Effectiveness of Sampling Strategies for One-shot Active Learning from Relational Data

Ragib Ahsan
rahsan3@uic.edu
University of Illinois at Chicago

Elena Zheleva
ezheleva@uic.edu
University of Illinois at Chicago

ABSTRACT

Relational classification exploits structural information in network data to improve predictive performance. However, the large sizes of real-world networks cause two main scalability issues for relational classification. First, training supervised models on large networks is computationally expensive. Second, label acquisition for large samples can be costly and unrealistic. The goal of active learning is to query informative labels and reduce labeling cost. However, state-of-the-art active learning strategies require multiple iterations of learning, in order to pick the best labels at each iteration, which incurs higher computational cost. In this work, we focus on a constrained version of the problem, named *one-shot active learning* where the active learner has to decide which nodes to sample in one shot, rather than iteratively. We consider several simple and network-based sampling strategies as potential solutions to this problem and propose a novel sampling method (WLS) based on Weisfeiler-Lehman graph labeling algorithm. In our experiments, we show a comprehensive evaluation of eleven different sampling methods on four real world network datasets using four relational classifiers (wvRN, ICA, SGC, GraphSage), offering the first comparison between collective classification and neural network approaches for one-shot active learning. Our sampling method (WLS) shows the strongest performance on average across classifiers and datasets. It performs particularly well with GNN-based classifiers whereas edge-based sampling performs best with wvRN and ICA. We also show that some of the computationally cheaper one-shot active learning approaches can achieve comparable Micro-F1 scores to existing active learning methods that require multiple iterations.

ACM Reference Format:
Ragib Ahsan and Elena Zheleva. 2020. Effectiveness of Sampling Strategies for One-shot Active Learning from Relational Data. In *Proceedings of MLG'20: 16th International workshop on Mining and Learning in Graphs (MLG'20).* ACM, San Diego, CA, USA, 8 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

One of the main factors for the success of relational classification is the ability to harness the properties of relational structure in data. Real-world networks have millions of nodes. Often, acquiring labels for a large sample can be difficult and sometimes unrealistic. A standard and practical solution to this problem is active learning which allows the learner itself to select samples to be labeled by an oracle [31]. State-of-the-art active learning strategies repeatedly selects batch of samples in multiple iterations until a prespecified budget of labels is reached. ALFNET [7], RAL [18] and ANRMAB [12] are some examples of such active learning approaches for relational data. However, these strategies generally learn a model at each iteration, in order to compute utility scores for all the unlabeled samples over all iterations which incurs a substantial computational cost. In order to address this issue for large networks we consider a constrained problem setup where the active learner is allowed only a single iteration to select samples to label. We refer to it as *one-shot active learning*.

A popular approach to active learning is carefully selecting a representative sample of the source data. Prior works have empirically evaluated the effectiveness of sampling methods for one-shot active learning in the context of relational classification [4, 6]. However, the main difference between previous works with ours is twofold. First, they consider only the labeled subgraph for classification whereas we use the full source graph which gives us the opportunity to utilize the structural properties better. Second, they primarily consider a naive relational classifier, wvRN [23] whereas we compare performance of collective classification and modern neural network based approaches.

In this study we consider a wide variety of sampling strategies to compare their relative performance for one-shot active learning. We consider both graph sampling algorithms as well as sampling strategies specifically designed for semi-supervised node classification [39]. We also propose a sampling approach based on the Weisfeiler-Lehman algorithm which shows promising results in empirical evaluation. Our proposed Weisfeiler-Lehman Sampling (WLS) relies solely on the structural role of nodes for label acquisition decisions. One of its main advantages is that it is computationally efficient and yet harnesses structural information effectively. Our empirical evaluation shows that even though there isn't one sampling method that performs the best consistently across datasets and classifiers, Weisfeiler-Lehman ranks the highest on average.

The main contributions of our paper are:

- We propose a *novel node sampling method*, WLS for one-shot active learning and empirically show that it achieves competitive Micro-F1 scores while being particularly effective on larger graphs with lower clustering coefficient.
- We show *the most comprehensive evaluation* of sampling methods for one-shot active learning, including eleven different sampling methods on four real-world network datasets using four popular relational classifiers: wvRN, ICA, SGC, GraphSage.

MLG'20, August 24, 2020, San Diego, CA
© 2020 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

- We empirically show that some of the computationally cheaper one-shot active learning approaches achieve competitive Micro-F1 scores when compared to the existing *multi-shot* active learning approaches.

2 RELATED WORK

The most common scenario for active learning is the pool-based scenario where a pool of labeled and unlabeled samples are present and the learner can choose from the unlabeled pool to query for labels [31]. An established pool-based active learning algorithm for relational data is ALFNET [7] which is based on disagreement-based active learning [32] using *Iterative Classification Algorithm (ICA)* as relational classifier. Even though it performs well, it is computationally expensive specifically for larger graphs due to the cost of iterative training. Recently proposed neural network-based approaches [8, 12] follow similar expensive procedure of re-training the model over iterations without showing significant improvement over ALFNET. One-shot active learning circumvents this cost of re-training by allowing a single chance to decide which nodes to label. Note that, one-shot active learning is different from one-shot learning [11] or active one-shot learning [37]. One-shot learning refers to learning from one or few samples and active one-shot learning refers to active learning where one or few samples can be labeled in each iteration. In contrast, one-shot active learning refers to active learning with one iteration to label nodes. To the best of our knowledge there is no prior work investigating one-shot active learning on relational data.

Graph sampling algorithms have been studied for a long time. Kolaczyk [17] investigated sample properties from a social science perspective. Other works analyzed the statistical properties of sampled subgraphs and how sampling changes topological network properties [19, 35, 41]. Several studies analyzed representativeness [20], correlations of graph properties [2], biases of topological approaches [24] and impact on A/B testing [5].

