HES: Edge Sampling for Heterogeneous Graphs

Le Fang* and Chuan Wu*
*The University of Hong Kong, Email: lefang@connect.hku.hk, cw@cs.hku.hk

***Abstract*—In light of the success of graph neural networks (GNNs), recent years have seen significant developments in modeling graph-structured data. Heterogeneous graphs have been widely adopted to model complex systems for various ML tasks. However, although some researchers have proposed methods for heterogeneous graphs, they merely focus on node features while neglecting the effectiveness of edge features. Besides, some research projects attempted incorporating edges into GNNs, but most regarded edge features as shared weights between node pairs. In this paper, we propose a two-stage method HES to learn the correlations among edge neighbors: (1) Graph Transformation: We convert the original heterogeneous graph into an undirected graph while preserving the orientation information, (2) Group Edge Sampling: To reduce the computation cost for the edge sampling in a heterogeneous graph, we propose to sample the most important edges over a group of edge neighbors instead of the whole graph, which leverages the edge features based on the one-hot encodings to describe the mutual influences between any adjacent edges. Finally, the experimental results on multiple public datasets show that HES outperforms existing state-of-the-art (SOTA) graph sampling methods. We further apply our approach to some existing GNN models as a pre-training process, demonstrating that HES can augment GNN-based models effectively.**

***Index Terms*—Edge Sampling, Graph Neural Networks**

I. Introduction

Recent years have seen significant developments of deep learning models for graph-structured data [1]–[3], as a heterogeneous graph, with nodes and oriented edges, is naturally suitable to describe the entities and their inner-relationships among data [4]. Generally, a heterogeneous graph contains nodes and oriented edges, in which objects of different types interact in various ways [5]. Considering a citation network [6], there are five types of nodes representing multiple characters: authors, papers, venues, institutes, and fields. Besides, the oriented edges are used to present high-dimensional relationships between two neighboring nodes, e.g., *paper-cite-paper, author-write-paper, author-in-institute, paper-in-field, paper-show-venue*, etc. For instance, the edge between a paper and an author, indicates "the author writes this paper". However, for a paper with a group of co-authors, it is necessary to measure the importance of each co-author on this paper. Besides, the edge orientation is commonly ignored [7], [8], which usually contains important information about the graph [9]. For example, machine learning papers cite other theoretical papers in a citation network. However, theoretical papers may rarely cite machine learning papers. In a graph, each edge is used to describe the relationship strengths or other properties between two edges, which is commonly shared by these two neighboring nodes [7]–[9]. Nevertheless, sharing the weights is not practical in most cases, as the edge orientation information affects both endpoints differently in a heterogeneous graph.

Neural networks were firstly adopted into graphs by Scarselli et al. [10]. Recently some graph-structured neural networks were proposed to provide better performance for graph learning, e.g., graph convolutional network (GCNs) [7], graph attention network(GAT) [8]. But these models commonly neglected the edge information, which are not designed for heterogeneous graphs. Several works [4], [9], [11] attempted to incorporate the edge features into the graph neural networks but still faced a key limitation: they merely regard edges features as sharing weights between two neighboring nodes, making these frameworks hard to implement in heterogeneous graphs. For heterogeneous graph learning, Wang et al. [12] introduced heterogeneous graph attention networks (HANs) based on GAT [8], and Hu et al. [5] recently proposed the heterogeneous graph transformer (HGT) inspired by the Transformer [13]. However, their works still focused more on node features while neglecting the importance of edges. So it is important to explore a better method to augment graph learning in a heterogeneous graph based on the edge information.

To empower the influences of edge information in a graph, edge sampling can be applied to sample a subgraph, in which more influential (high-degree) nodes with more edges can be sampled [14]. Nevertheless, we observe only a limited number of research works based on edge sampling for heterogeneous graphs. Edge sampling has been well-studied for undirected graphs (UG), or homogeneous graphs [15]–[17]. Penalized likelihood-based methods have been widely adopted to fit a UG on a given dataset, which preserves symmetry in the adjacency matrix constructed based on the data, e.g., the least angle regression (LARS) [15], the least absolute shrinkage and selection operator (LASSO) [16]. Specifically, the symmetry of the adjacent matrix is typically compulsory for edge selection [18]. However, guaranteeing symmetry is challenging on a heterogeneous graph, as the oriented edge often affects two adjacent edges differently. Furthermore, most studies [15], [16], [18] mainly focused on a small UG and sampled edges from the entire graph, which made them difficult to implement for a complex heterogeneous graph directly.

In light of these challenges, we tend to propose a more effective edge sampling method based on the edge information in a heterogeneous graph. Firstly, we consider transforming the original heterogeneous graph into an undirected graph while maintaining the orientation information of edges for each node

This work was supported in part by grants from Hong Kong RGC under the contracts HKU 17208920, 17207621 and 17203522.

and preserving the symmetry. Secondly, we partially select a neighbor set for each edge and catch the most correlated neighbors based on the edge selection function, to achieve more effective edge sampling. Our work can also be extended to existing neural networks (like GCNs [7], GAT [8]) as the pre-training stage to select effective neighboring edges. As each sampled edge links two neighboring nodes, we can construct a neighbor set for each node instead of searching for all neighbors by adopting random walks. [7].

In summary, the contributions of this work are given as follows:

- We propose **HES**, an edge sampling method for heterogeneous graphs that can be verified as an effective way for graph sampling as the most correlated edge neighbors can be selected.
- We demonstrate the effectiveness of HES on several public datasets by comparing it with other sampling methods, including the standard and the state-of-the-art methods.
- We apply our method to GNN-based models as the pre-training process. The experimental results can verify that HES can perform better than the random walks for the classification tasks.

II. Background and Related Work

A. Current Sampling Methods

Generally, we can divide the sampling methods into three categories, i.e., node-based sampling and edge-based sampling, and topology-based sampling.

