THESAURUS: Contrastive Graph Clustering by Swapping Fused Gromov-Wasserstein Couplings

Bowen Deng[1] [*], **Tong Wang**[2] [*], **Lele Fu**[1], **Sheng Huang**[1], **Tao Zhang**[1] [†], **Chuan Chen**[2] [†],

[1]School of Systems Science and Engineering, Sun Yat-sen University, Guangzhou, China
[2]School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, China
{dengbw3, wangt, full, huangs}@mail2.sysu.com, {chenchuan,zhangt358}@mail.sysu.edu.cn

Abstract

Graph node clustering is a fundamental unsupervised task. Existing methods typically train the encoder through self-supervised learning and then apply K-means to the encoder representations. Some methods use this clustering result directly as the final output, while others initialize centroids based on this initial clustering and then finetune both the learnable centroids and the encoder. However, due to their reliance on K-means, these methods inherit its drawbacks when the cluster separability of representations is low, facing challenges from the Uniform Effect and Cluster Assimilation. We summarize three main reasons for the low cluster separability in existing methods: (1) lack of contextual information prevents discrimination between similar nodes from different clusters; (2) training tasks are not sufficiently aligned with the downstream clustering task; (3) the cluster information in the graph structure is not appropriately exploited. To address these issues, we propose conTrastive grapH clustEring by SwApping fUsed gRomov-wasserstein coUplingS (THESAURUS). THESAURUS introduces semantic prototypes to provide contextual information; employs a cross-view "clustering" assignment prediction pretext task that aligns well with the downstream clustering task; and utilizes Gromov-Wasserstein Optimal Transport (GW-OT) along with the proposed prototype graph to exploit cluster information in the graph structure thoroughly. To adapt to diverse real-world data, THESAURUS updates the prototype graph and the prototype marginal distribution used in OT using momentum and the observed data. Extensive experiments demonstrate that THESAURUS achieves higher cluster separability than prior art, significantly mitigating the Uniform Effect and Cluster Assimilation issues.

1 Introduction

Graph node clustering (Wang et al. 2024; Liu et al. 2023b), a fundamental unsupervised task, aims to group similar nodes into clusters. Recently, methods based on Graph Self-Supervised Learning (Graph SSL) (Liu et al. 2022a) have become predominant (Liu et al. 2023b). Despite their success, these methods, e.g., Dink-Net (Liu et al. 2023a), heavily rely on K-means to guide the representation learning process and/or to get the final clustering results, and thus inherit the shortcomings of K-means. Clusters that contain significantly more samples than others are called majority clusters, and conversely, those with fewer samples minority clusters. When the input node representations exhibit low cluster separability, K-means results may show **(1) Uniform Effect**: samples from majority clusters being assigned to neighboring minority clusters (Xiong, Wu, and Chen 2009), and **(2) Cluster Assimilation**: minority clusters being merged into majority clusters (Lu, Cheung, and Tang 2021). In contrast, high cluster separability, an ideal clustering outcome characterized by large inter-cluster distances and small intra-cluster distances, can alleviate these two issues (Lu, Cheung, and Tang 2021), as demonstrated by the Dink-Net finetune effect experiment presented below.

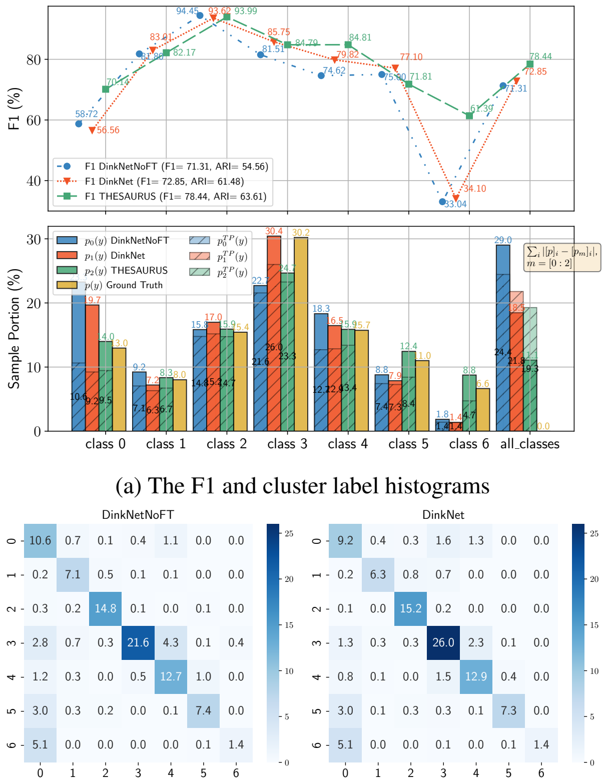
Uniform Effect & Cluster Assimilation in Dink-Net

The current state-of-the-art (SOTA) model, Dink-Net, is pretrained with a binary classification to distinguish between original data and randomly corrupted and shuffled data. The pretrained Dink-Net is referred to as Dink-Net-NoFT. After pretraining, K-means is employed to perform a clustering on Dink-Net-NoFT output, initializing the learnable centroids $\{\mathbf{c}_i\}_{i=0}^{C-1}$. In the later finetune stage, the encoder and centroids are adjusted by minimizing the cluster dilation and shrink losses to enhance cluster separability. These two losses are respectively defined as $\mathcal{L}_d = \frac{-1}{(C-1)C}\sum_i \sum_{i\neq i} \|\mathbf{c}_i - \mathbf{c}_j\|_2^2$ and $\mathcal{L}_s = \frac{1}{BC}\sum_{i=0}^{B-1}\sum_{j=0}^{C-1} \|\mathbf{z}_i - \mathbf{c}_j\|_2^2$, where B is the node batch size and $\mathbf{z}_i$ is the representation of node i. For a node i, its predicted cluster is $\hat{\mathbf{y}}_i = \arg\min_j \|\mathbf{z}_i - \mathbf{c}_j\|_2$. Fig. 1 shows the finetune impact on Dink-Net.

Before finetune, two phenomena are observed on the Cora dataset. **(1) Uniform Effect**: the largest cluster (cluster 3) has many external edges with the much smaller clusters 0 and 4, causing them to be close in the embedding space. This is evidenced in Fig. 1b, where many nodes from cluster 3 are misclassified into clusters 4 and 0. **(2) Cluster Assimilation**: The smallest cluster (cluster 6) has the most external edges with cluster 0, leading to its merging into cluster 0.

After finetuning towards high separability, some of the nodes from cluster 3 that were misclassified into clusters 4 and 0 are returned to cluster 3, resulting in a significant improvement in the F1 score for cluster 3. However, since the

[*]These authors contributed equally.
[†]Corresponding authors.



(a) The F1 and cluster label histograms

DinkNetNoFT

	0	1	2	3	4	5	6
0	10.6	0.7	0.1	0.4	1.1	0.0	0.0
1	0.2	7.1	0.5	0.1	0.1	0.0	0.0
2	0.3	0.2	14.8	0.1	0.0	0.1	0.0
3	2.8	0.7	0.3	21.6	4.3	0.1	0.4
4	1.2	0.3	0.0	0.5	12.7	1.0	0.0
5	3.0	0.3	0.2	0.0	0.1	7.4	0.0
6	5.1	0.0	0.0	0.0	0.0	0.1	1.4

DinkNet

	0	1	2	3	4	5	6
0	9.2	0.4	0.3	1.6	1.3	0.0	0.0
1	0.2	6.3	0.8	0.7	0.0	0.0	0.0
2	0.1	0.0	15.2	0.2	0.0	0.0	0.0
3	1.3	0.3	0.3	26.0	2.3	0.1	0.0
4	0.8	0.1	0.0	1.5	12.9	0.4	0.0
5	3.0	0.1	0.3	0.3	0.1	7.3	0.0
6	5.1	0.0	0.0	0.1	0.0	0.1	1.4

(b) The confusion matrices

Figure 1: The effect of Dink-Net finetune towards cluster separability on Cora. **(a)** The **top** row illustrates the F1 scores for each class before and after finetune, as well as the average F1 score over all classes. The **second** row shows the distribution of predicted labels by three models, along with the ground-truth labels.It also presents the distribution of predicted labels for true-positive (TP) samples, denoted as $p_i^{TP}(y), i \in \{0,1,2\}$. The final set of bars shows the differences between the predicted and ground-truth distributions. **(b)** displays the confusion matrices of Dink-Net before and after finetune, normalized by the number of nodes.

cluster separability is still insufficient, the smallest cluster (cluster 6) remains merged into cluster 0, leading to a low F1 score on class 6 and a low Macro-F1 score. This experiment underscores the critical role of high cluster separability and reveals that even the current SOTA struggles to achieve sufficient cluster separability to effectively address the challenges of Uniform Effect and Cluster Assimilation.

Current Model Limitations and Contributions

We summarize three **limitations (Ls)** impeding current methods achieving high cluster separability. **L1: Lack of contextual information hinders distinguishing synonymous nodes (i.e., similar nodes from different classes).** When a graph has many inter-class edges, low-pass graph filters (e.g., some popular GNNs) tend to generate similar embeddings for neighboring nodes from different classes (NT and Maehara 2019; Chen et al. 2020). Distinguishing such nodes based solely on embedding distances is challenging. Just as synonyms in a thesaurus need context to be properly understood, "childish" implies immaturity in "Stop being so childish!" while "childlike" conveys innocence in "She has a childlike wonder." Nodes also require contextual information for accurate clustering. Current methods depend only on the distances between embeddings and thus no contextual details are provided. As a result, they fail to differentiate between closely embedded nodes from different classes, leading to the mixup of nodes from clusters 0, 5, and 6 in Fig. 1. **L2: Training tasks are not well aligned with downstream clustering tasks.** The alignment between pretext and downstream tasks is crucial for SSL performance (Lee et al. 2021; Wei et al. 2021; Li et al. 2023). However, current pretext tasks often lack this alignment. **(1)** SDCN (Bo et al. 2020) reconstructs attributes and adopts a dual self-supervised strategy derived from DEC (Xie, Girshick, and Farhadi 2016). DFCN (Tu et al. 2021) reconstructs both attributes and structures with a DEC-like triplet self-supervised task. The reconstruction tasks do not optimize cluster separability explicitly and fail to align closely with clustering. DEC-style tasks use K-means to cluster pretrained representations initially. However, the pretrained representations often exhibit low separability, incurring Uniform Effect and Cluster Assimilation. Since finetune only refines the initial clustering, substantial improvements in addressing these issues are not available via DEC-style tasks. **(2)** Regarding the contrastive ones, DCRN (Liu et al. 2022b) and HSAN (Liu et al. 2023d) only preserve self-correlations, not aligned with clustering. SCGC (Liu et al. 2023c) and S^{3}GC (Devvrit et al. 2022) treat neighbors as positive pairs, but neighbors are not always of the same class, introducing clustering noise. Although the finetune task of (Liu et al. 2023a) aligns with the clustering objective, the unrelated pretrain task limits the representation separability. Thus, Dink-Net finetune cannot resolve all challenges, as shown in Fig. 1. **L3: The cluster information in graph structure is not appropriately extracted.** Existing methods primarily integrate structure information into embeddings via encoding with GCNs (Kipf and Welling 2017) or graph filters (Shuman et al. 2013). However, over-smoothing and over-squashing (Oono and Suzuki 2020; Nguyen et al. 2023) may hurt structure information. HSAN (Liu et al. 2023d) processes structure through a linear layer, yet it lacks permutation invariance, unduly emphasizing node indices over structure information. DFCN (Tu et al. 2021) and DCRN (Liu et al. 2022b) reflect the structure information through adjacency reflection loss. SCGC (Liu et al. 2023c) and S^{3}GC (Devvrit et al. 2022) take neighbors as positive samples and maximizes their similarities. These four methods implicitly treat neighbors as belonging to the same class, misleading the clustering on adjacent nodes from different classes.

