Variational Graph Contrastive Learning

Shifeng Xie
LTCI, Télécom Paris
Institut Polytechnique de Paris
`shifeng.xie@telecom-paris.fr`

Jhony H. Giraldo
LTCI, Télécom Paris
Institut Polytechnique de Paris
`jhony.giraldo@telecom-paris.fr`

Abstract

Graph representation learning (GRL) is a fundamental task in machine learning, aiming to encode high-dimensional graph-structured data into low-dimensional vectors. Self-supervised learning (SSL) methods are widely used in GRL because they can avoid expensive human annotation. In this work, we propose a novel Subgraph Gaussian Embedding Contrast (SGEC) method. Our approach introduces a subgraph Gaussian embedding module, which adaptively maps subgraphs to a structured Gaussian space, ensuring the preservation of graph characteristics while controlling the distribution of generated subgraphs. We employ optimal transport distances, including Wasserstein and Gromov-Wasserstein distances, to effectively measure the similarity between subgraphs, enhancing the robustness of the contrastive learning process. Extensive experiments across multiple benchmarks demonstrate that SGEC outperforms or presents competitive performance against state-of-the-art approaches. Our findings provide insights into the design of SSL methods for GRL, emphasizing the importance of the distribution of the generated contrastive pairs[1].

1 Introduction

Graph representation learning (GRL) aims to effectively encode high-dimensional sparse graph-structured data into low-dimensional dense vectors, which is a fundamental task that has been widely studied in a range of fields, including machine learning and data mining [Ju et al., 2024].

Self-supervised learning (SSL) offers a promising approach to GRL by mitigating the need for extensive human annotation [Jaiswal et al., 2020]. Particularly, contrastive learning is a prominent approach in SSL that leverages the similarities and differences between data samples or data embeddings to learn representations. In this context, positive sample pairs are typically two augmented views of the same data point that should be close in the representation space, while negative sample pairs are original and augmented views of different data samples [Chen et al., 2020]. Current graph-based contrastive learning methods primarily generate positive and negative sample pairs through perturbations [Zhu et al., 2020a][2]. However, t-SNE visualizations of current graph-based contrastive learning methods, like GCA [Zhu et al., 2021] and GSC [Han et al., 2022], reveal uneven node distributions within the same graph, with sharp boundaries and erroneous node clusters as illustrated in Figure 1. We observe that those characteristics of previous models penalize their performance in GRL tasks.

In this paper, we propose the Subgraph Gaussian Embedding Contrast (SGEC) model. In our method, a subgraph Gaussian embedding (SGE) module is proposed to generate the features of the subgraphs. The SGE maps input subgraphs to a structured Gaussian space, where the features of the output subgraphs tend towards a Gaussian distribution by using the Kullback–Leibler (KL) divergence. SGE is a learnable mapping that controls the distribution of the embeddings. The

[1]Our code is provided at `https://github.com/ShifengXIE/SGEC`
[2]A complete review of the related work can be found in Appendix A.

NeurIPS 2024 Workshop: Self-Supervised Learning– Theory and Practice.

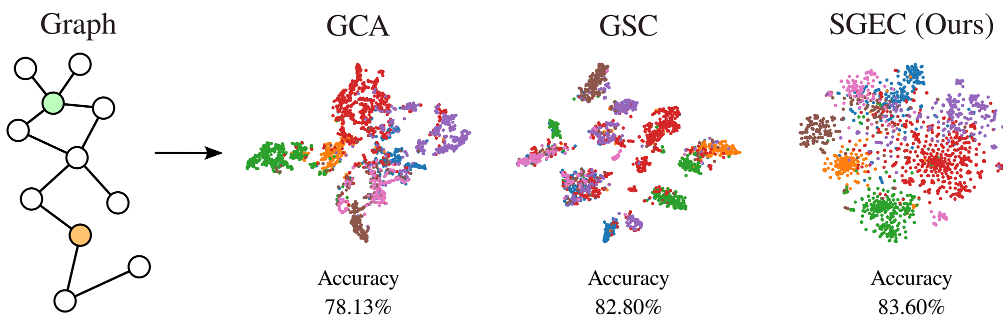

Figure 1: T-SNE visualizations of the graph contrastive learning method GCA [Zhu et al., 2021], GSC [Han et al., 2022], and our method SGEC on the Cora dataset [McCallum et al., 2000]. Each point in the visualization corresponds to a node embedding, with colors indicating classes. SGEC maps the node representations into a dense, uniform, and more linearly separable space.

embedded subgraphs, paired with the original subgraphs to form positive and negative pairs, then conduct contrastive learning with optimal transport measurements to obtain the graph representation. The embedded subgraphs offer diversity to prevent mode collapse (also called positive collapse [Jing et al., 2022]) where the embeddings shrink into a low-dimensional subspace, by controlling the embedding distribution.

For our contributions, firstly, we propose the Subgraph Gaussian Embedding Contrast (SGEC) model that demonstrates its advantages across eight distinct benchmarks. We also provide insights into the importance of the distribution of subgraphs generated for contrastive positive and negative pairs. Finally, we provide different validation and ablation studies to assess the effectiveness of each design choice in SGEC.

2 Subgraph Gaussian Embedding Contrast (SGEC) Method

Mathematical Notation. Consider an undirected graph $G = (\mathcal{V}, \mathcal{E})$ with vertex set $\mathcal{V}$ and edge set $\mathcal{E}$. The feature matrix $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^\top \in \mathbb{R}^{N\times C}$ contains node features $\mathbf{x}_i \in \mathbb{R}^C$, where N is the number of nodes and C is the feature dimension. The adjacency matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$ represents the graph topology, and $\mathbf{D}$ is the diagonal degree matrix. For some node i, let $H^i = (\mathcal{V}^i, \mathcal{E}^i)$ be its induced breadth-first search (BFS) subgraph with k nodes, adjacency matrix $\mathbf{A}^i \in \mathbb{R}^{k\times k}$, and feature matrix $\mathbf{X}^i \in \mathbb{R}^{k\times C}$. Our method embeds this subgraph into a structured space, yielding the embedded subgraph $\tilde{H}^i = (\mathcal{V}^i, \tilde{\mathcal{E}}^i)$ with adjacency matrix $\tilde{\mathbf{A}}^i$ and feature matrix $\tilde{\mathbf{X}}^i$. We provide the definitions of KL divergence, Wasserstein, and Gromov-Wasserstein distances in Appendix B.

