DPGS: Degree-Preserving Graph Summarization

Houquan Zhou*[1], Shenghua Liu†[1], Kyuhan Lee[2], Kijung Shin[2], Huawei Shen[1] and Xueqi Cheng[1]

[1]Institute of Computing Technology, Chinese Academy of Sciences‡
[2]Graduate School of AI, KAIST

Abstract

Given a large graph, how can we summarize it with fewer nodes and edges while maintaining its key properties, e.g. node degrees and graph spectrum? As a solution, graph summarization, which aims to find the compact representation for optimally describing and reconstructing a given graph, has received much attention, and numerous methods have been developed for it. However, many existing methods adopt the *uniform reconstruction* scheme, which is an unrealistic assumption as most real-world graphs have highly skewed node degrees, even within communities. Therefore we propose a degree-preserving graph summarization model, DPGS, with a novel reconstruction scheme based on the configuration model. To optimize the Minimum Description Length of our model, we deisgn a linearly scalable algorithm using hashing techniques. We theoretically show that the minimized reconstruction error bounds the perturbation of graph spectral information. Extensive experiments on real-world datasets show that DPGS yields more accurate summary graphs than several well-known baselines. Moreover, our reduced summary graphs can effectively train graph neural networks (GNNs) while saving computational cost.

1 Introduction

Recent years have witnessed the explosive growth of data size, and large-scale graphs have become ubiquitous, including social networks, computer networks, and protein interactions network. Since they are hard to process, analyze, and understand, this poses significant challenges to graph mining applications.

An effective technique to tackle such challenges is **graph summarization**. Given a graph G, it aims to find a compact representation of G in the form of a *summary graph* with *supernodes* (i.e., subsets of nodes in G) and *superedges* (i.e., subsets of edges in G).

Generally, graphs are expected to be reconstructed from summary graphs by a summarization model, and thus the reconstruction scheme is the heart of most summarization models. Closely related studies [15, 21, 4, 14] reconstructed connections of original nodes based on the uniform assignment of superedges, namely the *uniform reconstruction* scheme. However, many real-world graphs contain communities of degree-skewed nodes [3]. That is, popular nodes or leaders in each community usually have more connections than the others. Thus, summarization models based on the uniform reconstruction scheme often result in a large gap between the reconstructed graph and the original graph in terms of node degrees, losing related properties, such as graph spectrum (i.e. eigenvalues), and the authorities and hubnesses of nodes [11].

Therefore, we propose a **D**egree-**P**reserving **G**raph **S**ummarization model, named DPGS, which assigns superedges proportional to node degrees, as in the well-known configuration model. Such a configuration-based assignment as a scheme (CR scheme) can generally be plugged in and improve existing related summarization methods. DPGS uses minimum description length (MDL) from information theory as the principle to minimize the cost of summary graphs and reconstruction error. Theoretically, we show that DPGS bounds the perturbation of Laplacian's eigenvalues by minimized reconstruction error. A fast algorithm, named the DPGS algorithm, is designed for DPGS to summarize large graphs based on LSH (Locality Sensitive Hashing) to group candidate nodes, and perform greedy merging within groups.

Our empirical study on synthetic datasets (random graphs with both uniform and skewed degree distributions) validates that the proposed scheme tends to reconstruct the original graph more accurately than the uniform reconstruction scheme, especially in highly degree-skewed graphs. Extensive experiments on 8 real-world datasets show that the DPGS algorithm outperforms several state-of-the-art algorithms. Moreover, we also show that our summary graphs can efficiently yet effectively help to train a graph neural network, while preserving high accuracy in a node classification task.

*Email: zhouhouquan18@mails.ucas.ac.cn

†Corresponding author. Email: liushenghua@ict.ac.cn

‡They are also with University of Chinese Academy of Sciences, Beijing 100049, China, and CAS Key Laboratory of Network Data Science & Technology, CAS.

In summary, our contributions include:

- **Novel reconstruction scheme**: We propose a graph summarization model, named DPGS, with a novel reconstruction scheme based on the configuration model. We theoretically show that our DPGS bounds the perturbation of graph spectrum with the reconstruction error.

- **Compatibility**: Our proposed scheme can be generally applied to other graph summarization models, and improve their summarization quality.

- **Effectiveness**: Experiments on both synthetic and real-world graphs verify the superiority of the proposed reconstruction scheme, and show that our DPGS algorithm outperforms several state-of-the-art methods with better summary. Moreover, summary graphs can help to train graph neural networks efficiently yet effectively.

- **Scalability**: Our DPGS algorithm runs fast, and theoretical analysis shows that the complexity is linear in number of edges.

Reproducibility : Our DPGS algorithm is open-sourced at `https://github.com/HQJo/DPGS`.

2 Related Work

Graph summarization methods can be categorized based on many aspects. See the comprehensive survey [16] for more knowledge about this topic. In this section, we categorize some graph summarization methods according to the objective functions.