Random node (RN) sampling [19] is one of the most apparent node-based sampling methods, which samples a fixed fraction of nodes from the whole graph, where nodes are chosen independently and randomly. Then PageRank algorithm [20] adopted by Google can be regarded as the state-of-the-art node-based sampling method, which measures the importance of each node within the graph.

Edge sampling focus on sampling edges rather than nodes, where an edge is sampled with two nodes selected at each step. Compared with node-based sampling, each node is selected in proportion to its degree in the sampled graph, as high-degree nodes tend to have more connected edges with other nodes. We omit the discussions of edge sampling methods here and present more details and comparisons in the next subsection to avoid repetitive conversations.

The most popular topology-based sampling method is the random walk sampler (RW) [21], which randomly picks a starting node, simulates a random walk on the graph, and then collects nodes after a number of walking steps. Recently a common neighbor-aware random walk sampler (CNARW) [22] was proposed by leveraging weighted walking at each step, which can be regarded as a state-of-the-art algorithm.

B. Edge Sampling for Graphs

Some previous works have studied edge sampling in undirected graphs (UG). Tan et al. [16] presented a LASSO-based method, which can fit in a small homogeneous graph and estimate all correlations among each edge pair. Ong et al. [15] proposed a LARS-based method to sample edges in an undirected graph that preserves the symmetry. However, these two methods can not be applied to a large graph, as it is too computationally costly to estimate all distances of each edge pair within the whole graph. Ahmed et al. [23] optimized classic edge sampling and proposed a totally induced edge sampling (TIES) method, in which they induced all the edges between the sampled nodes instead of the sampled edges. Yousuf et al. [24] recently presented a totally induced weighted edge sampling (TIWES) method by introducing the weights, which randomly samples an edge and then increases the sampling probability of its neighbors. Nevertheless, TIWES may increase the sampling bias sharply, as only the neighbors of sampled edges have more sampling opportunities while ignoring the inner correlations among edges.

For directed graphs, some hybrid method methods considered the effectiveness of edges. Ribeiro et al. [25] adopted directed unbiased random walk (DURW) to sample subgraphs, which neglected those multiple attributes for edges in a heterogeneous graph. Xu et al. [26] and Voudigari et al. [27] only focused on the nodal properties in a directed graph while ignoring the edge features. Furthermore, these works can not sample these correlated edges in a heterogeneous graph.

To fill in the gap of edge sampling for heterogeneous graphs, we propose an efficient way to transform a heterogeneous graph into a UG, namely graph transformation (GT). Then we sample these important edges from the neighbor set of each edge instead of the whole graph, which assures that our method can be applied to big graphs.

C. Incorporating Edges into GNNs

As edge features contain some important information in a graph, some researchers attempted to exploit edge information in GNN models. Schlichtkrull et al. [11] introduced relational graph convolutional networks (R-GCNs) with edges based on GCNs [7]. However, R-GCNs can not accept edges with multiple attributes. Gong et al. [9] proposed edge enhanced graph neural network (EGNN) to augment GCNs [7] and GAT [8] with edges features. However, EGNN merely regards these edges as weights between any neighboring nodes. Chen et al. [4] presented edge-featured graph attention networks (EGAT) based on GAT [8], which processed edges as the same as nodes powered by attention networks. However, EGAT is not enough to process heterogeneous graphs as neighboring nodes share the same edge attributes. Recently Gong et al. [28] adopted an edge selection function when building the graph neural networks, which solely selects the edges based on the features of the adjacent points.

Compared with these works, we aim to incorporate multiple edge features into GNNs and leverage edges to show the mutual influences between any node pairs.

D. Heterogeneous Graphs

There are some existing works to adopt GNNs to learn with heterogeneous graphs. Wang et al. [12] firstly introduced het-

erogeneous graph attention networks (HANs), which focused more on nodes while neglecting the edge features. Yun et al. [29] proposed graph transformer networks (GTNs) based on Transformer [13], which still focused more on learning effective node representations while edges are merely used to design meta-paths. Zhang et al. [30] presented a heterogeneous graph neural network (HetGNN) model, which mainly sampled correlated neighbors for each node and grouped them based on the node types while neglecting the edge features. Hu et al. [5] recently proposed the heterogeneous graph transformer (HGT) architecture to process various heterogeneous graphs, which sampled sub-graphs in which different types of nodes are with similar proportion while neglecting the importance of selected edge features.

We observe that all these works only adopted the edge features in the learning process of node presentations but ignored selecting essential edge features in a heterogeneous graph. Unlike them, we tend to explore the correlations of edges and sample correlated edges for graph learning.

TABLE I: Summary of Notations

s, t	refers to a node	e	refers to an edge
$\mathcal{N}$	the set of nodes	$\mathcal{E}$	the set of edges
$\mathcal{A}$	the set of node types	$\mathcal{R}$	the set of edge types
M	the number of nodes	D	the number of edges
J	the edge sample size	S	upper bound of edge neighbors
W	number of hidden layers	H	the dimension of hidden layer

$\mathcal{G}, \mathcal{G}'$	the original graph, and the reconstructed graph by GT
U_s, E_s	the neighboring node/edge set for node s
$\tau(s)$, $\phi(e)$	node/edge type mapping function
$\sigma(s,t)$	the orientation information between nodes s and t
Q_e	the quadruplet representing the meta relation for edge e
Q	the set of all edge meta relations
i_s, o_s	the in-degree and out-degree values for node s
d_s	the degree vector for node s, $d_s = [i_s, o_s]$
$d_{(e,j)}$	the shared degree vector for both edges e and j
X_e, X	the edge feature for edge e, the set of edge features
$N_e^{(1)}$, $N_e^{(2)}$	the first-order/second-order edge neighbors for edge e
E_{out}, $E_{out}^{(e)}$	the sampled edge set, the sampled edge set for edge e
$N_{out}^{(e)}$	the reconstructed neighboring node set based on $E_{out}^{(e)}$
N_{out}, $N_{out}^{(s)}$	the sampled node set, the sampled node set for node s

