GLNCD: Graph-Level Novel Category Discovery

Bowen Deng
Sun Yat-sen University
bowen.deng20@gmail.com

Lele Fu
Sun Yat-sen University

Sheng Huang
Sun Yat-sen University

Tianchi Liao
Sun Yat-sen University

Jialong Chen
Sun Yat-sen University

Tao Zhang
Sun Yat-sen University
zhangt358@mail.sysu.edu.cn

Chuan Chen *
Sun Yat-sen University
chenchuan@mail.sysu.edu.cn

Abstract

Graph classification has long assumed a closed-world setting, limiting its applicability to real-world scenarios where new categories often emerge. To address this limitation, we introduce Graph-Level Novel Category Discovery (GLNCD), a new task aimed at identifying unseen graph categories without supervision from novel classes. We first adapt classical Novel Category Discovery (NCD) methods for images to the graph domain and evaluate these baseline methods on four diverse graph datasets curated for the GLNCD task. Our analysis reveals that these methods suffer a notable performance degradation compared to their image-based counterparts, due to two key challenges: (1) insufficient utilization of structural information in graph self-supervised learning (SSL), and (2) ineffective pseudo-labeling strategies based on ranking statistics (RS) that neglect graph structure. To alleviate these issues, we propose ProtoFGW-NCD, a framework consisting of two core components: ProtoFGW-CL, a novel graph SSL framework, and FGW-RS, a structure-aware pseudo-labeling method. Both components employ a differentiable Fused Gromov-Wasserstein (FGW) distance to effectively compare graphs by incorporating structural information. These components are built upon learnable prototype graphs, which enable efficient, parallel FGW-based graph comparisons and capture representative patterns within graph datasets. Experiments on four GLNCD benchmark datasets demonstrate the effectiveness of ProtoFGW-NCD.

1 Introduction

Graph neural networks [70, 10, 1, 6] have become important in modern machine learning, finding applications in diverse domains such as protein function prediction [55], malware detection [2], and social network analysis [71]. However, most existing methods assume either a closed-world setting [70, 68, 74], where all test categories are known during training, or an unsupervised setting, where no categories are known [43, 42, 12, 64, 22]. In real-world scenarios, this assumption often breaks down: new graph categories may emerge dynamically (e.g., novel protein structures or previously unseen software behaviors), requiring models to adapt to an open-world setting. Despite the importance of this challenge, the problem of discovering novel graph categories remains unexplored.

We introduce graph-level novel category discovery (GLNCD), a new task that extends graph classification to open-world scenarios. Different from node-level NCD [32, 34, 11] aiming to discover new

*Corresponding author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

node categories, GLNCD is to discover novel graph classes (i.e., cluster unlabeled graphs) without explicit label supervision for these classes. In Section 3.1, we construct four diverse GLNCD datasets spanning various graph classification domains, including bioinformatics [55], program analysis [21], social networks [71], and graph-based image classification [15]. These datasets serve as benchmarks for evaluating GLNCD methods and highlight the unique challenges posed by graph data.

We adapt classical visual NCD methods (originally designed for image data) [78, 27, 80, 60] to the GLNCD setting. A typical visual NCD method is comprised of self-supervised learning (SSL), supervised learning on known classes, and category discovery. By replacing these components with the counterparts in graph domain (e.g, replacing SSL [23, 24, 5, 7, 9, 76] with Graph SSL [56, 75, 73, 66, 41, 72, 81, 82, 73, 12, 35]), we get the graph variants of visual NCD methods (detailed in Section 3.2). Surprisingly, our experiments (see Figure 1 and Table 2) reveal that these adapted methods perform significantly worse on graph data than on image data. Through systematic investigation presented in Section 4, we identify two key limitations contributing to this underperformance: 1) Existing graph-level SSL methods fail to fully exploit the structural information in graphs, resulting in suboptimal graph encoders. 2) The unlabeled sample pseudo-labeling strategy commonly used in visual NCD, ranking statistics (RS) [27], does not account for the structural information of graphs, leading to low-quality pseudo-labels for unlabeled graph samples.

To address the above limitations, we propose ProtoFGW-CL, a new graph SSL framework, and FGW-RS, a structure-aware pseudo-labeling method. They constitute our GLNCD method, ProtoFGW-NCD (Section 5), and both rely on the Fused Gromov-Wasserstein (FGW) distance [59, 39], which enables structural comparison between attributed graphs—going beyond readout vectors in baseline methods. However, computing pairwise FGW distances between two batches of graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ with varying sizes is inefficient using the BAPG solver [39] from existing tools [20]. To address this, we design a differentiable BAPG layer that supports efficient parallel computation during both forward and backward passes (Section B). Our BAPG layer assumes that the second group of graphs has uniform size. To handle variable-sized input graphs, we introduce K learnable prototype graphs $\mathcal{G}^{(B)}$ of identical size. We then compute the FGW distance matrix $\mathbf{V}_1 \in \mathbb{R}^{b_1 \times K}$ between $\mathcal{G}_1$ and $\mathcal{G}^{(B)}$, and $\mathbf{V}_2$ between $\mathcal{G}_2$ and $\mathcal{G}^{(B)}$. By comparing $\mathbf{V}_1$ and $\mathbf{V}_2$, we derive the pairwise relationships between $\mathcal{G}_1$ and $\mathcal{G}_2$, which are central to graph SSL and unlabeled graph pseudo-labeling. To make $\mathcal{G}^{(B)}$ more than just computational intermediaries, we treat them as learnable parameters following [5]. These prototypes are updated through BAPG layer during training, enabling them to capture representative structural patterns from the dataset, thereby benefiting graph SSL and pseudo-labeling.

Contributions: **1)** We introduce graph-level novel category discovery (GLNCD), a new task that extends graph-level classification to open-world scenarios, along with four diverse benchmark datasets spanning multiple domains. **2)** We adapt classical visual NCD methods to the graph domain and conduct a systematic analysis, revealing their limitations in sufficiently capturing structural information in graphs. **3)** To address these limitations, we propose ProtoFGW-NCD, a novel GLNCD method built upon a differentiable Fused Gromov-Wasserstein (FGW) distance. **4)** We design a parallel, differentiable BAPG layer for extremely efficient pairwise FGW distance computation.

2 Related Work

2.1 Visual Novel Category Discovery for Image Data

Novel Category Discovery (NCD) for image data has been well studied. AutoNovel [27] generates binary pairwise pseudo-labels for unknown samples using ranking statistics (RS), providing learning signals for the MLP-based clustering head. The quality of these pseudo-labels is closely tied to the representation quality output by the encoder. Incorporating contrastive loss [80] can enhance both representation and pseudo-label quality, thereby improving NCD performance. Exploiting multi-scale representation can improve pseudo-label reliability; for instance, DualRS [78] leverages both global and local branches to capture large-scale and fine-grained visual information, respectively. UNO [19] introduces a Sinkhorn-based approach to generate pseudo-labels without requiring pairwise comparisons. To better leverage knowledge from known categories, rKD [25] employs a fixed, supervised encoder with a known category head to constrain model outputs on known classes during the discovery training phase. Vaze et al. [61] and Cao et al. [4] propose a more challenging setting where the model must classify known-category samples and cluster unknown-category samples simultaneously during inference. While these methods perform well on regular image data, adapting

them to graph-structured data requires tailored modifications. We find that straightforward adaptations, such as replacing a CNN encoder with a GNN encoder, yield poor performance (see Figure 1 and Table 2), suggesting the need for more exploration.

2.2 Open World Graph Learning

Recent graph learning research has begun tackling open-world settings, where a node's class may be one that was unseen during training. In the node classification context, this gives rise to open-set recognition (OSR) problem: the model must classify nodes from known classes while identifying any nodes belonging to novel classes as "unknown." The seminal OpenWGL method [67] tackles this by learning uncertainty-aware node embeddings via a variational GNN, so that nodes from novel classes yield high predictive uncertainty and can be automatically rejected during inference. Hoffmann et al. [31] propose a meta-model that aggregates multiple confidence scores and employs a weakly-supervised thresholding strategy to decide when to label a node as unknown. Zhang et al. [77] propose to generate proxy nodes for unknown categories, tackling inductive node OSR problem. Beyond merely rejecting unknown nodes, NCD methods to cluster unlabeled nodes are proposed. ORAL [34] detects and remove the edges linking old and novel categories to alleviate the bias towards old categories, and uses multi–layer predictions to generate pseudo labels for unknown nodes. Hou et al. [32] consider a different NCD task that provides old-class nodes in the first training stage and then novel-class nodes in the second one. It is worth noting that open-world graph learning has been explored almost exclusively at the node level so far, and our work is the first to focus on graph-level.

3 Datasets and Baselines Adapted from Visual NCD

3.1 Prepare GLNCD datasets

Novel Category Discovery (NCD) can be viewed as a relaxed version of multi-class classification, where the goal is to group samples from the same novel category into the same cluster, without requiring them to be assigned to a specific label (e.g., $y = 0$). Therefore, one common approach to constructing an NCD dataset is to treat a subset of classes in a standard multi-class classification dataset as novel categories.

Although over 120 public graph-level datasets [50, 15] are available, we observe that only about five real-world multi-class graph classification datasets exist outside of computer vision scenarios. From these, we select three representative datasets spanning diverse domains, which, together with CIFAR10 (graph) [15], form the benchmark for GLNCD (Table 1). **Bioinformatics**: Graph structures naturally model protein molecules, where nodes represent secondary structure elements and edges indicate either sequential or spatial proximity between elements. The ENZYMES dataset [55, 50] contains six types of proteins with different catalytic functions. We treat the first three classes as old categories and the remaining three as novel categories. **Program Analysis**: The MalNet-Tiny dataset [21] consists of function call graphs (FCGs) derived from Android APK static disassembly. Each graph corresponds to a program type (Addisplay, Adware, Benign, Downloader, Trojan). We use the first three classes as old categories and the last two as novel categories. **Social Networks**: In the REDDIT12K dataset [71, 50], each graph represents a discussion thread, where nodes correspond to users and edges denote comment responses. The dataset contains 12 types of graphs, each corresponding to a subreddit. We designate the first six classes as old categories and the last five as novel categories. **Computer Vision**: An image can be decomposed into super-pixels, each forming a node whose features are computed as the average RGB values and spatial coordinates of the constituent pixels. Each node is connected to its eight nearest neighbors via edges weighted by Gaussian similarity. The CIFAR10 (graph) dataset [15] is constructed from CIFAR10 (image) [38] in this way. And we treat the first five classes as old categories and the rest as novel ones.

Luo et al. [46] incorporated various modern deep learning techniques, e.g., batch normalization [33] and residual connections [30], into GCN [37] and GIN [70] for graph-level tasks. More importantly, they employed Random Walk Structural Encoding (RWSE) [40, 16] as a preprocessing step to extract structural information, which is then used as part of the node and edge features. For all four datasets used in this work, we also apply RWSE-based preprocessing with a maximum path length of 32.

Table 1: Overview of GLNCD datasets. # steps is the maximal random walking length [46].

Dataset	# graphs	Avg. # nodes	Avg. # edges	# node/edge feats	# steps	# old/new classes
ENZYMES	600	32.6	124.3	21/0	32	3/3
MalNet-Tiny	5000	1410.3	2859.9	0/0	32	2/3
REDDIT12K	11929	391.41	456.89	0/0	32	5/6
CIFAR10	60000	117.6	941.1	5/1	32	5/5

3.2 Design GLNCD Baselines with Visual NCD Methods

A number of NCD methods have been developed in the field of computer vision. If these can be directly adapted to the graph-level setting, it would offer a convenient solution to GLNCD. To explore this, we adapt three representative visual NCD methods, i.e., AutoNovel [27], NCL [80], and DualRS [78], to the graph domain. These models typically consist of an encoder f_θ and two classification heads: h_n (for novel categories) and h_o (for old known categories). The encoder is trained using SSL, the old-class head is trained with standard supervised signals from labeled samples, and the new-class head relies on specifically designed pseudo-labeling strategies for training. By replacing each component with its counterpart in the graph domain, we achieve a straightforward adaptation.