Error of adjacency matrix: Methods in this category try to minimize some error metrics between the original and reconstructed adjacency matrices. k-Gs [15] aimed to find a summary graph with at most k supernodes, such that the L1 reconstruction error is minimized. Riondato et al. [21] revealed the connection between the geometric clustering problem and the graph summarization problem under multiple error metrics (including L1 error, L2 error and cut-norm error), and they proposed a polynomial-time approximate graph summarization method based on geometric clustering algorithms. Beg et al. [4] developed an randomized algorithm SAA-Gs using weighted sampling and count-min sketch [5] techniques to find promising node pairs efficiently.

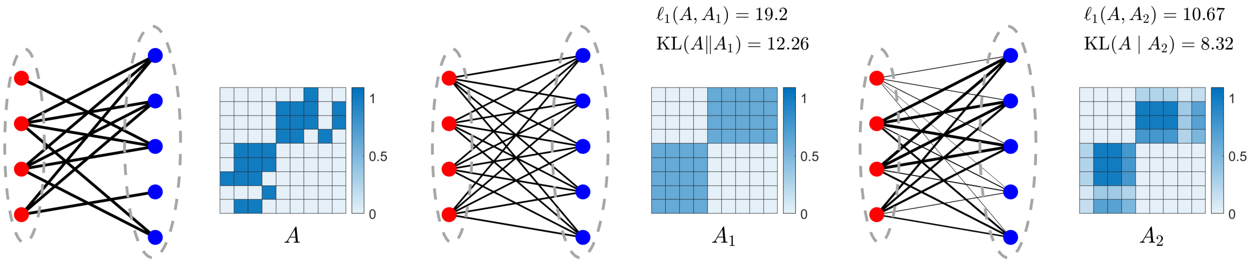
Total edge number: In this kind of methods, the objective function is defined as number of edges in summary graph plus edge corrections. In [17], Navlakha et al. proposed two algorithms: Greedy and Randomized. The former considers all possible node pairs at each step, and merges the best pair (u, v) which results in the greatest decrease of the total edge number. The latter samples a supernode as u randomly at each step, checks all other supernodes, finds the best v and merges them together. This process continues until the summary graph becomes smaller than a given size. However, both algorithms are computationally expensive. To address this problem, SWeG [24] reduces the search space by grouping supernodes, according to their shingle values, and only considers merging node pairs in the same groups. Combined with parallelization schemes, SWeG scales to large graphs with tens of billions of edges.

Table 1: Notations

Notation	Description
G	Input simple undirected graph
$\mathcal{V}, \mathcal{E}$	Node/edge set of G
n, m	Size of $\mathcal{V}$ and $\mathcal{E}$
$A_{n\times n}$	Adjacency matrix of G
G_S	Summary graph
$\mathcal{V}_S, \mathcal{E}_S$	Supernode/superedge set of G_S
n_s, m_s	Size of $\mathcal{V}_S$ and $\mathcal{E}_S$
$(A_S)_{n_s\times n_s}$	Adjacency matrix of G_S
$A'_{n\times n}$	Reconstructed adjacency matrix
d_i	Degree of node i
D_k	Degree of supernode S_k, i.e., the sum of degrees of the nodes in S_k

Encoding length: This kinds of methods often adopt the MDL principle and use the total encoding length as the objective function. They typically optimize the total description length under their proposed encoding scheme. LeFevre and Terzi [15] formulated the graph summarization problem Gs based on the MDL principle, and they proposed three algorithms Greedy, SamplePairs and LinearCheck. Lee et al. [14] designed a dual-encoding scheme and proposed a sparse summarization algorithm SSumM, which reduces the number of node and sparsifies the graph simultaneously. By dropping less important edges and encoding them as errors, SSumM is able to obtain a compact and sparse summary graph. Different from methods mentioned above, VoG [13] adopted an vocabulary-based encoding scheme, which encodes the graph using frequent patterns in real-world graphs, such as cliques, stars, and bipartite cores.

Methods mentioned above mainly focus on static simple graphs. There are works aiming to summarize other types of graphs, including dynamic graphs [23, 1, 20], attributed graphs [10, 8, 26], and streaming graphs [25, 12].



(a) Original graph

(b) Reconstructed (uniform)

(c) Reconstructed (CR)

Figure 1: Comparison between the CR and uniform schemes. The original graph and graphs reconstructed using the two reconstruction schemes are shown in the figure. Heatmaps of the corresponding adjacency matrices are plotted on the right side of graphs. Red nodes and blues nodes are two supernodes, and the width of edges reflects the edge weight. Visually and in terms of quantitative measures, the proposed CR scheme achieves better reconstruction results than the uniform scheme.

3 Proposed Method

In this section, we describe our DPGS model. To enhance the readability, we list frequently-used symbols in Table 1.

3.1 Reconstruction Scheme Given a summary graph, we can reconstruct the original graph based on a graph summarization model. Existing summarization models reconstruct the adjacency matrix using the *uniform reconstruction* scheme [15].

Definition 1. (Uniform Reconstruction Scheme) *Denote A_S and A' as the adjacency matrices of the summary graph and the reconstructed graph respectively. The uniform reconstruction scheme calculates $A'(i, j)$ as follows:*

$$A'(i,j) = \begin{cases} \dfrac{A_S(k,l)}{|S_k| \cdot |S_l|} & k \neq l \\ \dfrac{A_S(k,l)}{|S_k|(|S_k|-1)}, & k = l\,. \end{cases} \tag{3.1}$$

where S_k and S_l are the supernodes to which node i and node j belong, respectively.

