_t(\mathcal{H}_j) = (1-\gamma)\frac{w_{t+1}(j)}{\sum_i w_{t+1}(i)} + \gamma/n$$

.

for some learning rate $\eta > 0$ and $\gamma \in (0, 1]$.

During the exploitation phase, the above algorithm prioritizes arms with higher weights, followed by uniform random arm selection. The weights are updated after receiving the regret feedback. The use of exponential growth function can lead to an increase the weight of good arms.

5.1 Theoretical Analysis

In this section, we present a theoretical analysis of our bandit estimator and its development process. We also provide a theoretical evaluation of its performance and compare it to the optimal sampling scheme. Since the optimal sampler involves computations among all the neighbors, it is crucial to approximate the optimal sampling distribution without hindering the convergence. Gradient variance reduction methods have become increasingly popular in stochastic optimization frameworks [3, 8, 12]. Our goal is to minimize the accumulated variance based on the sampling algorithm. Following the analysis in [2], define the cost at each round t as follows:

$$\mathbb{E}_{p_t}\left[\left\|\hat{\nabla}\mathcal{L}^{(\mathcal{H}_j)}(\theta_t)\right\|_2^2\right] = \sum_j \frac{\left\|\hat{\nabla}\mathcal{L}^{(\mathcal{H}_j)}(\theta_t)\right\|_2^2}{p(\mathcal{H}_j)}. \tag{8}$$

At each round t, the optimal sampling strategy is to select a cluster with probabilities proportional to:

$$p_t^*(\mathcal{H}_j) \propto \left\|\hat{\nabla}\mathcal{L}^{(\mathcal{H}_j)}(\theta_t)\right\| \tag{9}$$

where

$$p_t^* = \arg\min \mathbb{E}_p\left[\left\|\hat{\nabla}\mathcal{L}^{(\mathcal{H}_j)}(\theta_t)\right\|_2^2\right] \tag{10}$$

This can be derived by looking at the Lagrangian expression of the optimization problem. To minimize the performance gap between the designed distribution and the optimal one, the regret can be defined as:

$$\left\langle p_t - p^*, \nabla_{p_t}\mathbb{E}\left[\left\|\hat{\nabla}L^{(H_j)}(\theta_t)\right\|_2^2\right]\right\rangle, \tag{11}$$

which can be bounded by the optimal distribution within a factor of 3, similar to the work by [12]. Due to page limitations, we present a proof sketch: Note that the expectation term is convex in p_t. To minimize the performance gap between our proposed distribution and the optimal one, we assume that the gradient is Lipschitz continuous, which enables us to bound the accumulated gradient variance under the bandit distribution. This observation follows from the multiplicative weight update algorithms for EXP3 regret [3].

6 EXPERIMENTS

In this section, we evaluate the MAB-based sampling method proposed in Section 5 on the Open Academic Graph (OAG), the largest publicly available heterogeneous dataset. We perform Paper-Field prediction, Paper-Venue prediction, Author Disambiguation tasks. All the experiments for the three tasks are evaluated in terms of Mean Reciprocal Rank (MRR). All baselines are implemented via PyTorch.

Table 1: Open Academic Graph (OAG) Statistics

Dataset	#nodes	#edges	#papers (P)	#authors (A)	#fields (F)	#venues (V)	#institutes (I)	#P-A	#P-F	#P-V	#A-I	#P-P
OAG	178,663,927	2,236,196,802	89,606,257	88,364,081	615,228	53,073	25,288	300,853,688	657,049,405	89,606,258	167,449,933	1,021,237,518

6.1 Datasets

Open Academic Graph (OAG) [17] consists of more than 178 million nodes and 2.236 billion edges, which is far more distinguishable than widely-adopted small citation graphs in the GNN literature. We summarize the statistics in Table 1, where P-A, P-F, P-V, A-I, and P-P represent the edges between paper and author, paper and field, paper and venue, author and institute, and the citation links between two papers.

6.2 Tasks and Evaluation

We evaluate our sampling method on three distinct real-world downstream tasks: Paper-Field prediction, Paper-Venue prediction, and Author Disambiguation. The objective of the first two tasks is to predict the fields to which each paper belongs to, or the venue to which each paper is published at, respectively. For author disambiguation, we gather all authors sharing identical names and their associated papers. The goal is to conduct link prediction between these papers and candidate authors. After obtaining the representations from GNNs, we leverage a Neural Tensor Network to compute the probability of each author-paper pair to be linked.

For all three tasks, we utilize papers that were published prior to the 2015 as the training set, papers between 2015 and 2016 as the validation set, and papers between 2016 and 2019 as testing. The main purpose of this paper is to conduct a comparative analysis between our samplers and the existing training algorithms. We aim to evaluate the distinct impacts of our samplers in contrast to other GNN models.

6.3 Baselines

We compare our MAB-based sampling method proposed in Section 5 with several state-of-the-art GNNs, including homogeneous GCN [7] and heterogeneous GNNs — HetGNN [16] and HGT [10]. We summarize the baselines as follows: HetGNN proposes a heterogeneous sampling method based on random walk with restart (RWR). HGT leverages heterogeneous graphs' meta-relations to parameterize weight matrices for several critical steps: heterogeneous mutual attention, heterogeneous message passing, and target specific aggregation. Following the implementation details as in HGT [10], we set the hidden dimension for the neural networks to be 256 for all baselines and the head number as 8. For each model, we train it for 200 epoches, the one with the lowest validation loss is selected as the model. We adhere to the default parameters as in the literature without hyperparameter tuning.

6.4 Results

We summarize the experimental results of the proposed method and baselines in Table 2. It shows that the proposed MAB-based sampling method significantly outperforms the homogeneous baseline for all the tasks and stays on par with the heterogeneous baselines. Moreover, our method is faster than all baselines. It suggests that the bandit sampling speed-up the whole training process.