III. METHOD

A. Basic Notations

Consider a heterogeneous graph $\mathcal{G} = (\mathcal{N}, \mathcal{E}, \mathcal{A}, \mathcal{R})$ ($\mathcal{N} \in R^F, \mathcal{E} \in R^P$) with M nodes and D edges, in which each node $s \in \mathcal{N}$ and each edge $e \in \mathcal{E}$, $\tau(n) : \mathcal{N} \rightarrow \mathcal{A}$ and $\phi(e) : \mathcal{E} \rightarrow \mathcal{R}$ are type mapping functions. For node s, we denote its neighboring edge set is E_s and its neighboring node set is U_s. For an edge e linking two neighboring nodes s and t with orientation information $\sigma(s,t)$ (e.g., from s to t), let a quadruplet $Q_e = [\tau(s), \phi(e), \tau(t), \sigma(s,t)]$ be the meta relation for edge e. Note that $\sigma(s,t) \in \{-1, 0, 1\}$, where $\sigma(s,t) = -1$ $(or\ 1)$ indicates that s points to t (or t points to 1), $\sigma(s,t) = 0$ indicates that there is no directed edge between s with t. And Q denotes the set of all meta relations. Besides, we denote that n is an F-dimensional vector presenting the node, and e is a P-dimensional vector indicating the edge. A high-dimensional vector for presenting a node has been well-adopted [5], [9], [12], and a high-dimensional vector for presenting an edge can be an advantage for augmenting the edge vector with more important features. For example, in an "*author-write-paper*" network, a higher dimensional vector can be used to present the authorship order and author type, e.g., the first author, the last author, and the corresponding author, which can be an important factor that indicates the contribution of each author.

B. Problem Formulation

Given a larger heterogeneous graph, we tend to study the correlations of edges and sample some important edges, as edges contain multiple important information [5]. However, traditional edge sampling methods [15], [16], [24] can only work for undirected graphs because of the limitation of preserving the symmetry. So our **first challenge** is maintaining symmetry for a heterogeneous graph. Besides, these works are not efficient in a graph as they tend to sample edges in the whole graph, which can be pretty challenging facing a large graph. Then our **second challenge** is to propose a more efficient sampling method in a heterogeneous graph.

To address these challenges, we propose a two-stage method for sampling effective edges in a heterogeneous graph:

1 **Graph Transformation (GT)**. We notice that the biggest difference for a UG and a heterogeneous graph is that the latter contains each edge's orientation information linking two neighboring nodes. We propose an effective way of converting the original heterogeneous graph into a UG while preserving the orientation information.

2 **Group Edge Sampling**. To reduce the computation cost for the edge sampling in a heterogeneous graph, we tend to sample the most important edges over a group of edge neighbors instead of the whole graph.

C. Graph Transformation

To convert a heterogeneous graph into a UG, prior methods [31], [32] just simply replace the oriented edges into undirected edges, in which any neighboring node pairs share the same influence of the edge. However, such transformation can cause avoidable mistakes. For example, in a citation network in Fig. 1, paper B (node 1) cites paper A (node 0). If we just remove the orientation information, the citation information can be misunderstood in the new graph. Based on the directed graph theory [33], we observe that for each node, it contains both *in-degree* and *out-degree* values, where *in-degree* indicates the number of incoming edges, while *outdegree* presents the number of outgoing edges. To describe the orientation information for each node s, we adopt a two-dimensional vector $d_s = [i_s, o_s]$, in which $i_n = \sum_{j \in U_s} \sigma(s,j)$ is the in-degree value and $i_n = \sum_{j \in U_s} \sigma(j,s)$ is the out-degree value. As shown in Fig. 1, we illustrate these two steps for graph transformation:

- **STEP 1:** We replace the original oriented edges with undirected edges and preserve both in-degree and out-

degree values for each node by adding a two-dimensional feature $[i_s, o_s]$ $(s \in \mathcal{N})$;

- **STEP 2:** We convert the positions of nodes and edges, and then obtain a new reconstructed graph $\mathcal{G}'$. Note that we preserve prior notations to avoid any conflicts. In the graph $\mathcal{G}'$, any neighboring edges e and j share the same degree vector $d_{(e,j)} = d_s$, where $e, j \in E_s$.

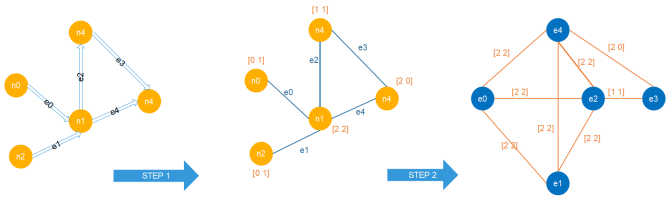


Fig. 1: Graph Transformation (GT) consists of two steps. STEP 1: Use node degrees to preserve the orientation information; STEP 2: Convert the positions of nodes and edges.

D. Group Edge Sampling

After the graph transformation, we tend to sample a fixed number of edges for each target edge in the newly constructed graph $\mathcal{G}'$. Prior work [34] tended to find the weight distribution matrix from all edges, which presents all correlation weights for any edge pairs, which is computationally expensive for a large-scale graph with millions of edges.

To address this challenge, sampling the edges on a partial graph [15] can be a promising solution. For example, we can only sample the first-order neighbors (and second-order neighbors) for computing the weight distribution for a target edge. Note that first-order neighbors mean these edges connected directly with the target edge. Second-order neighbors indicate these edges that walk through another edge before arriving at the target edge. However, sampling some edges on a partial graph can not guarantee symmetry all the time [35], as if edge j is selected in the sampled set of edge e, there is no guaranty that the edge t would be selected as a sampled significant edge of j. Recall that the set of edges is E, the set of edge meta relations is Q, the first-order neighborhood for edge e is $N_e^{(1)}$ (while $N_e^{(2)}$ includes both the first-order and second-order neighbors). Λ is the learnt parameters that present the edge weight distribution for each edge, and $\beta_{ej} = -\Lambda_{ej}/\sqrt{\Lambda_{ee}\Lambda_{jj}}$. Inspired by the "MB-OR" method in [18], we adopt a consistency procedure (CP) to preserve symmetry when sampling the edges, in which edges j or e can be sampled at each side when $\beta_{ej} \neq 0$ or $\beta_{je} \neq 0$.