AutoNovel: (*Stage1*) Originally employs a RotNet approach [23] for pre-training, where the encoder learns to recognize rotation angles (0°, 90°, 180°, or 270°) to capture semantical image features. (*Stage 2*) After pretraining, the GNN encoder and the old-class head is trained on old-class samples. (*Stage 3*) Finally, the new-class head is trained using pairwise ranking statistics (RS) pseudo-labels generated with new-class sample representations. ***Adaptation***: Since graph data consists of abstract topological structures without spatial orientation, we replace RotNet with GraphCL [75], a contrastive learning method tailored for graph-level tasks. Other stages remain unchanged.

NCL: Builds upon AutoNovel by introducing a MoCo-style [29] contrastive loss in (*Stage 3*), where negative samples are drawn from queues representing labeled and unlabeled data. Positive pairs include augmented views of the same sample and samples sharing the same old class label [36]. A hard-negative mining strategy is also incorporated [54]. ***Adaptation***: The readout outputs of the encoder are Euclidean, so the changes of NCL from AutoNovel can directly fit into graph domain.

DualRS: *(Stage 1)* Abandons the RotNet-pretrained ResNet18 and instead uses a ResNet50 backbone pretrained on ImageNet [13] with MoCoV2 [8]. *(Stage 2)* is removed. *(Stage 3)* Both classification heads are equipped with global and local branches. The global branch captures coarse-grained representations, while the local branch focuses on fine-grained details. Each branch generates pseudo-labels via RS to train its respective head, and then distilled to the other branch. ***Adaptation***: We use GraphCL to pretrain the GNN encoder. All other components remain unchanged.

We largely retain the original names of the visual NCD methods when referring to their adaptations for the graph domain. To avoid possible confusion in some cases, we prefix them, in these instances, with "G-" and refer to them as G-AutoNovel, G-NCL, and G-DualRS, respectively.

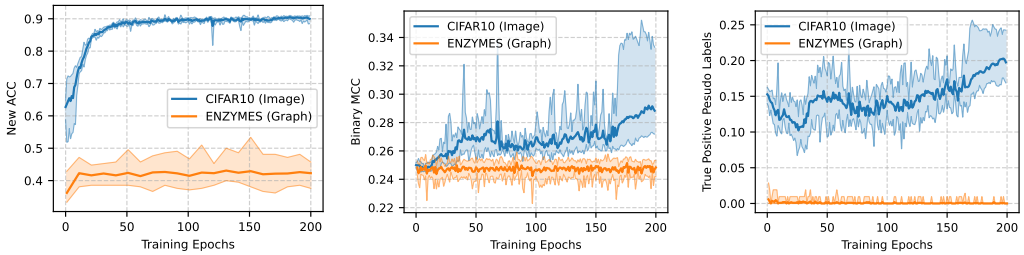
4 Challenges in NCD Method Adaptation: From Image to Graph Data

In this section, we reveal the failure of adapting the visual NCD methods to the graph domain like Section 3.2, and analyze the underlying reasons.

4.1 Why Direct Adaptation Fails? Ranking Statistics (RS) Fails

Table 2 shows the performance of AutoNovel on three image datasets, as well as its adapted version for graph-level tasks on four GLNCD datasets (Section 3.1), following the modifications described in Section 3.2. The New ACC (Train) on graph datasets, which is the primary metric of interest in NCD, is significantly lower than that on image datasets. More importantly, we observe a substantial performance gap between novel-class and old-class samples on graph datasets (Table 2), a phenomenon not present in image datasets. This suggests that the direct adaptation of AutoNovel fails to effectively leverage knowledge from old classes to aid in clustering unlabeled novel-class samples.

Given that pseudo-label quality plays a critical role in the performance of NCD methods, we hypothesize that the observed failure is due to the inability of RS to produce reliable pseudo-labels



(a) The clustering accuracy on unlabeled new-class samples.

(b) The MCC (quality) of RS pseudo labels for sample pairs.

(c) The ratio of true positive RS pseudo labels for sample pairs.

Figure 1: Training dynamics of AutoNovel [27] on CIFAR10 (image) and ENZYMES (graph). **(a)** The performance on unlabeled training dataset. **(b)** The Matthews Correlation Coefficient (MCC) to evaluate the quality of pairwise pseudo-labels generated via ranking statistics (RS). **(c)** The ratio of samples for which at least one true positive (same-class) pair is identified by RS. The definitions of these two pseudo-label metrics and the rationale for their selection are provided in Appendix A.

Table 2: The average performance (over 10 runs) of AutoNovel [27] on image and graph datasets. Old ACC (Test) is the accuracy on the old-class samples in test dataset. New ACC (Train) is the clustering accuracy on the unlabeled training dataset. The last row is the gap between these metrics.

Datasets	Image			Graph			
	CIFAR10	CIFAR100	SVHN	ENZYMES	MalNet-Tiny	REDDIT12K	CIFAR10
Old ACC (Test)	95.34	74.51	98.10	73.00	93.30	67.59	61.36
New ACC (Train)	88.50	74.28	94.21	41.90	74.51	39.21	41.67
Old (row1) - New (row2)	6.84	0.23	3.89	31.10	18.79	28.38	19.69

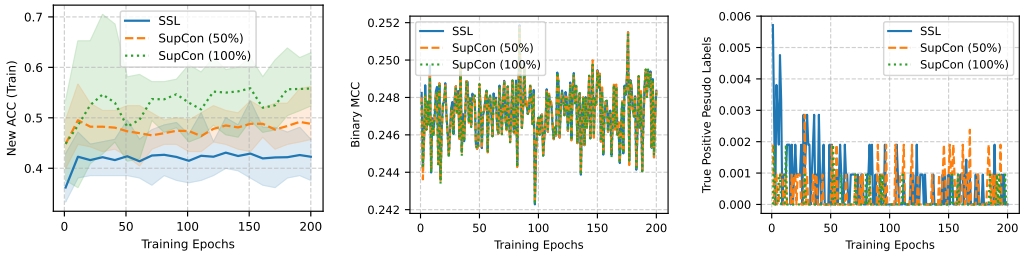
on graph-based datasets. To verify this, we track the New ACC (Train) along with RS pseudo-label quality on CIFAR-10 (image) and ENZYMES (graph), as shown in Figure 1. The results indicate that RS produces higher-quality pseudo-labels on CIFAR-10, with an overall increasing trend during training (Figure 1b and 1c). In contrast, on ENZYMES, the quality of pseudo-labels remain consistently low throughout training, suggesting that RS fails on graph-structured data in the direct adaptation described in Section 3.2.

4.2 Why RS Fails? Insufficient Exploration of Graph Structure

Ranking statistics (RS) generates pseudo-labels based on the representations output by the encoder (see Appendix A for the details of generation). If the encoder produces high-quality representations (i.e., samples from the same class are similar and those from different classes are dissimilar), the resulting RS pseudo-labels are typically of higher quality.

We therefore suspect that the representation learned by GraphCL in G-AutoNovel is not sufficiently discriminative. This hypothesis would be supported if higher-quality representations led to better pseudo-labels. However, according to recent reports [69, 26, 79], GraphCL performs as well or better than other graph SSL methods across a wide range of datasets, suggesting that replacing GraphCL with alternative SSL methods would not yield a significant improvement in representation quality for comparison purposes. To validate our hypothesis in another way, we pretrain three GNN encoders with increasing representation quality using the Supervised Contrastive (SupCon) loss [36], under settings where 0% (i.e., standard SSL), 50%, and 100% of the ground-truth binary pairwise labels are known. We then evaluate how the quality of RS pseudo-labels and NCD performance vary across these three levels of representation qualities.

Contrary to expectations, the quality of RS pseudo-labels (Figure 2b and 2c) shows little difference across the three representation qualities. Strikingly, despite nearly identical pseudo-label quality, the final NCD performance varies significantly across the three setups. **1)** This suggests that representation quality primarily influences the pseudo-label utilization rather than the pseudo-label quality itself: higher-quality representations provide a better data manifold, which facilitates learning better decision boundaries even when pseudo-labels are of comparable quality. **2)** This result does not imply that



(a) The clustering accuracy on unlabeled new-category samples.

(b) The MCC (quality) of RS pseudo labels for sample pairs.

(c) The raito of true positive RS pseudo labels for sample pairs.

Figure 2: Training dynamics of G-AutoNovel [27] on ENZYMES (graph). Three GIN encoders are pretrained with 0% (SSL), 50% (SupCon), and 100% (SupCon), true binary pairwise labels. **(a)** The performance on unlabeled training dataset. **(b)** The Matthews Correlation Coefficient (MCC) to evaluate the quality of RS pseudo-labels. **(c)** The ratio of samples for which at least one true positive pair is identified by RS. The details about of these pseudo-label metrics are provided in Appendix A.

representation quality has no impact on pseudo-label quality: as shown in Figure 1, on CIFAR-10 (image), RS pseudo-label quality improves substantially as the encoder learns better representations during training. Therefore, the nearly identical pseudo-label quality is likely due to the fact that RS is not well-suited to graph data and thus cannot effectively leverage improvements in representation quality to generate better pseudo-labels. We attribute the failure of RS to two principal factors: **1)** insufficient GNN pretraining that limits the representation quality and thus RS pseudo-label utilization; **2)** RS is unsuitable for graph data and fails to improve pseudo-label quality even when representation quality increases. Both issues stem from insufficient exploitation of graph structure:

- *Methods like GraphCL do not make sufficient use of structural information, resulting in suboptimal representations;*
- *RS operates solely on the readout graph representation vectors, thereby discarding part of valuable graph-structure information.*

5 Proposed Method: ProtoFGW-NCD

To test our hypothesis from Section 4.2—that the absence of structural information adversely affects both the SSL (i.e., GraphCL) and RS modules of AutoNovel—we developed ProtoFGW-NCD (Figure 3). This method was designed as a controlled experiment; it mirrors the architecture of AutoNovel exactly, with the sole exception of a Bregman Alternating Projected Gradient (BAPG) layer that injects structural information into the SSL and RS pipelines. Crucially, this integration requires no auxiliary loss functions or sophisticated mechanisms. This design provides a clean and direct means to validate our claim by allowing any performance difference to be attributed solely to the inclusion of structural information. The differentiable BAPG layer functions by solving the optimal transport problem between batches of attributed graphs [59, 65, 53], yielding rich structural information for graph comparison. With the inclusion of this layer, the original SSL and RS components of AutoNovel are adapted into our proposed ProtoFGW-CL and FGW-RS modules, respectively.

5.1 ProtoFGW-CL: Graph-level Representation Learning by Swapping Transport Couplings

GraphCL [75] utilizes graph structure only in the message-passing phase of the GNN encoder. The contrastive loss is computed solely on Euclidean vectors obtained via readout operations (e.g., min, sum), and does not explicitly compare structural differences between graphs G_i and G_j at this critical stage. Other graph-level SSL methods not only perform similarly to GraphCL but also fail to make full use of graph structure. For instance, some approaches generate views based on structural information (e.g., graph diffusion, communities, or motifs) [62, 57, 28, 58], while others employ multi-channel encoders to exploit more the original graph [69, 47]. However, like GraphCL, they all ignore structures when computing the distances between different graphs or graph views.

To address this, we propose ProtoFGW-CL, a graph-level contrastive learning method that explicitly computes the distances considering graph structures. Specifically, we compute the Fused Gromov-Wasserstein (FGW) distances between each graph and prototype graphs, normalizes it into codes, and enforces consistency across views. These randomly initialized prototype graphs $\left\{G_k^{(B)} = (\mathbf{A}_k^{(B)}, \mathbf{Z}_k^{(B)}) | \mathbf{A}_k^{(B)} \in \mathbb{R}_{\geq 0}^{N_k \times N_k}, \mathbf{Z}_k^{(B)} \in \mathbb{R}^{N_k \times d_k}\right\}_{k=1}^{K}$ establish the coordinates for cross-view structure-aware graph comparisons, and are learned with backpropagation to capture representative graph patterns in dataset. The FGW distance (Definition 2) [59, 39] between two graphs is defined as the optimal transport between their probability measures given by Definition 1.