It can be seen from Equation (3.1) that the edges between two supernodes S_k and S_l, i.e. $A_S(k, l)$, are equally assigned to each node pair between them, and each node pair has the same connection weight. Thus, this approach assumes the random graph model (or Erdős-Rényi model equivalently) [7]. However, real-world graphs have highly **skewed degree distributions.** Thus, this uniform reconstruction scheme is not suitable for real-world graphs.

Different from the uniform reconstruction scheme, we reconstruct A' based on degrees of nodes:

Definition 2. (CR Scheme) *Denote A_S and A' as the adjacency matrices of the summary graph and the reconstructed graph respectively. The configuration-based reconstruction scheme (CR scheme) calculates $A'(i, j)$ as follows:*

$$A'(i,j) = \frac{d_i}{D_k} A_S(k,l) \frac{d_j}{D_l}\,. \tag{3.2}$$

where S_k and S_l are the supernodes to which node i and node j belong respectively. We use d_i and d_j to denote the degrees of nodes i and j; and we use D_k and D_l to denote the degrees of supernodes S_k and S_l.

In this way, the reconstructed edge weight $A'(i, j)$ is proportional to the product of endpoints' degrees. This approach is based on the configuration model [18], which has proved successful in modularity-based community detection [19].

Note that the proposed CR scheme is able to preserve the degrees of nodes.

Property 3.1. (Degree Preservation)

$$\sum_{j=1}^{n} A'(i,j) = d_i = \sum_{j=1}^{n} A(i,j)\,. \tag{3.3}$$

Proof.

$$\begin{aligned} \sum_j A'(i,j) &= \sum_l \sum_{j \in S_l} \frac{d_i}{D_k} A_S(k,l) \frac{d_j}{D_l} \\ &= \sum_l \frac{d_i}{D_k} A_S(k,l) = d_i\,. \end{aligned}$$

■

Figure 1 demonstrates the difference between these two reconstruction schemes. It can be seen that our CR scheme can yield more accurate results, restoring the graph topology and the adjacency matrix better than the uniform reconstruction scheme.

3.2 Our proposed DPGS We use the MDL principle to find a summary graph. That is, we minimize the total description length while assuming that one needs both the summary graph and the errors for exactly reconstructing the original graph. Then, the objective $L(M, D)$ of our DPGS has two parts: the description length of summary graph $L(M)$, and the description length of errors $L(D \mid M)$:

$$L(M, D) = L(M) + L(D \mid M)\,. \tag{3.4}$$

To encode the errors between original and reconstructed adjacency matrices A and A', we use the generalized KL-divergence, an instance of the Bregman divergence [6], as in [9]:

$$\begin{aligned} L(D \mid M) &= \mathrm{KL}(A \| A') \\ &= \sum_{i,j} A(i,j) \ln \frac{A(i,j)}{A'(i,j)} - A(i,j) + A'(i,j) \\ &= \sum_{i,j} A(i,j) \ln \frac{A(i,j)}{A'(i,j)}, \end{aligned} \tag{3.5}$$

where the last two terms $-A(i,j) + A'(i,j)$ originate from the first-order term in the Bregman divergence. Due to the degree-preservation property, the last two terms are cancelled out when summing over i and j.

Moreover, we show that the eigenvalue perturbation are bounded by the reconstruction error $L(D \mid M)$, as formalized in the following theorem:

Theorem 3.1. (Eigenvalue perturbation) *Denote the normalized Laplacian matrices of the original graph and the reconstructed graph as $\mathcal{L}$ and $\mathcal{L}'$. Then, the total squared errors of their eigenvalues (denoted by $\lambda(i)$ and $\lambda'(i)$) are bounded as follows:*

$$\sum_{i=1}^{n} (\lambda(i) - \lambda'(i))^2 \leq 2 \cdot L(D \mid M) \tag{3.6}$$

Proof. See Appendix. ■

For the encoding length of model, we have:

$$L(M) = \mathrm{L}_{\mathbb{N}}(n_s) + n\,\mathrm{L}_{\mathbb{N}}(n_s) + \sum_{i=1}^{n} \mathrm{L}_{\mathbb{N}}(d_i) + L(A_S)\,, \tag{3.7}$$

where $\mathrm{L}_{\mathbb{N}}$ is the optimal encoding length for a positive integer [22]. The first two term encodes the number of supernodes and the supernode index to which each node belongs, respectively (we use $\mathrm{L}_{\mathbb{N}}(n_s)$ for all nodes for simplicity). The third term encodes the degrees of nodes in the original graph, which are required for reconstruction. Since the degree distribution is skewed, the encoding length for degrees is not large. Finally, $L(A_S)$ encodes the adjacency matrix of the summary graph in the following way:

$$\begin{aligned} L(A_S) = \mathrm{L}_{\mathbb{N}}(m_s) &+ \sum_{i=1}^{m_s} \mathrm{L}_{\mathbb{N}}(w_i) \\ &+ \mathrm{BE}\left(\frac{n_s(n_s+1)}{2}, m_s\right)\,, \end{aligned} \tag{3.8}$$

where w_i is the weight of superedge i, and $\mathrm{BE}(\frac{n_s(n_s+1)}{2}, m_s)$ encodes m_s superedges' endpoints (there are $\frac{n_s(n_s+1)}{2}$ possible superedges) using the **binomial encoding**, as in Equation (3.9).

$$\mathrm{BE}(a, b) = -b \log_2 \frac{b}{a} - (a - b) \log_2 (1 - \frac{b}{a})\,. \tag{3.9}$$

In summary, $L(A_S)$ encodes the size, endpoints and weights of superedges.