As Q_e is a quadruplet presenting diverse attributes, we can pre-trained each attribute of the edge meta relation ($Q_e = [\tau(s), \tau(t), \phi(i), \sigma(s,t)]$) into an edge feature separately using one-hot encoding method [36] and finally concatenate these vectors as the edge feature $X_e \in R^{d_E}$ (d_E is the dimension of edge features). For example, considering $Q_e = [\tau(s), \phi(e), \tau(t), \sigma(s,t)]$ after using one-hot encoding, $\tau(s)$, $\phi(e)$, $\tau(t)$ and $\sigma(s,t)$ can be represented as "0001", "0100", "1000", "10" separately. So the final edge representation can be set as "00010100100010". After getting all edge representations, we scale them into the set of edge features X, in which X_e for edge e satisfies the Gaussian distribution $(X_e = (X_{e1}, X_{e2}, ..., X_{ed_G}) \sim \mathcal{N}(0, \Sigma)$, where each element in Σ satisfies $\Sigma_{pq} = 1)$ and d_G is the dimension of edge features. Inspired by the LARS-based method in [15], we measure the distance between the original edge meta relation and its edge neighbors. Then we can obtain the original objective function by minimizing $L(\Lambda)$ into the form with only the partial neighboring edges as follows:

$$L(\Lambda) = \frac{1}{2}\sum_{e\in E}(d(e,j)||X_e - \sum_{j\in N_e^{\cdot}} \beta_{ej}\sqrt{\frac{\Lambda_{jj}}{\Lambda_{ee}}}X_j||_2^2) + \kappa\sum_{x<j}|f_{pen}|, \tag{1}$$

where a shrinkage parameter κ is adopted to avoid the overfitting issue [15] by multiplying the non-negative penalty function f_{pen} (normally $f_{pen} = \Lambda_{tj}$), in which κ can be determined by cross-validation [37]. Note that the "$\cdot$" in $N_e^{\cdot}$ can be set to (1) or (2), which indicates the first-order (and second-order) neighborhood for edge e.

As $d(e,j)$ is a two-dimensional vector, we investigate the influence for both in-degree and out-degree values separately. Recall that it is essential to describe the physical meanings from both dimensions in the citation network: the first-dimension weight means the number of cited papers, and the other weight represents the citation number by other articles. So we adopt $d_{(e,j)} = [v_{ej1}, v_{ej2}]$. Besides, a novel method [38] was recently proposed to optimize the loss function by introducing the edge sample size J. Based on these prior studies, then we can optimize the original objective (Equation 1) and obtain the following:

$$\begin{aligned} L_o(\Lambda) = &\frac{1}{2}\sum_{k\in\{0,1\}}\sum_{e\in E}(-J\log v_{ejk} + v_{ejk}||X_e \\ &- \sum_{j\in N_e^{\cdot}} \beta_{ej}\sqrt{\frac{\Lambda_{jj}}{\Lambda_{ee}}}X_j||_2^2) + \kappa\sum_{x<j}|f_{pen}|, \end{aligned} \tag{2}$$

in which we calculate the distances weighted on both in-degree and out-degree values for each neighboring edge and finally regard their sum as the final loss.

Theorem 1. *Given the sampled size J and shrinkage size κ, $L_o(\Lambda)$ is a convex function, which converges to a minimum that satisfies all edge neighbors.*

Proof. Here we can regard $L_o(\Lambda)$ as a special case of the objective function in [38], where we sample a group of neighbors $N_e^{(\cdot)}$ instead of the whole edge set. Since $-J\log v_{ejk}$ and $\kappa\sum_{x<j}|f_{pen}|$ are obviously convex functions, we mainly focus on the second term $v_{ejk}||X_e - \sum_{j\in N_e^{\cdot}}\beta_{ej}X_j||_2^2$. For the second term, we can obtain the following:

$$
\begin{aligned}
& v_{ejk}||X_e - \sum_{j \in N_e^{\cdot}} \beta_{ej} \sqrt{\frac{\Lambda_{jj}}{\Lambda_{ee}}} X_j||_2^2 \\
&= v_{ejk}||X_e + \sum_{j \in N_e^{\cdot}} \frac{\Lambda_{ej}}{\Lambda_{ee}} X_j||_2^2 \\
&= v_{ejk}||\frac{1}{\Lambda_{ee}}(\Lambda_{ee} X_e + \sum_{j \in N_e^{\cdot}} \Lambda_{ej} X_j) \\
&= \frac{v_{ejk}}{\Lambda_{ee}^2}|| \sum_{j \in N_e^{\cdot} \bigcup \{e\}} \Lambda_{ej} X_j||_2^2 \\
&= \frac{v_{ejk}}{\Lambda_{ee}^2}(\Lambda_{\cdot e}^{(e)'} X^{(e)'} X^{(e)} \Lambda_{\cdot e}^{(e)}),
\end{aligned}
\tag{3}
$$

where $\Lambda_{\cdot j}^{(e)'} X^{(e)'} X^{(e)} \Lambda_{\cdot j}^{(e)}$ is the quadratic form that is computed on set $N_e^{\cdot} \bigcup \{e\}$. Note that $\Lambda^{(e)}$ is the partial matrix in Λ for all edge's neighbors (including itself).