Definition 1. (Graph as Probability Measure) An attributed graph $G_1 = (\mathbf{C}_1, \mathbf{X}_1)$ with $N_1 = |V_1|$ nodes defines a metric-measure (MM) space $(V_1, \mathbf{C}_1, \mu_1)$. Each node v_i has an explicit node feature vector $\mathbf{x}_{1,i} \in \Omega_x \subset \mathbb{R}^d$ and an implicit structural feature $\mathbf{s}_{1,i} \in \Omega_s$. The pairwise relationship among structural features is encoded in $\mathbf{C}_1$, where $C_{1,ij} = D_{\Omega_s}(\mathbf{s}_{1,i}, \mathbf{s}_{1,j})$ typically represented as an adjacency or Laplacian matrix. The probability measure associated with G_1 is defined as $\mu_1 = \sum_{i=1}^{N_1} h_i^{(1)} \delta_{(\mathbf{s}_{1,i}, \mathbf{x}_{1,i})}$, where $h^{(1)} \in \mathcal{H}_N = \left\{h \mid h \in \mathbb{R}_{>0}^N, \sum_{i=1}^N h_i = 1\right\}$ is a simplex histogram, and $\delta_{(\mathbf{s}_{1,i}, \mathbf{x}_{1,i})}$ denotes the Dirac delta function located at $(\mathbf{s}_{1,i}, \mathbf{x}_{1,i})$.

Definition 2. (FGW distance) Given two attributed graphs $G_1 = (\mathbf{C}_1, \mathbf{X}_1)$ and $G_2 = (\mathbf{C}_2, \mathbf{X}_2)$, their corresponding probability measures are $\mu_1 = \sum_{i=1}^{N_1} h_i^{(1)} \delta_{(\mathbf{s}_{1,i}, \mathbf{x}_{1,i})}$ and $\mu_2 = \sum_{j=1}^{N_2} h_j^{(2)} \delta_{(\mathbf{s}_{2,j}, \mathbf{x}_{2,j})}$, respectively. The FGW distance between them is defined as

$$\inf_{\pi \in \Pi} \left\{ \sum_{i,k=1}^{N_1} \sum_{j,l=1}^{N_2} \left[(1-\alpha) D_{\Omega_x}(\mathbf{x}_{1,i}, \mathbf{x}_{2,j}) + \alpha |\mathbf{C}_1(i,k) - \mathbf{C}_2(j,l)|^2 \right] \mathbf{T}_{i,j} \mathbf{T}_{k,l} \right\}, \tag{1}$$

where $\Pi = \{\mathbf{T} \in \mathbb{R}_{\geq 0}^{N_1 \times N_2} \mid \mathbf{T}\mathbf{1}_{N_2} = h^{(1)}, \mathbf{T}^\top \mathbf{1}_{N_1} = h^{(2)}\}$ is the feasible set of transport plans, $h^{(1)}$ and $h^{(2)}$ denote marginal distributions and are typically set to uniform distributions in practice, D_{Ω_x} is a metric in Ω_x, and $\alpha \in [0, 1]$ balances the contributions of node features and graph structures.

5.2 FGW-RS: Ranking Statistics with More Graph Structure Information

G-AutoNovel applies ranking statistics (RS) to the Euclidean readout summary vectors, while graph structure is only implicitly used during message passing. Given the importance of structural information in graph-level tasks, this utilization is insufficient. To enable RS to more directly incorporate graph structure information, we construct two attributed graphs from the representations before readout $G_1^z = (\mathbf{A}_1, \mathbf{Z}_1)$ and $G_2^z = (\mathbf{A}_2, \mathbf{Z}_2)$, corresponding to the two graph samples. We then compute their FGW distances to a set of prototype graphs, resulting in distance vectors $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{R}_{\geq 0}^K$, which are subsequently fed into classic RS used in [27].

The prototype graphs, learned during training, capture diverse representative structural patterns in the dataset. The FGW distances $\mathbf{v}_1, \mathbf{v}_2$ explicitly encode both the input samples' structures and their relations to global patterns in the dataset. This allows RS to generate pseudo-labels that better reflect structural information, thereby improving the quality of pseudo-labels.

5.3 ProtoFGW-NCD: Integrating ProtoFGW-CL and FGW-RS

Unlike AutoNovel [27], which separates pretraining, supervised learning, and category discovery from each other, ProtoFGW-NCD performs representation learning, supervised learning and category discovery within a single training process. Figure 3 illustrates the architecture and pipelines. At the beginning of training, we randomly initialize a set of prototype graphs $\left\{G_k^{(B)} = (\mathbf{A}_k^{(B)}, \mathbf{Z}_k^{(B)})\right\}_{k=1}^{K}$. Labeled old-class samples and unlabeled new-class samples from the training set are mixed and randomly sampled into mini-batches. For each graph sample $G_i = (\mathbf{A}_i, \mathbf{X}_i, \mathbf{E}_i)$, $\mathbf{A}_i \in \mathbb{R}_{\geq 0}^{N_i \times N_i}$ is the adjacency matrix, $\mathbf{X}_i \in \mathbb{R}^{N_i \times d_n}$ is the node feature matrix, and $\mathbf{E}_i \in \mathbb{R}^{E_i \times d_e}$ is the edge feature matrix. **Data Augmentation**: We randomly remove $p\%$ of the nodes (and their edges) to get the subgraph $\tilde{G}_i = (\tilde{\mathbf{A}}_i, \tilde{\mathbf{X}}_i, \tilde{\mathbf{E}}_i)$ induced by the rest nodes. **Encoding**: Both G_i and $\tilde{G}_i$ are passed through a learnable feature encoder to refine node and edge features, followed by a GNN$^+$ encoder f_θ [46]. The outputs at the final GNN layer are denoted as $(\mathbf{A}_i, \mathbf{Z}_i)$ and $(\tilde{\mathbf{A}}_i, \tilde{\mathbf{Z}}_i)$, where $\mathbf{Z}_i$ and $\tilde{\mathbf{Z}}_i$ are the learned node representations. **FGW Codes**: We compute the FGW distances between these

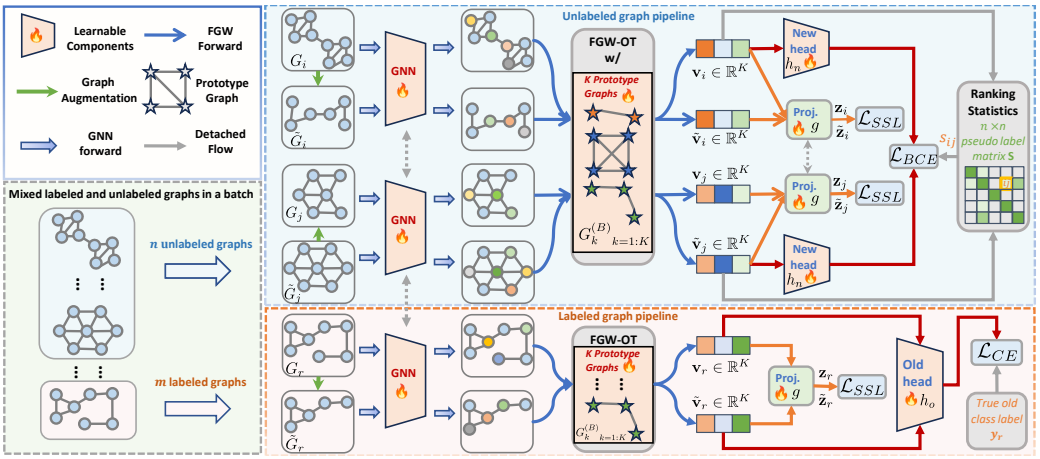


Figure 3: Illustration of **ProtoFGW-NCD**. Unlike previous graph SSL methods that directly compare Euclidean representations, **ProtoFGW-CL** maps graphs to codes by aligning them with learnable prototype graphs using the Fused Gromov-Wasserstein (FGW) distance. Under the supervision of $\mathcal{L}_{SSL}$, these codes are encouraged to be consistent across views, such that the code from one view can be predicted using the other. **FGW-RS**: The codes of n unlabeled graphs are fed into RS to generate pseudo-label matrix $\mathbf{S}$, and also into the new-class head to obtain predictions. The pairwise similarities between these predictions are guided by $\mathbf{S}$ with $\mathcal{L}_{BCE}$. **Supervised Learning**: The codes of m labeled graphs and their augmentations are fed into the old-class head to compute $\mathcal{L}_{CE}$.

outputs and the K prototype graphs, resulting in distance vectors $\mathbf{v}_i, \tilde{\mathbf{v}}_i \in \mathbb{R}^K$. Let m and n denote the number of labeled and unlabeled samples in a batch, respectively. The overall training loss is composed of three terms

$$\mathcal{L} = \mathcal{L}_{CE} + \mathcal{L}_{BCE} + \mathcal{L}_{SSL} \tag{2}$$

$$\mathcal{L}_{CE} = -\frac{1}{2m}\sum_{r=1}^{m} \log\left[h_o(\mathbf{v}_r)\right]_{y_r} + \log\left[h_o(\tilde{\mathbf{v}}_r)\right]_{y_r} \tag{3}$$

$$\mathcal{L}_{BCE} = -\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\left[s_{ij}\log h_n\left(\mathbf{v}_i\right)^\top h_n\left(\mathbf{v}_j\right) + \left(1-s_{ij}\right)\log\left(1 - h_n\left(\mathbf{v}_i\right)^\top h_n\left(\mathbf{v}_j\right)\right)\right] \tag{4}$$

$$\mathcal{L}_{SSL} = \frac{1}{2(m+n)}\sum_{i=1}^{m+n}\sum_{k=1}^{K}\left[\mathrm{SM}(\tilde{\mathbf{z}}_i)\right]_k \log\left(\left[\mathrm{SM}(\mathbf{z}_i/\tau)\right]_k\right) + \left[\mathrm{SM}(\mathbf{z}_i)\right]_k \log\left(\left[\mathrm{SM}(\tilde{\mathbf{z}}_i/\tau)\right]_k\right) \tag{5}$$

where $\mathrm{SM}(\cdot)$ is the softmax operator, $g(\cdot)$ is the projector for SSL, $\tilde{\mathbf{z}}_i = g(\tilde{\mathbf{v}}_i)$, $\mathbf{z}_i = g(\mathbf{v}_i)$, $\mathcal{L}_{CE}$ provides supervised signals from old-class samples, $\mathcal{L}_{BCE}$ guides the new-class head h_n, and $\mathcal{L}_{SSL}$ encourages the cross-view consistency of FGW codes. Here, $s_{ij} \in \{0, 1\}$ is the binary pseudo-label generated by ranking statistics (RS) based on structure-aware codes $\mathbf{v}_i$ and $\mathbf{v}_j$, and $s_{ij} = 1$ if RS considers i and j to belong to the same class. FGW is typically solved using iterative algorithms. If backpropagation is naively handled by PyTorch, it would record many intermediate steps, leading to an excessively large and computationally intractable computational graph. To address this, we design a parallel, differentiable FGW module, BAPG layer. This layer supports parallel differentiable FGW distances between b_1 `torch_sparse.SparseTensor` [18] and b_2 dense `torch.Tensor` on GPU. In contrast, previous differentiable FGW module [63] only supports CPU backend and relies on slow nested loops for pairwise FGW solving.

6 Experiments

6.1 Experimental Setup

We adopt the latest improved versions of two classic GNN architectures, i.e., GCN$^+$ and GIN$^+$ [46], as the backbones for all GLNCD methods. These models achieve or approach state-of-the-art

Table 3: GLNCD results with GIN$^+$ encoder. The 1st and 2nd results are highlighted.