3.3 Algorithm Our DPGS algorithm is based on greedy merging operations. Firstly, each node is initialized as a supernode containing itself alone. Then, the algorithm finds promising supernode pairs (u, v) and merges them together consecutively. Here, the goodness of a node pair $gain(u, v)$ is defined as the decrease of the total description length when merging them.

Ideally, the algorithm is expected to find the best pairs at each step, which leads to the greatest decrease of the description length. However, this exhaustive search takes $O(|V|^2)$ time for each step, which is time-consuming and not scalable to large graphs. Thus, we need to find a way to reduce the search space.

The total description length is comprised of two parts, the model part and the error part. The model part contains several discrete function and is hard to analyze quantitatively. Generally, the smaller the m_s and n_s are, the smaller the model length is. For the error part, denote the change of the error part when merging supernodes S_i and S_j as $\Delta L_E(i, j)$. We have the following theorem.

Theorem 3.2. (Merging Cost) $\Delta L_E(i, j) \geq 0$.

Proof. See Appendix. ■

Theorem 3.2 says that merging two nodes never decreases the error part. This matches our intuition since merging two nodes retains or loses information. Thus, $\Delta L_E(i, j)$ can be seen as the cost of merging supernode S_i and S_j. While the the error part increases due to mergers, the model part may decrease, and this leads to the reduction of the total description length.

By analyzing $\Delta L_E(i, j)$, we have the following observation.

Algorithm 1 DPGS

```
Input: $G = (\mathcal{V}, \mathcal{E})$, iteration $T$
Output: $G_S = (\mathcal{V}_S, \mathcal{E}_S, A_S)$
$G_S \leftarrow G, \mathcal{V}_S \leftarrow \mathcal{V}, \mathcal{E}_S \leftarrow \mathcal{E}$
$t \leftarrow 0$
while $t < T$ do
    $t \leftarrow t + 1$
    Update LSH
    Divide supernodes into disjoint groups by LSH
    for each group $g$ do
        MergeGroup($g$)
    end for
end while
return $G_S$
```

Algorithm 2 MergeGroup

```
Input: $g \subset V_S$
$times \leftarrow \log_2|g|$
$nskip \leftarrow 0$
while $nskip < times$ and $|g| \geq 1$ do
    $pairs \leftarrow$ Sample $\log_2|g|$ node pairs from $g$
    $u, v \leftarrow \arg\max_{(i,j)\in pairs} gain(i, j)$
    if $gain(u, v) > 0$ then
        Merge $u$ and $v$
        $nskip \leftarrow 0$
    else
        $nskip \leftarrow nskip + 1$
    end if
end while
```

Observation 1. *The more common neighbors supernodes S_i and S_j have, the more likely their merging cost is small.*

Due to the space limit, the full analysis is placed in Appendix. This observation provides a guideline for our algorithm. That is, it is better to **merge nodes with more common neighbors**.

To do so, we utilize the LSH technique [2] to group nodes with similar neighborhoods together. The overview of the DPGS algorithm is given in Algorithm 1. The DPGS algorithm runs T iterations. In each iteration, DPGS seperates current nodes into disjoint groups according to LSH, and tries merging node pairs separately within each group. Inspired by SSumM [14], in each group g, we sample $\log_2|g|$ node pairs and merge the one with maximum gain. Here the gain is defined as the reduction of the total description length. This process is repeated until the size of group is less than 2, or the algorithm fails to find node pairs decreasing the description length for $\log_2|g|$ times. The full merging algorithm is described in Algorithm 2. According to Observation 1, DPGS is able to find and merge promising node pairs raising great reduction of description length.

3.4 Complexity Analysis

The DPGS algorithm scales linearly with the number of edges of the input graph, as formalized in Theorem 3.3.

Theorem 3.3. *The time complexity of Algorithm 1 is $O(T \cdot |E|)$.*

Proof. In each iteration, updating LSH costs $O(|E|)$ time. During the group merging step, at most $O(\log^2|g|)$ pairs are sampled, and calculating the merging gain of a node pair (u, v) and the remaining part of the merging step costs $O(d_u + d_v)$ time. If we limit the size of group not larger than a constant C (for example, 500), the expected running time of merging a group is $O(\sum_{u\in g} d_u)$. Thus, merging all the groups costs $O(\sum_g \sum_{u\in g} d_u) = O(|E|)$. In conclusion, the algorithm runs T iterations, and the total time complexity is $O(T \cdot |E|)$. ■