As the quadratic form in Equation 3 is jointly convex of $\Lambda^{(e)}$, combining with other two convex terms $-J \log v_{ejk}$ and $\kappa \sum_{x<j} |f_{pen}|$, we can obtain that the objective function $L_o(\Lambda)$ is a jointly convex function of Λ. □

E. Complete Algorithm

Then we present the heterogeneous edge sampling algorithm (HES) based on Stochastic Gradient Descent (SGD) [39] in Algorithm 1. Suppose the epoch is Z, and each step is indexed by z. Note that the complexity of Algorithm 1 is $O(MD + ZD^2 + JD)$. However, as we calculate the gradients based on the adjacent edges for each edge, the edge neighborhood's size is upper bound by $S \ll D$, indicating that $O(ZD^2)$ can reduce to $O(ZDS)$ in our scheme. So the complexity can reduce to $O(MD)$ by ignoring the small constants, i.e., Z, J, and S. According to the complexity analysis, it costs more computation resources on graph transformation while reducing the computation cost for edge sampling. Note that the sampled set is $E_{out} \in R^{D \times J}$, and the correlated edge set for edge e is $E_{out}^{(e)} \in R^J$.

F. Augmenting GNN-based Models

Our heterogeneous edge sampling algorithm can be adopted as the pre-training process to augment these standard GNN-based models, e.g., GCNs [7], GAT [8]. Note that we omit further discussion for other frameworks discussed in related work, as most of them relied on the designs of GCNs [7] and GAT [8] and commonly sampled the node/edge neighbors by random walks. Instead of sampling nodes by adopting random walks among a group of nodes, we can optimize the sampling process and further reduce the number of node neighbors based on the sampled edge set E_{out}. For example, for a sampled edge set $E_{out}^{(e)}$ of edge e, we can collect all neighboring nodes $N_{out}^{(e)} = \{s : d_s = (e, j), j \in E_{out}^{(e)}\}$ based on the degree vector $d_{(e,j)}$, in which e and j share the degree vector d_s for node s. Considering the size of edge sampling is J, the final size of total neighboring nodes N_{out} ($N_{out}^{(e)} \in N_{out}$) is smaller than JD. Then we can sample all the neighboring nodes $N_{out}^{(s)}$ ($N_{out}^{(s)} \in N_{out}$) for node s in N_{out}.

Algorithm 1: Heterogeneous Edge Sampling (HES)

Input: The original graph $\mathcal{G}$, the meta relation quadruplet set Q, learning rate η, randomly selected Λ that preserve symmetry, the total training rounds Z, edge sample size J

Output: Sampled edge set E_{out}

```
Compute both the in-degree and out-degree values for each node s in G: d_s = [i_S, o_s];
Transform G into G' by converting the positions of nodes and edges;
Transform the edge meta relation into edge features: Q → X;
for z ∈ {1, 2, .., Z} do
    Compute the gradients: g ← ▽L_o(Λ);
    Update the Λ: Λ ← Λ − ηg;
    for each symmetric elements Λ_pq and Λ_qp in Λ do
        if Λ_pq = 0 AND Λ_qp = 0 then
            Continue;
        else
            Λ_pq = Λ_gp = (Λ_pq + Λ_gp)/2;
        end if
    end
end
Sample the edges with Top − J correlation weights in each row of Λ and construct the sampled set E_out;
return Sampled edge set E_out.
```

IV. Evaluation

In this section, we evaluate our proposed method (HES) using several public datasets, in which we tend to answer the following questions:

- Q1 What is the effectiveness of HES comparing with other edge sampling methods?
- Q2 How does the sampling fraction (the portion of sampled subgraph over the whole graph) affect the sampling performance?
- Q3 Can HES augment other GNN-based methods for some downstream tasks?

A. Setup

1) Datasets: To evaluate the effectiveness of our method, we adopt four public datasets: two citation network datsets **Cora** [40] and **Pubmed** [41], a social network dataset **Facebook** [42] The main statistics of these datasets are summarized in Table II. Note that all networks are directed. For all datasets, we split 60%, 20%, and 20% of the whole dataset for training, validation, and testing, respectively. For HES, we set the edge sample size J to 10 for Cora and Pubmed, and set it to 20 for Facebook if not mentioned, which has covered most first-order and second-order neighbors for each edge.

TABLE II: Summary of Datasets

Dataset	# of Nodes	# of Edges	Entries
Cora	2,708	5,429	Seven Classes of Publications
Pubmed	19,717	44,338	Three Classes of Medical Papers
Facebook	46,952	876,993	Facebook Users, Posts

TABLE III: Summary of Implementation Details

Dataset	Epoch	Learning Rate	Edge Sampling Size
Cora	10	0.01	10
Pubmed	20	0.01	10
Facebook	30	0.005	20

2) Metrics: To measure the sampling performance of HES, we adopt the normalized root mean squared error (NRMSE) [25], [26] as follows:

$$NRMSE(\widehat{\chi}_e(J)) = \frac{\sqrt{(\widehat{\chi}_e(J) - \chi_e)^2}}{\chi_e}, e \in \mathcal{E}, \quad (4)$$

which measures the relative error of the estimate $\widehat{\chi}_e(J)$ concerning its true value $\chi_e(J)$ when the sample size is J. NRMSE has been verified as a better metric than the mean squared error (MSE) in prior studies [25], [26], as it allows different approaches to compare on a standard scale. Note that the estimates $\widehat{\chi}_e(J)$ and χ_e can be evaluated as follows:

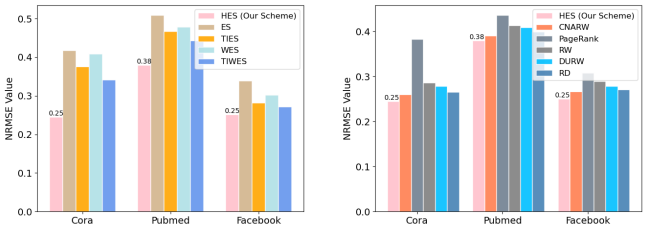
$$\widehat{\chi}_e(J) = \sum_{j \in E_{out}^{(e)}} X_j, \chi_e = \sum_{j \in N_e} X_j \quad (5)$$

To evaluate how well HES augments GNN-based models, we run each model several rounds and record the mean and standard deviation of the classification accuracy, which measures if the test sample is classified into the right class.