A few recent works studied the effectiveness of sampling methods for relational classification [3, 4, 6, 22]. Ahmed et al. [4] provided a comprehensive analysis of a variety of graph sampling methods and their effectiveness on relational classification. They sampled subgraphs from a given source graph based on each of the baseline sampling methods. They evaluated the sampling methods based on accuracy of supervised classification models trained with corresponding subgraphs. Their experiments on four real-world networks show that induced edge sampling [1] produces better accuracy than any other graph sampling methods [3]. In a more recent work, Berton et al. [6] experimentally evaluated effectiveness of centrality-based sampling methods for relational classification. They showed that sampling based on clustering coefficient provides greater accuracy in general. Note that, both these studies considered a supervised classification task and trained the classification model only on the sampled graph. In contrast, our evaluation is based on semi-supervised classification where the full source graph is used in creating the features for training. Moreover, they only considered network-based sampling methods whereas we consider network-based, non-network-based and hybrid methods. We also use state-of-the-art relational classifiers like ICA [21], GCN [16] and GraphSage [15] whereas they used a simple classifier, wvRN [23], which relies on label aggregates and has no learning component. Earlier work by Macskassy [22] is closely related to ours which is motivated to speed up active learning on graph by sampling a small candidate set of nodes using structural properties from which an Empirical Risk Minimization (ERM) [29] method chooses the top candidate to be labeled. However, they also followed the standard active learning procedure of multiple shots for active querying which is costlier than the one-shot active learning we are considering.

One of the sampling methods we propose in our work is based on the Weisfeiler-Lehman algorithm which is a graph labeling algorithm widely used for graph isomorphism testing. One key benefit it offers is the relative representation of the vertices based on their structural roles in the graph. Because of this feature, it has inspired several works in the network domain, especially for graph classification [33], graph embedding [34] and link prediction [42].

3 PRELIMINARIES

3.1 Basic Notations

We consider an undirected graph $G = (\mathbf{V}, \mathbf{E})$ where $\mathbf{V}$ and $\mathbf{E}$ are the set of vertices and edges correspondingly. Each node V_i is associated with a feature vector $\vec{X_i}$ and a corresponding class label Y_i which may be unknown, $V_i = \langle \vec{X_i}, Y_i \rangle$. A set of individual attributes comprises the vector $\vec{X_i} = \langle X_i^1, X_i^2, ..., X_i^p \rangle$ where $1, 2, ..., p$ are feature dimensions. The set of node features for all nodes is denoted by $\mathbf{X} = \{\vec{X_i} | V_i \in V\}$ and the set of class labels for all nodes is denoted by $\mathbf{Y} = \{Y_i | V_i \in V\}$. The domain of class labels Y_i is discrete and the set of possible labels is denoted by $\mathcal{Y} = \{y_1, y_2, ..., y_m\}$. The domain for $\vec{X_i}$ can be either discrete or continuous. An edge $E_{ij} = \langle V_i, V_j \rangle$ represents an explicit link between two nodes V_i and V_j in the network. Let $\mathcal{N}_i$ denote the set of neighboring nodes of V_i, $\mathcal{N}_i = \{V_j | \langle V_i, V_j \rangle \in E\}$.

3.2 Relational classification

In contrast to standard classification tasks where data samples are i.i.d, relational classification deals with interconnected samples. The fundamental idea behind relational classification is to effectively exploit the attribute and label correlations between linked nodes to achieve better accuracy in predicting the labels of individual samples. A trivial relational classifier is wvRN [23] which simply infers class association probability based on strong homophily assumption. It requires no learning, rather it classifies the entities of a relational network based only on the relational structure [23]. Modern relational classifiers can be categorized into two families: 1) Collective classification and 2) Graph neural networks.

Collective classification refers to the combined classification of a set of connected objects [30]. The fundamental assumption is that the label Y_i of a node V_i not only depends on its own node attributes $\vec{X_i}$, but also depends on the labels Y_j and attributes $\vec{X_j}$ of its neighboring nodes in the network. Collective classifiers use a vector-based classifier such as logistic regression which is trained iteratively. It learns the conditional probability $P(Y_i | X_i, aggr(\mathcal{N}_i))$ for estimating node labels. Here, $aggr(\mathcal{N}_i)$ refers to the aggregate of class labels of neighboring nodes $\mathcal{N}_i$. The two most common algorithms for collective classification are Iterative classification algorithm (ICA) [21, 25] and Gibbs Sampling (GS) [13, 30].

Graph neural networks (GNN) emerged with the popularity of deep learning architectures. GNN is inspired by the success of convolutional neural networks (CNN) in computer vision. Most of the GNN models are primarily based on redefined notions of convolution for graph data [40]. These convolutional GNNs can be divided into two main categories: spectral and spatial. The most important difference between these fundamental approaches lies in their treatment of the graph laplacian matrix. Spectral methods utilizes eigen-decomposition of the graph laplacian to extract useful information about the graph structure. Spatial methods treat it as spatial connectivity of nodes [9]. The two most representative algorithms from these two categories are GCN [16] and GraphSage [15].

3.3 Weisfeiler-Lehman Algorithm

The Weisfeiler-Lehman algorithm [36] is a graph labeling algorithm that generates canonical ordering of the vertices of a given graph. The classic Weisfeiler-Lehman algorithm is presented in Algorithm 1. The algorithm starts by assigning the same initial label to all vertices (line 1). For each node, it forms a multiset of labels from its direct neighbors' color labels (line 4). After sorting the elements in the multiset and concatenating it to the node's label, it generates signature strings (line 5-6). These signature strings are then sorted and compressed and used to assign new labels to the nodes (line 8-9). This process continues until the labels have stabilized.