3.5 Compatibility with Existing Methods

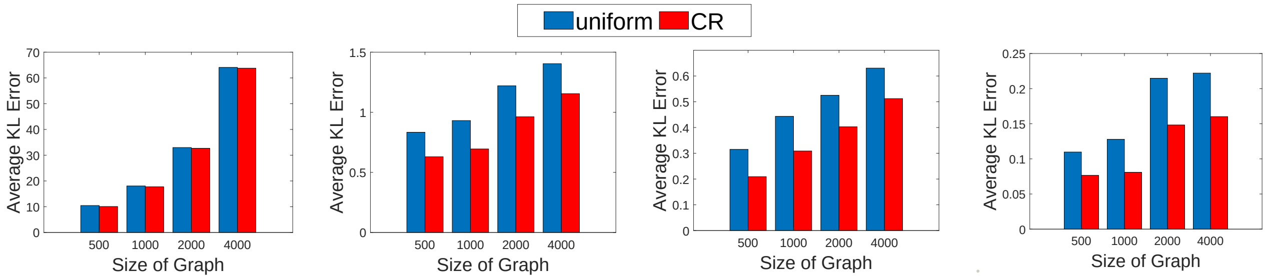
As a reconstruction scheme is the heart of most summarization models, our novel CR reconstruction scheme can replace the commonly-used uniform reconstruction scheme, and upgrade the existing models and then the corresponding algorithms. We show in the following two examples:

- **k-Gs (CR):** k-Gs [15] greedily merges (super)node pairs that lead to the least increase of the L1 error between the uniformly reconstructed graph and the original one at every step. k-Gs (CR) replaces the uniform reconstruction with our CR scheme and our KL-divergence encoding scheme in Equation (3.5).
- **SSumM(CR):** SSumM[14] greedily merges (super)node pairs and sparsifies (i.e. drops) (super)edges simultaneously to obtain a sparse summary graph. As SSumM greedily finds a summary graph based on the MDL principle and the uniform reconstruction scheme, we can smoothly upgrade it with our CR scheme and error encoding scheme used in the optimization objective and algorithm.

As such, k-Gs (CR) and SSumM(CR) yield degree-preserved summary graphs, and inherit the bounding graphs' spectra as our theoretical analysis.

4 Experiments

In this section, we design experiments to answer the following questions:



(a) Erdős-Rényi ($p = 0.02$) (b) Power-law ($\alpha = 3.0$) (c) Power-law ($\alpha = 3.5$) (d) Power-law ($\alpha = 4.0$)

Figure 2: Average reconstruction error[1] (KL-divergence) of two reconstruction schemes. Ours achieves lower error when the degrees of nodes are highly skewed (larger α indicates higher skewness), as in many real-world graphs.

- **Q1. Comparison of reconstruction schemes** Does our CR scheme outperform the uniform reconstruction scheme?
- **Q2. Effectiveness** Does DPGS yield better summarization compared to the baselines on real-world datasets?
- **Q3. Compatibility** How much can existing graph summarization methods be improved when they are equipped with our reconstruction scheme?
- **Q4. GNN application** Can we train graph neural networks effectively on summary graphs?
- **Q5. Scalability** Does DPGS scale well with the size of the input graph?

4.1 Q1. Comparison of reconstruction schemes

We generate synthetic data using the following random graph models:

- $G(n, p)$ model (also known as Erdős-Rényi model) [7], with connection probability $p = 0.02$.
- Random graph model with power-law degree distribution, with parameter $\alpha = 3.0$, 3.5, and 4.0

The main difference between these two models is that the degrees of nodes in Erdős-Rényi graphs are nearly uniform, while those in power-law graphs are highly skewed (larger α implies higher skewness).

We compare our CR scheme with the uniform reconstruction scheme using the summary graphs obtained by SSumM [14]. Note that we fix the optimization method to fairly compare the two reconstruction schemes. As shown in Figure 2, our CR scheme achieves lower reconstruction error (i.e. KL-divergence error) than the uniform reconstruction scheme. The margin of improvement is significantly large when the degree distribution is highly skewed, as in many real-world graphs. Simply put, our CR scheme gives better summaries of real-world graphs than the uniform reconstruction scheme.

Table 2: Dataset statistics

Dataset	#Nodes	#Edges	Description
ppi	14,755	228,431	Protein
ppi-large	56,944	818,786	Protein
soc-Epinions1	75,879	405,740	Social
flickr	89,250	449,878	Social
reddit	232,965	11,606,919	Social
yelp	716,847	7,335,833	Social
amazon	1,569,960	132,954,714	Co-purchase
amazon2m	2,449,029	61,859,140	Co-purchase