Overview of the Proposed Method. Figure 2 shows an overview of our methodology, where our process begins with an encoder of the input graph. Subsequently, we obtain subgraphs utilizing BFS-based sampling. The node representations and topology information within these subgraphs are adaptively embedded into a latent space tending towards a Gaussian distribution, which is enforced by the KL divergence regularization. Finally, we use the Wasserstein and the Gromov-Wasserstein distance to measure dissimilarities in the subgraphs for contrastive learning.

Graph Encoder. We begin by employing a graph encoder to preprocess the graph data [Kingma and Welling, 2014, Han et al., 2022]. The graph encoder comprises some graph convolutions layers and activation functions $\sigma(\cdot)$. We use two graph convolution layers [Kipf and Welling, 2017] in SGEC. Further details on the implementation of these layers are provided in the Appendix B.3.

Subgraph Gaussian Embedding (SGE). The SGE module comprises GraphSAGE [Liu et al., 2020] and GAT (graph attention) [Veličković et al., 2018] layers. Appendix B.4 provides further details about GraphSAGE. We first randomly sample a set of nodes $\mathcal{S}$ and get their BFS-induced subgraphs, where $|\mathcal{S}|$ is a hyperparameter of SGEC. We then apply the combination of GraphSAGE and GAT layers on these subgraphs. The latent means and variances are encoded by separate GAT networks as follows:

$$\boldsymbol{\mu}_i = \text{GATConv}_\mu(\mathbf{h}_i), \quad \log \boldsymbol{\sigma}_i^2 = \text{GATConv}_\sigma(\mathbf{h}_i), \quad \tilde{\mathbf{x}}_i = \boldsymbol{\mu}_i + \exp(\log(\boldsymbol{\sigma_i})) \odot \boldsymbol{\epsilon},$$

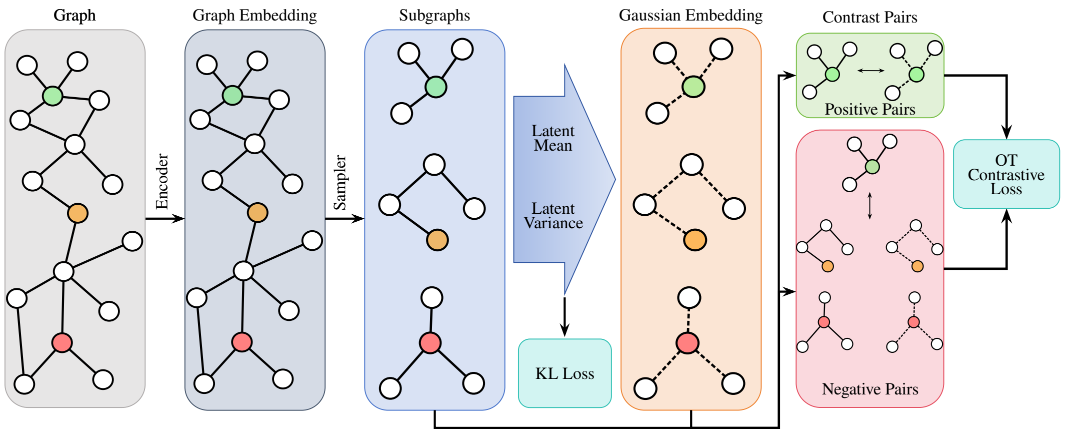

Figure 2: Overview of the Subgraph Gaussian Embedding Contrast (SGEC) method. The SGEC method begins with a graph encoder to generate embeddings and employs a breadth-first search to sample subgraphs for some set of nodes $\mathcal{S}$. A Gaussian embedding module then produces contrastive samples in a Gaussian space of these subgraphs. Positive pairs consist of subgraphs with the same central node, while negative pairs have different central nodes. SGEC introduces the Wasserstein and Gromov-Wasserstein distances to compute distances between subgraphs for contrastive learning.

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ represents Gaussian noise drawn from the standard normal distribution, $\mathbf{h}_i$ represents the output of the last GraphSAGE layer, $\tilde{\mathbf{x}}_i$ denote the embedded features of node i. Note that i is some node in the set of $|\mathcal{S}|$ induced subgraphs. In this configuration, $\boldsymbol{\mu}_i$ and $\log \boldsymbol{\sigma}_i^2$ are derived through separate GAT layers. SGE effectively capture both the node attributes and the graph topology. In our method, we preserve the original graph topology, so the embedded subgraphs have the same adjacency matrices as the original subgraphs.

Regularization in the Subgraph Gaussian Embedding. In our approach, we introduce a regularization constraint to the SGE module to guide the embedded subgraph node features towards a Gaussian distribution. For the motivation of introducing the Gaussian regularization, please see the Appendix C. This regularization is implemented using the Kullback-Leibler divergence, denoted as $\mathrm{KL}(q(\cdot)\|p(\cdot))$, between the embedding distribution $q(\cdot)$ and a predefined Gaussian prior. The prior $p(\tilde{\mathbf{X}})$ is taken as a product of independent Gaussian distributions for each latent variable $\tilde{\mathbf{x}}_i$, given by $p(\tilde{\mathbf{x}}_i) = \mathcal{N}(\tilde{\mathbf{x}}_i|\mathbf{0}, \mathbf{I})$. The expression for the KL divergence then simplifies to:

$$\mathrm{KL}\left(q(\tilde{\mathbf{X}} \mid \mathbf{X}, \mathbf{A}) \,\|\, p(\tilde{\mathbf{X}})\right) = \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} \sum_{j=1}^{\tilde{C}} \left(\boldsymbol{\mu}_{ij}^2 + \boldsymbol{\sigma}_{ij}^2 - 1 - 2\log \boldsymbol{\sigma}_{ij}\right),$$

where $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}^2$ are the latent mean and latent variance of the ith embedded node, respectively, and $\mathcal{P}$ is the set of nodes in the $|\mathcal{S}|$ induced subgraphs. $\tilde{C}$ is the dimension of the latent features.