Algorithm 1 Weisfeiler-Lehman Graph Labeling

```
Input: Graph $G = (V, E)$, initial labels $l^0(v) = 1$ for all $v \in V$
Output: Final labels $l(v)$ for all $v \in V$
1:  Let $l(v) = l^0(v)$ for all $v \in V$
2:  while $l(v)$ has not converged do
3:      for each $v \in V$ do
4:          Build a multiset $\{l(v')|v' \in \Gamma(v)\}$ concatenating
            its neighbor's labels
5:          Sort elements in the multiset in ascending order
6:          Concatenate the sorted multiset to $l(v)$ to generate
            a signature string $s(v) = \langle l(v), \{l(v')|v' \in \Gamma(v)\}\rangle$
7:      end for
8:      Sort all of the strings $s(v)$ for all $v$ in ascending order
9:      Map each string $s(v)$ to a new compressed label,
        using a function $f$ such that $f(s(v)) = f(s(w))$ if and
        only if $s(v) = s(w)$.
10: end while
```

The effectiveness of the Weisfeiler-Lehman algorithm has been demonstrated for graph classification [34] and link prediction [42]. Weisfeiler-Lehman is used to encode subgraph property for a given link for link prediction. This subgraph property is then used as input features to a neural network model to predict existence of links, exploiting the ability of Weisfeiler-Lehman to encode relative structural roles of nodes in subgraph [42]. Our work is the first to study the application of Weisfeiler-Lehman encoding in the context of active learning sampling.

4 ONE-SHOT ACTIVE LEARNING FOR RELATIONAL CLASSIFICATION

In this section, we first formulate the problem and then describe the sampling methods we consider as potential solutions.

4.1 Problem Definition

One-shot active learning is a constrained version of active learning problem. The main difference between them is that in case of one-shot active learning the learner can query only once to acquire labels from the oracle.

Problem 1 (One-shot Active Learning for Relational Classification). *Given an undirected graph $G = (\mathbf{V}, \mathbf{E})$, node features $\mathbf{X}$, a labeling budget B, a relational classifier C and a labeling oracle, select a set of nodes of size B to be labeled by the oracle in one shot such that Micro-F1 score of classifier C on unseen data is maximized upon training on the labeled set.*

Note that, the active learning budget B is typically much less than the size of the available pool of unlabeled nodes. In such a scenario the classifier can either exploit the full graph structure or restrict itself to the subgraph induced by the labeled nodes. In this work, we focus on the first option with the availability of the full graph structure, making it a semi-supervised classification problem.

4.2 Sampling for one-shot active learning

Since the initial data has no labels available, smart sampling strategies are key to solving the one-shot active learning problem. The existing sampling methods for relational data can be categorised into three groups: 1) Network sampling methods, 2) Non-network sampling methods and 3) Hybrid sampling methods.

4.2.1 ***Network Sampling Methods.*** This category of sampling methods is the state-of-the-art sampling for relational data. Their effectiveness for preserving the structural properties of networks makes them good candidates for one-shot active learning. These methods can be grouped into four major types:

- **Node Sampling:** This is a standard sampling strategy where the algorithm can sample nodes based on their structural properties (e.g., highest or lowest degree). One of the deficiencies of this sampling strategy is that it doesn't preserve the connectivity of the original graph. We consider two different node sampling methods. NS-DC-H refers to a sampling method which prioritizes nodes with high *degree centrality* [6]. We also consider sampling proposed by Berton et al. [6] which prioritizes nodes with high clustering coefficients (NS-CT-H).
- **Edge Sampling:** This is another standard sampling strategy where one can sample edges instead of nodes. Generally, all nodes incident to the edges are subsequently added to form the induced subgraph. The advantage of this method is that unlike *Node Sampling*, it can preserve the connectivity of original graph better. However, due to the independent selection of edges it fails to preserve clustering properties. We consider random edge sampling (ES-RS) where edges are selected at random.
- **Topological Sampling:** Both *Node Sampling* and *Edge Sampling* methods exhibit shortcomings in preserving the structural properties of the graph. In order to overcome these shortcomings, several topology-based sampling algorithms have been proposed. These algorithms mostly utilize either breadth-first search or random walks over the graph to construct a representative sample [4]. For example, *Snowball Sampling* (SS) selects nodes and edges following a breadth-first search from a randomly selected seed node. It stops when a certain threshold is reached. Another

example is *Forest Fire Sampling* (FFS) which also follows breadth-first search, but only considers a proportion (we consider 70%) of the neighborhood for exploration.

- **Graph Clustering:** Nguyen and Smeulders [27] showed that clustering the data can help improve the performance of an active learning strategy. Inspired by this, we consider sampling based on graph clustering. We choose modularity-based clustering [26] since it is a standard method for community detection in networks and has been used for active learning in the past [7]. We generate modularity-based clusters and then iteratively select random nodes from each cluster until the labeling budget is exhausted. We refer to this sampling method as MS.

4.2.2 *Non-network sampling methods.*

We consider sampling methods that are not based on networks for a comprehensive comparison. The primary reason for choosing this is to evaluate how much the structural information helps in prediction. The trivial choice for this category is random sampling (RS). Moreover, we consider sampling based on k-means clustering over the node features $\mathbf{X}$ in order to utilize the strength of clustering in active learning [7]. We create the k-means clusters for a given k value. Then in each iteration we pick a random node from each cluster until the labeling budget is exhausted. We refer to this method as KMS.

4.2.3 *Hybrid sampling methods.*

Several recent works use an intuitive idea of combining the power of both structural properties and node features. Most of those works follow a standard active learning strategy with multiple iterations. FeatProp proposed by Wu et al. [39] can be considered a hybrid approach for one-shot active learning. It clusters the samples using K-Means clustering based on a distance function derived by both propagated node features and graph structure. Then it iteratively selects the closest nodes to the cluster centers until the active learning budget is exhausted.

4.2.4 *Weisfeiler-Lehman Sampling.*

We propose a new sampling algorithm, *Weisfeiler-Lehman Sampling (WLS)* for one-shot active learning for networks. It is adapted from the Weisfeiler-Lehman node labeling algorithm [36]. WLS considers the structural role of a node as the main predictor for label acquisition. Since Weisfeiler-Lehman method has been proven to be useful for encoding relative structural roles of nodes in link prediction problem [42], we investigate its effectiveness for one-shot active learning.