4.2 Q2. Effectiveness

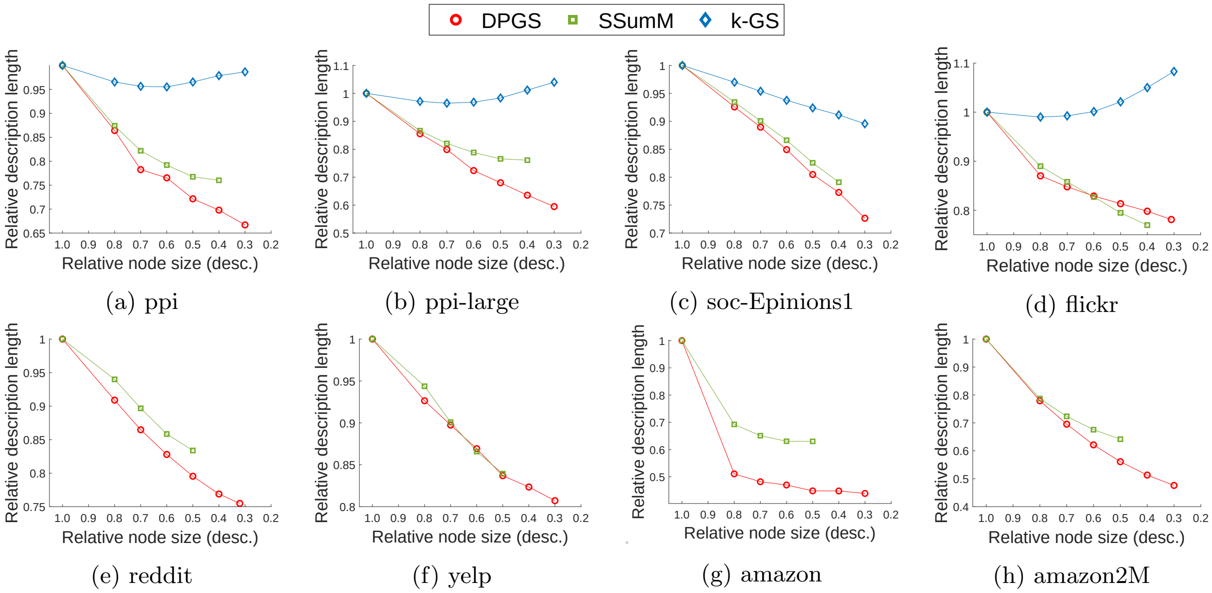
We compare our DPGS algorithm with k-Gs [15] and SSumM [14] on eight real-world datasets listed in Table 2.

We implement k-Gs in C++ and adopt the SamplePairs strategy for summarizing large graphs. For SSumM, we use the open-sourced code[2]. The number of iterations in SSumM and DPGS is set to 30. The number of bands (i.e., b) in LSH grows gradually as the iteration proceeds, and the minimum and maximum numbers are set to 3 and 8 respectively.

We summarize the graphs aiming at different fractions of node sizes (from 80% to 20%), while reducing the graph size, and compare the relative description lengths[3] of different methods. The results are shown in Figure 3. It can be seen that the proposed DPGS algorithm achieves better summarization in most of the datasets, except the flickr dataset where the perfor-

[1] The values are normalized by the size of graph, i.e. $|V|$.

[2] `https://github.com/KyuhanLee/SSumM`



(a) ppi

(b) ppi-large

(c) soc-Epinions1

(d) flickr

(e) reddit

(f) yelp

(g) amazon

(h) amazon2M

Figure 3: Relative description length[3] (the smaller, the better) of different methods on 8 datasets. DPGS yields the best summarization result on most datasets. Due to the high complexity of k-Gs , it fails on the large datasets in the second row.

mance gap is marginal. However, in the other datasets, our DPGS algorithm achieves up to 28.85% improvement in terms of the relative description length.

4.3 Q3. Compatibility In this experiment, we show how much our CR scheme improves the existing graph summarization methods. We compare k-Gs and SSumM, which are based on the uniform reconstruction scheme, with k-Gs (CR) and SSumM(CR), which are upgraded as described in subsection 3.5. Note that the algorithms with the upgraded models search summary graphs by optimizing the new objective functions, which are different from simply reconstructing a given summary graph with the configuration-based scheme in Section 4.1. Since k-Gs searches for the best summary graph subject to a given target number of supernodes, we measure the KL-divergence error on average while changing the target number of supernodes (from 10% to 90% of the number of nodes in the original input graph). On the other hand, SSumM searches for the best summary graph subject to a given target size in-bits. Thus, we measure the average KL-divergence error while changing the target size in bits (from 10% to 80% of the original input graph size).[4]