To address the above limitations, we propose a novel contrastive graph clustering method, THESAURUS. **(1)** THESAURUS establishes semantic prototypes in the embedding space, each representing a semantic category. The relationships between one node and these prototypes constitute its context. **(2)** Inspired by SwAv (Caron et al. 2020), THESAURUS considers semantic prototypes as centroids and learns by predicting the node clustering assignments across different data augmentation views. **(3)** To explore the structure cluster information, we encode the relationships between prototypes as prototype graph, and then match it with the data graph using GW-OT (Mémoli 2011; Peyré and Cuturi 2019). **(4)** To exploit structure and attribute information comprehensively, the last two designs are unified by our Task and Structure Alignment (TSA) module based on Fused Gromov-Wasserstein OT (FGW-OT) (Titouan et al. 2019). **(5)** A momentum module for prototype graph and marginal distribution is developed for data adaptability.

Our main contributions are as follows. **(1)** We identify that prior methods has insufficient cluster separability and face the Uniform Effect and Cluster Assimilation challenges. **(2)** We propose a novel graph contrastive learning framework THESAURUS, which leverages semantic prototypes to provide contextual information. **(3)** We design the TSA module to align the pretext task to clustering and exploit the cluster information in graph structure. **(4)** We develop a momentum strategy for the prototype graph and prototype marginal distribution for data adaptability. **(5)** Extensive experiments demonstrate that THESAURUS achieves high cluster separability and significantly outperforms existing methods.

2 Related Work

Deep Graph Clustering

Early graph clustering methods use Autoencoder (AE) and Graph Autoencoder (GAE) (Kipf and Welling 2016) for feature extraction, followed by K-means or spectral clustering (Cao, Lu, and Xu 2016; Wang et al. 2017; Pan et al. 2018). Later methods such as DAEGC (Wang et al. 2019), SDCN (Bo et al. 2020), AGCN (Peng et al. 2021), and DFCN (Tu et al. 2021) incorporate DEC-style tasks to better align with clustering objectives. With the advent of graph contrastive learning (Liu et al. 2022a), contrastive graph clustering gained popularity. AGE (Cui et al. 2020) uses dynamic positive and negative sample pairs constructed according to dynamic pair similarities, DCRN (Liu et al. 2022b) introduces DICR loss to reduce cross-view correlations between a node and the other nodes, SCGC (Liu et al. 2023c) maximizes neighbor similarity, and S^{3}GC (Devvrit et al. 2022) optimizes a one-layer GNN with InfoNCE-style loss (van den Oord, Li, and Vinyals 2019). DinkNet (Liu et al. 2023a) maximizes differences between original and adversarial data, akin to maximizing the JSD lower bound of mutual information (Shrivastava et al. 2023), and then finetunes towards cluster separability via minimizing $\mathcal{L}_d$ and $\mathcal{L}_s$.

Optimal Transport

Optimal transport (OT) (Monge 1781; Villani et al. 2009) is a mathematical framework for measuring distances between distributions, finding the most cost-efficient way to transform one distribution into another. It has gained prominence in machine learning for tasks like supervised learning (Frogner et al. 2015), domain adaptation (Courty et al. 2017), and generative modeling (Tolstikhin et al. 2018). The optimal transport cost deduces the Wasserstein Distance, which does not require two probability distributions to have overlapping support sets. However, when distributions are defined in different or incomparable spaces, classical OT is not applicable. For example, transporting $s \in \mathbb{R}^2$ to $t \in \mathbb{R}^3$ lacks a meaningful cost measurement. Considering the well-defined distances within two distinct spaces $\mathcal{S}$ and $\mathcal{T}$, denoted as $D_{\mathcal{S}} : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$ and $D_{\mathcal{T}} : \mathcal{T} \times \mathcal{T} \to \mathbb{R}$, GW-OT (Mémoli 2011; Peyré, Cuturi, and Solomon 2016) regards the cost of transporting $s_i \in (\mathcal{S}, D_{\mathcal{S}})$ to $t_j \in (\mathcal{T}, D_{\mathcal{T}})$ as the difference between the relative relationship between s_i and other elements s_k in $\mathcal{S}$, and that between t_j and other elements t_l in $\mathcal{T}$. Such ability to transport across incomparable spaces makes it useful in tasks such as cross-lingual alignment (Alvarez-Melis and Jaakkola 2018) and cross-modal alignment (Gong, Nie, and Xu 2022).

3 Methodology

In this section, we present the proposed graph contrastive learning framework in detail, discussing how each component addresses the corresponding limitation in prior art.

THESAURUS Overview and Notations

The attribute graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$, consisting of N nodes from $\mathcal{V}$ and E edges from $\mathcal{E}$, can be summarized by the tuple $\mathcal{G} = (\mathbf{A}, \mathbf{X})$. Here, $\mathbf{A} \in \mathbb{R}^{N \times N}$ is the (binary) adjacency matrix, and $\mathbf{X} \in \mathbb{R}^{N \times d_0}$ is the node attribute matrix. For convenience, we will use these two graph notations interchangeably in the following sections.

Our framework is illustrated in Fig. 2. Initially, part of edges and feature dimensions of the original graph are masked to generate two distinct (but similar) augmented views $\mathcal{G}_1 = (\mathbf{A}_1, \mathbf{X}_1)$ and $\mathcal{G}_2 = (\mathbf{A}_2, \mathbf{X}_2)$. Subsequently, a GCN encoder f_θ (Kipf and Welling 2017) and an MLP projector $f_\omega : \mathbb{R}^{d_1} \to \mathbb{R}^d$ are employed to map these views into representations $\mathbf{Z}_1 = f_\omega \circ f_\theta(\mathbf{A}_1, \mathbf{X}_1) \in \mathbb{R}^{N \times d}$ and $\mathbf{Z}_2$, where $f_1 \circ f_2$ denotes the function composition and the whole neural network. The cosine similarities between $\mathbf{z}_{1,i} = [\mathbf{Z}_1]_i$ and S semantic prototypes $\{\mathbf{s}_i \in \mathbb{R}^{1 \times d}\}_{i=1}^{S}$ form the context-aware representation $\mathbf{r}_{1,i} \in \mathbb{R}^{1 \times S}$ of node i in view 1, where $[\cdot]_i$ is the i-th row of matrix. Similarly, the context-aware vector $\mathbf{r}_{2,i}$ in view 2 is obtained. Let $\mathbf{S} = [\mathbf{s}_1, \cdots, \mathbf{s}_S] \in \mathbb{R}^{S \times d}$ be the learnable prototype matrix, $\mathbf{R}_1 = \mathbf{Z}_1 \mathbf{S}^\top \in \mathbb{R}^{N \times S}$ be the context-aware representation matrix, and $\mathbf{B}_1 \in \mathbb{R}^{S \times S}$ be the prototype graph in view 1. THESAURUS computes the (optimal) node-prototype assignment $\mathbf{Q}_1 \in \mathbb{R}^{N \times S}$ for view 1 by solving the FGW-OT problem involving $\mathbf{R}_1$, $\mathbf{B}_1$, and $\mathbf{A}_1$. Similarly, the assignment $\mathbf{Q}_2$ is obtained. Once the assignments are got, we train the network $f_\omega \circ f_\theta$ to predict $\mathbf{Q}_2$ from $\mathbf{Z}_1$ and vice versa. After training, the $\mathbf{R}$ of the original graph $\mathcal{G}$ is fed into K-means to get the final clustering result $\Phi \in \{0, 1\}^{N \times C}$.