The basic idea behind WLS is to explore different local neighborhoods of the graph and pick the nodes based on their relative structural roles. In order to achieve this, WLS utilizes the final color labels produced by Weisfeiler-Lehman algorithm. The color labels encode relative neighborhood properties of nodes which plays an important role in isomorphism testing. For any two isomorphic graphs the first nodes exhibit similar structural properties in the corresponding orderings. The key benefit of this process is that the algorithm is able to pick the structural roles specific to each network, so they do not have to be defined apriori. Note that, in our case, a neighborhood is formed around a given seed node. An exploration budget B_e is introduced to limit the number of hops of neighborhood the algorithm can explore from the given seed node.

The WLS algorithm is presented in Algorithm 2. It starts with an empty set of labeled nodes ($\mathcal{L}$) and a set of all nodes $\mathcal{U}$ (line 1).

Algorithm 2 Weisfeiler-Lehman Sampling

```
Input: A network $G = (\mathbf{V}, \mathbf{E})$
Parameter: Batch size $k$, labeling budget $B$, exploration budget $B_e$
Output: Set of labeled nodes $\mathcal{L}$
1:  $\mathcal{L} = \emptyset, \mathcal{U} = \mathbf{V}$
2:  while $|\mathcal{L}| < B$ do
3:      $\mathcal{L}^k = \emptyset$
4:      Pick $k$ random seed nodes, $\mathcal{S}^k$ from $\mathcal{U}$
5:      for each $V_i \in \mathcal{S}^k$ do
6:          $\mathcal{N}_i \leftarrow$ Up to $B_e$ hop neighborhood subgraph of $V_i$
7:          $R_i \leftarrow$ Weisfeiler-Lehman($\mathcal{N}_i$)
8:          $v_i \leftarrow$ top ranked $v \in (\mathcal{N}_i \cap \mathcal{U})$ in $R_i$
9:          $\mathcal{L}^k = \mathcal{L}^k \cup v_i$
10:     end for
11:     $\mathcal{L} = \mathcal{L} \cup \mathcal{L}^k$
12:     $\mathcal{U} = \mathcal{U} \setminus \mathcal{L}^k$
13: end while
14: return $\mathcal{L}$
```

Then the algorithm keeps selecting k informative nodes until the labeling budget B is exhausted (lines 3-10). Note that, the number k here is batch size for WLS. The true labels for this selected nodes are acquired from the oracle. Then the labeled set $\mathcal{L}$ and unlabeled set $\mathcal{U}$ are updated accordingly (lines 11-12). At the end of the iterations, the labeled set $\mathcal{L}$ is ready to be used as training samples for classification. The core functionality of the algorithm lies in lines 3-10. Here, it first chooses k distinct random seed nodes from the unlabeled pool $\mathcal{U}$ (line 4). Then, for each of the seed node, it constructs a subgraph with up to B_e hop neighbors of the seed node V_i (line 6). This subgraph is sent to the Weisfeiler-Lehman method (line 7) to produce the labels which we will consider as canonical ordering of nodes based on structural properties. The algorithm always picks the first node in the produced ordering of the subgraph (line 8). It breaks ties arbitrarily. The selected node is added to the current set $\mathcal{L}^k$ (line 9). The algorithm selects k nodes for label acquisition for corresponding k seed nodes. These nodes may or may not have overlap with the seed nodes. In order to avoid duplication, once a node is picked it is no longer considered in the neighborhood of any other nodes. At the end of the iterations, the algorithm returns the set of k selected nodes (line 14). We use exploration budget $B_e = 2, 3$ for our evaluation and WLS-2, WLS-3 represents the corresponding versions of our algorithm.

The computational complexity of WLS directly depends on the complexity of Weisfeiler-Lehman algorithm. Let's denote the complexity of Weisfeiler-Lehman algorithm as $W(n)$ where n is number of nodes to label. Also, let $N_i(h)$ refer to the average number of nodes in the h-hop neighborhood of any node V_i. The time complexity of WLS becomes $O(B * W(N_i(h)))$.

5 EXPERIMENTAL EVALUATION

5.1 Data

We conduct experiments on four real world datasets, three of which are based on citation networks: Cora, Citeseer, and Pubmed [1]. The

[1]All datasets available at https://linqs.soe.ucsc.edu/data.

Dataset	$\|V\|$	$\|E\|$	Number of Features	Number of Classes	Class Entropy	Average Degree	Clustering Coefficient	Homophily	Class Label
Citeseer	3312	4660	3703	6	1.71	2.81	0.1711	0.74	Topic
Cora	2708	5278	1433	7	1.83	3.90	0.2376	0.80	Topic
Hateful	3218	9620	1036	2	0.47	5.98	0.0785	0.72	Hateful/normal
PubMed	19717	44327	500	3	1.06	4.50	0.0602	0.80	Topic

Table 1: Properties of the datasets used in experimental evaluation.

first two corresponds to publications in computer science and the third one is based on publications on Diabetes diseases. The fourth dataset is sampled from the *Hateful Users on Twitter* dataset [28]. The original network contains around $100k$ users where around $5k$ users are annotated as either "hateful" or "normal". Our sample consists of the annotated nodes and the edges between them.

We pre-process all datasets by removing all nodes that are not connected to the largest connected component. Table 1 summarizes the properties of the datasets after pre-processing. The column titled *Class Entropy* represents the entropy of distribution of classes in a dataset. The higher the entropy means more balanced class distribution and vice versa. Cora and Hateful seems to be the best and worst datasets in terms of class balance.

The next three columns of Table 1 shows several important network properties. We can see all the datasets exhibit reasonably good amount of homophily where homophily is measured by the proportion of edges that connect two nodes from the same class. One interesting property to notice here is the distinction of clustering coefficient among the datasets. Based on these properties we can categorize the datasets into two groups. Citeseer and Cora fall into *Group I* with smaller number of nodes and edges, relatively high clustering coefficient and low average degree. They also consist of higher number of classes with reasonable class balance. On the other hand, Hateful and Pubmed forms *Group II* with higher number of nodes and edges, high average degree and low clustering coefficient. They exhibit strong class imbalance, especially Hateful.