	ENZYMES		MalNet-Tiny		REDDIT12K		CIFAR10		*Avg. Rank* ↓		
	Old ACC	New ACC	Old ACC	New ACC	Old ACC	New ACC	Old ACC	New ACC	Old ACC	New ACC	All
K-means	50.33±5.19	39.90±4.01	54.75±0.00	40.24±0.11	35.22±1.58	30.56±2.04	42.16±3.52	40.76±4.18	5.00	4.00	4.500
AutoNovel	73.00±1.39	41.90±1.62	90.75±2.66	68.93±6.29	67.59±0.77	39.21±0.92	61.36±3.95	41.67±1.90	3.00	2.75	2.875
NCL	70.00±2.04	45.81±5.82	92.55±1.14	69.76±7.44	70.11±0.75	36.57±1.59	67.54±1.24	38.21±2.01	2.25	3.00	2.625
DualRS	76.33±0.75	39.52±1.12	92.85±1.05	69.09±3.37	66.05±0.58	39.68±1.77	65.78±1.39	40.23±0.66	2.25	3.25	2.750
Ours	74.00±5.68	37.95±3.16	93.30±0.76	74.51±7.05	67.39±0.48	39.94±2.42	56.44±1.43	44.04±1.58	2.50	2.00	**2.250**

Table 4: GLNCD results with GCN$^+$ encoder. The 1st and 2nd results are highlighted.

	ENZYMES		MalNet-Tiny		REDDIT12K		CIFAR10		*Avg. Rank* ↓		
	Old ACC	New ACC	Old ACC	New ACC	Old ACC	New ACC	Old ACC	New ACC	Old ACC	New ACC	All
K-means	39.67±1.83	38.86±2.89	66.60±9.26	58.72±2.76	42.34±4.28	37.05±2.11	42.27±0.12	40.72±1.42	5.00	3.75	4.375
AutoNovel	71.33±2.74	41.52±1.86	80.30±6.79	62.43±5.40	68.91±0.33	39.08±1.34	61.10±3.12	41.26±1.31	3.00	2.00	2.500
NCL	67.67±1.49	39.71±3.77	85.50±2.35	62.23±1.68	69.35±1.11	37.01±1.20	70.63±0.46	39.64±0.82	2.25	3.50	2.875
DualRS	64.67±3.21	39.33±5.07	68.75±7.45	49.53±3.74	66.47±0.87	40.76±2.77	70.90±0.86	39.17±0.86	3.50	3.75	3.625
Ours	72.17±5.67	44.84±3.07	80.95±6.16	63.35±1.19	69.43±3.74	40.81±2.16	71.25±0.61	38.92±0.49	1.25	2.00	**1.625**

performance on graph-level supervised tasks [46]. In the Fused Gromov-Wasserstein (FGW) distance Eq. (1), we use the Euclidean distance as node feature comparison metric D_{Ω_x}, and the adjacency matrix $\mathbf{A}$ to represent pairwise structural relationship $\mathbf{C}$. The attributed graphs used in computing FGW distances are constructed by the node hidden features $\mathbf{Z}$ of the final GNN layer and the input graph adjacency matrix $\mathbf{A}$. Inspired by [63], we set the trade-off parameter α as learnable. For data graphs, the marginal probability distributions are set to uniform, while for prototype graphs the marginals are made learnable. ProtoFGW-CL aims to incorporate structural information into the contrastive loss. So for graph data augmentation, we follow GraphCL and simply apply random node dropping: a fraction $p\%$ of nodes are randomly removed, along with all edges connected to them.

6.2 Main Results

The GLNCD datasets constructed in Section 3.1 each include a training set, validation set, and test set, as detailed in Table 7. We train the model on all labeled old-class samples and unlabeled new-class samples from the training set. During evaluation, we report two performance metrics: the clustering accuracy on unlabeled (new-class) training samples, denoted as **New ACC** and the classification accuracy on old-class test samples, denoted as **Old ACC**. We report the GLNCD results using a GIN$^+$ encoder in Tables 3 while the results for the GCN$^+$ encoder are provided in Table 4. As shown in these tables, ProtoFGW-NCD, which incorporates more graph structural information, demonstrates decent NCD performance and achieves the highest average ranking on all datasets. This confirms the effectiveness of exploiting more graph structure in GLNCD tasks.

6.3 Ablation Study

Table 5: Ablation with GCN$^+$ on ENZYMES

Method	Old ACC	New ACC
wo FGW-RS	72.00±5.14	33.33±0.00
wo ProtoFGW-CL	70.50±3.69	42.10±4.52
wo Supervised learning	34.00±2.74	41.05±3.36
fixed proto. graphs	62.33±4.32	38.19±2.12
ProtoFGW-NCD	72.17±5.67	44.84±3.07

We conduct ablation studies on the components of ProtoFGW-NCD by reporting both Old ACC and New ACC to evaluate their effectiveness. ProtoFGW-NCD consists of four key components: ProtoFGW-CL, FGW-RS, supervised learning, and learnable prototype graphs. Removing the first three components can be achieved by excluding $\mathcal{L}_{SSL}$, $\mathcal{L}_{BCE}$, and $\mathcal{L}_{CE}$ from the training loss. To assess the importance of adaptively learned prototype graphs, we disable gradient updates for them.

The results in Table 5 demonstrate that each component contributes significantly, and removing any leads to noticeable degradation. Among the components, FGW-RS has a minor impact on Old ACC but plays a decisive role in New ACC, as its removal leads to a drop to 33.33%, equivalent to random guessing among the three unlabeled categories. In contrast, ProtoFGW-CL significantly affects Old ACC while having minimal influence on New ACC. This may be because training the encoder and new head with pseudo-labels from FGW-RS itself serves as a form of self-supervised learning (SSL) beneficial for novel category discovery. Without supervised signals, although Old ACC degrades to the level of random guessing, New ACC still achieves notable performance, indicating that ProtoFGW-CL effectively guides the model to learn meaningful SSL representations for NCD.

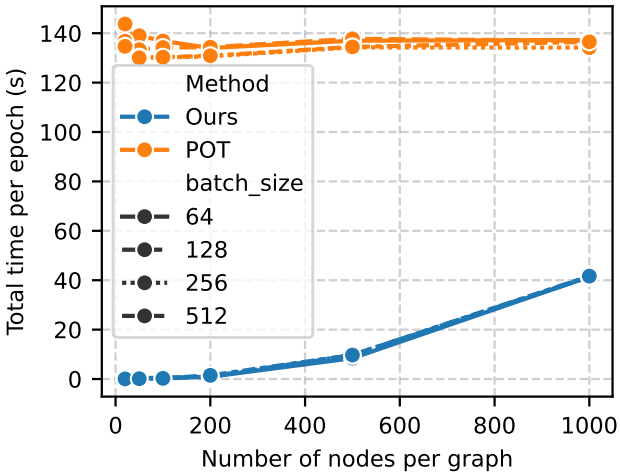


Figure 4: The epoch time (s) of different BAPG solver implementations for FGW problems. The x-axis indicates different synthetic datasets, i.e., CSBM-20-10, CSBM-50-10, ..., CSBM-1000-10.

Table 6: The average batch time (s) of different BAPG implementations. ↑ indicates speedup factor. The highest speedup for each dataset (row) across all batch sizes is shown in boldface.

Batch Size B		64			128			256			512	
Dataset	POT	Ours	↑	POT	Ours	↑	POT	Ours	↑	POT	Ours	↑
CSBM-20-10	8.90	0.04	250.7	17.96	0.02	719.6	34.19	0.03	1296.8	67.34	0.03	**2070.2**
CSBM-50-10	8.69	0.03	333.5	16.67	0.03	598.8	32.57	0.04	844.8	65.02	0.06	**1020.1**
CSBM-100-10	8.55	0.03	289.3	16.78	0.04	416.9	32.54	0.07	457.6	65.14	0.14	**474.6**
CSBM-200-10	8.38	0.06	142.6	16.79	0.11	**150.2**	32.66	0.28	117.9	65.43	0.74	88.0
CSBM-500-10	8.56	0.53	**16.2**	17.22	1.20	14.3	33.56	2.43	13.8	67.23	4.85	13.8
CSBM-1000-10	8.58	2.59	**3.3**	17.12	5.17	3.3	33.54	10.35	3.2	68.26	20.82	3.3

Learnable prototypes are critical for both Old ACC and New ACC, resulting in performance changes of approximately 10 and 6 points, respectively. This highlights the necessity of learnable prototypes and our efficient BAPG layer.

6.4 The Efficiency of Our BAPG Layer

In contrast to the BAPG operator in POT [20] which requires a brute-force loop, our proposed BAPG layer (Appendix B) can compute the Fused Gromov-Wasserstein (FGW) distance between B_1 and B_2 graphs in parallel. We compare their efficiency on Contextual Stochastic Block Model (CSBM) graphs [14, 48] with varying numbers of nodes. For a graph dataset with N nodes, we assume there are 10 prototype graphs, each with $\frac{N \log N}{2}$ nodes (see Appendix C.2 for more details) and denote by CSBM-N-10 the synthesized dataset. The computation times for FGW distance under different batch sizes are presented in Table 6 and Figure 4, where our method achieves a speedup of up to 2070.2x compared to the POT implementation.

7 Conclusion

This paper introduces Graph-Level Novel Category Discovery (GLNCD) task, aiming to identify unseen graph categories in an open-world setting. We present four diverse benchmark datasets across different scenarios and systematically adapt classical NCD methods for images to the graph domain. However, experimental results show that these direct adaptations perform poorly on graph data due to insufficient graph structure explorations. To address this issue, we propose ProtoFGW-NCD that consists of structure-aware SSL and pseudo-labeling, via a differentiable Fused Gromov-Wasserstein (FGW) module, BAPG layer. Experiments demonstrate that ProtoFGW-NCD matches or outperforms baseline methods. As a direction for future research, it would be promising to develop methods that explore graph structures more efficiently than FGW module for GLNCD tasks on large-scale datasets.

Acknowledgments

The research is supported by the National Key R&D Program of China (2023YFB2703700), the National Natural Science Foundation of China (62176269).

References

[1] Ali Behrouz and Farnoosh Hashemi. Graph Mamba: Towards Learning on Graphs with State Space Models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, pages 119–130, New York, NY, USA, August 2024. Association for Computing Machinery. 1

[2] Tristan Bilot, Nour El Madhoun, Khaldoun Al Agha, and Anis Zouaoui. A Survey on Malware Detection with Graph Representation Learning. *ACM Comput. Surv.*, 56(11):278:1–278:36, June 2024. 1

[3] Vincent D. Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, October 2008. C.2

[4] Kaidi Cao, Maria Brbic, and Jure Leskovec. Open-World Semi-Supervised Learning. In *International Conference on Learning Representations*, October 2021. 2.1

[5] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, pages 9912–9924, Red Hook, NY, USA, December 2020. Curran Associates Inc. 1

[6] Jialong Chen, Bowen Deng, Zhen WANG, Chuan Chen, and Zibin Zheng. Graph neural ricci flow: Evolving feature from a curvature perspective. In *The Thirteenth International Conference on Learning Representations*, 2025. 1

[7] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A Simple Framework for Contrastive Learning of Visual Representations. In *Proceedings of the 37th International Conference on Machine Learning*, pages 1597–1607. PMLR, November 2020. 1

[8] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved Baselines with Momentum Contrastive Learning, March 2020. 3.2

[9] Xinlei Chen and Kaiming He. Exploring Simple Siamese Representation Learning. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15745–15753, Nashville, TN, USA, June 2021. IEEE. 1

[10] Gabriele Corso, Hannes Stark, Stefanie Jegelka, Tommi Jaakkola, and Regina Barzilay. Graph neural networks. *Nature Reviews Methods Primers*, 4(1):17, March 2024. 1

[11] Bowen Deng, Lele Fu, Jialong Chen, Sheng Huang, Tianchi Liao, Zhang Tao, and Chuan Chen. Towards understanding parametric generalized category discovery on graphs. In *Forty-second International Conference on Machine Learning*, 2025. 1