3) Baselines: We compare HES with other four standard edge-based sampling methods: (i) the standard Edge Sampling (ES) [43], (ii), the Weighted Edge Sampling (WES) [44], (iii) an optimized version, Totally Induced Edge Sampling (TIES) [23], (iv) the state-of-the-art (SOTA), Totally Induced Weighted Edge Sampling (TIWES) [24]. Besides, we compare HES with two other hybrid methods for directed graphs: (i) an optimized graph sampling method based on the random walk, Directed Unbiased Random Walk (DURW) [25], (ii) the state-of-the-art, Rank Degree (RD) [27]. Furthermore, We further compare with the well-known method (i) random walk sampler (RW) [21], and other SOTA approaches: (ii) the node-based sampling method, PageRank [20], (iii) the topology-based sampling method, Common Neighbor Aware Random Walk sampler (CNARW) [22].

We select two GNN-based baselines to test the augmentation performance of HES on the classification task: (i) the graph convolutional network (GCN) [7], (ii) the graph attention network (GAT) [8].

4) Implementation: For these experiments on Cora, Pubmed and Facebook, we adopt TensorFlow 2.0 [45] to implement our method on a server equipped with one Nvidia GTX 1080 GPU, an Intel Xeon E5-1620 CPU with four cores, and 32GB memory. To implement HES based on the SGD algorithm effectively, we summarize the details concerning various datasets in Table III based on the hands-on experience when the trained model achieves minimal loss. Besides, we set the momentum of SGD to 0.9 [46] for accelerating the weight updates.



(a) HES vs. Edge-based Methods (b) HES vs. Other Methods

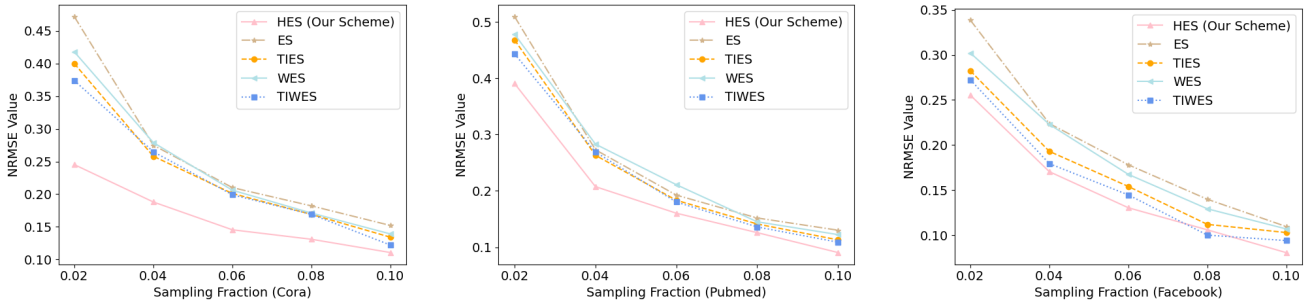
Fig. 2: NRMSE values of the degree: HES vs. Baseline Models

B. Evaluation Results

1) The Overall Results of HES (Q1): Note that the other four edge sampling baselines merely transform the directed graph into an undirected graph, so they combine both in-degree values and out-degree values when calculating the node degree. We first fix the sampling fraction to 2% of the whole dataset [27] and then repeat each experiment 20 times. We compute the averaged NRMSE values of the degree and summarize all experimental results on these benchmark datasets. Note that smaller NRMSE values indicate higher correlations of sampled edges. In Fig. 2a, we observe that HES outperforms other baselines on all datasets. Compared with ES and WES, TIES and TIWES also perform well as they add the neighbors of each sampled edge into the sampled list. However, they neglect the diverse weights of neighbors on the sampled edge, which can wrongly add the adjacent edges that are less correlated with the sampled one.

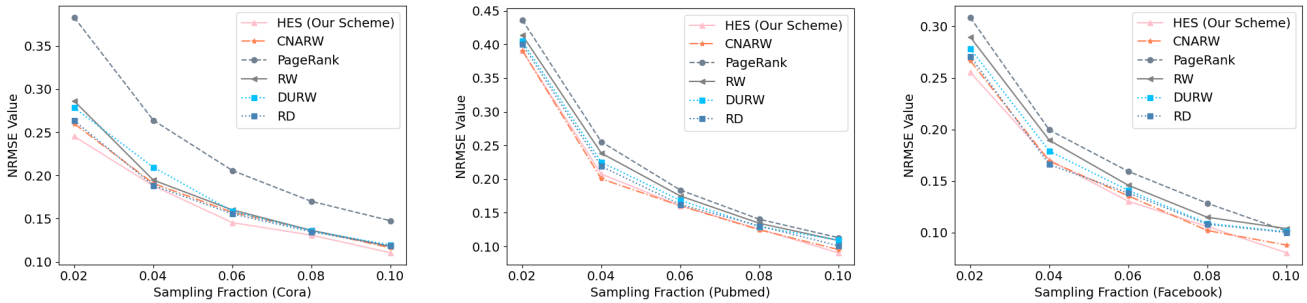
We further add experiments on other types of sampling methods and summarize the experimental results in Fig. 2b. We note that the PageRank algorithm performs worst among all methods, while HES can achieve the most competitive performance.