Optimal Transport Contrastive Loss. Our contrastive loss function integrates the Wasserstein and Gromov-Wasserstein distances into the InfoNCE loss formulation [van den Oord et al., 2018], addressing the complexities of graph-based data. The Wasserstein distance, $W(\mathbf{X}^i, \tilde{\mathbf{X}}^i)$, captures feature distribution representation within subgraphs, whereas the Gromov-Wasserstein distance, $GW(\mathbf{A}^i, \mathbf{X}^i, \mathbf{A}^i, \tilde{\mathbf{X}}^i)$, captures structural discrepancies, providing a topology-aware similarity measure. The complete contrastive loss $\mathcal{L}_{\text{contrast}}$ is the sum of $\mathcal{L}_{\text{W}}$ and $\mathcal{L}_{\text{GW}}$ provided as follows:

$$\mathcal{L}_{\text{W}} = \alpha \left(-\sum_{i\in\mathcal{S}} \log \frac{\exp(-W(\mathbf{X}^i,\tilde{\mathbf{X}}^i)/\tau)}{\sum_{j\in\mathcal{S},j\neq i}^{N}\left(\exp(-W(\mathbf{X}^i,\tilde{\mathbf{X}}^j)/\tau)+\exp(-W(\mathbf{X}^i,\mathbf{X}^j)/\tau)\right)}\right),$$

$$\mathcal{L}_{\text{GW}} = (1-\alpha) \left(-\sum_{i\in\mathcal{S}} \log \frac{\exp(-GW(\mathbf{A}^i,\mathbf{X}^i,\mathbf{A}^i,\tilde{\mathbf{X}}^i)/\tau)}{\sum_{j\in\mathcal{S},j\neq i}^{N}\left(\exp(-GW(\mathbf{A}^i,\mathbf{X}^i,\mathbf{A}^j,\tilde{\mathbf{X}}^j)/\tau)+\exp(-GW(\mathbf{A}^i,\mathbf{X}^i,\mathbf{A}^j,\mathbf{X}^j)/\tau)\right)}\right),$$

where α is a hyperparameter that balances the emphasis on feature distribution and structural fidelity, and τ is a temperature hyperparameter. The final loss $\mathcal{L}$ of our model incorporates both contrastive and regularization components balanced by a hyperparameter β as follows: $\mathcal{L} = \mathcal{L}_{\text{contrast}} + \beta \mathrm{KL}(q(\tilde{\mathbf{X}} \mid \mathbf{X}, \mathbf{A})\|p(\tilde{\mathbf{X}}))$.

Table 1: Comparison (accuracy %) of SGEC against state-of-the-art SSL method for graphs.

Method	Cora	Citeseer	Pubmed	Coauthor	Squirrel	Chameleon	Cornell	Texas
MUSE	69.90$_{\pm 0.41}$	66.35$_{\pm 0.40}$	79.95$_{\pm 0.59}$	90.75$_{\pm 0.39}$	40.15$_{\pm 3.04}$	55.59$_{\pm 2.21}$	83.78$_{\pm 3.42}$	83.78$_{\pm 2.79}$
GREET	**84.40**$_{\pm 0.77}$	**74.10**$_{\pm 0.44}$	80.29$_{\pm 0.24}$	**94.65**$_{\pm 0.18}$	39.76$_{\pm 0.75}$	60.57$_{\pm 1.03}$	78.36$_{\pm 3.77}$	78.03$_{\pm 3.94}$
GRACE	83.30$_{\pm 0.74}$	72.10$_{\pm 0.60}$	**86.70**$_{\pm 0.16}$	92.78$_{\pm 0.04}$	52.10$_{\pm 0.94}$	52.29$_{\pm 1.49}$	60.66$_{\pm 11.32}$	75.74$_{\pm 2.95}$
GSC	82.80$_{\pm 0.10}$	71.00$_{\pm 0.10}$	85.60$_{\pm 0.20}$	91.88$_{\pm 0.11}$	51.32$_{\pm 0.21}$	64.02$_{\pm 0.29}$	93.56$_{\pm 1.73}$	88.64$_{\pm 1.21}$
DGI	81.99$_{\pm 0.95}$	71.76$_{\pm 0.80}$	77.16$_{\pm 0.24}$	92.15$_{\pm 0.63}$	38.80$_{\pm 0.76}$	58.00$_{\pm 0.70}$	70.82$_{\pm 7.21}$	81.48$_{\pm 2.79}$
GCA	78.13$_{\pm 0.85}$	67.81$_{\pm 0.75}$	80.63$_{\pm 0.31}$	93.10$_{\pm 0.20}$	47.13$_{\pm 0.61}$	56.54$_{\pm 1.07}$	53.11$_{\pm 9.34}$	81.02$_{\pm 2.30}$
GraphMAE	84.20$_{\pm 0.40}$	73.40$_{\pm 0.40}$	81.10$_{\pm 0.40}$	80.63$_{\pm 0.15}$	48.26$_{\pm 1.21}$	**71.05**$_{\pm 0.36}$	61.93$_{\pm 4.59}$	67.80$_{\pm 3.37}$
SGEC (Ours)	83.60$_{\pm 0.10}$	73.14$_{\pm 0.14}$	84.60$_{\pm 0.10}$	92.34$_{\pm 0.04}$	**56.39**$_{\pm 0.57}$	69.14$_{\pm 1.12}$	**94.58**$_{\pm 2.13}$	**92.38**$_{\pm 0.81}$

Table 2: Validation and ablation studies on different modules of SGEC.