Address L1: Context via Sematic Prototypes

To capture the subtle differences between synonymous nodes (those similar nodes from different clusters), we draw inspiration from the human ability to accurately distinguish synonyms using textual context. For instance, consider the sentences "Stop being so childish!" and "She has a child-like wonder." In the first sentence, "stop being" conveys a

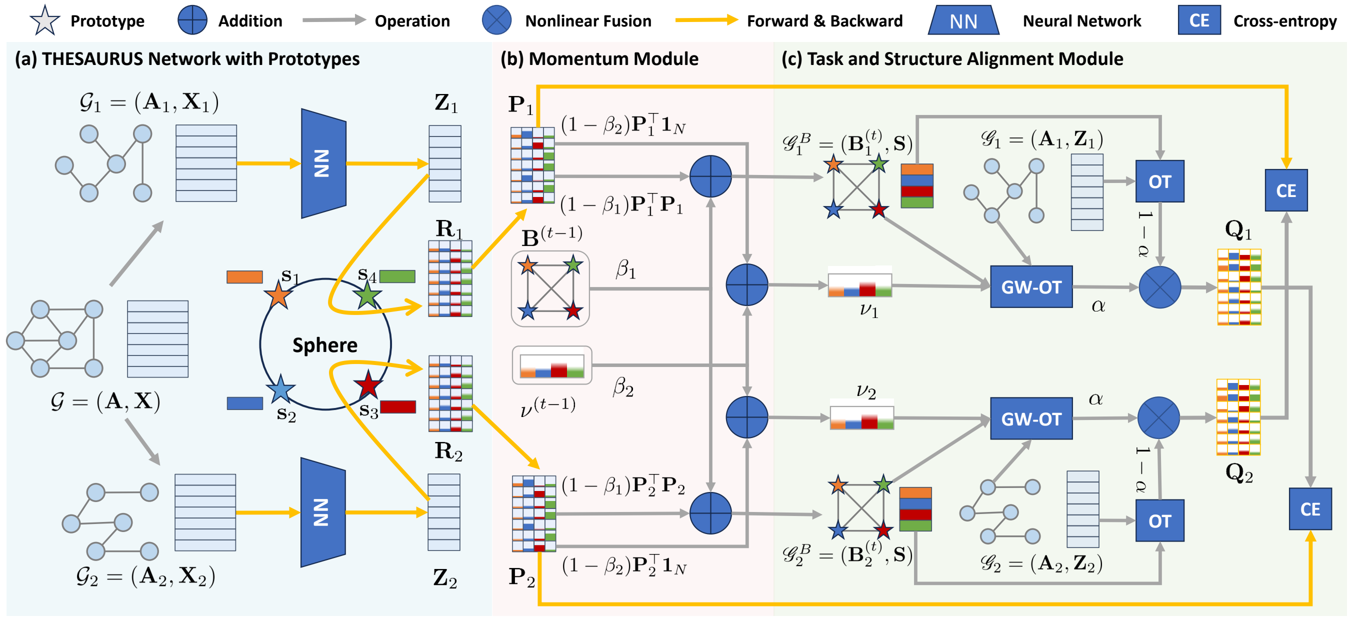

Prototype
Addition
Operation
Nonlinear Fusion
Forward & Backward
NN
Neural Network
CE
Cross-entropy
(a) THESAURUS Network with Prototypes
(b) Momentum Module
(c) Task and Structure Alignment Module
$\mathcal{G}_1 = (\mathbf{A}_1, \mathbf{X}_1)$
$\mathcal{G} = (\mathbf{A}, \mathbf{X})$
$\mathcal{G}_2 = (\mathbf{A}_2, \mathbf{X}_2)$
NN
NN
$\mathbf{Z}_1$
$\mathbf{R}_1$
$\mathbf{R}_2$
$\mathbf{Z}_2$
Sphere
$\mathbf{s}_1$
$\mathbf{s}_2$
$\mathbf{s}_3$
$\mathbf{s}_4$
$\mathbf{P}_1$
$\mathbf{P}_2$
$(1-\beta_2)\mathbf{P}_1^\top \mathbf{1}_N$
$(1-\beta_1)\mathbf{P}_1^\top \mathbf{P}_1$
$\mathbf{B}^{(t-1)}$
β_1
$\nu^{(t-1)}$
β_2
$(1-\beta_1)\mathbf{P}_2^\top \mathbf{P}_2$
$(1-\beta_2)\mathbf{P}_2^\top \mathbf{1}_N$
$\mathcal{G}_1^B = (\mathbf{B}_1^{(t)}, \mathbf{S})$
$\mathcal{G}_1 = (\mathbf{A}_1, \mathbf{Z}_1)$
$\mathcal{G}_2^B = (\mathbf{B}_2^{(t)}, \mathbf{S})$
$\mathcal{G}_2 = (\mathbf{A}_2, \mathbf{Z}_2)$
OT
OT
GW-OT
GW-OT
ν_1
ν_2
α
α
$1-\alpha$
$1-\alpha$
$\mathbf{Q}_1$
$\mathbf{Q}_2$
CE
CE

Figure 2: The illustration of our proposed THESAURUS. And the details are summarized in Algorithm 1 in the appendix.

negative connotation, while "so" is neutral. In the second sentence, "wonder" carries a positive connotation, and "She has a" is neutral. The word co-occurrence in these sentences indicates that "childish" is associated with negative and neutral semantics (prototypes), whereas "childlike" is linked with positive and neutral semantics (prototypes). This allows us to immediately infer that "childish" has a negative meaning related to children, while "childlike" has a positive connotation, thus distinguishing these two synonyms.

Similarly, in the embedding space, we define S semantic prototypes, each representing a specific semantic category. The positional relationships between nodes and these prototypes constitute their contextual semantics. Unlike existing methods that measure the association between two nodes solely by the distance between their embedding vectors, THESAURUS uses that between S-dimensional context-aware representations—derived from the distances between nodes and semantic prototypes—to represent their associations. With this approach, the input for K-means is not the encoder output, but the context-aware representations $\mathbf{R}$.

Task and Structure Alignment Module

To closely align with the downstream clustering task, we design the pretext task of predicting optimal clustering assignments cross views. To thoroughly exploit the cluster information in the graph structure, we align the prototype and data graph structures. These designs are integrated as the TSA module based on FGW-OT (Titouan et al. 2019).

Address L2: Pretext Task Aligned with Clustering THESAURUS treats semantic prototypes as clustering centroids providing clear semantic meanings and learns by predicting swapped clustering assignments across views.

The node-prototype assignment $\mathbf{Q}$ is derived from context-aware representations $\mathbf{R} = \mathbf{Z}\mathbf{S}^\top$, calculated via solving the problem

$$\min_{\pi\in\Pi} \mathrm{Tr}\left(-\mathbf{R}^\top \pi\right) - \epsilon H(\pi), \tag{1}$$

where $\pi \in \mathbb{R}_{\geq 0}^{N\times S}$ deduces $\mathbf{Q}$ with row-sum normalization, $H(\pi) = -\sum_{ij} \pi_{ij} \log \pi_{ij}$ denotes the entropy, and $\epsilon > 0$ controls the smoothness of the unnormalized assignment (i.e., the coupling π in OT). Here π is constrained by the node marginal distribution $\mu \in \mathbb{R}_{\geq 0}^{N}$ and the prototype marginal distribution $\nu \in \{\nu \in \mathbb{R}_{\geq 0}^{S} \mid \sum_i \nu_i = 1\}$

$$\Pi = \{\pi | \pi \mathbf{1}_S = \mu, \pi^\top \mathbf{1}_N = \nu\}, \tag{2}$$

where $\mathbf{1}_S$ is a S-dimensional all-one column vector. Problem (1) can be viewed as a relaxed and regularized K-means problem. K-means minimizes the sum of squared distances between data points and centroids, and we take the negative similarities $-\mathbf{R}$ as "distances" (and OT costs) here. Thanks to the entropy regularization, this problem can be efficiently solved by the scalable Sinkhorn (Cuturi 2013).

Feeding $\mathbf{R}_1$ of view 1 to the above procedure gives $\mathbf{Q}_1$, and similarly $\mathbf{Q}_2$ is got from view 2. The goal is to predict the assignment $\mathbf{Q}_2$ of view 2 from the representation $\mathbf{Z}_1$ of view 1, and $\mathbf{Q}_1$ from $\mathbf{Z}_2$. Since $\mathbf{Z}_1$ lacks interaction with prototypes $\mathbf{S}$, the prediction distribution $[\mathbf{P}_1^\tau]_n$ for node n is built on $\mathbf{R}_1 = \mathbf{Z}_1\mathbf{S}^\top$ instead of $\mathbf{Z}_1$

$$[\mathbf{P}_1^\tau]_{n,s} = \frac{\exp\left([\mathbf{Z}_1]_n [\mathbf{S}]_s^\top/\tau\right)}{\sum_{s'}\exp\left([\mathbf{Z}_1]_n [\mathbf{S}]_{s'}^\top/\tau\right)}, \tag{3}$$

where τ is the temperature that controls the distribution sharpness (Wu et al. 2018) and $\mathbf{P}_1^1$ is abbreviated to $\mathbf{P}_1$. The distributions of all nodes are stacked into $\mathbf{P}_1^\tau \in \mathbb{R}_{\geq 0}^{N\times S}$.

Similarly, $\mathbf{P}_2^\tau$ is obtained. The overall training loss is then computed as the averaged cross-entropy

$$\mathcal{L} = -\frac{1}{2N}\sum_{n=1}^{N}\sum_{s=1}^{S}\left([\mathbf{Q}_1]_{n,s}\log[\mathbf{P}_2^\tau]_{n,s} + [\mathbf{Q}_2]_{n,s}\log[\mathbf{P}_1^\tau]_{n,s}\right) \tag{4}$$

Address L3: Structure Alignment via GW-OT The above node-prototype clustering assignment is derived from the relationships between node embeddings $\mathbf{Z}$ and prototypes $\mathbf{S}$. Like prior methods, this approach does not explicitly extract structure cluster information. To fill this gap, we propose deriving the optimal assignment from the structure and using this assignment as a prediction target for $\mathbf{Z}$. One effective method to assign nodes to different semantic prototypes based on the structures is to perform GW-OT between $\mathbf{A}$ and an isolated graph $\mathbf{I}_S$ (Xu, Luo, and Carin 2019). In an isolated graph, each node represents a cluster with no inter-cluster edges. Such approach adheres to cluster definition but ignores inter-cluster relationships, which is unrealistic. Therefore, we replace the isolated graph with a complete prototype graph $\mathbf{B} \in \mathbb{R}_{\geq 0}^{S \times S}$, which is constructed as

$$\mathbf{B} = \mathbf{P}^\top \mathbf{P}. \tag{5}$$

GW-OT is formally defined in Def. 1. For an attribute graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$, each node v_i contains observable attribute $x_i = [\mathbf{X}]_{i,:} \in \Omega_x \subset \mathbb{R}^d$ and implicit structure embedding $s_i \in \Omega_s$. Although s_i is not known, the pairwise relationship $\mathbf{C} \in \mathbb{R}_{\geq 0}^{N \times N}$ determined by the metric $D_{\Omega_s} : \Omega_s \times \Omega_s \to \mathbb{R}_{\geq 0}$ on the space Ω_s is given by the adjacency matrix $\mathbf{A}$, the Laplacian $\mathbf{L}$, or the pairwise shortest path matrix (Chowdhury and Mémoli 2019). To use OT, the probability measure on the space $(\mathcal{V}, D_{\Omega_s})$ or equivalently $(\mathcal{V}, \mathbf{C})$ must be defined. Denote the importance of N nodes by the histogram

$$h \in \mathcal{H}_N = \left\{ h \mid h \in \mathbb{R}_{>0}^N, \sum_{i=1}^{N} h_i = 1 \right\}. \tag{6}$$

Then this space has a measure $\mu = \sum_i h_i \delta_{(s_i)}$, where $\delta_{(s_i)}$ denotes the Dirac delta function at s_i.