5.2 Experimental Setup

5.2.1 Hash function for WLS. We use a specific hashing function, Pallette-WL [42], compatible with the standard Weisfeiler-Lehman algorithm (1) for implementing WLS. Pallette-WL not only avoids higher computational cost by using refined normalized hash function, it also preserves vertex orders across iterations. This technique has been shown to be effective in link prediction for utilizing subgraph features [42]. In our work, we use the Pallette-WL method for the implementation of the Weisfeiler-Lehman algorithm. Note that, the complexity for Pallette-WL is $O(n^2)$ where n is the number of nodes to label [42]. So, according to the description in Section 4.2.4, the overall complexity of WLS becomes $O(k * W(N_i(h)^2))$.

5.2.2 Relational Classifiers. We consider *Logistic Regression* as the local classifier for ICA and *Count* function as the aggregator. We choose *Simplified GCN* (SGC) [38] and GraphSage [15] classifiers as representatives of GNN. SGC is a faster approximation of the popular relational classifier GCN.

5.2.3 Evaluation Methodology. We randomly split 80% of the nodes for training and keep the other 20% for testing. We run all our experiments 5 times and take the average.

Most of the datasets in this study contain multiple classes and there is a considerable class imbalance present in the data as shown in Table 1. To reduce the impact of class imbalance in evaluation, we used stratification while splitting the train and test samples. We also considered class weighted loss functions for the classifiers. We considered Micro-F1 score as the evaluation metric. This is a popular metric used to evaluate multi-class classification.

We varied active learning budget B from 32 up to 224. We used the same budget for all datasets for consistency. The maximum budget, 224 represents 10% of the training nodes for all datasets except PubMed. We consider batch size $k = 8$ for some sampling methods in our experiments. It represents the number of nodes selected per iteration for WLS whereas for MS and KMS it represents the number of clusters.

5.2.4 Packages and Hardware. We use NetworkX 2.3 [14] for representing and processing graphs. Scikit-Learn library is used for implementation of *Logistic Regression* and K-means clustering. We use StellarGraph [10] package for implementing SGC and GraphSage [2]. For running all our experiments and recording execution time we use Ubuntu 18.04 OS running on a 96 core Intel(R) Xeon(R) Platinum 8275CL @ 3.00GHz processor with 185GB memory.

5.3 Results

Figures 1 shows results for all the candidate sampling methods using four classifiers (wvRN, ICA, SGC, GraphSage) on four different datasets. In the figure, the rows represent different datasets and the columns represent different classifiers used. Moreover, the y-axis represents Micro-F1 score and the x-axis shows the number of training nodes considered as active learning budget B. We can observe a great deal of variance in terms of performance of different sampling methods. To better understand the relative performance, we list down all the Micro-F1 scores for the highest budget (224) in Table 2. Each row in this table corresponds to a specific dataset and a specific relational classifier. The bold cell represents the best Micro-F1 score in the corresponding row. For example, in the first row, for Citeseer dataset and wvRN classifier, NS-DC-H performs the best. We rank the algorithms based on this table and present the final ranking in Table 3. We used the following process to generate the final ranking: first, rank all sampling algorithms for each row of Table 2 separately and then calculate the average rank of each sampling algorithm over all 16 rows. First column of Table 3 represents the sampling methods sorted by their average ranks over all 16 combinations. The next three columns show the average rank along with standard deviation of the corresponding sampling methods based on different categories of relational classifiers. The column *All* shows the average ranks for all classifiers whereas the