Methodology	Cora	Citeseer	Pubmed	Coauthor	Squirrel	Chameleon	Cornell	Texas
W.O. Regularization	83.00$_{\pm 0.07}$	71.88$_{\pm 0.07}$	**85.46**$_{\pm 0.04}$	91.96$_{\pm 0.09}$	42.99$_{\pm 0.17}$	64.25$_{\pm 0.21}$	**94.58**$_{\pm 0.22}$	75.72$_{\pm 0.40}$
With Decoder	82.80$_{\pm 0.07}$	73.00$_{\pm 0.08}$	80.26$_{\pm 0.37}$	92.03$_{\pm 0.13}$	35.47$_{\pm 0.21}$	60.30$_{\pm 0.21}$	93.56$_{\pm 0.64}$	88.98$_{\pm 0.52}$
With reconstruction	81.60$_{\pm 0.99}$	79.60$_{\pm 0.10}$	67.54$_{\pm 0.35}$	87.03$_{\pm 0.65}$	30.52$_{\pm 0.48}$	43.26$_{\pm 0.66}$	53.29$_{\pm 0.13}$	63.45$_{\pm 0.48}$
Dropout	79.00$_{\pm 0.21}$	70.60$_{\pm 3.52}$	80.84$_{\pm 0.05}$	91.52$_{\pm 0.37}$	44.24$_{\pm 0.50}$	58.94$_{\pm 0.72}$	85.76$_{\pm 0.24}$	87.98$_{\pm 0.15}$
SGEC (Ours)	**83.60**$_{\pm 0.10}$	**73.14**$_{\pm 0.14}$	84.60$_{\pm 0.10}$	**92.34**$_{\pm 0.04}$	**56.39**$_{\pm 0.57}$	**69.14**$_{\pm 1.12}$	94.57$_{\pm 2.13}$	**92.38**$_{\pm 0.81}$

3 Experimental Evaluation

Comparison of Classification Performance. We compare our model against state-of-the-art SSL for node classification methods: GREET [Liu et al., 2023], GRACE [Zhu et al., 2020a], GSC [Han et al., 2022], and MUSE [Yuan et al., 2023]. Additionally, we include classic SSL baselines for a comprehensive comparison: DGI [Veličković et al., 2019], GCA [Zhu et al., 2021], and GraphMAE [Hou et al., 2022]. We test SGEC on standard benchmark datasets for node classification Cora, Citeseer, Pubmed, Coauthor, Squirrel, Chameleon, Cornell, and Texas [Giraldo et al., 2023]. Please see the Appendix D for more information about the datasets and experimental setup. The evaluation results for these eight datasets are summarized in Table 1. We observe that SGEC achieves the highest accuracies on the Squirrel, Cornell, and Texas datasets, outperforming existing state-of-the-art SSL methods. While not always leading on other datasets, SGEC consistently demonstrates competitive performance, highlighting its robustness and effectiveness in GRL.

Validation and Ablation Studies. We design four network configurations for validation and ablation studies of SGEC in Table 2. The first, termed "W.O. Regularization", omits the KL divergence component, retaining the original network structure but mapping node features into an unstructured space. This ablation demonstrates the efficacy of the added regularization term encouraging a Gaussian distribution. The second study, "with decoder", incorporates a decoder of two fully connected layers. This allows the contrastive learning loss to substitute for the reconstruction loss and effectively models the network as a Variational Autoencoder (VAE) generating contrastive pairs. This study illustrates that SGEC is not merely a combination of a VAE model with contrastive learning. The third study, "with reconstruction", includes an ℓ_1 reconstruction loss calculated as the norm of the difference between input and output features, adding a constraint to enforce similarity between them. The fourth study, "Dropout", replaces the KL divergence with dropout as the regularization technique. Our method outperforms dropout, validating its superiority in SSL tasks for GRL.

4 Conclusion

We introduced SGEC, which embeds subgraphs into a Gaussian space to enhance self-supervised graph representation learning. By controlling embedding distributions and utilizing optimal transport distances, SGEC achieves superior or competitive performance. Our results emphasize the importance of distribution control in generating contrastive pairs.

Future work will involve integrating spectral-based contrastive learning methods [Chen et al., 2024] to further validate and enhance our proposed approach. Additionally, we intend to explore the applicability of our framework to other modalities, extending beyond graph data. This would involve validating our underlying ideas in different contexts, potentially broadening the impact and utility of our approach in various domains of representation learning.

References

Wei Ju, Zheng Fang, Yiyang Gu, Zequn Liu, Qingqing Long, Ziyue Qiao, Yifang Qin, Jianhao Shen, Fang Sun, Zhiping Xiao, Junwei Yang, Jingyang Yuan, Yusheng Zhao, Yifan Wang, Xiao Luo, and Ming Zhang. A comprehensive survey on deep graph representation learning. *Neural Networks*, 2024.

A Jaiswal, A Ramesh Babu, M Zaki Zadeh, D Banerjee, and F Makedon. A survey on contrastive self-supervised learning. *Technologies*, 2020.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*, 2020.

Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Deep graph contrastive representation learning. In *International Conference on Machine Learning - Workshop*, 2020a.

Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Graph contrastive learning with adaptive augmentation. In *The Web Conference*, 2021.

Yuehui Han, Le Hui, Haobo Jiang, Jianjun Qian, and Jin Xie. Generative subgraph contrast for self-supervised graph representation learning. In *European Conference on Computer Vision*, 2022.

Li Jing, Pascal Vincent, Yann LeCun, and Yuandong Tian. Understanding dimensional collapse in contrastive self-supervised learning. In *International Conference on Learning Representations*, 2022.