Definition 1. *Let* $(\mathcal{V}_1, \mathbf{C}_1, \mu)$ *and* $(\mathcal{V}_2, \mathbf{C}_2, \nu)$ *be Metric-Measure (MM) spaces defined on* $\mathcal{G}_1 = (\mathbf{C}_1, \emptyset)$ *and* $\mathcal{G}_2 = (\mathbf{C}_2, \emptyset)$*, respectively.* $\mu = \sum_i h_i^{(1)} \delta_{(s_i)}, h^{(1)} \in \mathcal{H}_{N_1}$ *and* $\nu = \sum_i h_i^{(2)} \delta_{(s_i)}, h^{(2)} \in \mathcal{H}_{N_2}$ *are the probability measures on* $\mathcal{V}_1$ *and* $\mathcal{V}_2$*, separately. The Gromov-Wasserstein distance* $GW_p(\mathcal{G}_1, \mathcal{G}_2)$ *between these two measures is given by*

$$\inf_{\pi \in \Pi(\mu,\nu)} \sum_{i,k=1}^{N_1} \sum_{j,l=1}^{N_2} \left([\mathbf{C}_1]_{i,k} - [\mathbf{C}_2]_{j,l}\right)^p \pi_{i,j}\pi_{k,l}, \tag{7}$$

where $\Pi = \left\{ \pi \in \mathbb{R}_{\geq 0}^{N_1 \times N_2} \mid \pi \mathbf{1}_{N_2} = h^{(1)}, \pi^\top \mathbf{1}_{N_1} = h^{(2)} \right\}$.

We utilize the GW_1 optimal transport between the non-attribute data graph $(\mathbf{A}, \emptyset)$ and prototype graph $(\mathbf{B}, \emptyset)$ to get the optimal node-prototype assignment $\mathbf{Q}$. It is row-normalized from the solution π of following OT problem

$$\min_{\pi \in \Pi(\mu,\nu)} \sum_{i,k=1}^{S} \sum_{j,l=1}^{N} \left| [\mathbf{A}]_{i,k} - [\mathbf{B}]_{j,l} \right| \pi_{i,j}\pi_{k,l}, \tag{8}$$

where μ is the uniform node marginal distribution and ν is the prototype marginal. After the structure-induced assignments $\mathbf{Q}_1, \mathbf{Q}_2$ of two views $\mathcal{G}_1, \mathcal{G}_2$ are separately got via Eq. (8), they can be used in the loss Eq. (4).

Fused Clustering Assignment via FGW-OT The above introduce two kinds of "clustering" assignments respectively acquired from the context-aware node representation $\mathbf{R}$ and the graph structure $\mathbf{A}$. The assignment from $\mathbf{R}$ focuses on attribute information, while that from $\mathbf{A}$ emphasizes structural information. We fuse them with FGW-OT for more comprehensive graph mining. We build $\mathscr{G} = (\mathbf{A}, \mathbf{Z})$ with the embeddings $\mathbf{Z}$ as node attributes. And we add prototypes $\mathbf{S} \in \mathbb{R}^{S \times d}$ as attributes to the prototype graph $\mathbf{B}$, resulting in $\mathscr{G}^B = (\mathbf{B}, \mathbf{S})$. The optimal transport between $\mathscr{G}^B$ and $\mathscr{G}$ encapsulate both attribute and structure cluster information, and can be achieved via FGW-OT defined below.

Definition 2. *Let* $(\mathcal{V}_1, \mathbf{C}_1, \mu)$ *and* $(\mathcal{V}_2, \mathbf{C}_2, \nu)$ *be MM-spaces on* $\mathcal{G}_1 = (\mathbf{C}_1, \mathbf{X}_1)$ *with measure* $\mu = \sum_i h_i^{(1)} \delta_{(s_i, x_i)}, h^{(1)} \in \mathcal{H}_{N_1}$ *and on* $\mathcal{G}_2 = (\mathbf{C}_2, \mathbf{X}_2)$ *with measure* $\nu = \sum_i h_i^{(2)} \delta_{(s_i, x_i)}, h^{(2)} \in \mathcal{H}_{N_2}$*, respectively. The Fused Gromov-Wasserstein distance* $FGW_{p,\alpha}(\mathcal{G}_1, \mathcal{G}_2)$ *is*

$$\inf_{\pi \in \Pi(\mu,\nu)} \left\{ \sum_{i,k=1}^{N_1} \sum_{j,l=1}^{N_2} \Big[(1-\alpha) D_{\Omega_x}(x_{1,i}, x_{2,j}) \right.$$

$$\left. + \alpha |\mathbf{C}_1(i,k) - \mathbf{C}_2(j,l)|^p \Big] \pi_{i,j}\pi_{k,l} \right\}^{\frac{1}{p}} - \epsilon H(\pi) \tag{9}$$

where $\Pi = \left\{ \pi \in \mathbb{R}_{\geq 0}^{N_1 \times N_2} \mid \pi \mathbf{1}_{N_2} = h^{(1)}, \pi^\top \mathbf{1}_{N_1} = h^{(2)} \right\}$; $H(\pi) = -\sum_{ij} \pi_{ij} \log \pi_{ij}$ *is the coupling entropy and* ϵ *weights this regularization;* $x_{1,i}, x_{2,j} \in \Omega_x \subset \mathbb{R}^d$ *are the attributes of nodes* $v_i^{(1)} \in \mathcal{V}_1$ *and* $v_j^{(2)} \in \mathcal{V}_2$*, respectively.*

The optimal coupling of $FGW_{1,\alpha}(\mathscr{G}, \mathscr{G}^B)$ encapsulates the information in $\mathbf{R}$ and $\mathbf{A}$. In view 1, the assignment $\mathbf{Q}_1$ of $FGW_{1,\alpha}(\mathscr{G}_1, \mathscr{G}_1^B)$ is obtained via normalizing the optimal coupling π_1 at each row $n \in \{1, 2, \ldots, N\}$

$$[\mathbf{Q}_1]_{n,s} = \frac{[\pi_1]_{n,s}}{\sum_{s'=1}^{S} [\pi_1]_{n,s'}}, \tag{10}$$

where $\mathscr{G}_1 = (\mathbf{A}_1, \mathbf{Z}_1)$ and $\mathscr{G}_1^B = (\mathbf{B}_1, \mathbf{S})$. Similarly, $\mathbf{Q}_2$ is got from view 2. Finally, these fused assignments are used to guide the training via Eq. (4).

Prototype Momentum Module

Prototype graph and marginal should adapt to the data. So momentum update is adopted here. Let $\mathbf{B}^{(t-1)}$ be the prototype graph of last forward step $t-1$. The step t has

$$\mathbf{B}^{(t)} = \beta_1 \mathbf{B}^{(t-1)} + (1-\beta_1)\mathbf{P}^\top \mathbf{P}. \tag{11}$$

To reflect the common graph homophily, $\mathbf{B}$ is initialized as an identity matrix and $\beta_1 \in [0, 1]$ is set to a high value.

Meanwhile, the logits $\mathbf{P}$, e.g, Eq. (3), are used to update prototype marginal, which is initialized as an uniform distribution $\nu^{(0)} = \mathbf{1}_S / S$ to reflect the presumed even cluster size before data exposure.

$$\nu^{(t)} = \beta_2 \nu^{(t-1)} + (1-\beta_2)(\mathbf{P}^\top \mathbf{1}_N) \tag{12}$$

4 Experiments

Experimental Setup

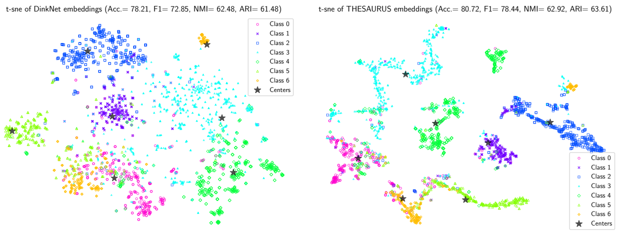
To evaluate the performance of THESAURUS, we run the proposed method on nine attribute graph datasets, including Cora, Citeseer, Pubmed, Amazon-Photo (A-Photo), Cora-Full, ACM, DBLP, UAT, and Wiki. The baselines are K-means, DEC, GRACE (Zhu et al. 2020), SDCN, DFCN, DCRN, S^{3}GC, SCGC, HSAN, and Dink-Net.

Our evaluation protocol follows that of the previous SOTA Dink-Net (Liu et al. 2023a). Besides Normalized Mutual Information (NMI) and Adjusted Rand Index (ARI), the metrics include Accuracy (ACC) and the Macro-F1 score (F1), computed after mapping the clusters to the ground-truth classes with linear assignment (Lovasz 1986; Crouse 2016). The F1 score, defined as the harmonic mean of precision and recall, balances the effects of false positives and false negatives. Meanwhile, ARI quantifies the number of true positive and true negative pairs and normalizes these values to ensure that the assessment is not influenced by variations in cluster sizes. Therefore, F1 and ARI are more effective than ACC and NMI on imbalanced data.

Overall Performance

Part of the results are summarized in Table 1, with OOM indicating out-of-memory failures on one RTX 4090 GPU. And the rest results are presented in Tables 4 and 5 in the appendix. These results demonstrate that the proposed THESAURUS significantly outperforms existing methods across all datasets. And several key observations can be made.

The contextual information from semantic prototypes is important. Existing methods do not achieve sufficient cluster separability and are affected by the Uniform Effect and Cluster Assimilation. This results in suboptimal performance on clusters of varying sizes, particularly minority clusters, adversely impacting F1 and ARI. In contrast, THESAURUS, utilizing semantic prototype contexts, achieves better distinction between synonymous nodes, leading to higher cluster separability. This effectively mitigates Uniform Effect and Cluster Assimilation, evidenced by THESAURUS's F1 and ARI significantly surpassing existing methods. **Pretext task aligned with clustering is vital.** Contrastive clustering methods like S^{3}GC and HSAN outperform DEC-style methods such as SDCN and DFCN, likely due to the implicit but insufficient alignment between InfoNCE (**?**) and (spectral) clustering (HaoChen et al. 2021; Tan et al. 2023). DinkNet, which explicitly optimize towards clustering during finetune, is better aligned with clustering tasks, thus surpassing other baselines. Furthermore, THESAURUS representations are learned towards high cluster separability from start to finish, and thus far outperform all other methods. **The cluster information in structure matters.** Methods utilizing no graph structures, such as K-means and DEC, are unsuitable for graph clustering. Methods like HSAN and SCGC, which inject structure information with supervision signals, generally surpass those that only leverage structure through graph filters, such as SDCN and DAEGC. Furthermore, THESAURUS, which exhaustively exploits structural cluster information, surpasses all other methods.