[12] Bowen Deng, Tong Wang, Lele Fu, Sheng Huang, Chuan Chen, and Tao Zhang. Thesaurus: contrastive graph clustering by swapping fused gromov-wasserstein couplings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 16199–16207, 2025. 1

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, June 2009. 3.2

[14] Yash Deshpande, Subhabrata Sen, Andrea Montanari, and Elchanan Mossel. Contextual Stochastic Block Models. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. 6.4, C.2

[15] Vijay Prakash Dwivedi, Chaitanya K Joshi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Benchmarking Graph Neural Networks. *Journal of Machine Learning Research*, 24(43):1–48, 2023. 1, 3.1

[16] Vijay Prakash Dwivedi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Graph Neural Networks with Learnable Structural and Positional Representations. In *International Conference on Learning Representations*, October 2021. 3.1

[17] M. Fey and J.E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *International Conference on Learning Representations*, 2019. C.1

[18] Matthias Fey. Rusty1s/pytorch_sparse: PyTorch Extension Library of Optimized Autograd Sparse Matrix Operations. https://github.com/rusty1s/pytorch_sparse. 5.3

[19] Enrico Fini, Enver Sangineto, Stéphane Lathuilière, Zhun Zhong, Moin Nabi, and Elisa Ricci. A Unified Objective for Novel Class Discovery. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9264–9272, October 2021. 2.1

[20] Rémi Flamary, Nicolas Courty, Alexandre Gramfort, Mokhtar Z. Alaya, Aurélie Boisbunon, Stanislas Chambon, Laetitia Chapel, Adrien Corenflos, Kilian Fatras, Nemo Fournier, Léo Gautheron, Nathalie T.H. Gayraud, Hicham Janati, Alain Rakotomamonjy, Ievgen Redko, Antoine Rolet, Antony Schutz, Vivien Seguy, Danica J. Sutherland, Romain Tavenard, Alexander Tong, and Titouan Vayer. POT: Python optimal transport. *Journal of Machine Learning Research*, 22(78):1–8, 2021. 1, 6.4, C.1

[21] Scott Freitas, Yuxiao Dong, Joshua Neil, and Duen Horng Chau. A Large-Scale Database for Graph Representation Learning. In *Thirty-Fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, June 2021. 1, 3.1

[22] Lele Fu, Bowen Deng, Sheng Huang, Tianchi Liao, Chuanfu Zhang, and Chuan Chen. Learn from global rather than local: Consistent context-aware representation learning for multi-view graph clustering. In James Kwok, editor, *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, pages 5145–5153. International Joint Conferences on Artificial Intelligence Organization, 8 2025. Main Track. 1

[23] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised Representation Learning by Predicting Image Rotations. In *International Conference on Learning Representations*, February 2018. 1, 3.2

[24] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent a new approach to self-supervised learning. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, pages 21271–21284, Red Hook, NY, USA, December 2020. Curran Associates Inc. 1

[25] Peiyan Gu, Chuyu Zhang, Ruijie Xu, and Xuming He. Class-relation Knowledge Distillation for Novel Class Discovery. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 16428–16437. IEEE Computer Society, October 2023. 2.1

[26] Xiaojun Guo, Yifei Wang, Zeming Wei, and Yisen Wang. Architecture matters: Uncovering implicit mechanisms in graph contrastive learning. In A. Oh, T. Neumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 28585–28610. Curran Associates, Inc., 2023. 4.2

[27] Kai Han, Sylvestre-Alvise Rebuffi, Sébastien Ehrhardt, Andrea Vedaldi, and Andrew Zisserman. AutoNovel: Automatically Discovering and Learning Novel Visual Categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(10):6767–6781, October 2022. 1, 2.1, 3.2, 1, 2, 2, 5.2, 5.3, A, A, C.1

[28] Kaveh Hassani and Amir Hosein Khasahmadi. Contrastive Multi-View Representation Learning on Graphs. In *Proceedings of the 37th International Conference on Machine Learning*, pages 4116–4126. PMLR, November 2020. 5.1

[29] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum Contrast for Unsupervised Visual Representation Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020. 3.2

[30] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 3.1

[31] Marcel Hoffmann, Lukas Galke, and Ansgar Scherp. Open-World Lifelong Graph Learning. In *2023 International Joint Conference on Neural Networks (IJCNN)*, pages 1–9, June 2023. 2.2

[32] Yue Hou, Xueyuan Chen, He Zhu, Ruomei Liu, Bowen Shi, Jiaheng Liu, Junran Wu, and Ke Xu. NC2D: Novel Class Discovery for Node Classification. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, CIKM '24, pages 849–859, New York, NY, USA, October 2024. Association for Computing Machinery. 1, 2.2

[33] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 448–456. PMLR, June 2015. 3.1

[34] Yucheng Jin, Yun Xiong, Juncheng Fang, Xixi Wu, Dongxiao He, Xing Jia, Bingchen Zhao, and Philip S. Yu. Beyond the Known: Novel Class Discovery for Open-World Graph Learning. In Makoto Onizuka, Jae-Gil Lee, Yongxin Tong, Chuan Xiao, Yoshiharu Ishikawa, Sihem Amer-Yahia, H. V. Jagadish, and Kejing Lu, editors, *Database Systems for Advanced Applications*, pages 117–133, Singapore, 2024. Springer Nature. 1, 2.2

[35] Wei Ju, Yifan Wang, Yifang Qin, Zhengyang Mao, Zhiping Xiao, Junyu Luo, Junwei Yang, Yiyang Gu, Dongjie Wang, Qingqing Long, Siyu Yi, Xiao Luo, and Ming Zhang. Towards Graph Contrastive Learning: A Survey and Beyond, May 2024. 1

[36] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 18661–18673, 2020. 3.2, 4.2

[37] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. 3.1

[38] A. Krizhevsky. Learning Multiple Layers of Features from Tiny Images. 2009. 3.1

[39] Jiajin Li, Jianheng Tang, Lemin Kong, Huikang Liu, Jia Li, Anthony Man-Cho So, and Jose Blanchet. A Convergent Single-Loop Algorithm for Relaxation of Gromov-Wasserstein in Graph Data. In *The Eleventh International Conference on Learning Representations*, September 2022. 1, 5.1, B.1, 1, C.1

[40] Pan Li, Yanbang Wang, Hongwei Wang, and Jure Leskovec. Distance encoding: Design provably more powerful neural networks for graph representation learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 4465–4478. Curran Associates, Inc., 2020. 3.1

[41] Yixin Liu, Ming Jin, Shirui Pan, Chuan Zhou, Yu Zheng, Feng Xia, and Philip Yu. Graph Self-Supervised Learning: A Survey. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–1, 2022. 1

[42] Yue Liu, Ke Liang, Jun Xia, Sihang Zhou, Xihong Yang, Xinwang Liu, and Stan Z. Li. Dink-Net: Neural clustering on large graphs. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *ICML'23*, pages 21794–21812. JMLR.org, July 2023. 1

[43] Yue Liu, Jun Xia, Sihang Zhou, Xihong Yang, Ke Liang, Chenchen Fan, Yan Zhuang, Stan Z. Li, Xinwang Liu, and Kunlun He. A Survey of Deep Graph Clustering: Taxonomy, Challenge, Application, and Open Resource, September 2023. 1

[44] Ilya Loshchilov and Frank Hutter. SGDR: Stochastic Gradient Descent with Warm Restarts. In *International Conference on Learning Representations*, November 2016. C.1

[45] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*, September 2018. C.1

[46] Yuankai Luo, Lei Shi, and Xiao-Ming Wu. Unlocking the Potential of Classic GNNs for Graph-level Tasks: Simple Architectures Meet Excellence, February 2025. 3.1, 1, 5.3, 6.1

[47] Zhenfei Luo, Yixiang Dong, Qinghua Zheng, Huan Liu, and Minnan Luo. Dual-channel graph contrastive learning for self-supervised graph-level representation learning. *Pattern Recognition*, 139:109448, 2023. 5.1

[48] Yao Ma, Xiaorui Liu, Neil Shah, and Jiliang Tang. Is Homophily a Necessity for Graph Neural Networks? In *International Conference on Learning Representations*, March 2022. 6.4, C.2

[49] B. W. Matthews. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta (BBA) - Protein Structure*, 405(2):442–451, October 1975. A

[50] C. Morris, N.M. Kriege, F. Bause, K. Kersting, P. Mutzel, and M. Neumann. Tudataset: A collection of benchmark datasets for learning with graphs. In *Proceedings of the International Conference on Machine Learning Workshop on Graph Representation Learning and Beyond*, 2020. 3.1

[51] M. E. J. Newman. Finding and evaluating community structure in networks. *Physical Review E*, 69(2):026113, 2004. C.2

[52] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. October 2017. C.1

[53] Gabriel Peyré, Marco Cuturi, and Justin Solomon. Gromov-Wasserstein Averaging of Kernel and Distance Matrices. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 2664–2672. PMLR, June 2016. 5, B.1

[54] Joshua David Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. Contrastive learning with hard negative samples. In *International Conference on Learning Representations*, 2021. 3.2

[55] Ida Schomburg, Antje Chang, Christian Ebeling, Marion Gremse, Christian Heldt, Gregor Huhn, and Dietmar Schomburg. BRENDA, the enzyme database: Updates and major new developments. *Nucleic Acids Research*, 32(suppl_1):D431–D433, January 2004. 1, 3.1

[56] Fan-Yun Sun, Jordan Hoffman, Vikas Verma, and Jian Tang. InfoGraph: Unsupervised and Semi-supervised Graph-Level Representation Learning via Mutual Information Maximization. In *International Conference on Learning Representations*, September 2019. 1

[57] Susheel Suresh, Pan Li, Cong Hao, and Jennifer Neville. Adversarial Graph Augmentation to Improve Graph Contrastive Learning. In *Advances in Neural Information Processing Systems*, November 2021. 5.1

[58] Shiyin Tan, Dongyuan Li, Renhe Jiang, Ying Zhang, and Manabu Okumura. Community-Invariant Graph Contrastive Learning. In *Proceedings of the 41st International Conference on Machine Learning*, pages 47579–47606. PMLR, July 2024. 5.1

[59] Vayer Titouan, Nicolas Courty, Romain Tavenard, and Rémi Flamary. Optimal transport for structured data with application on graphs. In *International Conference on Machine Learning*, pages 6275–6284. PMLR, 2019. 1, 5, 5.1, C.1

[60] Colin Troisemaine, Vincent Lemaire, Stéphane Gosselin, Alexandre Reiffers-Masson, Joachim Flocon-Cholet, and Sandrine Vaton. Novel Class Discovery: An Introduction and Key Concepts, February 2023. 1

[61] Sagar Vaze, Kai Han, Andrea Vedaldi, and Andrew Zisserman. Generalized category discovery. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7492–7501, 2022. 2.1

[62] 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. 5.1

[63] Cédric Vincent-Cuaz, Rémi Flamary, Marco Corneli, Titouan Vayer, and Nicolas Courty. Template based Graph Neural Network with Optimal Transport Distances. In *Advances in Neural Information Processing Systems*, October 2022. 5.3, 6.1

[64] Jing Wang, Songhe Feng, Gengyu Lyu, and Jiazheng Yuan. Surer: Structure-adaptive unified graph neural network for multi-view clustering. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 15520–15527, 2024. 1

[65] Yejiang Wang, Yuhai Zhao, Daniel Zhengkui Wang, and Ling Li. GALOPA: Graph transport learning with optimal plan alignment. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 5

[66] Yejiang Wang, Yuhai Zhao, Zhengkui Wang, Ling Li, Jiapu Wang, Fangting Li, Miaomiao Huang, Shirui Pan, and Xingwei Wang. Equivalence is all: A unified view for self-supervised graph learning. In *Forty-second International Conference on Machine Learning*, 2025. 1

[67] Man Wu, Shirui Pan, and Xingquan Zhu. OpenWGL: Open-world graph learning for unseen class node classification. *Knowledge and Information Systems*, 63(9):2405–2430, September 2021. 2.2

[68] 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*, 32(1):4–24,, January 2021. 1