[2]The code is available online at https://github.com/edgeslab/sampling-osal

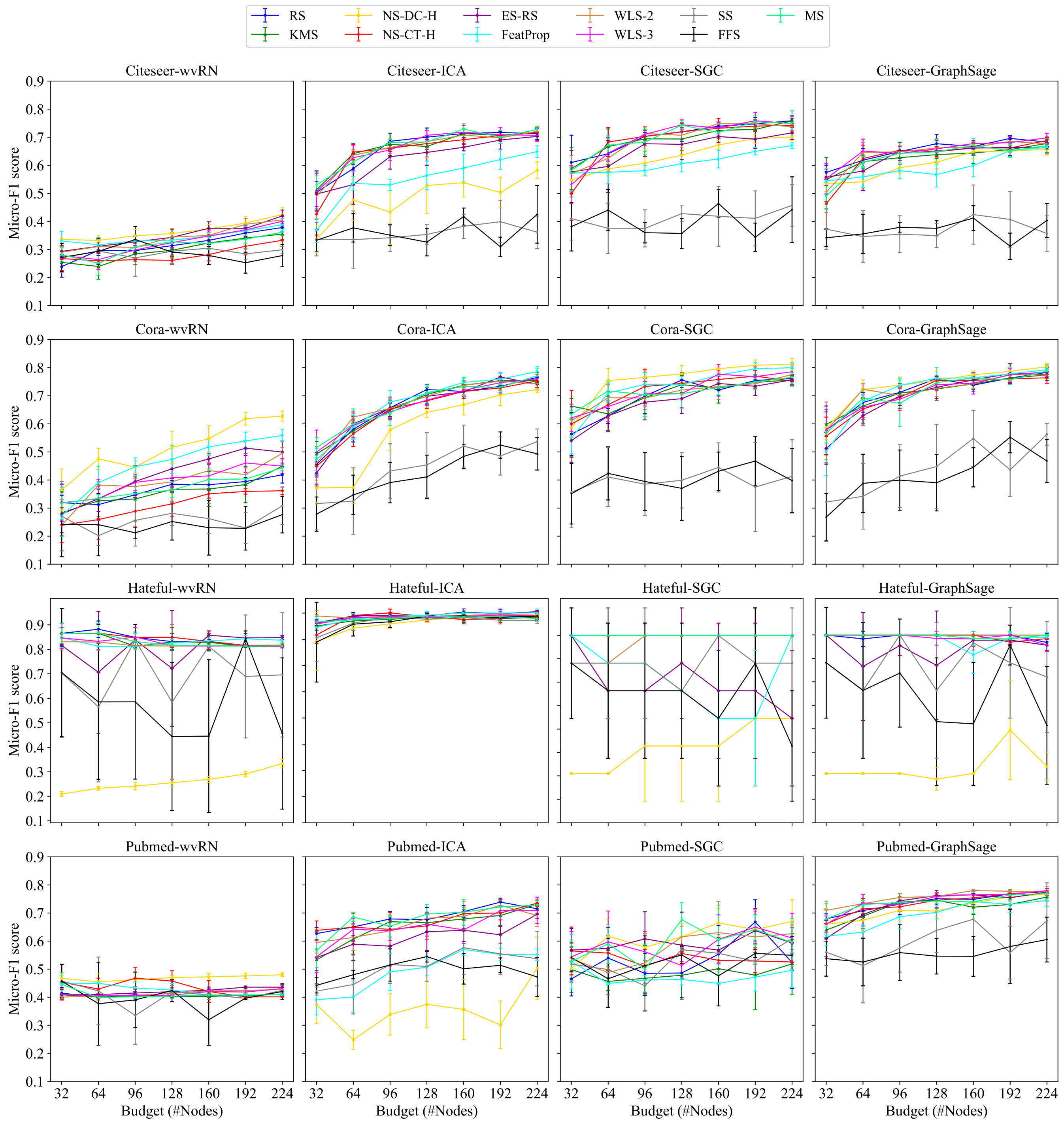


Figure 1: Comparison of Macro-F1 scores by different sampling methods using different relational classifiers.

column *GNN* shows average ranks for only GNN-based classifiers (over 8 rows). The last column shows average ranks for wvRN and ICA classifiers (over 8 rows). Both versions of our proposed method (WLS-3, WLS-2) top the overall ranking and show relatively low standard deviation. This establishes its robustness across multiple datasets and classifiers. Next, we present the main takeaways by analyzing the performance of the sampling methods from several different perspectives:

Group I vs Group II datasets We can observe from the results that our proposed method WLS performs relatively better in *Group*

Table 2: Micro-F1 scores of 11 sampling methods across 4 datasets and 4 classifiers for active learning budget of 224 nodes.

Dataset	Classifier	RS	NS-DC-H	NS-CT-H	ES-RS	FeatProp	WLS-2	WLS-3	SS	FFS	KMS	MS
Citeseer	wvRN	0.378199	**0.424171**	0.332701	0.419431	0.388132	0.405213	0.400474	0.298803	0.278156	0.354976	0.361611
Citeseer	ICA	0.712322	0.581991	0.716588	0.702844	0.648728	0.733745	**0.737602**	0.361347	0.425220	0.718483	0.727962
Citeseer	SGC	**0.758588**	0.700552	0.742451	0.715568	0.670169	0.737108	0.748460	0.457079	0.441471	0.758101	0.749000
Citeseer	GSAGE	0.682464	0.656398	0.688152	0.661611	0.663981	0.676303	**0.697630**	0.356634	0.403149	0.677725	0.669668
Cora	wvRN	0.418511	**0.627767**	0.361617	0.499396	0.558551	0.495372	0.449497	0.308099	0.276676	0.449253	0.443461
Cora	ICA	0.765392	0.721932	0.753320	0.741247	**0.786318**	0.742455	0.760161	0.536877	0.493188	0.754930	0.771429
Cora	SGC	0.762181	**0.812475**	0.752918	0.756856	0.800000	0.775742	0.785630	0.412851	0.397802	0.756111	0.765896
Cora	GSAGE	0.783501	**0.803219**	0.764185	0.776660	0.793159	0.773843	0.785111	0.562559	0.467780	0.774245	0.788330
Hateful	wvRN	0.813704	0.333333	0.817037	**0.848519**	0.837407	0.812593	0.810741	0.695926	0.456296	0.813704	0.814444
Hateful	ICA	0.888519	0.896296	0.888519	**0.905556**	0.900741	0.890000	0.889259	0.872593	0.888889	0.884444	0.879259
Hateful	SGC	**0.899083**	0.545549	**0.899083**	0.545549	**0.899083**	**0.899083**	**0.899083**	0.781238	0.427704	**0.899083**	**0.899083**
Hateful	GSAGE	0.867598	0.340110	0.884077	0.856113	0.888896	**0.899083**	0.857594	0.720681	0.514434	**0.899083**	**0.899083**
Pubmed	wvRN	0.409381	**0.479817**	0.401217	0.435446	0.409381	0.427840	0.430680	0.424074	0.420188	0.409432	0.408773
Pubmed	ICA	0.715112	0.503296	**0.734888**	0.695538	0.550963	0.690974	0.709533	0.537221	0.472110	0.730781	0.727890
Pubmed	SGC	0.523560	**0.669704**	0.525552	0.592995	0.496082	0.626141	0.610570	0.608615	0.549057	0.518602	0.594904
Pubmed	GSAGE	0.775913	0.767546	0.776318	0.773986	0.744371	0.775355	**0.779513**	0.672577	0.605269	0.756542	0.769320

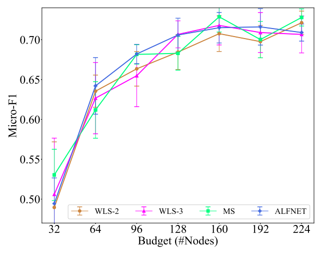
Table 3: Average ranks of sampling methods for different categories of relational classifiers over all datasets.