(a) Dink-Net on Cora (b) THESAURUS on Cora

Figure 3: The visualization of Dink-Net and THESAURUS

Class-wise Performance & Visualization

While Dink-Net's finetune step mitigates the Uniform Effect in the majority cluster (cluster 3), it fails to address Cluster Assimilation in the minority cluster (cluster 6), as indicated by the experiments presented in Section Introduction. This section evaluates the class-wise performance of Dink-Net and THESAURUS, demonstrating that THESAURUS achieves high cluster separability and addresses the failure cases of Dink-Net. Fig. 3 visualizes the final representations to cluster and the centroids on Cora using t-SNE (van der Maaten and Hinton 2008). Fig. 4 displays the class-wise performance on Pubmed, in a style like Fig. 1a.

Overall, the predicted label distribution of THESAURUS exhibits a lower deviation from the ground-truth distribution than that of DinkNet, as shown in Fig. 1a and Fig. 4. This suggests fewer mis-clustered nodes and corresponds with the higher cluster separability observed in THESAURUS's representation space in Fig. 3, resulting in a Macro-F1 score up to 5.5 percentage points higher than Dink-Net on Cora and 11.65 points higher on Pubmed.

Uniform Effect Fig. 3a (right part) shows that Dink-Net does not separate clusters 3 and 4 as well as THESAURUS in Fig. 3b (top-left). So although Dink-Net reduces the Uniform Effect in cluster 3, the F1 score for cluster 4 is significantly lower than THESAURUS, as shown by Fig. 1a. Besides, Dink-Net's slightly higher F1 score for cluster 3 comes at the cost of many false positives, which severely compromises the performance of other clusters (e.g., clusters 4 and 0). In contrast, THESAURUS doesn't favor the majority cluster and performs well overall.

Cluster Assimilation Fig. 3a (bottom-left) shows Dink-Net mixing clusters 0, 5, and 6, leading to a low F1 score of about 34% for cluster 6 due to samples being merged into other clusters. Fig. 3b shows that THESAURUS effectively separates these clusters, significantly mitigating Cluster Assimilation, as evidenced by a 27.29 percentage point higher F1 score for cluster 6 compared to Dink-Net (see Fig. 1a).

Ablation Study

To validate the effectiveness of the complete prototype graph, we set the prototype graph $\mathbf{B}$ as isolated graph $\mathbf{I}_S$

Datasets	Metrics	K-means	DEC	SDCN	GRACE	DAEGC	DFCN	DCRN	HSAN	S^{3}GC	SCGC	Dink-Net*	Dink-Net	Ours
Cora	ACC	33.80	46.50	35.60	73.90	70.43	36.33	61.93	77.21	74.21	73.88	75.55	78.21	**80.72**
	NMI	14.98	23.54	14.28	57.10	52.89	19.36	45.13	59.56	58.80	56.10	60.03	62.48	**62.92**
	ARI	8.60	15.13	7.78	52.70	49.63	4.67	33.15	57.93	54.43	51.79	54.56	61.48	**63.61**
	F1	30.26	39.23	24.37	72.50	68.27	26.16	49.50	75.13	72.10	70.81	71.31	72.85	**78.44**
Citeseer	ACC	39.32	46.51	65.96	63.10	64.54	69.50	69.86	71.05	68.81	71.02	69.34	69.91	**71.99**
	NMI	16.94	23.54	38.71	39.91	36.41	43.90	44.86	45.62	44.11	45.25	44.36	45.29	**47.37**
	ARI	13.43	15.13	40.17	37.70	37.78	45.50	45.64	48.22	44.80	46.29	45.65	46.29	**48.99**
	F1	36.08	39.23	63.62	60.30	62.24	64.30	64.83	64.52	64.30	64.80	65.54	65.79	**66.42**
Pubmed	ACC	59.83	60.14	64.20	63.72	68.73	68.89	69.87	OOM	71.31	45.12	67.32	67.51	**79.64**
	NMI	31.05	22.14	22.87	30.86	28.26	31.43	32.20		33.35	7.04	32.49	33.01	**41.43**
	ARI	28.10	19.55	22.30	27.61	29.84	30.64	31.41		34.52	7.04	29.95	30.44	**48.25**
	F1	58.88	61.45	65.01	62.85	68.23	68.10	68.94		70.33	44.54	67.12	67.35	**79.00**
A-Photo	ACC	27.22	47.22	53.44	67.66	75.96	76.82	79.94	77.02	75.15	77.48	77.19	80.71	**84.42**
	NMI	13.23	37.35	44.85	53.46	65.25	66.23	73.70	67.54	59.78	67.67	68.94	70.50	**74.99**
	ARI	5.50	18.59	31.21	42.74	58.12	58.28	63.69	58.05	56.13	58.48	60.20	66.54	**72.01**
	F1	23.96	46.71	50.66	60.32	69.87	71.25	73.82	72.60	72.85	72.22	71.23	73.09	**76.49**
CoraFull	ACC	26.27	31.92	26.67	32.38	34.35	37.51	38.80	OOM	36.46	41.89	38.51	39.45	**42.94**
	NMI	34.68	41.31	37.83	50.42	49.16	51.30	51.91		52.82	53.21	53.39	54.41	**55.83**
	ARI	9.35	16.89	22.60	20.64	22.60	24.46	25.25		24.78	24.23	26.53	27.45	**30.07**
	F1	22.57	27.77	22.14	27.82	26.96	31.22	31.68		29.78	32.98	30.90	31.95	**37.01**

Table 1: Clustering performance (%). The best result is in bold. Dink-Net* denotes Dink-Net-NoFT.

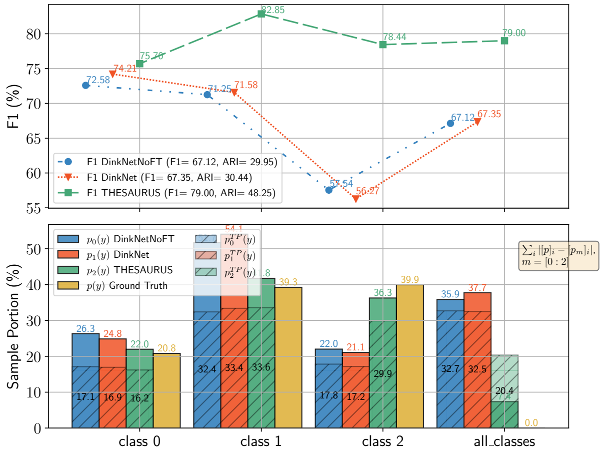


Figure 4: Dink-Net and THESAURUS on Pubmed. The **top** figure illustrates the F1 scores for each category, as well as the Macro-F1. The **bottom** shows the distribution of predicted labels by Dink-Net and THESAURUS, along with the ground-truth labels. It also presents the distribution of predicted labels for true-positive (TP) samples, denoted as $p_i^{TP}(y), i \in \{0, 1, 2\}$. The final set of bars shows the differences between the predicted and ground-truth distributions.

(w/o $\mathbf{B}$). To test the impact of structural information extraction, we replace $\mathbf{A}_1$ and $\mathbf{A}_2$ in FGW-OT with $\mathbf{I}_N$ (w/o $\mathbf{A}$). Additionally, to assess the effectiveness of the proposed momentum module, we set the momentum to 1 (fixed ν & $\mathbf{B}$). The results in Table 2 indicate that the complete prototype graph outperforms the isolated one, structural information extraction is effective, and momentum updates for the prototype graph and marginal is useful.

Datasets	Metrics	w/o $\mathbf{B}$	w/o $\mathbf{A}$	fixed ν & $\mathbf{B}$	Ours
Citeseer	ACC	70.27	71.54	71.24	**71.99**
	NMI	46.41	46.74	46.37	**47.37**
	ARI	47.82	47.92	47.57	**48.99**
	F1	65.02	65.75	65.68	**66.42**
Pubmed	ACC	75.73	78.35	75.84	**79.64**
	NMI	35.44	39.19	35.50	**41.43**
	ARI	39.25	45.47	40.62	**48.25**
	F1	75.27	77.81	75.39	**79.00**

Table 2: THESAURUS ablation. The best is in bold.

5 Conclusion

This work identifies that prior deep graph clustering methods face challenges from the Uniform Effect and Cluster Assimilation due to the low cluster separability in the learned embedding space. To achieve a highly separable representation space, we propose a novel contrastive graph learning framework, THESAURUS. Our method utilizes semantic prototypes to provide contextual information crucial for distinguishing similar nodes from different classes, leverages a pretext task well-aligned with the downstream clustering task for better feature transferability, employs GW-OT to exploit the cluster information in the graph structure thoroughly, and uses a momentum module for data adaptability. Experimental results demonstrate the effectiveness and superiority of THESAURUS over prior methods.

References

Alvarez-Melis, D.; and Jaakkola, T. 2018. Gromov-Wasserstein Alignment of Word Embedding Spaces. In Riloff, E.; Chiang, D.; Hockenmaier, J.; and Tsujii, J., eds., *Proceedings of the 2018 Conference on Empirical Methods* *in Natural Language Processing*, 1881–1890. Brussels, Belgium: Association for Computational Linguistics.

Bo, D.; Wang, X.; Shi, C.; Zhu, M.; Lu, E.; and Cui, P. 2020. Structural Deep Clustering Network. In *Proceedings of the Web Conference 2020*, Www '20, 1400–1410. New York, NY, USA: Association for Computing Machinery. ISBN 978-1-4503-7023-3.

Bojchevski, A.; and Günnemann, S. 2018. Deep Gaussian Embedding of Graphs: Unsupervised Inductive Learning via Ranking. In *International Conference on Learning Representations*.

Cao, S.; Lu, W.; and Xu, Q. 2016. Deep Neural Networks for Learning Graph Representations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Caron, M.; Misra, I.; Mairal, J.; Goyal, P.; Bojanowski, P.; and Joulin, A. 2020. Unsupervised Learning of Visual Features by Contrasting Cluster Assignments. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, 9912–9924. Red Hook, NY, USA: Curran Associates Inc. ISBN 978-1-71382-954-6.