2) The Effectiveness of Sampling Fraction (Q2): To evaluate the effectiveness of sampling fraction, we choose it from the set $\theta = \{0.02, 0.04, 0.06, 0.08, 0.1\}$ [24] and show the NRMSE value versus the sampling fraction for all methods on each dataset. We do not increase the sampling fraction over 0.1 as extracting a subgraph with more samples is too costly to serve the sampling purpose [26]. We first compare all methods on the Cora dataset. In Fig. 3a, we show that our scheme (HES) always outperforms other baselines when varying the sampling fraction. Besides, we observe that the NRMSE values of the degree for these methods tend to decrease with a larger sampling fraction as more highly correlated adjacent edges can be sampled. Furthermore, the NRMSE values tend to decrease more slightly, which indicates that most correlated edges of sampled edges have been sampled. Similar observations can be found in Fig. 3b and Fig. 3c when we test these methods



(a) Cora Dataset (b) Pubmed Dataset (c) Facebook Dataset

Fig. 3: (Edge-based sampling methods) NRMSE values vs. the sampling fraction on public datasets



(a) Cora Dataset (b) Pubmed Dataset (c) Facebook Dataset

Fig. 4: (Other Methods) NRMSE values vs. the sampling fraction on public datasets

on Pubmed and Facebook datasets.

Furthermore, we also observe that WES performs worse than ES at some sampling fractions, which further indicates that it is not adequate to apply the same weights on the neighbors of sampled edges as the influences of these neighbors on the sampled edge vary a lot.

Finally, we compare HES with other baselines and summarize all experimental results in Fig. 4a, Fig. 4b, and Fig. 4c. We observe that our proposed method HES outperforms PageRank all the time. Compared with other SOTA methods, HES can also provide comparable performances under various settings.

3) The Effectiveness of Augmenting GNNs (Q3): As these GNN-based baselines [7], [8] adopt the random walk sampler (RW) [21] to sample the neighbors of each node and generate node representations, we replace the neighbor set of each node with the selected neighbor set by HES instead. Then we conduct experiments on Cora and Pubmed with these four models for classification problems. To achieve more equal comparisons, we select the first-order and second-order neighbors by random walks for baselines advised in [7]. We run each model 20 times and summarize the mean and standard deviation of the classification accuracies in Fig. IV. Here the GCN-based models with HES are denoted as GCN-HES and GAT-HES separately. From the experimental results in Fig. IV, we observe that GCN-HES and GAT-HES outperform the original baselines on both datasets considering the classification accuracies. Furthermore, we can obtain a smaller standard deviation of accuracy on each augmented model, indicating HES brings higher stability as the selected neighbors correlate more with the sampled node.

TABLE IV: Classification Accuracies on Two Citation Networks

Model	GCN	GCN-HES	GAT	GAT-HES
Cora	72.0 ± 1.2%	**75.0 ± 0.5%**	79.0 ± 1.0%	**82.1 ± 0.4%**
Pubmed	83.4 ± 0.2%	**84.1 ± 0.1%**	83.4 ± 0.2%	**84.2 ± 0.1%**

V. Conclusion

This paper introduces HES, an effective edge sampling method for heterogeneous graphs. We propose the graph transformation to fill in the gap of preserving the symmetry for a heterogeneous graph. Then we suggest sampling edges based on the edge neighbors group and verify that our objective can converge to a minimum from the theoretical perspective. We conduct extensive experiments on several benchmark datasets, and the experimental results show that our proposed HES outperforms other baselines significantly and consistently on various sizes of sampled graphs. Finally, we replace the random walk sampler with HES in GNNs, in which further evaluations verify that HES can augment GNNs effectively.

For future work, we plan to adopt HES in a web-scale graph, e.g., the open academic graph (OAG) [6] with more than 2 billion edges, explore existing challenges considering various downstream tasks and propose more practical and effective sampling solutions accordingly.

REFERENCES

[1] E. Isufi, F. Gama, and A. Ribeiro, "Edgenets: Edge varying graph neural networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 11, pp. 7457–7473, 2022.

[2] H. Chen, C. Zhou, J. Zhang, and Q. Li, "Heterogeneous graph embedding based on edge-aware neighborhood convolution," in *2021 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, 2021.

[3] L. Peng, R. Hu, F. Kong, J. Gan, Y. Mo, X. Shi, and X. Zhu, "Reverse graph learning for graph neural network," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–12, 2022.

[4] J. Chen and H. Chen, "Edge-featured graph attention network," *arXiv preprint arXiv:2101.07671*, 2021.

[5] Z. Hu, Y. Dong, K. Wang, and Y. Sun, "Heterogeneous graph transformer," in *Proceedings of the World Wide Web Conference*, WWW '20, p. 2704–2710, 2020.

[6] F. Zhang, X. Liu, J. Tang, Y. Dong, P. Yao, J. Zhang, X. Gu, Y. Wang, B. Shao, R. Li, and K. Wang, "OAG: Toward linking large-scale heterogeneous entity graphs," in *Proceedings of the 25th ACM SIGKDD*, KDD '19, p. 2585–2595, 2019.

[7] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proceedings of the 5th International Conference on Learning Representations*, ICLR '17, 2017.

[8] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph Attention Networks," in *Proceedings of International Conference on Learning Representations*, 2018.

[9] L. Gong and Q. Cheng, "Exploiting edge features for graph neural networks," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9203–9211, 2019.

[10] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, 2009.

[11] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. van den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *Proceedings of European Semantic Web Conference*, pp. 593–607, 2018.

[12] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, and P. S. Yu, "Heterogeneous graph attention network," in *Proceedings of The World Wide Web Conference*, WWW '19, p. 2022–2032, 2019.

[13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of NeurIPS*, 2017.

[14] M. A. Hasan, "Methods and applications of network sampling," *Optimization Challenges in Complex, Networked and Risky Systems*, pp. 115–139, 2016.

[15] M. H. V. Ong, S. Chaudhuri, and B. A. Turlach, "Edge selection for undirected graphs," *Journal of Statistical Computation and Simulation*, vol. 88, no. 17, pp. 3291–3322, 2018.