Andrew Kachites McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval*, 2000.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

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

Jielun Liu, Ghim Ping Ong, and Xiqun Chen. GraphSAGE-based traffic speed forecasting for segment network with sparse data. *IEEE Transactions on Intelligent Transportation Systems*, 2020.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Yixin Liu, Yizhen Zheng, Daokun Zhang, Vincent Lee, and Shirui Pan. Beyond smoothing: Unsupervised graph representation learning with edge heterophily discriminating. In *AAAI Conference on Artificial Intelligence*, 2023.

Mengyi Yuan, Minjie Chen, and Xiang Li. MUSE: Multi-view contrastive learning for heterophilic graphs. In *ACM International Conference on Information and Knowledge Management*, 2023.

Petar Veličković, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. In *International Conference on Learning Representations*, 2019.

Zhenyu Hou, Xiao Liu, Yukuo Cen, Yuxiao Dong, Hongxia Yang, Chunjie Wang, and Jie Tang. Graphmae: Self-supervised masked graph autoencoders. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2022.

Jhony H Giraldo, Konstantinos Skianis, Thierry Bouwmans, and Fragkiskos D Malliaros. On the trade-off between over-smoothing and over-squashing in deep graph neural networks. In *ACM International Conference on Information and Knowledge Management*, 2023.

Jingyu Chen, Runlin Lei, and Zhewei Wei. PolyGCL: GRAPH CONTRASTIVE LEARNING via learnable spectral polynomial filters. In *International Conference on Learning Representations*, 2024.

Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in Neural Information Processing Systems*, 2017.

Yixin Liu, Ming Jin, Shirui Pan, Chuan Zhou, Yu Zheng, Feng Xia, and S Yu Philip. Graph self-supervised learning: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 2022.

Chun Wang, Shirui Pan, Guodong Long, Xingquan Zhu, and Jing Jiang. MGAE: Marginalized graph autoencoder for graph clustering. In *ACM International Conference on Information and Knowledge Management*, 2017.

Yanqiao Zhu, Yichen Xu, Feng Yu, Shu Wu, and Liang Wang. CAGNN: Cluster-aware graph neural networks for unsupervised graph representation learning. *arXiv preprint arXiv:2009.01674*, 2020b.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. DeepWalk: Online learning of social representations. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2014.

Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

Ziniu Hu, Yuxiao Dong, Kuansan Wang, Kai-Wei Chang, and Yizhou Sun. GPT-GNN: Generative pre-training of graph neural networks. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2020.

Yuning You, Tianlong Chen, Zhangyang Wang, and Yang Shen. When does self-supervision help graph convolutional networks? In *International Conference on Machine Learning*, 2020.

Ke Sun, Zhouchen Lin, and Zhanxing Zhu. Multi-stage self-supervised learning for graph convolutional networks on graphs with few labeled nodes. In *AAAI Conference on Artificial Intelligence*, 2020.

Alex Pothen. Graph partitioning algorithms with applications to scientific computing. In *Parallel Numerical Algorithms*. 1997.

Volker Roth, Julian Laub, Motoaki Kawanabe, and Joachim M Buhmann. Optimal cluster preserving embedding of nonmetric proximity data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2003.

Dorian Gailhard, Enzo Tartaglione, Lirida Naviner De Barros, and Jhony H Giraldo. HYGENE: A diffusion-based hypergraph generation method. *arXiv preprint arXiv:2408.16457*, 2024.

Teng Xiao, Huaisheng Zhu, Zhiwei Zhang, Zhimeng Guo, Charu C Aggarwal, and Suhang Wang. GraphECL: Towards efficient contrastive learning for graphs. 2024.

Ludger Rüschendorf. The Wasserstein distance and approximation theorems. *Probability Theory and Related Fields*, 1985.

Shreya Arya, Arnab Auddy, Ranthony Edmonds, Sunhyuk Lim, Facundo Memoli, and Daniel Packer. The Gromov-Wasserstein distance between spheres. *Foundations of Computational Mathematics*, 2024.

Tim Van Erven and Peter Harremos. Rényi divergence and Kullback-Leibler divergence. *IEEE Transactions on Information Theory*, 2014.

C. Lee Giles, Kurt D. Bollacker, and Steve Lawrence. CiteSeer: an automatic citation indexing system. In *ACM Conference on Digital Libraries*, 1998.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI Magazine*, 2008.

Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. *Advances in Neural Information Processing Systems - Workshop*, 2018.

Benedek Rozemberczki, Carl Allen, and Rik Sarkar. Multi-scale attributed node embedding. *Journal of Complex Networks*, 2021.

Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-GCN: Geometric graph convolutional networks. In *International Conference on Learning Representations*.

Mark Craven, Dan DiPasquo, Dayne Freitag, Andrew McCallum, Tom Mitchell, Kamal Nigam, and Seán Slattery. Learning to extract symbolic knowledge from the world wide web. AAAI Conference on Artificial Intelligence, 1998.

Johannes Gasteiger, Stefan Weißenberger, and Stephan Günnemann. Diffusion improves graph learning. *Advances in Neural Information Processing Systems*, 2019.

Jake Topping, Francesco Di Giovanni, Benjamin Paul Chamberlain, Xiaowen Dong, and Michael M. Bronstein. Understanding over-squashing and bottlenecks on graphs via curvature. In *International Conference on Learning Representations*, 2022.

Appendix

In the appendix, we provide supplementary material that includes a concise review of related work on graph representation learning and self-supervised learning on graphs. We also present essential mathematical preliminaries for our approach and discuss the motivation for introducing KL regularization into our method. Finally, we offer detailed experimental evaluations and dataset descriptions, implementation details, and sensitivity analyses of the regularization constraint and subgraph size.

A Related Work

Graph Neural Networks.