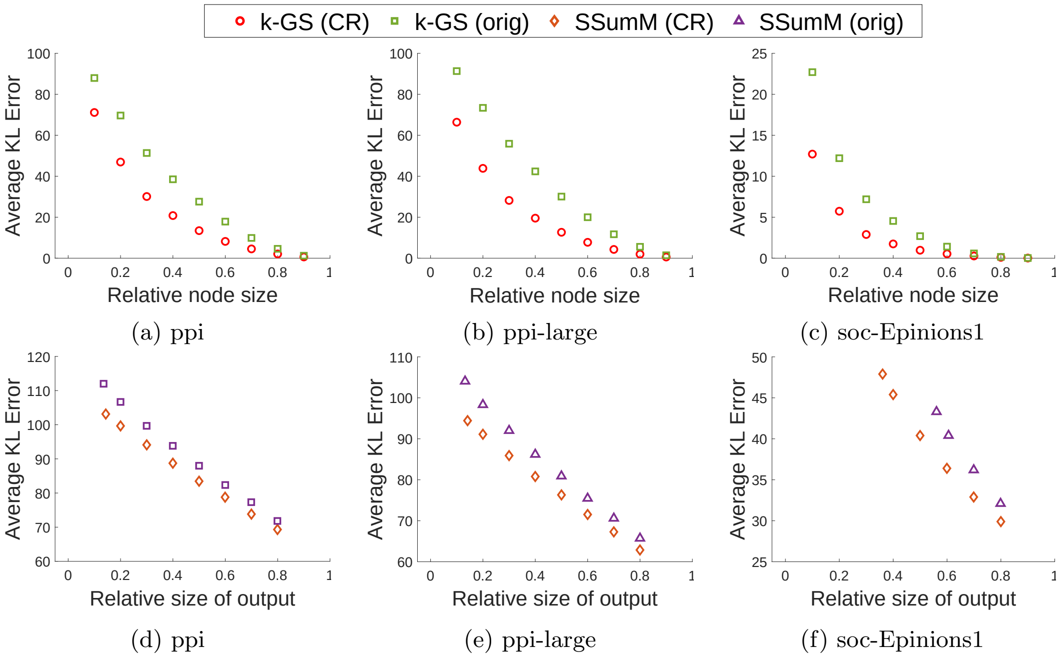
As seen in Figure 4, the variants with our CR scheme consistently yield more accurate summaries than the original methods based on the uniform reconstruction scheme in all three datasets. For example, k-Gs (CR) gives a 2.8× more accurate summary graph with the same number of supernodes than original k-Gs in the ppi-large dataset. Moreover, SSumM(CR) gives a summary graph with 1.2× smaller KL-divergence error but smaller sizes than original SSumM in the soc-Epinions1 dataset. Simply put, our CR scheme as the heart of the summarization models helps the existing methods to find a better solution.

4.4 Q4. Training GNNs on summary graphs

One important application of graph summarization is to accelerate graph mining algorithms, and graph neural networks (GNNs) are one of the most memory-consuming and time-consuming graph mining algorithms currently. Ideally, we can reduce the running time and required memory via graph summarization, without sacrificing the performance of GNNs much. Thus, we design experiments to check how our summarization method affects the performance of GNNs. Specifically, we choose the task of node classification to test the performance. Since the graphs we are deal-

[3] defined as $(L(M) + L(D \mid M))/L(D)$, where $L(D)$ is the encoding length of the original graph when it is encoded as the summary graph is encoded.

[4] The KL divergence error (i.e., Equation (3.5)) is not defined for dropped superedges, which may contain $(i, j) \in \mathcal{E}$ such that $\overline{A'(i,j)} = 0$ and $A(i,j) \neq 0$. Thus, we add the correction of each edge belonging to each dropped superedge to the model cost, and we increase the description length (and thus relative size) accordingly.



(a) ppi (b) ppi-large (c) soc-Epinions1

(d) ppi (e) ppi-large (f) soc-Epinions1

Figure 4: Our CR reconstruction scheme improves k-Gs and SSumM. Equipped with our scheme, k-Gs and SSumM consistently yield better summary graphs with smaller reconstruction errors[5] than the original methods. Larger improvement margins can be seen when aiming at a smaller summary graphs.

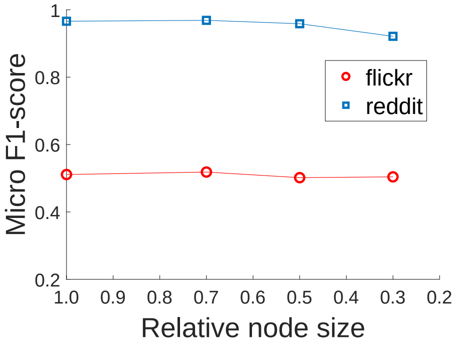


Figure 5: Micro F1-scores on the node classification task when using GNN models trained on summary graphs with different sizes.

ing with are large, we use GraphSAINT [27], a scalable GNN model based on subgraph sampling.

The procedure is as follows: We train GraphSAINT models on summary graphs, keep models' parameters, and test the node classification performance on the original graphs. Features of supernodes are obtained by simple aggregation (i.e., sum) of features of inside nodes, and class labels of supernodes are determined by a majority vote of inside nodes. We follow the original GraphSAINT paper's configurations and use the random walk sampler, which achieves the best performance in most datasets.

We perform experiments on two datasets: the flickr and reddit datasets. The micro F1-scores are reported in Figure 5. Surprisingly, summarization does not harm the performance, even when the size of the summary graphs is 30% of the size of the original ones.

4.5 Q5. Scalability We evaluate the scalability of our method on the largest amazon dataset, which contains 1,569,960 nodes and 132,169,374 edges. We run DPGS algorithm on a number of graphs that are obtained from the original dataset by randomly sampling different numbers of nodes. As seen in Figure 6, our proposed DPGS algorithm **scales linearly with the number of edges**.

5 Conclusion

In this work, we present DPGS, an efficient and effective graph summarization algorithm using a novel configuration-based reconstruction scheme, and we theoretically show that it bounds the perturbation of graph spectrum. The proposed reconstruction scheme can also be applied to existing graph summarization methods to improve their reconstruction accuracy. Extensive experiments on both synthetic and real-world datasets show

[5] The values are normalized by the number of nodes, i.e., $|V|$.