[69] Jun Xia, Lirong Wu, Jintao Chen, Bozhen Hu, and Stan Z. Li. SimGRACE: A Simple Framework for Graph Contrastive Learning without Data Augmentation. In *Proceedings of the ACM Web Conference 2022*, WWW '22, pages 1070–1079, New York, NY, USA, April 2022. Association for Computing Machinery. 4.2, 5.1

[70] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019. 1, 3.1

[71] Pinar Yanardag and S.V.N. Vishwanathan. Deep Graph Kernels. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pages 1365–1374, New York, NY, USA, August 2015. Association for Computing Machinery. 1, 3.1

[72] Liang Yang, Yukun Cai, Hui Ning, Jiaming Zhuo, Di Jin, Ziyi Ma, Yuanfang Guo, Chuan Wang, and Zhen Wang. Universal graph self-contrastive learning. In *IJCAI*, pages 3534–3542, 2025. 1

[73] Liang Yang, Zhenna Li, Jiaming Zhuo, Jing Liu, Ziyi Ma, Chuan Wang, Zhen Wang, and Xiaochun Cao. Graph contrastive learning with joint spectral augmentation of attribute and topology. In *AAAI*, pages 21983–21991, 2025. 1

[74] Zhenyu Yang, Ge Zhang, Jia Wu, Jian Yang, Quan Z. Sheng, Shan Xue, Chuan Zhou, Charu Aggarwal, Hao Peng, Wenbin Hu, Edwin Hancock, and Pietro Liò. State of the Art and Potentialities of Graph-level Learning. *ACM Computing Surveys*, 57(2):1–40, February 2025. 1

[75] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, pages 5812–5823, Red Hook, NY, USA, December 2020. Curran Associates Inc. 1, 3.2, 5.1

[76] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*, pages 12310–12320. PMLR, 2021. 1

[77] Qin Zhang, Zelin Shi, Xiaolin Zhang, Xiaojun Chen, Philippe Fournier-Viger, and Shirui Pan. G2Pxy: Generative open-set node classification on graphs with proxy unknowns. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, IJCAI '23, pages 4576–4583, August 2023. 2.2

[78] Bingchen Zhao and Kai Han. Novel visual category discovery with dual ranking statistics and mutual knowledge distillation. In *Advances in Neural Information Processing Systems*, volume 34, pages 22982–22994, 2021. 1, 2.1, 3.2, C.1

[79] Yuhai Zhao, Yejiang Wang, Zhengkui Wang, Wen Shan, Miaomiao Huang, and Xingwei Wang. Graph contrastive learning with progressive augmentations. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1*, KDD '25, pages 2079–2088, New York, NY, USA, 2025. Association for Computing Machinery. 4.2

[80] Zhun Zhong, Enrico Fini, Subhankar Roy, Zhiming Luo, Elisa Ricci, and Nicu Sebe. Neighborhood Contrastive Learning for Novel Class Discovery. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10862–10870. IEEE Computer Society, June 2021. 1, 2.1, 3.2, C.1

[81] Jiaming Zhuo, Can Cui, Kun Fu, Bingxin Niu, Dongxiao He, Chuan Wang, Yuanfang Guo, Zhen Wang, Xiaochun Cao, and Liang Yang. Graph contrastive learning reimagined: Exploring universality. In *WWW*, pages 641–651, 2024. 1

[82] Jiaming Zhuo, Yintong Lu, Hui Ning, Kun Fu, Bingxin Niu, Dongxiao He, Chuan Wang, Yuanfang Guo, Zhen Wang, Xiaochun Cao, and Liang Yang. Unified graph augmentations for generalized contrastive learning on graphs. In *NeurIPS*, 2024. 1

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Our main contributions and scope have been accurately summarized in the last paragraph of introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations of our work in Section D.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: We do not introduce theoretical results in the paper.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide all the hyper-parameter values or search spaces in Section C.1. We also include the description of the data we used in Sections 3.1 and C.1.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We plan to release the code, datasets, and pre-trained models with sufficient instructions to faithfully reproduce our results.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide these details in Section C.1.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The results in Section 6 include both the means and standard deviations of more than 5 repeating runs.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide the computer resources information in Section C.1.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: Our work conform with the Code of Ethics

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes] ,

    Justification: We discuss the broader impacts of our work in Sections 7 and D.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: Our work poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All codes and datasets used in this work are publicly available for research purposes.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Our code and pre-processed GLNCD datasets will be provided.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: We do not involve crowsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: We do not involve research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: LLM is used only for writing, editing, or formatting purposes in this paper.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.

A Measure the Quality of Pairwise Pseudo Labels for Unlabeled Samples

The pseudo labels from ranking statistics (RS) Methods such as AutoNovel [27] generate pairwise pseudo-labels using ranking statistics (RS). Instead of directly computing the similarity (e.g., cosine similarity) between d-dimensional readout vectors $\mathbf{v}_i = \mathrm{Readout}(f_\theta(G_i))$ and $\mathbf{v}_j = \mathrm{Readout}(f_\theta(G_j))$ of samples i and j, RS ranks the d elements of each vector based on the element magnitudes. If the element ranking orders of two readout vectors are consistent, the corresponding samples are likely to belong to the same novel category, and we assign a pseudo-label $s_{ij} = 1$. Otherwise, we set $s_{ij} = 0$. The new-class head h_n is then trained using binary cross-entropy loss and the pairwise binary labels:

$$\mathcal{L}_{\mathrm{BCE}} = -\frac{1}{M^2}\sum_{i=1}^{M}\sum_{j=1}^{M}\left[s_{ij}\log h_n\left(\mathbf{v}_i\right)^\top h_n\left(\mathbf{v}_j\right) + \left(1-s_{ij}\right)\log\left(1-h_n\left(\mathbf{v}_i\right)^\top h_n\left(\mathbf{v}_j\right)\right)\right], \quad (6)$$

where $h_n\left(\mathbf{v}_i\right)$ is the softmaxed prediction on sample i and M is the number of unlabeled samples involved.

Han et al. [27] found the loss Eq. 6 to be the most critical component for NCD. Therefore, we aim to quantify its quality to enable deeper analysis. Given M unlabeled new-class samples, RS generates a pseudo-label matrix $\mathbf{S} \in \{0,1\}^{M\times M}$. Let $\mathbf{T}$ denote the ground-truth pairwise label matrix, where $T_{ij} = 1$ if samples i and j belong to the same novel class, and $T_{ij} = 0$ otherwise. Evaluating the quality of RS can thus be framed as a binary classification problem.

Evaluate pseudo labels with Matthews Correlation Coefficient According to Proposition 3, there are much more zeros than ones in $\mathbf{T}$. So the RS pseudo label quality evaluation is an imbalanced binary classification task, making accuracy an unsuitable evaluation metric. Instead, we adopt the Matthews Correlation Coefficient (MCC) [49], a widely used measure for evaluating classification performance under class imbalance. Let the confusion matrix for the binary classification be

$$\begin{pmatrix} TP & FN \\ FP & TN \end{pmatrix}.$$

Then MCC is defined as

$$MCC = \frac{TP\cdot TN - FP\cdot FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}},$$

which takes into account all four components of the confusion matrix, i.e., true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN), and therefore provides a comprehensive measure of the agreement between generated pseudo-labels and the ground truth pairwise labels.

Proposition 3. *Consider a class-balanced dataset with C categories, each containing M samples. Define the pairwise comparison matrix $\mathbf{T} \in \{0,1\}^{CM\times CM}$ such that $T_{ij} = 1$ if samples i and j belong to the same class, and $T_{ij} = 0$ otherwise. Then, the fraction of positive entries (i.e., entries equal to 1) in $\mathbf{T}$ is given by*

$$\frac{CM^2}{(CM)^2} = \frac{1}{C}.$$

Evaluate pseudo labels with the portion of samples with at least one same-class pair As stated in Proposition 3, the majority of sample pairs are negative, i.e., with a label 0. Each sample encounters far fewer true positive pairs than negative ones, making true positive pseudo-labels more valuable. Therefore, we use the ratio of samples that have encountered at least one true positive pair to quantify the quality of the pairwise pseudo-labels (generated by RS) .

B Parallel Differentiable BAPG Layer for Efficient FGW Distance

B.1 Forward

For two attributed graphs $G_1 = (\mathbf{C}_1, \mathbf{X}_1)$ and $G_2 = (\mathbf{C}_2, \mathbf{X}_2)$, the Fused Gromov Wasserstein distance (Definition 2) between them amounts to, given $\mathbf{M} \in \mathbb{R}^{n_s\times n_t}$ defined by $M_{ij} = D_{\Omega_x}(\mathbf{x}_{1,i}, \mathbf{x}_{2,j})$,

the following quadratic optimization problem

$$FGW = \min_{\mathbf{T}} \quad (1-\alpha)\langle \mathbf{T}, \mathbf{M} \rangle_F + \alpha \sum_{i,j,k,l} L(\mathbf{C}_{1i,k}, \mathbf{C}_{2j,l}) \mathbf{T}_{i,j} \mathbf{T}_{k,l}$$
$$s.t. \quad \mathbf{T} \in [0,1]^{n_s \times n_t} \qquad \mathbf{T}\mathbf{1} = \mathbf{p} \qquad \mathbf{T}^T\mathbf{1} = \mathbf{q}.$$

According to Proposition 1 in [53], it holds that

$$\sum_{i,j,k,l} L(\mathbf{C}_{1i,k}, \mathbf{C}_{2j,l}) \mathbf{T}_{i,j} \mathbf{T}_{k,l} = \left\langle \left[\mathbf{C}_1^{\odot 2} \mathbf{p} \mathbf{1}_{N_2}^\top + \mathbf{1}_{N_1} \mathbf{q}^\top \left(\mathbf{C}_2^{\odot 2}\right)^\top - 2\mathbf{C}_1 \mathbf{T} \mathbf{C}_2^\top \right], \mathbf{T} \right\rangle_F,$$

where $\mathbf{C}_1^{\odot 2}$ is the elementwise square and $\mathbf{1}_{N_2} \in \mathbb{R}^{n_t}$ is the all-one vector. Then we have

$$FGW = \min_{\mathbf{T}} \quad (1-\alpha)\langle \mathbf{T}, \mathbf{M} \rangle_F + \alpha \left\langle \left[\mathbf{C}_1^{\odot 2} \mathbf{p} \mathbf{1}_{N_2}^\top + \mathbf{1}_{N_1} \mathbf{q}^\top \left(\mathbf{C}_2^{\odot 2}\right)^\top - 2\mathbf{C}_1 \mathbf{T} \mathbf{C}_2^\top \right], \mathbf{T} \right\rangle_F$$
$$s.t. \quad \mathbf{T} \in [0,1]^{n_s \times n_t} \qquad \mathbf{T}\mathbf{1} = \mathbf{p} \qquad \mathbf{T}^T\mathbf{1} = \mathbf{q}.$$

We take $L(\mathbf{C}_{1i,k}, \mathbf{C}_{2j,l}) = (\mathbf{C}_{1i,k} - \mathbf{C}_{2j,l})^2$ and employ the Bregman Alternating Projected Gradient (BAPG) method proposed in [39] to solve the optimal transport plan $\mathbf{T}$ and the corresponding FGW distance. The pseudo code of BAPG can be found in Algorithm 1.