Graph Neural Networks (GNNs) play a crucial role in learning representations that effectively capture both node features and graph topology [Wu et al., 2021]. Among various architectures, Graph Convolutional Networks (GCNs), pioneered by Kipf and Welling [2017], utilize a special approach inspired by spectral graph convolutions. GraphSAGE, introduced by Hamilton et al. [2017], employs multiple aggregator functions and enables inductive learning on graphs. Veličković et al. [2018] proposed Graph Attention Networks (GAT), which incorporate attention mechanisms to dynamically weigh the significance of neighboring nodes.

Self-Supervised Learning on Graphs. Self-supervised learning (SSL) on graphs can be categorized into four main frameworks [Liu et al., 2022]: generation-based [Kingma and Welling, 2014, Wang et al., 2017], auxiliary property-based [Zhu et al., 2020b], contrastive-based [Perozzi et al., 2014, Grover and Leskovec, 2016], and hybrid methods [Hu et al., 2020], each utilizing distinct pretext tasks and objective functions. Generation-based methods like Graph Autoencoders (GAEs) [Kingma and Welling, 2014] use node attributes and graph structure for learning representations. In auxiliary property-based SSL [Zhu et al., 2020b], properties such as node roles are predicted to facilitate learning, while contrastive methods like *Deep Graph Infomax* (DGI) [Veličković et al., 2019] maximize mutual information between different graph patches. Hybrid approaches combine these strategies to enhance model robustness and adaptability [Liu et al., 2022].

Generative Self-supervised Learning on Graphs. Graph data requires additional processing compared to standard vector-based data due to the interrelationship of edge and node features. Generative SSL methods often take full graphs or subgraphs as input, aiming to reconstruct either node features or the graph's topology. This splits into feature generation focusing on node or edge features [You et al., 2020, Sun et al., 2020] and structure generation [Pothen, 1997, Roth et al., 2003, Gailhard et al., 2024] targeting the graph's topological aspects, thus deepening the understanding of graph properties crucial for downstream tasks.

Augmented Contrastive Learning. Innovative methodologies in augmented contrastive learning for graphs have included: 1) graph augmentations that generate diverse graph instances [Xiao et al., 2024]; 2) various contrastive pretext tasks designed for non-Euclidean spaces [Han et al., 2022]; and 3) mutual information estimation techniques that underpin learning objectives alongside specific pretext tasks [Veličković et al., 2019, Zhu et al., 2021]. These approaches leverage the structural nuances of graphs to improve the quality and utility of the learned representations.

Positioning of SGEC. Our approach integrates subgraph feature generation with optimal transport measurement to perform subgraph-wise contrastive learning. In our approach, the features of the sampled subgraphs are generated by SGE, compared with generative SSL methods, we adopt contrastive learning as the training paradigm. Furthermore, we use the Wasserstein and Gromov-Wasserstein distances to measure the distance between subgraphs. To the best of our knowledge, the most similar approach to ours is GSC [Han et al., 2022], focusing on the distribution of generated features, our method aims to avoid the sparse of generated node vector features, achieving superior graph representations. This methodology not only captures the intrinsic properties of nodes and topological information but also maintains a distribution that enhances generalizability and effectiveness in downstream applications.

B Preliminaries

B.1 Optimal Transport Distance

The Wasserstein distance [Rüschendorf, 1985], commonly used in optimal transport theory, serves as a robust metric for comparing probability distributions defined over metric spaces. For subgraphs H^i and H^j, their corresponding feature matrices are denoted as $\mathbf{X}^i \in \mathbb{R}^{k^i \times C}$ and $\mathbf{X}^j \in \mathbb{R}^{k^j \times C}$, where k^i and k^j represent the number of nodes in the respective subgraphs, and C is the feature dimension. The Wasserstein distance between the feature distributions of these subgraphs is defined as:

$$W(\mathbf{X}^i, \mathbf{X}^j) = \min_{\mathbf{T} \in \pi(\mathbf{u}, \mathbf{v})} \sum_{p=1}^{k^i} \sum_{q=1}^{k^j} T_{pq} c(\mathbf{X}_p^i, \mathbf{X}_q^j),$$

where $\mathbf{T} \in \pi(\mathbf{u}, \mathbf{v})$ represents the set of all possible transport plans with marginals $\mathbf{u}$ and $\mathbf{v}$, which are the node feature distributions in subgraphs H^i and H^j, respectively. T_{pq} is the transportation plan between nodes p and q, and $c(\mathbf{X}_p^i, \mathbf{X}_q^j)$ represents the cost function, often defined as $\exp\left(-\frac{\langle \mathbf{X}_p^i, \mathbf{X}_q^j \rangle}{\tau}\right)$, where $\langle \cdot, \cdot \rangle$ denotes the cosine similarity between node features, and τ is a temperature parameter. This distance captures the minimal cost of transforming one subgraph's node feature distribution into another.

Similarly, the Gromov-Wasserstein distance [Arya et al., 2024] extends this idea to compare graph-structured data, where the internal distances between nodes are taken into account. For two subgraphs H^i and H^j with adjacency matrices $\mathbf{A}^i \in \mathbb{R}^{k^i \times k^i}$ and $\mathbf{A}^j \in \mathbb{R}^{k^j \times k^j}$, and feature matrices $\mathbf{X}^i$ and $\mathbf{X}^j$, the Gromov-Wasserstein distance is defined as:

$$GW(\mathbf{A}^i, \mathbf{A}^j, \mathbf{X}^i, \mathbf{X}^j) = \min_{\mathbf{T} \in \pi(\mathbf{u}, \mathbf{v})} \sum_{p, \tilde{p}, q, \tilde{q}} T_{pq} T_{\tilde{p}\tilde{q}} \left| d_{\mathbf{A}^i}(\mathbf{X}_p^i, \mathbf{X}_{\tilde{p}}^i) - d_{\mathbf{A}^j}(\mathbf{X}_q^j, \mathbf{X}_{\tilde{q}}^j) \right|,$$

where $d_{\mathbf{A}^i}(\mathbf{X}_p^i, \mathbf{X}_{\tilde{p}}^i)$ and $d_{\mathbf{A}^j}(\mathbf{X}_q^j, \mathbf{X}_{\tilde{q}}^j)$ represent the shortest path distances between node pairs $(p, \tilde{p})$ in subgraph H^i, and $(q, \tilde{q})$ in subgraph H^j, respectively. The matrix $\mathbf{T} \in \pi(\mathbf{u}, \mathbf{v})$ is the optimal transport plan that matches node pairs from the two subgraphs. This metric assesses the discrepancy between the internal node pair distances of the two subgraphs, aligning their intrinsic geometric structures.