Sampling	Avg. Rank		
	All	GNN	wvRN, ICA
WLS-3	**4.19 ± 1.81**	**3.38 ± 1.80**	5.00 ± 1.41
WLS-2	**4.56 ± 1.87**	**4.38 ± 1.87**	**4.75 ± 1.85**
MS	**4.88 ± 2.60**	**4.25 ± 1.39**	5.50 ± 3.28
FeatProp	5.28 ± 3.20	6.00 ± 3.24	**4.56 ± 2.99**
RS	5.31 ± 2.21	4.62 ± 2.29	6.00 ± 1.89
ES-RS	5.44 ± 2.71	7.12 ± 1.45	**3.75 ± 2.63**
KMS	5.66 ± 2.54	5.62 ± 2.83	5.69 ± 2.22
NS-DC-H	5.75 ± 4.07	5.88 ± 3.92	5.62 ± 4.21
NS-CT-H	5.88 ± 2.98	5.50 ± 2.69	6.25 ± 3.19
SS	9.19 ± 1.94	9.00 ± 2.06	9.38 ± 1.80
FFS	9.88 ± 1.76	10.25 ± 1.30	9.50 ± 2.06

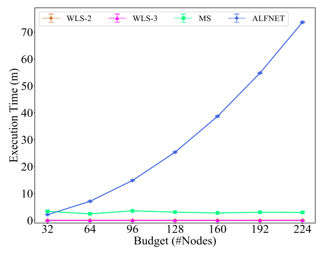
II where the network is larger and exhibits higher average degree. On the other hand random sampling (RS) and Degree Centrality (NS-DC-H) work better in *Group I*. This indicates that, in smaller networks with high clustering coefficient, simple node sampling or even random sampling is good enough for one-shot active learning. On the other hand, for larger graphs with low clustering coefficient it requires more sophisticated methods like WLS.

Network Sampling vs others. Next, we observe how different categories of sampling methods perform across all setups. Figure 1 shows that graph sampling methods like *Snowball Sampling* (SS) or *Forest Fire Sampling* (FFS) exhibit poor performance for relational classification. This is intuitive provided that these methods are heavily biased in spatial exploration. They fail to explore diverse local regions of the graph. Another expected observation is that the non-network sampling approach KMS suffers in almost all the cases since it can not exploit any of the relational information. Note that even though it shows high Micro-F1 score for the Hateful dataset, that could be due to the high class imbalance in that dataset. Node sampling approaches seem to do best in Cora and Pubmed where higher homophily is observed. In general graph-based clustering method (MS) shows a consistently good performance across all setups. Surprisingly, the hybrid approach FeatProp only performs great in Cora but produce relatively poor results for other cases.

GNN vs others. Most of the sampling methods produces higher Micro-F1 score when used with GNN approaches compared to wvRN and ICA. GNN approaches show consistently better performance across all datasets except for Hateful. Surprisingly, FeatProp works best with ICA even though it was primarily designed for SGC. It is interesting to note that certain sampling methods show significant variation in performance based on the classifier. For example, degree centrality (NS-DC-H) shows good result using SGC (3rd row) but quite poor using ICA (2nd row) on Citeseer dataset. In contrast, WLS-2 and WLS-3 show less variance across different classifiers and datasets. The last two columns of Table 3 show the difference in performance for GNN-based classifiers versus previous relational classifiers. The cells marked bold represent the top three average ranks in each category and the top three overall sampling methods also perform the best for GNN-based classifiers. However, ES-RS takes the top spot in the last column which supports the findings by Ahmed et al. [3].



(a) Micro-F1



(b) Execution time in minutes

Figure 2: Comparison of top three sampling methods vs ALFNET using ICA classifier on Citeseer dataset.

One-shot vs Multi-shot. In order to show the effectiveness of one-shot active learning, we compare the sampling methods with ALFNET, a state-of-the-art active learning algorithm for relational data which requires iterative training over the acquired samples. Figure 2 shows both Micro-F1 score (2a) and execution times (2b) for the sampling methods and ALFNET on the Citeseer dataset. We choose only the top 3 (second row of Table 2) sampling methods (WLS-3, WLS-2, MS) for convenience of comparison. In Figure 2a, we can see the sampling methods show competitive results compared to ALFNET. However, in Figure 2b the difference in execution time is significant. Note that, the execution times presented here is in minutes and lines for WLS-2, WLS-3 overlap with each other. This big difference in execution time and competitive Micro-F1 score justifies the motivation behind one-shot active learning.

6 CONCLUSIONS

We address a constrained classification problem, *one-shot active learning* for relational data. The objective is to reduce both labeling and computation cost of relational classification in large real world network datasets. We explore a wide variety of sampling methods as solutions and proposed a node sampling method based on Weisfeiler-Lehman algorithm. We experimentally evaluate all these sampling methods on four real world network datasets and four popular relational classifiers. The main takeaways are as follows:

- WLS performs best with GNN-based classifiers whereas ES-RS shows best results for wvRN and ICA classifier. WLS also shows overall best performance across all setups.
- Network-based node sampling methods work well for smaller networks with high clustering coefficient.
- One-shot active learning methods produce competitive results compared to state-of-the-art multi-shot active learning methods with much smaller computational cost.

This work shows a comprehensive analysis of the effectiveness of sampling methods for one-shot active learning on relational data. A promising next step could be developing a cost-effective multi-shot active learning method that can effectively solve the scaling issue of relational classifiers.

REFERENCES

[1] Nesreen Ahmed, Jennifer Neville, and Ramana Rao Kompella. 2011. Network sampling via edge-based node selection with graph induction. (2011).

[2] Nesreen K Ahmed, Jennifer Neville, and Ramana Kompella. 2010. Reconsidering the foundations of network sampling. In *Proceedings of the 2nd Workshop on Information in Networks*.

[3] Nesreen K Ahmed, Jennifer Neville, and Ramana Kompella. 2012. Network sampling designs for relational classification. In *Sixth International AAAI Conference on Weblogs and Social Media*.

[4] Nesreen K Ahmed, Jennifer Neville, and Ramana Kompella. 2013. Network sampling: From static to streaming graphs. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 8, 2 (2013), 1–56.

[5] Lars Backstrom and Jon Kleinberg. 2011. Network bucket testing. In *Proceedings of the 20th international conference on World wide web*. 615–624.

[6] Lilian Berton, Didier Augusto Vega-Oliveros, Jorge Carlos Valverde-Rebaza, Andre Tavares da Silva, and Alneu de Andrade Lopes. 2016. The Impact of Network Sampling on Relational Classification.. In *SIMBig*. 62–72.