Algorithm 1: POT BAPG Solver [39] for Fused Gromov-Wasserstein Distance (Forward)

Input: Node feature cost matrix $\mathbf{M} \in \mathbb{R}^{n_s \times n_t}$. Sparse structure matrices $\mathbf{C}_1 \in \mathbb{R}^{n_s \times n_s}$. $\mathbf{C}_2 \in \mathbb{R}^{n_t \times n_t}$. Distributions $\mathbf{p} \in \mathbb{R}^{n_s}$, $\mathbf{q} \in \mathbb{R}^{n_t}$. Trade-off $\alpha \in (0,1)$, entropy $\epsilon > 0$. Max iterations T, Tolerance tol

Output: Optimal transport matrix $\mathbf{T} \in \mathbb{R}^{n_s \times n_t}$ and FGW distance

Initialize $\mathbf{T} \leftarrow \mathbf{p}\mathbf{q}^\top$;
Elementwise function: $f(\mathbf{C}_1) \leftarrow \mathbf{C}_1^{\odot 2}, h(\mathbf{C}_1) \leftarrow \mathbf{C}_1, f(\mathbf{C}_2) \leftarrow \mathbf{C}_2^{\odot 2}, h(\mathbf{C}_2) \leftarrow \mathbf{C}_2$;
Precompute constants: $R \leftarrow f(\mathbf{C}_1)\mathbf{p}\mathbf{1}_{n_t}^\top + \mathbf{1}_{n_s}(f(\mathbf{C}_2)\mathbf{q})^\top$;
Define gradient operator $\nabla_{\mathbf{T}} \leftarrow 2\alpha(R - 2h(\mathbf{C}_1)\mathbf{T}h(\mathbf{C}_2)^\top + (1-\alpha)\mathbf{M}$;
for $t \leftarrow 1$ **to** T **do**
  $\mathbf{T}_{\text{prev}} \leftarrow \mathbf{T}$;
  Row projection:
    Update: $\mathbf{T} \leftarrow \mathbf{T} \odot \exp\left(-\nabla_{\mathbf{T}}/\epsilon\right)$
    Normalize rows: $\mathbf{T}_{a,:} \leftarrow \frac{p_a}{\sum_j \mathbf{T}_{a,b}} \mathbf{T}_{a,:}$
  Column projection:
    Update: $\mathbf{T} \leftarrow \mathbf{T} \odot \exp\left(-\nabla_{\mathbf{T}}/\epsilon\right)$
    Normalize columns: $\mathbf{T}_{:,b} \leftarrow \frac{q_b}{\sum_a \mathbf{T}_{a,b}} \mathbf{T}_{:,b}$
  if $t \bmod 10 = 0$ **then**
    Compute error: $err \leftarrow \|\mathbf{T} - \mathbf{T}_{\text{prev}}\|_F$;
    if $err < tol$ **then**
      break;
    end
  end
end
return $\mathbf{T}$ and FGW;

Our proposed method, ProtoFGW-NCD, requires computing pairwise FGW distances between b_1 sparse adjacency matrices of varying sizes (represented as `torch_sparse.SparseTensor`) and b_2 dense adjacency matrices of prototype graphs with uniform size. Ideally, we aim to solve these FGW distances in parallel. However, both the official implementation [2] and the POT implementation [3] only support solving the FGW distance between one pair of dense matrices at one time. When b_1 and b_2 are large, invoking these implementations $b_1 b_2$ times within nested loops results in prohibitively long computational time.

[2] https://github.com/squareRoot3/Gromov-Wasserstein-for-Graph

[3] https://pythonot.github.io/gen_modules/ot.gromov.html#ot.gromov.BAPG_fused_gromov_wasserstein

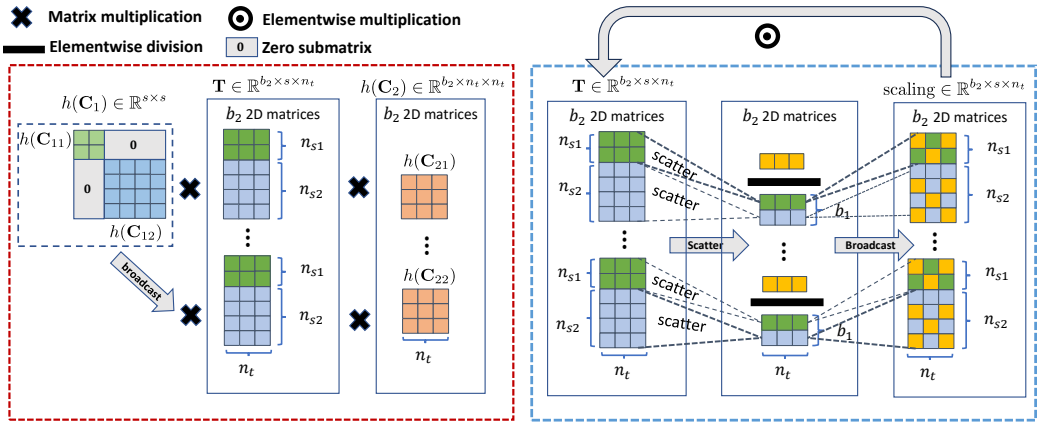


Figure 5: The illustration of two key operators in the forward of our BAPG layer, which parallels POT BAPG solver. The left is to parallel $h(\mathbf{C}_{1i})\mathbf{T}_{ij}h(\mathbf{C}_{2j})^\top$ (**Line 4** in Algorithm 1) and the right is to parallel $[\mathbf{T}_{ij}]_{:,b} \leftarrow \frac{q_b}{\sum_a [\mathbf{T}_{ij}]_{a,b}} [\mathbf{T}_{ij}]_{:,b}$(**Line 12** in Algorithm 1), for $i \in [1:b_1]$ and $j \in [1:b_2]$.

Observing Algorithm 1, we find that the main challenge in parallelizing the fused Gromov-Wasserstein (FGW) computation across $b_1 b_2$ groups lies in the irregular sizes of the b_1 sparse matrices. To address this, we propose constructing a sparse giant graph $\mathbf{C}_1 \in \mathbb{R}^{s\times s}$ (where $s = \sum_{i=1}^{b_1} n_{si}$) by arranging the sparse matrices $\mathbf{C}_{11}, \mathbf{C}_{12}, \ldots, \mathbf{C}_{1b_1}$ as diagonal blocks:

$$\mathbf{C}_1 = \begin{bmatrix} \mathbf{C}_{11} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_{12} & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{C}_{1b_1} \end{bmatrix} \in \mathbb{R}^{s\times s}.$$

This allows for intra-batch parallelization even under irregular matrix dimensions. Meanwhile, the b_2 dense matrices $\mathbf{C}_{21}, \mathbf{C}_{22}, \ldots, \mathbf{C}_{2b_2}$ are stacked into a tensor $\mathbf{C}_2 \in \mathbb{R}^{b_2\times n_t\times n_t}$, enabling parallelization along the first dimension via broadcasting rules. This design enables the $b_1 b_2$ pairs of FGW computations to be efficiently implemented using existing operators from `torch_sparse` and `torch_scatter`.

Since we need to solve for $b_1 b_2$ transport plans $\{\mathbf{T}_{ij} \in \mathbb{R}^{n_{si}\times n_t}\}, i \in [1:b_1], j \in [1:b_2]$, we represent them using a whole tensor $\mathbf{T} \in \mathbb{R}^{b_2\times s\times n_t}$. Placing the batch size b_2 of dense matrices along the first dimension facilitates efficient sparse-dense matrix multiplication (via `spmm`) over all b_2 matrices. Our goal is to parallelize $b_1 b_2$ FGW with the operations among $\mathbf{C}_1 \in \mathbb{R}^{s\times s}$, $\mathbf{C}_2 \in \mathbb{R}^{b_2\times n_t\times n_t}$, and $\mathbf{T} \in \mathbb{R}^{b_2\times s\times n_t}$. The key computational steps in Algorithm 1 to parallelize are

- **Line 4** $h(\mathbf{C}_{1i})\mathbf{T}_{ij}h(\mathbf{C}_{2j})^\top$ that can be parallelized using the `spmm` operator from `torch_sparse`[4]
- **Line 12** $[\mathbf{T}_{ij}]_{:,b} \leftarrow \frac{q_b}{\sum_a [\mathbf{T}_{ij}]_{a,b}} [\mathbf{T}_{ij}]_{:,b}$ that requires computing the normalization sums $\sum_a [\mathbf{T}_{ij}]_{a,b}$ for each pair $(\mathbf{C}_{1i}, \mathbf{C}_{2j})$ using `scatter_sum` from `torch_scatter`[5], followed by grouped broadcasting across the first two dimensions of $\mathbf{T} \in \mathbb{R}^{b_2\times s\times n_t}$.

We illustrate the above two parallelization designs in Figure 5. The forward process of our parallel BAPG layer is concluded in Algorithm 2, in a pytorch style.

[4]https://github.com/rusty1s/pytorch_sparse
[5]https://github.com/rusty1s/pytorch_scatter

Algorithm 2: Our BAPG layer for Fused Gromov-Wasserstein Distance (Forward)

Input: (1) Node feature cost matrix $\mathbf{M} \in \mathbb{R}^{b_2 \times b_1 \times n_s \times n_t}$. (2) Diagonal block matrix $\mathbf{C}_1 \in \mathbb{R}^{s \times s}$ stacked by b_1 sparse structure matrices $\mathbf{C}_{11}, \mathbf{C}_{12}, \ldots, \mathbf{C}_{1b_1}$ where $s = \sum_{i=1}^{b_1} n_{si}$. (3) The tensor $\mathbf{C}_2 \in \mathbb{R}^{b_2 \times n_t \times n_t}$ stacked by b_2 dense structure matrices $\mathbf{C}_{21}, \mathbf{C}_{22}, \ldots, \mathbf{C}_{2b_2}$. (4) Marginal distributions of b_1 sparse matrices $\mathbf{p} = [\mathbf{p}_1 \mid \mathbf{p}_2 \mid \cdots \mid \mathbf{p}_{b_1}] \in \mathbb{R}^s$ where $\mathbf{p}_i \in \mathbb{R}^{n_{si}}$. (5) Marginal distributions of b_2 dense matrices $\mathbf{q} \in \mathbb{R}^{b_2 \times n_t}$. (6) $\mathbf{g} \in [1 : b_1]^s$ where $\mathbf{g}[a]$ is the graph index of node a in the giant graph $\mathbf{C}_1$. (7) Trade-off $\alpha \in (0, 1)$, entropy $\epsilon > 0$. Max iterations T, tolerance tol

Operator : @: @ in pytorch, matrix/tensor multiplication. $*$: elementwise multiplication. $.view(...)$: pytorch tensor .view(...). $[:, \text{ind}, :]$: pytorch tensor slice.

Output: Optimal transport plans $\mathbf{T} \in \mathbb{R}^{b_2 \times s \times n_t}$ of and FGW distances between $b_1 b_2$ pairs

Initialize $\mathbf{T} \leftarrow \mathbf{p}.view(s, 1) * \mathbf{q}.view(b2, 1, n_t)$;
Elementwise function: $f(\mathbf{C}_1) \leftarrow \mathbf{C}_1^{\odot 2}, h(\mathbf{C}_1) \leftarrow \mathbf{C}_1, f(\mathbf{C}_2) \leftarrow \mathbf{C}_2^{\odot 2}, h(\mathbf{C}_2) \leftarrow \mathbf{C}_2$;
Precompute constants:
$R \leftarrow f(\mathbf{C}_1)@\mathbf{p}.view(s, 1)@\mathbf{1}_{n_t}^\top + \mathbf{1}_{n_s}.view(s, 1)@\mathbf{q}.view(b_2, 1, n_t)@f(\mathbf{C}_2)$;
Define gradient operator $\nabla_{\mathbf{T}} \leftarrow 2\alpha(R - 2h(\mathbf{C}_1)@\mathbf{T}@h(\mathbf{C}_2) + (1 - \alpha)\mathbf{M}$;
for $t \leftarrow 1$ **to** T **do**
  $\mathbf{T}_{\text{prev}} \leftarrow \mathbf{T}$;
  /* Row projection */
  $\mathbf{T} \leftarrow \mathbf{T} * \exp(-\nabla_{\mathbf{T}}/\epsilon)$ // Update
  $\mathbf{S}_r = \mathbf{p}/(\mathbf{T}.sum(-1))$ // Row scaling factor
  $\mathbf{T} \leftarrow \mathbf{S}_r.view(b2, s, 1) * \mathbf{T}$ // Normalize rows
  /* Column projection */
  $\mathbf{T} \leftarrow \mathbf{T} * \exp(-\nabla_{\mathbf{T}}/\epsilon)$ // Update
  /* Sum the elements from the same sparse matrix along the 2nd dim */
  group_sum = scatter_sum$(\mathbf{T}, \mathbf{g}, dim = 1)$
  /* Broadcast to each group and get columnm scaling factor */
  $\mathbf{S}_c = \mathbf{q}.view(b_2, 1, n_t)/\text{group\_sum}[:, \mathbf{g}, :]$
  $\mathbf{T} \leftarrow \mathbf{S}_c * \mathbf{T}$ // Normalize columns
  if $t \bmod 10 = 0$ **then**
    Compute error: $err \leftarrow \|\mathbf{T} - \mathbf{T}_{\text{prev}}\|_F$;
    if $err < tol$ **then**
      break;
    end
  end
end
return $\mathbf{T}$ and corresponding FGW distances