Chen, D.; Lin, Y.; Li, W.; Li, P.; Zhou, J.; and Sun, X. 2020. Measuring and Relieving the Over-Smoothing Problem for Graph Neural Networks from the Topological View. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 3438–3445.

Chowdhury, S.; and Mémoli, F. 2019. The Gromov–Wasserstein Distance between Networks and Stable Network Invariants. *Information and Inference: A Journal of the IMA*, 8(4): 757–787.

Courty, N.; Flamary, R.; Habrard, A.; and Rakotomamonjy, A. 2017. Joint Distribution Optimal Transportation for Domain Adaptation. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, 3733–3742. Red Hook, NY, USA: Curran Associates Inc. ISBN 978-1-5108-6096-4.

Crouse, D. F. 2016. On Implementing 2D Rectangular Assignment Algorithms. *IEEE Transactions on Aerospace and Electronic Systems*, 52(4): 1679–1696.

Cui, G.; Zhou, J.; Yang, C.; and Liu, Z. 2020. Adaptive Graph Encoder for Attributed Graph Embedding. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, Kdd '20, 976–985. New York, NY, USA: Association for Computing Machinery. ISBN 978-1-4503-7998-4.

Cuturi, M. 2013. Sinkhorn Distances: Lightspeed Computation of Optimal Transport. *Advances in neural information processing systems*, 26.

Devvrit, F.; Sinha, A.; Dhillon, I.; and Jain, P. 2022. S3GC: Scalable Self-Supervised Graph Clustering. In *Advances in Neural Information Processing Systems*, volume 35, 3248–3261.

Douze, M.; Guzhva, A.; Deng, C.; Johnson, J.; Szilvasy, G.; Mazaré, P.-E.; Lomeli, M.; Hosseini, L.; and Jégou, H. 2024. The Faiss Library.

Fey, M.; and Lenssen, J. 2019. Fast Graph Representation Learning with PyTorch Geometric. In *International Conference on Learning Representations*.

Frogner, C.; Zhang, C.; Mobahi, H.; Araya, M.; and Poggio, T. A. 2015. Learning with a Wasserstein Loss. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Gan, G.; Ma, C.; and Wu, J. 2020. *Data Clustering: Theory, Algorithms, and Applications*. SIAM.

Gong, F.; Nie, Y.; and Xu, H. 2022. Gromov-Wasserstein Multi-modal Alignment and Clustering. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, CIKM '22, 603–613. New York, NY, USA: Association for Computing Machinery. ISBN 978-1-4503-9236-5.

HaoChen, J. Z.; Wei, C.; Gaidon, A.; and Ma, T. 2021. Provable Guarantees for Self-Supervised Deep Learning with Spectral Contrastive Loss. In Ranzato, M.; Beygelzimer, A.; Dauphin, Y.; Liang, P.; and Vaughan, J. W., eds., *Advances in Neural Information Processing Systems*, volume 34, 5000–5011. Curran Associates, Inc.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In *2015 IEEE International Conference on Computer Vision (ICCV)*, 1026–1034.

Hunter, J. D. 2007. Matplotlib: A 2D Graphics Environment. *Computing in Science & Engineering*, 9(3): 90–95.

Johnson, J.; Douze, M.; and Jégou, H. 2019. Billion-Scale Similarity Search with GPUs. *IEEE Transactions on Big Data*, 7(3): 535–547.

Kingma, D. P.; and Ba, J. 2017. Adam: A Method for Stochastic Optimization. arXiv:1412.6980.

Kipf, T. N.; and Welling, M. 2016. Variational Graph Auto-Encoders. In *NIPS Workshop on Bayesian Deep Learning*.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations*.

Lee, J. D.; Lei, Q.; Saunshi, N.; and ZHUO, JIACHENG. 2021. Predicting What You Already Know Helps: Provable Self-Supervised Learning. In *Advances in Neural Information Processing Systems*, volume 34, 309–323. Curran Associates, Inc.

Li, M.; Wu, J.; Wang, X.; Chen, C.; Qin, J.; Xiao, X.; Wang, R.; Zheng, M.; and Pan, X. 2023. AlignDet: Aligning Pre-training and Fine-tuning in Object Detection. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, 6843–6853. Paris, France: IEEE. ISBN 9798350307184.

Liu, Y.; Jin, M.; Pan, S.; Zhou, C.; Zheng, Y.; Xia, F.; and Yu, P. 2022a. Graph Self-Supervised Learning: A Survey. *IEEE Transactions on Knowledge and Data Engineering*, 1–1.

Liu, Y.; Liang, K.; Xia, J.; Zhou, S.; Yang, X.; Liu, X.; and Li, S. Z. 2023a. Dink-Net: Neural Clustering on Large Graphs. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *ICML'23*, 21794–21812. JMLR.org.

Liu, Y.; Tu, W.; Zhou, S.; Liu, X.; Song, L.; Yang, X.; and Zhu, E. 2022b. Deep Graph Clustering via Dual Correlation Reduction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 7603–7611.

Liu, Y.; Xia, J.; Zhou, S.; Yang, X.; Liang, K.; Fan, C.; Zhuang, Y.; Li, S. Z.; Liu, X.; and He, K. 2023b. A Survey of Deep Graph Clustering: Taxonomy, Challenge, Application, and Open Resource. arXiv:2211.12875.

Liu, Y.; Yang, X.; Zhou, S.; Liu, X.; Wang, S.; Liang, K.; Tu, W.; and Li, L. 2023c. Simple Contrastive Graph Clustering. *IEEE Transactions on Neural Networks and Learning Systems*, 1–12.

Liu, Y.; Yang, X.; Zhou, S.; Liu, X.; Wang, Z.; Liang, K.; Tu, W.; Li, L.; Duan, J.; and Chen, C. 2023d. Hard Sample Aware Network for Contrastive Deep Graph Clustering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 8914–8922.

Lovasz, L. 1986. *Matching Theory (North-Holland Mathematics Studies)*. GBR: Elsevier Science Ltd. ISBN 0-444-87916-1.

Lu, Y.; Cheung, Y.-M.; and Tang, Y. Y. 2021. Self-Adaptive Multiprototype-Based Competitive Learning Approach: A k-Means-Type Algorithm for Imbalanced Data Clustering. *IEEE Transactions on Cybernetics*, 51(3): 1598–1612.

Mémoli, F. 2011. Gromov–Wasserstein Distances and the Metric Approach to Object Matching. *Foundations of Computational Mathematics*, 11(4): 417–487.

Monge, G. 1781. Mémoire Sur La Théorie Des Déblais et Des Remblais. *Mem. Math. Phys. Acad. Royale Sci.*, 666–704.

Mrabah, N.; Bouguessa, M.; Touati, M. F.; and Ksantini, R. 2023. Rethinking Graph Auto-Encoder Models for Attributed Graph Clustering. *IEEE Transactions on Knowledge and Data Engineering*, 35(9): 9037–9053.

Nguyen, K.; Nong, H.; Nguyen, V.; Ho, N.; Osher, S.; and Nguyen, T. 2023. Revisiting Over-Smoothing and over-Squashing Using Ollivier-Ricci Curvature. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *ICML'23*, 25956–25979. Honolulu, Hawaii, USA: JMLR.org.

NT, H.; and Maehara, T. 2019. Revisiting Graph Neural Networks: All We Have Is Low-Pass Filters. arXiv:1905.09550.

Oono, K.; and Suzuki, T. 2020. Graph Neural Networks Exponentially Lose Expressive Power for Node Classification. In *International Conference on Learning Representations*.

Pan, S.; Hu, R.; Long, G.; Jiang, J.; Yao, L.; and Zhang, C. 2018. Adversarially Regularized Graph Autoencoder for Graph Embedding. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, IJCAI'18, 2609–2615. Stockholm, Sweden: AAAI Press. ISBN 978-0-9992411-2-7.

Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; and Lerer, A. 2017. Automatic Differentiation in PyTorch.

Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; and Duchesnay, E. 2011. Scikit-Learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12: 2825–2830.

Peng, Z.; Liu, H.; Jia, Y.; and Hou, J. 2021. Attention-Driven Graph Clustering Network. In *Proceedings of the 29th ACM International Conference on Multimedia*, Mm '21, 935–943. New York, NY, USA: Association for Computing Machinery. ISBN 978-1-4503-8651-7.

Peyré, G.; and Cuturi, M. 2019. Computational Optimal Transport: With Applications to Data Science. *Foundations and Trends® in Machine Learning*, 11(5-6): 355–607.

Peyré, G.; Cuturi, M.; and Solomon, J. 2016. Gromov-Wasserstein Averaging of Kernel and Distance Matrices. In *Proceedings of The 33rd International Conference on Machine Learning*, 2664–2672. PMLR.

Sen, P.; Namata, G.; Bilgic, M.; Getoor, L.; Galligher, B.; and Eliassi-Rad, T. 2008. Collective Classification in Network Data. *AI Magazine*, 29(3): 93–93.

Shchur, O.; Mumme, M.; Bojchevski, A.; and Günnemann, S. 2019. Pitfalls of Graph Neural Network Evaluation. arXiv:1811.05868.

Shrivastava, A.; Selvaraju, R. R.; Naik, N.; and Ordonez, V. 2023. CLIP-Lite: Information Efficient Visual Representation Learning with Language Supervision. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, 8433–8447. PMLR.

Shuman, D. I.; Narang, S. K.; Frossard, P.; Ortega, A.; and Vandergheynst, P. 2013. The Emerging Field of Signal Processing on Graphs: Extending High-Dimensional Data Analysis to Networks and Other Irregular Domains. *IEEE Signal Processing Magazine*, 30(3): 83–98.

Tan, Z.; Zhang, Y.; Yang, J.; and Yuan, Y. 2023. Contrastive Learning Is Spectral Clustering on Similarity Graph. In *The Twelfth International Conference on Learning Representations*.

Titouan, V.; Courty, N.; Tavenard, R.; and Flamary, R. 2019. Optimal Transport for Structured Data with Application on Graphs. In *International Conference on Machine Learning*, 6275–6284. PMLR.

Tolstikhin, I.; Bousquet, O.; Gelly, S.; and Schoelkopf, B. 2018. Wasserstein Auto-Encoders. In *International Conference on Learning Representations*.