[7] Mustafa Bilgic, Lilyana Mihalkova, and Lise Getoor. 2010. Active Learning for Networked Data. In *ICML*.

[8] Hongyun Cai, Vincent W Zheng, and Kevin Chen-Chuan Chang. 2017. Active learning for graph embedding. *arXiv preprint arXiv:1705.05085* (2017).

[9] Zhiqian Chen, Fanglan Chen, Lei Zhang, Taoran Ji, Kaiqun Fu, Liang Zhao, Feng Chen, and Chang-Tien Lu. 2020. Bridging the Gap between Spatial and Spectral Domains: A Survey on Graph Neural Networks. *arXiv preprint arXiv:2002.11867* (2020).

[10] CSIRO's Data61. 2018. StellarGraph Machine Learning Library. https://github.com/stellargraph/stellargraph.

[11] Li Fei-Fei, Rob Fergus, and Pietro Perona. 2006. One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence* 28, 4 (2006), 594–611.

[12] Li Gao, Hong Yang, Chuan Zhou, Jia Wu, Shirui Pan, and Yue Hu. 2018. Active discriminative network representation learning. In *IJCAI International Joint Conference on Artificial Intelligence*.

[13] Stuart Geman and Donald Geman. 1984. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-6 (1984), 721–741.

[14] Aric Hagberg, Pieter Swart, and Daniel S Chult. 2008. *Exploring network structure, dynamics, and function using NetworkX*. Technical Report. Los Alamos National Lab.(LANL), Los Alamos, NM (United States).

[15] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in neural information processing systems*. 1024–1034.

[16] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations (ICLR)*.

[17] Eric D Kolaczyk. 2009. Sampling and estimation in network graphs. In *Statistical Analysis of Network Data*. Springer, 1–30.

[18] Ankit Kuwadekar and Jennifer Neville. 2011. Relational Active Learning for Joint Collective Classification Models. In *ICML*.

[19] Sang Hoon Lee, Pan-Jun Kim, and Hawoong Jeong. 2006. Statistical properties of sampled networks. *Physical review E* 73, 1 (2006), 016102.

[20] Jure Leskovec and Christos Faloutsos. 2006. Sampling from large graphs. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. 631–636.

[21] Qing Lu and Lise Getoor. 2003. Link-based Classification. In *ICML*.

[22] Sofus A. Macskassy. 2009. Using graph-based metrics with empirical risk minimization to speed up active learning on networked data. In *KDD*.

[23] Sofus A Macskassy and Foster Provost. 2007. Classification in networked data: A toolkit and a univariate case study. *Journal of machine learning research* 8, May (2007), 935–983.

[24] Arun S Maiya and Tanya Y Berger-Wolf. 2011. Benefits of bias: Towards better characterization of network sampling. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. 105–113.

[25] Jennifer Neville and David Jensen. 2000. Iterative classification in relational data. In *Proc. AAAI-2000 Workshop on Learning Statistical Models from Relational Data*. 13–20.

[26] Mark EJ Newman. 2006. Modularity and community structure in networks. *Proceedings of the national academy of sciences* 103, 23 (2006), 8577–8582.

[27] Hieu Tat Nguyen and Arnold W. M. Smeulders. 2004. Active learning using pre-clustering. In *ICML*.

[28] Manoel Horta Ribeiro, Pedro H. Calais, Yuri A. Santos, Virgílio A. F. Almeida, and Wagner Meira. 2018. Characterizing and Detecting Hateful Users on Twitter. In *ICWSM*.

[29] Nicholas Roy and Andrew McCallum. 2001. Toward Optimal Active Learning through Sampling Estimation of Error Reduction. In *ICML*.

[30] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Gallagher, and Tina Eliassi-Rad. 2008. Collective Classification in Network Data. *AI Magazine* 29 (2008), 93–106.

[31] Burr Settles. 2009. *Active Learning Literature Survey*. Computer Sciences Technical Report 1648. University of Wisconsin–Madison.

[32] H. Sebastian Seung, Manfred Opper, and Haim Sompolinsky. 1992. Query by Committee. In *COLT*.

[33] Nino Shervashidze and Karsten Borgwardt. 2009. Fast subtree kernels on graphs. In *Advances in neural information processing systems*. 1660–1668.

[34] Nino Shervashidze, Pascal Schweitzer, Erik Jan van Leeuwen, Kurt Mehlhorn, and Karsten M. Borgwardt. 2011. Weisfeiler-Lehman Graph Kernels. *Journal of Machine Learning Research* 12 (2011), 2539–2561.

[35] Michael PH Stumpf, Carsten Wiuf, and Robert M May. 2005. Subnets of scale-free networks are not scale-free: sampling properties of networks. *Proceedings of the National Academy of Sciences* 102, 12 (2005), 4221–4224.

[36] Boris Weisfeiler and AA Lehman. 1968. A reduction of a graph to a canonical form and an algebra arising during this reduction. *Nauchno-Technicheskaya Informatsia* 2, 9 (1968), 12–16.

[37] Mark Woodward and Chelsea Finn. 2017. Active one-shot learning. *arXiv preprint arXiv:1702.06559* (2017).

[38] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying Graph Convolutional Networks. In *Proceedings of the 36th International Conference on Machine Learning*. PMLR, 6861–6871.

[39] Yuexin Wu, Yichong Xu, Aarti Singh, Yiming Yang, and Artur Dubrawski. 2019. Active Learning for Graph Neural Networks via Node Feature Propagation. In *Workshop on Graph Representation Learning, NeurIPS*.

[40] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S Yu. 2019. A comprehensive survey on graph neural networks. *arXiv preprint arXiv:1901.00596* (2019).

[41] Sooyeon Yoon, Sungmin Lee, Soon-Hyung Yook, and Yup Kim. 2007. Statistical properties of sampled networks by random walks. *Physical Review E* 75, 4 (2007), 046114.

[42] Muhan Zhang and Yixin Chen. 2017. Weisfeiler-Lehman Neural Machine for Link Prediction. In *KDD*.