B.2 Backward

With $L(\mathbf{C}_{1i,k}, \mathbf{C}_{2j,l}) = (\mathbf{C}_{1i,k} - \mathbf{C}_{2j,l})^2$, the gradients of 1-to-1 FGW w.r.t. the input elements are

$$\frac{\partial FGW}{\partial \mathbf{M}} = (1 - \alpha)\mathbf{T}$$
$$\frac{\partial FGW}{\partial \mathbf{C}_1} = 2\alpha\mathbf{C}_1 \odot (\mathbf{p}\mathbf{p}^\top) - 2\alpha\mathbf{T}\mathbf{C}_2\mathbf{T}^\top$$
$$\frac{\partial FGW}{\partial \mathbf{C}_2} = 2\alpha\mathbf{C}_2 \odot (\mathbf{q}\mathbf{q}^\top) - 2\alpha\mathbf{T}^\top\mathbf{C}_1\mathbf{T}$$
$$\frac{\partial FGW}{\partial \alpha} = \left\langle \left[\mathbf{C}_1^{\odot 2}\mathbf{T}\mathbf{1}_{N_2}\mathbf{1}_{N_2}^\top + \mathbf{1}_{N_1}\mathbf{1}_{N_1}^\top\mathbf{T}\left(\mathbf{C}_2^{\odot 2}\right)^\top - 2\mathbf{C}_1\mathbf{T}\mathbf{C}_2^\top\right], \mathbf{T}\right\rangle - \langle \mathbf{T}, \mathbf{M}\rangle_F$$
$$\frac{\partial FGW}{\partial \mathbf{p}} = \alpha\mathbf{C}_1^{\odot 2}\mathbf{p} \qquad \frac{\partial FGW}{\partial \mathbf{q}} = \alpha(\mathbf{C}_2^{\odot 2})^\top\mathbf{q}.$$

The backward computations of $b_1 b_2$ FGW distances can be parallelized via similar techniques introduced in parallel FGW computation described in the forward process of our BAPG layer (Section B.1).

C More Experiments

C.1 Implementation Details

The implementation is based on Pytorch2.5 [52] and PyG2.6 [17]. All experiments are conducted on Ubuntu 22.04 server equipped with an RTX 4090 GPU and Intel Xeon Gold 6240C CPU. For FGW [59, 39], we implement an extremely efficient version from scratch (Section B) instead of using Python Optimal Transport toolbox [20].

Many studies report test performance based on the epoch that achieves the best performance on the validation set. However, in the context of NCD, selecting the epoch based on clustering accuracy over unlabeled validation samples would require knowledge of the ground truth new-class labels, which violates the NCD assumption. Alternatively, using old-class accuracy on the validation set may reinforce model bias toward known classes, potentially harming performance on new classes. Therefore, we report results from the final training epoch. In addition to the three baseline methods designed in Section 3.2, we also implement a K-means baseline: after pretraining the GNN encoder with GraphCL, we apply K-means directly to the GNN representations of the unlabeled training samples and old-class test samples, and report the corresponding clustering accuracies. For all experiments, we use the AdamW optimizer [45] and cosine annealing scheduler [44] with warmup.

Table 7: The split information of four GLNCD datasets

Dataset	# train	# test	# all
ENZYMES	420	120	600
MalNet-Tiny	3500	1000	5000
REDDIT12K	8350	2386	11929
CIFAR10	35000	10000	60000

Following common practice in visual NCD [80, 78, 27], we first determine the hyperparameters for AutoNovel on a given dataset. These hyperparameters are then inherited by NCL and DualRS on the same dataset, and only the hyperparameters that differ from AutoNovel are subsequently tuned for these two methods. ProtoFGW-NCD has a fundamentally different architecture from the above baselines, and its hyperparameters are therefore not influenced by AutoNovel. The hyperparameter values or search spaces of these mehods are presented in Table 8.

C.2 Benchmarking Our BAPG Layer Implementation

Section B introduces our BAPG layer for efficient, differentiable FGW distance computation, which leverages `torch_sparse` and `torch_scatter` to parallels pairwise FGW computations between b_1 sparse matrices and b_2 dense matrices. Compared to the BAPG implementation in POT [6], our improved version supports sparse matrices, parallelized iterative solving, and efficient automatic differentiation. These enhancements significantly promote the broader application of the FGW distance in graph-level machine learning. In this section, we compare our parallel BAPG implementation with POT on a series of synthetic attributed graph datasets.

Graph sizes considered are 20, 50, 100, 200, 500, and 1000 nodes. For each size, we generate 1000 Contextual Stochastic Block Model (CSBM) graphs [14, 48]. Prototype graphs are typically used to represent large-scale topological relationships; thus, each node in a prototype graph corresponds to a block or community of nodes. While no universally accepted formula exists for determining the optimal number of communities in a network, empirical studies suggest that the number of communities typically grows sub-linearly with the number of nodes N, often approximated as $O(N/\log N)$, especially in scale-free or real-world networks where community sizes follow a power-law distribution [51, 3]. Therefore, for graphs of size N, we use 10 prototype graphs of size $N/2 \log N$ to capture patterns in the dataset, and the dataset is denoted CSBM-N-10. In Figure 6, we display 10 prototype graphs used to generate 1000 graphs in CSBM-100-10 dataset. Commonly used batch sizes in graph learning are 64, 128, 256, and 512. We evaluate the efficiency of different

[6] https://pythonot.github.io/gen_modules/ ot.gromov.html#ot.gromov.BAPG_fused_gromov_wasserstein

Table 8: Hyperparameter values and search spaces of GLNCD methods

Group	Hyperparameter	Value or Search Space
Common Hyperparameters		
Optimization	Learning rate	[0.001, 0.005, 0.01, 0.05, 0.1]
	Dropout	[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8]
	Weight decay	[0.0, 5e-5, 1e-4, 5e-4, 1e-3, 5e-3, 1e-2]
	Cosine warmup steps	[2, 5, 10]
	Batch size	[64, 128, 256, 512]
	Max epochs	[20, 50, 100, 300]
Neural Network Arch.	GNN encoder layer	[2, 3, 4, 5, 6]
	Hidden dimension	[32, 64, 128, 256]
	has_residual	[False, True]
	has_ffn	[False, True]
	Normalization	[batchborm, None]
Graph SSL	Node droprate	[0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6]
	Temp. for contrastive loss	[0.1, 0.3, 0.5, 0.7, 0.9, 1.1]
GLNCD method Hyperparameters		
All baselines	Encoder pooling readout	['mean', 'add', 'max']
AutoNovel	Topk in RS	[5, 10, 15]
	Rampup length	[10, 50, 80, 150, 300]
	Rampup coefficient	[1.0, 5.0, 25., 50.]
NCL	Labeled NCL loss weight	[0.2, 1]
	Unlabeled NCL loss weight	[0.2, 1]
	Queue length	[200, 2000]
DualRS	Memory bank length	[256, 512, 1024]
ProtoFGW-NCD	Epsilon	[0.01, 0.05, 0.07, 0.11, 0.15, 0.19]
	Prototype node feature std.	[0.5, 1.]
	# prototype graphs	range(10, 130, 10)
	# prototype graph nodes	20

BAPG implementations in comparing all 1000 graphs with the 10 prototype graphs under various batch sizes.

The average time for computing the FGW distance between a batch of graphs and all 10 prototype graphs, under different batch sizes across various datasets, is shown in Table 6. The total time required to traverse each entire dataset is presented in Figure 4. As summarized in Table 6, our parallel BAPG solver delivers dramatic runtime improvements over the POT implementation across all batch sizes and problem scales. While POT's runtime grows roughly linearly with batch size (e.g., from ≈8.9 s at B=64 to ≈67.3 s at B=512 on CSBM-20-10), our approach maintains a nearly constant per-batch cost (≈0.02–0.04 s), yielding speedups that increase from ≈250× to ≈2070× as B grows. Furthermore, the degree of acceleration decreases as the graph size N increases—exceeding 400×–2000× for small-to-medium sizes ($N \leq 100$) at B=512, yet still achieving 3×–14× for large-size graphs (N=500–1000). These findings demonstrate that our technique effectively amortizes overhead and exploits parallelism for batches of FGW problems, offering exceptional throughput for the tasks of small- and medium-size graphs while retaining nontrivial gains even in large-graph settings. This is also supported by the epoch time comparison displayed in Figure 4.

D Limitations

As the first work to consider graph-level NCD, *this paper aims to introduce the new task of GLNCD and examine whether existing visual NCD methods can be effectively adapted by simply replacing their components with graph-domain counterparts*. To address this question, we adapt three classic NCD methods from computer vision to establish GLNCD baseline approaches (Section 3.2) and evaluate their performance on four newly designed GLNCD datasets spanning different domains (Section 3.1).

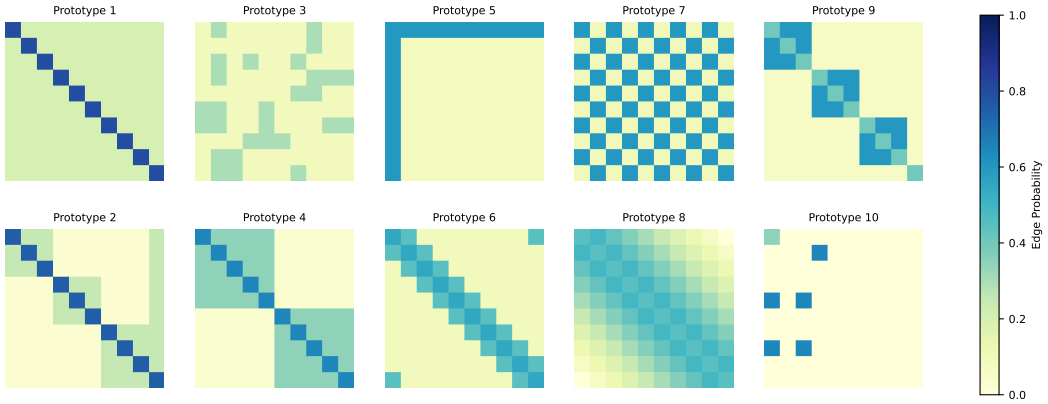


Figure 6: The prototype graphs used to synthesize the 1000 graphs in CSBM-100-10 dataset.

Our experimental results and analysis (Section 4) clearly indicate a negative answer, suggesting that current graph SSL methods and ranking statistics are insufficient in capturing structural information within graphs, thereby leading to low GLNCD performance. Although we do not adapt the most recent visual NCD methods, the limitations we observe from direct transfer can be generalized to them. This is because these recent visual NCD methods would still rely on existing graph SSL techniques for representation learning and employ pseudo-labeling strategies that neglect graph structure to train the new-class head. Additionally, despite the development of our parallel BAPG solver (Section B), which significantly improves computational efficiency in solving FGW for graph learning—achieving an impressive 2070× speedup on the CSBM-20-10 dataset where graphs have 20 nodes—the experimental results (Section C.2) show that the acceleration drops to only about 3.3× on larger graphs (CSBM-1000-10). Future work may focus on analyzing the causes of reduced speedup and improving the implementation, or exploring alternatives to FGW that more efficiently exploit graph structure to enhance both graph SSL and pseudo-labeling strategies.