Tu, W.; Zhou, S.; Liu, X.; Guo, X.; Cai, Z.; Zhu, E.; and Cheng, J. 2021. Deep Fusion Clustering Network. In *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence*, 9978–9987.

van den Oord, A.; Li, Y.; and Vinyals, O. 2019. Representation Learning with Contrastive Predictive Coding. arXiv:1807.03748.

van der Maaten, L.; and Hinton, G. 2008. Visualizing Data Using T-SNE. *Journal of Machine Learning Research*, 9(86): 2579–2605.

Villani, C.; et al. 2009. *Optimal Transport: Old and New*. Springer.

Wang, C.; Pan, S.; Hu, R.; Long, G.; Jiang, J.; and Zhang, C. 2019. Attributed Graph Clustering: A Deep Attentional Embedding Approach. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, IJCAI'19, 3670–3676. AAAI Press. ISBN 978-0-9992411-4-1.

Wang, C.; Pan, S.; Long, G.; Zhu, X.; and Jiang, J. 2017. Mgae: Marginalized Graph Autoencoder for Graph Clustering. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 889–898.

Wang, S.; Yang, J.; Yao, J.; Bai, Y.; and Zhu, W. 2024. An Overview of Advanced Deep Graph Node Clustering. *IEEE Transactions on Computational Social Systems*, 11(1): 1302–1314.

Wei, F.; Gao, Y.; Wu, Z.; Hu, H.; and Lin, S. 2021. Aligning Pretraining for Detection via Object-Level Contrastive Learning. In *Advances in Neural Information Processing Systems*.

Wu, Z.; Xiong, Y.; Yu, S. X.; and Lin, D. 2018. Unsupervised Feature Learning via Non-parametric Instance Discrimination. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3733–3742. Salt Lake City, UT: IEEE. ISBN 978-1-5386-6420-9.

Xie, J.; Girshick, R.; and Farhadi, A. 2016. Unsupervised Deep Embedding for Clustering Analysis. In Balcan, M. F.; and Weinberger, K. Q., eds., *Proceedings of the 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, 478–487. New York, New York, USA: PMLR.

Xiong, H.; Wu, J.; and Chen, J. 2009. K-Means Clustering Versus Validation Measures: A Data-Distribution Perspective. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(2): 318–331.

Xu, H.; Luo, D.; and Carin, L. 2019. Scalable Gromov-Wasserstein Learning for Graph Partitioning and Matching. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Zhu, Y.; Xu, Y.; Yu, F.; Liu, Q.; Wu, S.; and Wang, L. 2020. Deep Graph Contrastive Representation Learning. arXiv:2006.04131.

A Appendix A: Algorithm of THESAURUS

The pseudo-code for THESAURUS is given in Algorithm 1. Please also refer to the schematic Fig. 2 in the main content for a quick and comprehensive understanding of the details of THESAURUS.

Algorithm 1: Pseudocode of THESAURUS

Input: The graph $\mathcal{G} = (\mathbf{A}, \mathbf{X})$ with E edges and d_0 attribute dimensions; the number of clusters C; training epochs T; momentum weights β_1 and β_2; the edge drop rate pe and feature drop rate px of data augmentation module
Parameter: the trade-off weight α of FGW-OT; Softmax temperature τ; the number of prototypes S
Output: Clustering assignment Φ

```
Initialize current epoch $t = 1$, neural network $f_\omega \circ f_\theta$, prototypes $\mathbf{S}$, prototype graph $\mathbf{B}^{(0)} = \mathbf{I}_S$, and prototype marginal $\nu^{(0)} = \mathbf{1}_S / S$
while $t < T$ do
    //* Generate two augmented views
    Randomly drop $pe \cdot E$ edges and $px \cdot d_0$ feature dimensions to get $\mathcal{G}_1 = (\mathbf{A}_1, \mathbf{X}_1)$ and $\mathcal{G}_2 = (\mathbf{A}_2, \mathbf{X}_2)$
    //* Encode two views with $f_\omega \circ f_\theta$
    Encode $\mathcal{G}_1$ and $\mathcal{G}_2$ with $f_\omega \circ f_\theta$ into $\mathbf{Z}_1$ and $\mathbf{Z}_2$
    //* Process view 1
    $\mathbf{R}_1 = \mathbf{Z}_1 \mathbf{S}^\top$, compute $\mathbf{P}_1$ and $\mathbf{P}_1^\tau$ with Eq. (3)
    $\mathbf{B}_1^{(t)} = \beta_1 \mathbf{B}^{(t-1)} + (1 - \beta_1) {\mathbf{R}_1}^\top \mathbf{R}_1$
    $\nu_1^{(t)} = \beta_2 \nu^{(t-1)} + (1 - \beta_2)(\mathbf{P}_1^\top \mathbf{1}_N)$
    $\mathbf{B}^{(t-1)} \leftarrow \mathbf{B}_1^{(t)}$; $\nu^{(t-1)} \leftarrow \nu_1^{(t)}$
    //* Process view 2
    $\mathbf{R}_2 = \mathbf{Z}_2 \mathbf{S}^\top$, compute $\mathbf{P}_2$ and $\mathbf{P}_2^\tau$ like Eq. (3)
    $\mathbf{B}_2^{(t)} = \beta_1 \mathbf{B}^{(t-1)} + (1 - \beta_1) {\mathbf{R}_2}^\top \mathbf{R}_2$
    $\nu_2^{(t)} = \beta_2 \nu^{(t-1)} + (1 - \beta_2)(\mathbf{P}_2^\top \mathbf{1}_N)$
    $\mathbf{B}^{(t)} \leftarrow \mathbf{B}_2^{(t)}$; $\nu^{(t)} \leftarrow \nu_2^{(t)}$
    //* Prepare attribute graphs for FGW-OT
    $\mathbf{B}_1 \leftarrow \mathbf{B}_1^{(t)}$; $\mathbf{B}_2 \leftarrow \mathbf{B}_2^{(t)}$; $\nu_1 \leftarrow \nu_1^{(t)}$; $\nu_2 \leftarrow \nu_2^{(t)}$
    Construct graphs $\mathscr{G}_1 = (\mathbf{A}_1, \mathbf{Z}_1)$ and $\mathscr{G}_1^B = (\mathbf{B}_1, \mathbf{S})$
    Construct graphs $\mathscr{G}_2 = (\mathbf{A}_2, \mathbf{Z}_2)$ and $\mathscr{G}_2^B = (\mathbf{B}_2, \mathbf{S})$
    Set $\nu_1$ as the measure on $\mathscr{G}_1^B$ and $\nu_2$ on $\mathscr{G}_2^B$
    Set uniform measures $\mu_1$ for $\mathscr{G}_1$ and $\mu_2$ for $\mathscr{G}_2$
    //* Compute the optimal couplings of FGW-OT
    Get the optimal coupling $\pi_1$ of $FGW_{1,\alpha}(\mathscr{G}_1, \mathscr{G}_1^B)$
    Get the optimal coupling $\pi_2$ of $FGW_{1,\alpha}(\mathscr{G}_2, \mathscr{G}_2^B)$
    //* Compute the cross-view loss and backward
    Get $\mathbf{Q}_1, \mathbf{Q}_2$ with Eq. (10)
    Compute the loss Eq. (4) with $\mathbf{Q}_1, \mathbf{Q}_2, \mathbf{P}_1^\tau, \mathbf{Q}_2^\tau$
    Backward and update $f_\omega \circ f_\theta$ and $\mathbf{S}$
    $t \leftarrow t + 1$
end while
//* Inference with original data, and final clustering
Encode $\mathcal{G} = (\mathbf{A}, \mathbf{X})$ into $\mathbf{Z}$ with the trained $f_\omega \circ f_\theta$
Feed $\mathbf{R} = \mathbf{Z}\mathbf{S}^\top$ into K-means to get $\Phi$ over $C$ classes
```

B Appendix B: Implementation Details

Datasets

We use nine public datasets

- Cora, Citeseer, and Pubmed from (Sen et al. 2008)
- Amazon-photo (A-Photo) from (Shchur et al. 2019)
- CoraFull from (Bojchevski and Günnemann 2018)
- ACM and DBLP from (Bo et al. 2020)
- UAT from (Mrabah et al. 2023)
- Wiki from (Cao, Lu, and Xu 2016).

Cora, Citeseer, Pubmed, CoraFull, and Amazon-photo can be acquired via the dataset interface of PyG (Fey and Lenssen 2019) [1]. And ACM, DBLP, UAT, and Wiki are hosted by Liu et al. (2023b)[2].

Baseline Implementation

We adopt the open source implementations of baselines.

- K-means: Faiss implementation[3]
- DEC: official implementation[4]
- SDCN, DAEGC, and DFCN: the third party implementation[5]
- HSAN: official implementation[6]
- SCGC: official implementation[7]
- GRACE: official implementation[8]
- S^{3}GC: official implementation[9]
- DinkNet: official implementation[10]

Regarding DinkNet, we use the official pretrained model weights for the Cora, Citeseer, and Amazon-Photo datasets. For the other datasets, we pretrain and fine-tune DinkNet from scratch, selecting the optimal combination from 100 sets of hyperparameter settings.

For the other baselines, we use the official hyperparameters on all datasets. For datasets not reported in the original papers, we tune the baseline methods over 100 sets of hyperparameters and report the best results.

[1] https://pytorch-geometric.readthedocs.io/en/latest/modules/datasets.html

[2] https://github.com/yueliu1999/Awesome-Deep-Graph-Clustering

[3] https://github.com/facebookresearch/faiss

[4] https://github.com/piiswrong/dec

[5] https://github.com/Marigoldwu/A-Unified-Framework-for-Deep-Attribute-Graph-Clustering

[6] https://github.com/yueliu1999/HSAN

[7] https://github.com/yueliu1999/SCGC

[8] https://github.com/CRIPAC-DIG/GRACE

[9] https://drive.google.com/corp/drive/folders/18B_eWbdVhOURZhqwoBSsyryb4WsiYLQK

[10] https://github.com/yueliu1999/Dink-Net

THESAURUS Implementation

We implement THESAURUS using PyTorch 2.1 (Paszke et al. 2017), PyG 2.5 (Fey and Lenssen 2019), and Faiss-GPU 1.8 (Johnson, Douze, and Jégou 2019; Douze et al. 2024), which are compiled with CUDA 12.1. To ensure reproducibility, all graph convolution operators utilize SparseTensor from torch-sparse 0.6[11] for sparse matrix multiplications. We implement the FGW-OT (Titouan et al. 2019) from scratch based on PyTorch and add KL regularization to it.