B.2 The Kullback-Leibler (KL) Divergence

The Kullback-Leibler (KL) divergence is an asymmetry measure between two probability distributions [Van Erven and Harremos, 2014], P and Q. It quantifies the amount of information lost when Q is used to approximate P. In mathematical terms, the KL divergence is defined as:

$$D_{KL}(P \| Q) = \sum_{x \in \mathcal{X}} P(x) \log \frac{P(x)}{Q(x)},$$

where $P(x)$ and $Q(x)$ are the probability masses of P and Q at each point x in the sample space $\mathcal{X}$.

B.3 Graph Convolutional Network

The graph encoder uses two graph convolution layers, which are mathematically represented as follows:

$$\mathbf{H}_1 = \sigma\left((\mathbf{D}^{-\frac{1}{2}}(\mathbf{A} + \mathbf{I})\mathbf{D}^{-\frac{1}{2}}\mathbf{X}\mathbf{W}_1\right), \quad \mathbf{H}_2 = \sigma\left(\mathbf{D}^{-\frac{1}{2}}(\mathbf{A} + \mathbf{I})\mathbf{D}^{-\frac{1}{2}}\mathbf{H}_1\mathbf{W}_2\right).$$

B.4 GraphSAGE Layer

For GraphSAGE, we present expressions:

$$\mathbf{x}_v^{(l+1)} = \sigma\left(\mathbf{W}^{(l)} \cdot \text{MEAN}\left(\mathbf{x}_v^{(l)}, \{\mathbf{x}_u^{(l)} \mid u \in \mathcal{N}(v)\}\right) + \mathbf{b}^{(l)}\right),$$

where $\mathbf{x}_v^{(l)}$ is the feature vector of node v at layer l, $\mathbf{W}^{(l)}$ is the weight matrix at layer l, $\mathcal{N}(v)$ denotes the set of neighbors of node v.

Table 3: Overview of selected datasets used in the study.

Dataset	# Nodes	# Edges	# Features	Avg. # degree	# Classes
Cora [McCallum et al., 2000]	2,708	5,429	1,433	4.0	7
Citeseer [Giles et al., 1998]	3,312	4,732	3,703	2.9	6
Pubmed [Sen et al., 2008]	19,717	44,338	500	4.5	3
Coauthor.CS [Shchur et al., 2018]	18,333	163,788	6,805	17.9	15
Squirrel [Rozemberczki et al., 2021]	5,201	217,073	2,089	83.5	5
Chameleon [Pei et al.]	2,277	36,101	2,325	31.7	5
Cornell [Craven et al., 1998]	183	298	1,703	3.3	5
Texas [Craven et al., 1998]	183	325	1,703	3.6	5

C Motivation of Introducing Regularization

Our method involves forming positive and negative pairs from both the structured feature representations and the original input features for contrastive learning. This pairing strategy offers multiple advantages: it enriches the combination of features, reduces the risk of overfitting, and smooths and linearizes the feature distribution.

We incorporate a Gaussian structure into the embeddings by enforcing the embedded subgraph node features to approximate a Gaussian distribution. This is regulated using the Kullback-Leibler (KL) divergence from the distribution $q(\tilde{\mathbf{X}} \mid \mathbf{X}, \mathbf{A})$ of the embeddings to a standard Gaussian prior $p(\tilde{\mathbf{x}})$, given by:

$$-\log p(\tilde{\mathbf{x}} \mid \mathbf{x}, \mathbf{A}) = \frac{(\mathbf{x} - \tilde{\mathbf{x}})^2}{2\boldsymbol{\sigma}^2} + \text{const},$$

where "const" denotes a normalization constant, irrelevant to our optimization process. $p(\tilde{\mathbf{x}})$ is assumed to be a Gaussian distribution, and the formula represents the negative log-likelihood. This term is precisely what we need when calculating the KL divergence, as it evaluates the log-likelihood of the observation $\tilde{\mathbf{x}}$ relative to the assumed distribution $p(\tilde{\mathbf{x}})$. This formulation allows us to evaluate the expectation of the KL divergence as follows:

$$\mathbb{E}_{q(\tilde{\mathbf{X}} \mid \mathbf{X}, \mathbf{A})} \left[\frac{(\mathbf{x} - \tilde{\mathbf{x}})^2}{2\boldsymbol{\sigma}^2} \right].$$

To compute this expectation, we assume that the embedding distribution $q(\tilde{\mathbf{X}} \mid \mathbf{X}, \mathbf{A})$ centers around $\mathbf{X}$ with variance $\boldsymbol{\sigma}^2$, reflecting how $\tilde{\mathbf{X}}$ is encoded from $\mathbf{X}$. In our Optimal Transport Contrastive Loss section, the Wasserstein distance $W(\mathbf{X}^i, \tilde{\mathbf{X}}^i)$ plays a crucial role in measuring the cost of transporting one probability distribution to another, enhancing the model's sensitivity to the geometrical structure of the data:

$$W(\mathbf{X}^i, \tilde{\mathbf{X}}^i) = \min_{\mathbf{T} \in \pi(\mathbf{u}, \mathbf{v})} \sum_{p=1}^{k^i} \sum_{q=1}^{k^i} T_{pq} c(\mathbf{X}_p^i, \tilde{\mathbf{X}}_q^i) = \inf_{\gamma \in \Gamma(\mu_{\mathbf{X}^i}, \mu_{\tilde{\mathbf{X}}^i})} \int \|\mathbf{x} - \tilde{\mathbf{x}}\| d\gamma(\mathbf{x}, \tilde{\mathbf{x}}).$$

$\Gamma(\mu_{\mathbf{X}^i}, \mu_{\tilde{\mathbf{X}}^i})$ denotes the set of all joint distributions that relate the distributions $\mu_{\mathbf{X}^i}$ and $\mu_{\tilde{\mathbf{X}}^i}$. These distributions represent the probabilistic distributions of two different random variables, $\mathbf{X}^i$ and $\tilde{\mathbf{X}}^i$, respectively. This Wasserstein distance provides a robust measure of the discrepancy between the embedding distribution and the Gaussian prior, effectively guiding the embeddings to conform to the desired Gaussian structure.