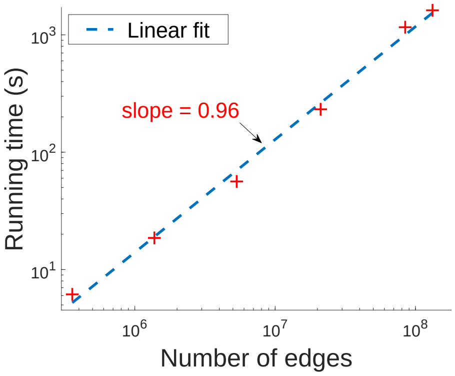


Figure 6: DPGS is scalable. The running time scales linearly with the number of edges.

that DPGS yields better summary graphs compared to several state-of-the-art methods. Moreover, we show that the summary graphs can be used to train GNN models with much less computational resources while maintaining comparable performance.

Acknowledgements

This paper is partially supported by Strategic Priority Research Program of Chinese Academy of Sciences, Grant No. XDA19020400, NSF of China No. 61772498, U1911401, 61872206, 91746301, and 2020 Tencent Wechat Rhino-Bird Focused Research Program.

References

[1] B. Adhikari, Y. Zhang, A. Bharadwaj, and B. A. Prakash, *Condensing temporal networks using propagation*, in ICDM, 2017.

[2] A. Andoni and P. Indyk, *Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions*, in FOCS, 2006.

[3] M. Araujo, S. Günnemann, G. Mateos, and C. Faloutsos, *Beyond blocks: Hyperbolic community detection*, in ECML-PKDD, 2014.

[4] M. A. Beg, M. Ahmad, A. Zaman, and I. Khan, *Scalable approximation algorithm for graph summarization*, in PAKDD, 2018.

[5] G. Cormode and S. Muthukrishnan, *An improved data stream summary: the count-min sketch and its applications*, Journal of Algorithms, (2005), pp. 58–75.

[6] I. S. Dhillon and S. Sra, *Generalized nonnegative matrix approximations with bregman divergences*, in NIPS, 2005.

[7] P. Erdös and A. Rényi, *On random graphs i*, Publicationes Mathematicae Debrecen, 6 (1959), p. 290.

[8] N. Hassanlou, M. Shoaran, and A. Thomo, *Probabilistic graph summarization*, in WAIM, 2013.

[9] K. Henderson, B. Gallagher, T. Eliassi-Rad, H. Tong, S. Basu, L. Akoglu, D. Koutra, C. Faloutsos, and L. Li, *Rolx: Structural role extraction & mining in large graphs*, in KDD, 2012.

[10] K. U. Khan, W. Nawaz, and Y.-K. Lee, *Set-based unified approach for summarization of a multi-attributed graph*, WWW, 20 (2017), pp. 543–570.

[11] J. M. Kleinberg, *Authoritative sources in a hyperlinked environment*, in SODA, 1998.

[12] J. Ko, Y. Kook, and K. Shin, *Incremental lossless graph summarization*, in KDD, 2020.

[13] D. Koutra, U. Kang, J. Vreeken, and C. Faloutsos, *Vog: Summarizing and understanding large graphs*, in SDM, 2014.

[14] K. Lee, H. Jo, J. Ko, S. Lim, and K. Shin, *Ssumm: Sparse summarization of massive graphs*, in KDD, 2020.

[15] K. LeFevre and E. Terzi, *Grass: Graph structure summarization*, in ICDM, 2010.

[16] Y. Liu, T. Safavi, A. Dighe, and D. Koutra, *Graph summarization methods and applications: A survey*, ACM Computing Surveys, 51 (2018), pp. 1–34.

[17] S. Navlakha, R. Rastogi, and N. Shrivastava, *Graph summarization with bounded error*, in SIGMOD, 2008.

[18] M. Newman, *Networks: An Introduction*, Oxford University Press, Inc., 2010.

[19] M. E. J. Newman, *Modularity and community structure in networks*, PNAS, 103 (2006), pp. 8577–8582.

[20] Q. Qu, S. Liu, C. S. Jensen, F. Zhu, and C. Faloutsos, *Interestingness-driven diffusion process summarization in dynamic networks*, in ECML-PKDD, 2014.

[21] M. Riondato, D. García-Soriano, and F. Bonchi, *Graph summarization with quality guarantees*, Data mining and knowledge discovery, (2017), pp. 314–349.

[22] J. Rissanen, *Modeling by shortest data description*, Automatica, 14 (1978), pp. 465–471.

[23] N. Shah, D. Koutra, T. Zou, B. Gallagher, and C. Faloutsos, *Timecrunch: Interpretable dynamic graph summarization*, in KDD, 2015.

[24] K. Shin, A. Ghoting, M. Kim, and H. Raghavan, *Sweg: Lossless and lossy summarization of web-scale graphs*, in WWW, 2019.

[25] N. Tang, Q. Chen, and P. Mitra, *Graph stream summarization: From big bang to big crunch*, in SIGMOD, 2016.

[26] Y. Wu, Z. Zhong, W. Xiong, and N. Jing, *Graph summarization for attributed graphs*, in ICIEE, 2014.

[27] H. Zeng, H. Zhou, A. Srivastava, R. Kannan, and V. Prasanna, *GraphSAINT: Graph sampling based inductive learning method*, in ICLR, 2020.