Table 2: Experimental results of different methods on Open Academic Graph

GNN Models	GCN	HetGNN	HGT	This paper
Paper-Field	.527	.601	.636	.647
Paper-Venue	.262	.255	.323	.311
Author Disambiguation	.724	.802	.827	.793

Table 3 reports the runtime to reach the performance in Table 2. We evaluate the time taken by our proposed method in comparison to all the baselines. Since our proposed method possesses the capability to achieve faster convergence, it is significantly faster than both HetGNN and HGT, it achieves a with a speed-up of 1.5x for Paper-Venue.

Table 3: Runtime (mins) of different methods on Open Academic Graph

GNN Models	GCN	HetGNN	HGT	This paper
Paper-Field	281	354	398	279
Paper-Venue	370	516	427	361
Author Disambiguation	91	173	115	85

In summary, our proposed method outperforms the state-of-the-art baselines — both HetGNN and HGT — in terms of the runtime on various tasks, including Paper-Field, Paper-Venue, and Author Disambiguation, while achieving good performance. This empirically show that our sampling algorithm could converge to better results faster than existing baselines.

7 CONCLUSION

In this paper, we proposed a multi-armed bandit (MAB)-based framework with a convergence guarantee to model the neighbor sampling process in heterogeneous GNNs. We showed that our MAB-based sampling method guarantees that the accumulated gradient variance approaches the optimal distribution within a constant factor under practical assumptions. To the best of our knowledge, our MAB-based sampling method is the first bandit sampling method for heterogeneous GNNs with provable convergence. Experiments on the OAG dataset demonstrate that our proposed method significantly outperforms state-of-the-art sampling methods in terms of training time.

ACKNOWLEDGMENTS

This work has been supported by the U.S. National Science Foundation (NSF) under grant OAC-2209563 and CNS-2009057, as well as the DEVCOM Army Research Lab (ARL) under grant W911NF2220159. Distribution Statement A: Approved for public release. Distribution is unlimited.

REFERENCES

[1] Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. 2002. The nonstochastic multiarmed bandit problem. *SIAM journal on computing* 32, 1 (2002), 48–77.

[2] Ghadir Ayache, Thomas Hugues, Chris Xu, Julia Zhang, and Diane Hu. 2022. Adaptive Bandit Cluster Selection for Graph Neural Networks. In *2022 56th Asilomar Conference on Signals, Systems, and Computers*. IEEE, 1385–1392.

[3] Cenk Baykal, Vamsi K Potluru, Sameena Shah, and Manuela M Veloso. 2022. Bandit Sampling for Multiplex Networks. *arXiv preprint arXiv:2202.03621* (2022).

[4] Jie Chen, Tengfei Ma, and Cao Xiao. 2018. Fastgcn: fast learning with graph convolutional networks via importance sampling. *arXiv preprint arXiv:1801.10247* (2018).

[5] Xia Chen, Guoxian Yu, Jun Wang, Carlotta Domeniconi, Zhao Li, and Xiangliang Zhang. 2019. Activehne: Active heterogeneous network embedding. *arXiv preprint arXiv:1905.05659* (2019).

[6] Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. 2019. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 257–266.

[7] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems* 30 (2017).

[8] Samuel Horváth and Peter Richtárik. 2019. Nonconvex variance reduced optimization with arbitrary sampling. In *International Conference on Machine Learning*. PMLR, 2781–2789.

[9] Binbin Hu, Chuan Shi, Wayne Xin Zhao, and Philip S Yu. 2018. Leveraging meta-path based context for top-n recommendation with a neural co-attention model. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 1531–1540.

[10] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. 2020. Heterogeneous graph transformer. In *Proceedings of the web conference 2020*. 2704–2710.

[11] Xin Liu, Mingyu Yan, Lei Deng, Guoqi Li, Xiaochun Ye, and Dongrui Fan. 2021. Sampling methods for efficient training of graph convolutional networks: A survey. *IEEE/CAA Journal of Automatica Sinica* 9, 2 (2021), 205–234.

[12] Ziqi Liu, Zhengwei Wu, Zhiqiang Zhang, Jun Zhou, Shuang Yang, Le Song, and Yuan Qi. 2020. Bandit samplers for training graph neural networks. *Advances in Neural Information Processing Systems* 33 (2020), 6878–6888.

[13] Xiang Ren, Jialu Liu, Xiao Yu, Urvashi Khandelwal, Quanquan Gu, Lidan Wang, and Jiawei Han. 2014. Cluscite: Effective citation recommendation by information network-based clustering. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 821–830.

[14] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 974–983.

[15] Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna. 2019. Graphsaint: Graph sampling based inductive learning method. *arXiv preprint arXiv:1907.04931* (2019).

[16] Chuxu Zhang, Dongjin Song, Chao Huang, Ananthram Swami, and Nitesh V Chawla. 2019. Heterogeneous graph neural network. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 793–803.

[17] Fanjin Zhang, Xiao Liu, Jie Tang, Yuxiao Dong, Peiran Yao, Jie Zhang, Xiaotao Gu, Yan Wang, Bin Shao, Rui Li, et al. 2019. Oag: Toward linking large-scale heterogeneous entity graphs. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2585–2595.

[18] Qingru Zhang, David Wipf, Quan Gan, and Le Song. 2021. A biased graph neural network sampler with near-optimal regret. *Advances in Neural Information Processing Systems* 34 (2021), 8833–8844.

[19] Difan Zou, Ziniu Hu, Yewen Wang, Song Jiang, Yizhou Sun, and Quanquan Gu. 2019. Layer-dependent importance sampling for training deep and large graph convolutional networks. *Advances in neural information processing systems* 32 (2019).