D Experimental Evaluation

Datasets. In this study, we have selected several widely used datasets for graph node classification to conduct a comprehensive evaluation of our proposed method. These datasets encompass various types of networks including academic citation networks, collaboration networks, and web page networks, providing a diverse set of challenges and characteristics. Table 3 summarizes the basic statistics of these datasets:

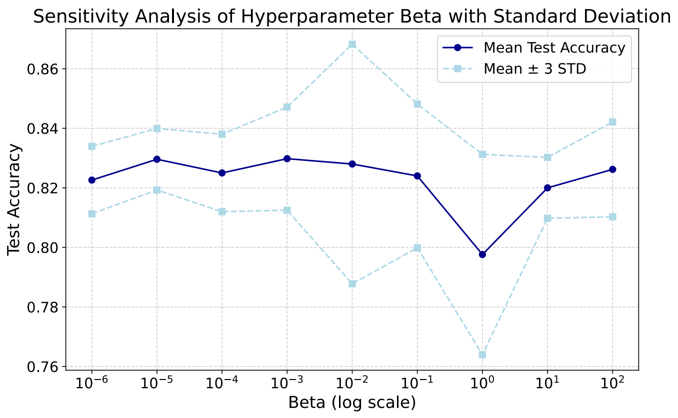

Figure 3: Sensitivity analysis of hyperparameter beta with standard deviation.

Implementation Details. The model's architecture leverages the strengths of well-established libraries such as `PyTorch`, `PyTorch Geometric`, and `Optuna`. It is optimized for execution on contemporary GPU architectures, including the RTX 3060 and A40, ensuring broad applicability. Our approach adopts a self-supervised scheme evaluated via linear probing. The model is trained using the official training subsets of the referenced datasets. Hyperparameter tuning involves a random search on the validation set to determine optimal values for learning rate, alpha, and beta. The identified best configuration is subsequently employed for tests on the dataset. Loss minimization is performed using the Adam optimizer, which produces a series of node embeddings. These embeddings are crucial for the downstream task of node classification, demonstrating the model's efficacy and adaptability to diverse hardware and data scenarios.

Hyperparameter Random Search. Informed by the findings of Gasteiger et al. [2019], Topping et al. [2022], Giraldo et al. [2023], which indicate the sensitivity of graph neural networks to hyperparameter settings, we undertake an extensive random search to optimize our model. The training proceeds on the official splits of the train datasets, with the random search conducted on the validation dataset to pinpoint the best configurations. These settings are then implemented to evaluate the model on the test dataset, thereby confirming the robustness of our model under optimal hyperparameter settings. The ranges of the hyperparameters explored and the code implementation of our approach will be made publicly available to facilitate replication and further research.

D.1 Sensitivity Analysis of the Regularization Constraint

To investigate the impact of the regularization constraint on our method, experiments were conducted on the Cora dataset. The influence of regularization within the loss function was controlled by varying the hyperparameter Beta, which ranged from 10^{-6} to 10^{2}. The results, as illustrated in Figure 3, indicate significant sensitivity to changes in Beta. Specifically, it was observed that values of Beta greater than or equal to 10^{-5} had a pronounced effect on model performance. Optimal results on the Cora dataset were achieved when Beta was set within the magnitude of 10^{-3}. This analysis highlights that Beta critically balances the strength of regularization and the intensity of contrastive learning, ensuring an optimal trade-off that enhances model performance without compromising generalization capabilities.

D.2 Sensitivity Analysis of the Subgraph Size

To assess the impact of subgraph size on the performance of our SGEC model, we conducted a sensitivity analysis using the Cora dataset with subgraph sizes $k = 5, 15, 25$, and 35. These results, summarized in Figure 4, indicate that a moderate subgraph size ($k = 15$) yields the highest accuracy and lowest variability, suggesting that it effectively captures essential structural information without introducing excessive noise. Larger subgraphs might include redundant or irrelevant information, leading to slight decreases in performance. Therefore, selecting an appropriate subgraph size is crucial for optimizing the representation learning capability of SGEC.

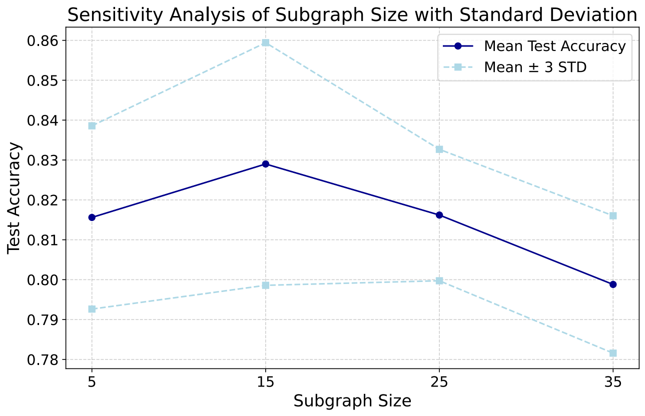

Figure 4: Sensitivity analysis of subgraph size k on the Cora dataset. The plot shows the mean test accuracy (dark blue line) with error bars representing the mean $\pm$ 3 standard deviations (light blue dashed lines) for different subgraph sizes.