[16] K. M. Tan, P. London, K. Mohan, S.-I. Lee, M. Fazel, and D. Witten, "Learning graphical models with hubs," *Journal of Machine Learning Research*, vol. 15, no. 95, pp. 3297–3331, 2014.

[17] T. Eden and W. Rosenbaum, "On Sampling Edges Almost Uniformly," in *1st Symposium on Simplicity in Algorithms (SOSA)*, 2018.

[18] S. Zhou, P. Rütimann, M. Xu, and P. Bühlmann, "High-dimensional covariance estimation based on gaussian graphical models," *Journal of Machine Learning Research*, vol. 12, no. null, p. 2975–3026, 2011.

[19] M. P. H. Stumpf, C. Wiuf, and R. M. May, "Subnets of Scale-Free Networks Are Not Scale-Free: Sampling Properties of Networks," in *Proceedings of the National Academy of Sciences of the United States of America*, 2005.

[20] J. Leskovec and C. Faloutsos, "Sampling from large graphs," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2006.

[21] M. Gjoka, M. Kurant, C. T. Butts, and A. Markopoulou, "Walking in facebook: A case study of unbiased sampling of osns," in *2010 Proceedings IEEE INFOCOM*, pp. 1–9, 2010.

[22] Y. Li, Z. Wu, S. Lin, H. Xie, M. Lv, Y. Xu, and J. C. Lui, "Walking with perception: Efficient random walk sampling via common neighbor awareness," in *Proceedings of 2019 IEEE 35th International Conference on Data Engineering (ICDE)*, pp. 962–973, 2019.

[23] N. Ahmed, J. Neville, and R. Kompella, "Network sampling via edge-based node selection with graph induction," *Technical Report, Purdue University*, 2011.

[24] M. I. Yousuf and R. Anwar, "Weighted edge sampling for static graphs," 2019.

[25] B. Ribeiro, P. Wang, F. Murai, and D. Towsley, "Sampling directed graphs with random walks," in *Proceedings of IEEE INFOCOM*, pp. 1692–1700, 2012.

[26] X. Xu, C.-H. Lee, and D. Y. Eun, "A general framework of hybrid graph sampling for complex network analysis," in *Proceedings of IEEE INFOCOM*, pp. 2795–2803, 2014.

[27] E. Voudigari, N. Salamanos, T. Papageorgiou, and E. J. Yannakoudakis, "Rank degree: An efficient algorithm for graph sampling," in *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pp. 120–129, 2016.

[28] P. Gong, C. Wang, and L. Zhang, "Mmpoint-gnn: Graph neural network with dynamic edges for human activity recognition through a millimeter-wave radar," in *Proceedings of 2021 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7, 2021.

[29] S. Yun, M. Jeong, R. Kim, J. Kang, and H. J. Kim, "Graph transformer networks," in *Proceedings of NeurIPS*, 2019.

[30] C. Zhang, D. Song, C. Huang, A. Swami, and N. V. Chawla, "Heterogeneous graph neural network," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, p. 793–803, 2019.

[31] A. Hagberg, P. Swart, and D. Chult, "Exploring network structure, dynamics, and function using networkx," in *Proceedings of the 7th Python in Science Conference*, 2008.

[32] "NetworkX." https://networkx.org/, 2020.

[33] Y. Shang, "Focusing of maximum vertex degrees in random faulty scaled sector graphs," *Panamerican Mathematical Journal*, vol. 22, 09 2012.

[34] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of Computational Statistics*, COMPSTAT '10, 2010.

[35] N. Meinshausen and P. Bühlmann, "High-dimensional graphs and variable selection with the Lasso," *The Annals of Statistics*, vol. 34, no. 3, pp. 1436 – 1462, 2006.

[36] J. T. Hancock and T. M. Khoshgoftaar, "Survey on categorical data for neural networks," *Journal of Big Data*, vol. 7, no. 28, pp. 1–41, 2020.

[37] J. Z. Huang, N. Liu, M. Pourahmadi, and L. Liu, "Covariance matrix selection and estimation via penalised normal likelihood," *Biometrika*, vol. 93, no. 1, pp. 85–98, 2006.

[38] K. Khare, S.-Y. Oh, and B. Rajaratnam, "A convex pseudo-likelihood framework for high dimensional partial correlation estimation with convergence guarantees," *Journal of the Royal Statistical Society*, vol. 77, no. 4, pp. 803–825, 2015.

[39] L. Bottou and O. Bousquet, "The tradeoffs of large scale learning," in *Proceedings of the 20th International Conference on Neural Information Processing Systems*, NIPS'07, p. 161–168, 2007.

[40] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI Magazine*, vol. 29, p. 93, Sep. 2008.

[41] G. M. Namata, B. London, L. Getoor, and B. Huang, "Query-driven active surveying for collective classification," in *Workshop on Mining and Learning with Graphs*, 2012.

[42] B. Viswanath, A. Mislove, M. Cha, and K. P. Gummadi, "On the evolution of user interaction in facebook," in *Proceedings of the 2nd ACM Workshop on Online Social Networks*, WOSN '09, p. 37–42, 2009.

[43] V. Krishnamurthy, M. Faloutsos, M. Chrobak, L. Lao, J. H. Cui, and A. G. Percus, "Reducing large internet topologies for faster simulations," in *Proceedings of NETWORKING 2005. Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; Mobile and Wireless Communications Systems*, pp. 328–341, 2005.

[44] N. K. Ahmed, N. Duffield, T. L. Willke, and R. A. Rossi, "On sampling from massive graph streams," in *Proceedings of the VLDB Endowment*, 2017.

[45] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: A system for large-scale machine learning," in *Proceedings of USENIX OSDI*, pp. 265–283, 2016.

[46] Z. Xie, L. Yuan, Z. Zhu, and M. Sugiyama, "Positive-negative momentum: Manipulating stochastic gradient noise to improve generalization," in *Proceedings of the 38th International Conference on Machine Learning*, vol. 139, pp. 11448–11458, 18–24 Jul 2021.