The weights of neural network $f_\omega \circ f_\theta$ and prototypes $\mathbf{S}$ are initialized with Kaiming uniform initializer (He et al. 2015). And THESAURUS is trained with Adam optimizer (Kingma and Ba 2017).

THESAURUS Hyperparameters

The hyperparameters of THESAURUS for all used datasets are listed in Table 3, including:

- S: the number of prototypes
- α: the weight of graph structure cost in FGW-OT
- τ: the softmax temperature of prediction in Eq. (3)
- pe: the edge drop rate of data augmentation
- px: the attribute drop rate of data augmentation
- T: the training epochs
- lr: the learning rate
- wd: the weight decay of Adam optimizer.

Moreover, the momentum weight for the prototype graph β_1 is consistently set to 0.99 across all datasets, while the momentum weight for the prototype marginal β_2 is set to 0.999.

Dataset	S	α	τ	pe	px	T	lr	wd
Cora	18	0.70	0.60	0.4	0.4	200	5e-4	5e-5
Citeseer	27	0.55	0.80	0.1	0.3	200	5e-4	5e-4
Pubmed	98	0.65	0.15	0.2	0.0	200	1e-3	5e-3
A-photo	63	0.45	0.60	0.3	0.3	200	1e-3	0
CoraFull	494	0.25	0.40	0.4	0.4	100	5e-4	1e-4
ACM	15	0.25	0.25	0.5	0.4	220	5e-3	5e-4
DBLP	57	0.25	0.2	0.5	0.3	200	1e-3	5e-3
UAT	5	0.95	0.65	0.2	0.5	220	5e-3	0
Wiki	240	0.75	0.25	0.3	0.5	50	1e-2	5e-5

Table 3: The hyperparameters of THESAURUS

C Appendix C: More Experiment Details

Evaluation Metrics

Clustering Accuracy and Macro-F1 Score Clustering accuracy (ACC) and Macro-F1 score (F1) are not directly applicable as clusters do not inherently have labels that can be directly compared with the ground truth. Hence, it typically involves finding the best match between cluster labels and the ground truth labels using an optimal relabeling strategy. This strategy involves solving a linear sum assignment (LSA) problem (Lovasz 1986; Crouse 2016) where each cluster label is mapped to a ground truth label to maximize the number of correct predictions.

The cost matrix $\mathbf{M} \in \mathbb{R}^{C \times C}$ is constructed such that each element $\mathbf{M}_{i,j}$ represents the "cost" of assigning cluster i to true label j. In the context of maximizing clustering accuracy, this cost is defined inversely as the number of data points in cluster i that belong to the true label j. Therefore, solving the problem involves maximizing the total number of correct classifications. Formally, we have

$$\max_{\sigma} \sum_{i=1}^{k} \mathbf{M}_{i,\sigma(i)}, \tag{13}$$

where σ is a permutation of the set $\{1, 2, \ldots, C\}$ and C is the number of classes. A fast sovler for LSA problems is scipy.optimize.linear_sum_assignment. With the solution σ, the cluster predictions of all nodes are first mapped to class predictions and then the normal accuracy and Macro-F1 computed like in classification tasks are taken as ACC and F1 for clustering.

NMI and ARI NMI measures the amount of information that is shared between the predicted clustering and the ground truth clustering. It is a normalization of the Mutual Information (MI) to scale the results between 0 (no mutual information) and 1 (perfect correlation). ARI is an adjustment of the Rand Index that corrects for the chance grouping of elements, providing a more accurate measure of how well the clustering has performed. For their formal definitions, please refer to any textbooks, e.g., (Gan, Ma, and Wu 2020), discussing clustering algorithms.

Environment

All experiments were conducted on two computers. The first is an Ubuntu 22.04 server equipped with an RTX 4090 GPU (24GB), an Intel i7-12700 CPU, and 64GB of RAM. The second is an Ubuntu 20.04 server equipped with an RTX 4090 GPU (24GB), two Intel Xeon Gold 6240C CPUs, and 126GB of RAM.

Both computers have the same Conda virtual environment, with PyTorch 2.1, PyG 2.5, and Faiss-GPU 1.8, all built on CUDA 12.1.

The Results on ACM, DBLP, UAT, and Wiki

To follow the evaluation protocol of Dink-Net (Liu et al. 2023a) and meet the page space limit, we only report the results on five datasets in the main content (Table 1). And to align with Dink-Net, we only report the results of one run in the main content.

To assess the stability of THESAURUS, we carry out experiments on ACM, DBLP, UAT, and Wiki, computing the mean and standard deviation across five runs. The results, displayed in Tables 4 and 5, demonstrate that THESAURUS consistently outperforms other methods across all four metrics. Moreover, the observed standard deviations are within a reasonable range, further confirming the stability of THESAURUS.

[11] https://github.com/rusty1s/pytorch_sparse

Datasets	Metrics	K-means	DEC	SDCN	DAEGC	DFCN	DCRN	Ours
ACM	ACC	67.31±0.71	84.33±0.76	90.45±0.18	86.94±2.83	90.90±0.20	91.72±1.44	**92.52±0.18**
	NMI	32.44±0.46	54.54±1.51	68.31±0.25	56.18±4.15	69.40±0.40	71.41±1.27	**73.26±0.27**
	ARI	30.60±0.69	60.64±1.87	73.91±0.40	59.35±3.89	74.90±0.40	77.18±1.28	**79.06±0.45**
	F1	67.57±0.74	84.51±0.74	90.42±0.19	87.07±2.79	90.80±0.20	83.92±0.76	**92.54±0.18**
DBLP	ACC	38.65±0.65	58.16±0.56	68.05±1.81	62.05±0.48	75.00±0.31	60.86±1.53	**82.25±0.32**
	NMI	11.45±0.38	29.51±0.28	39.50±1.34	32.49±0.45	42.70±1.03	25.87±0.41	**53.10±0.38**
	ARI	6.97±0.39	23.92±0.39	39.15±2.01	21.03±0.52	45.98±1.20	22.05±0.73	**59.29±0.62**
	F1	31.92±0.27	59.38±0.51	67.71±1.51	61.75±0.67	74.21±0.89	62.51±0.48	**81.71±0.33**
UAT	ACC	42.47±0.15	45.61±1.84	52.25±1.91	52.29±0.49	33.61±0.09	49.92±1.25	**62.25±0.92**
	NMI	22.39±0.69	16.63±2.39	21.61±1.26	21.33±0.44	26.49±0.41	24.09±0.53	**30.93±0.95**
	ARI	15.71±0.76	13.14±1.97	21.63±1.49	20.50±0.51	11.87±0.23	17.17±0.69	**31.12±1.33**
	F1	36.12±0.22	44.22±1.51	45.59±3.54	50.33±0.64	25.79±0.29	44.81±0.87	**61.29±0.71**
Wiki	ACC	31.92±2.42	33.58±0.95	42.12±0.28	38.14±0.52	44.37±0.75	48.52±1.31	**57.84±1.91**
	NMI	29.4±3.20	30.51±0.82	40.95±0.62	31.42±0.33	41.94±1.13	45.81±0.27	**53.41±0.96**
	ARI	4.11±1.43	14.19±0.59	27.26±0.38	17.94±0.21	27.33±0.49	27.38±0.57	**40.63±2.59**
	F1	20.29±1.32	22.38±0.47	37.55±0.54	24.51±0.19	37.53±0.72	37.47±0.63	**47.84±1.86**

Table 4: The performance (%) mean and standard deviation over 5 runs. The best is in bold.

Datasets	Metrics	HSAN	S^{3}GC	SCGC	Dink-Net*	Dink-Net	Ours
ACM	ACC	89.79±0.16	89.08±0.53	89.97±0.56	90.11±0.44	90.54±0.02	**92.52±0.18**
	NMI	66.97±0.50	64.22±1.10	67.08±0.97	68.32±0.93	69.00±0.05	**73.26±0.27**
	ARI	72.41±0.37	70.32±1.29	72.83±1.27	73.25±1.04	74.24±0.03	**79.06±0.45**
	F1	89.73±0.18	89.08±0.53	89.89±0.58	90.06±0.43	90.48±0.02	**92.54±0.18**
DBLP	ACC	73.18±0.77	76.89±0.31	66.82±1.45	73.31±2.41	74.71±0.02	**82.25±0.32**
	NMI	43.60±0.45	44.74±0.53	38.05±1.40	43.17±2.32	44.61±0.13	**53.10±0.38**
	ARI	44.80±0.80	49.14±0.67	35.42±1.82	43.51±3.26	45.55±0.86	**59.29±0.62**
	F1	72.84±0.76	76.47±0.25	66.60±1.50	73.20±2.28	74.65±0.54	**81.71±0.33**
UAT	ACC	56.04±0.67	49.55±4.36	56.58±1.62	54.84±1.83	58.07±0.17	**62.25±0.92**
	NMI	26.99±2.11	22.99±0.22	28.07±0.71	24.51±1.44	26.60±0.03	**30.93±0.95**
	ARI	25.22±1.96	20.49±1.36	24.80±1.85	23.45±1.55	25.66±0.06	**31.12±1.33**
	F1	54.20±1.84	46.19±5.54	55.52±0.87	52.32±2.08	55.55±0.11	**61.29±0.71**
Wiki	ACC	53.10±0.72	47.03±1.59	56.63±0.30	52.99±0.95	53.26±0.32	**57.84±1.91**
	NMI	50.11±0.32	22.49±0.59	52.62±0.49	49.91±1.11	51.06±0.27	**53.41±0.96**
	ARI	35.59±0.33	18.57±1.15	36.91±1.01	35.57±1.41	37.10±0.95	**40.63±2.59**
	F1	44.83±0.71	44.54±3.94	47.52±0.89	44.77±0.44	45.01±0.39	**47.84±1.86**

Table 5: The performance (%) mean and standard deviation over 5 runs. The best is in bold.

More Visualization Results

To elucidate the advantages of THESAURUS, we visualize the representations utilized for clustering along with the resultant cluster centers from both the previous SOTA, Dink-Net, and THESAURUS. This visualization is facilitated by dimensionality reduction to 2D using t-SNE, enabling detailed visual analysis.

A concise comparison between Dink-Net and THESAURUS on Cora is initially presented in Fig. 3, resized to conform to the page constraints of the main text. An expanded version of this visualization is provided in Fig. 5, along with additional comparisons across various datasets presented in Fig. 6 and Fig. 7. We do not visualize the results for CoraFull and Wiki due to the large number of classes, which makes clear representation difficult.

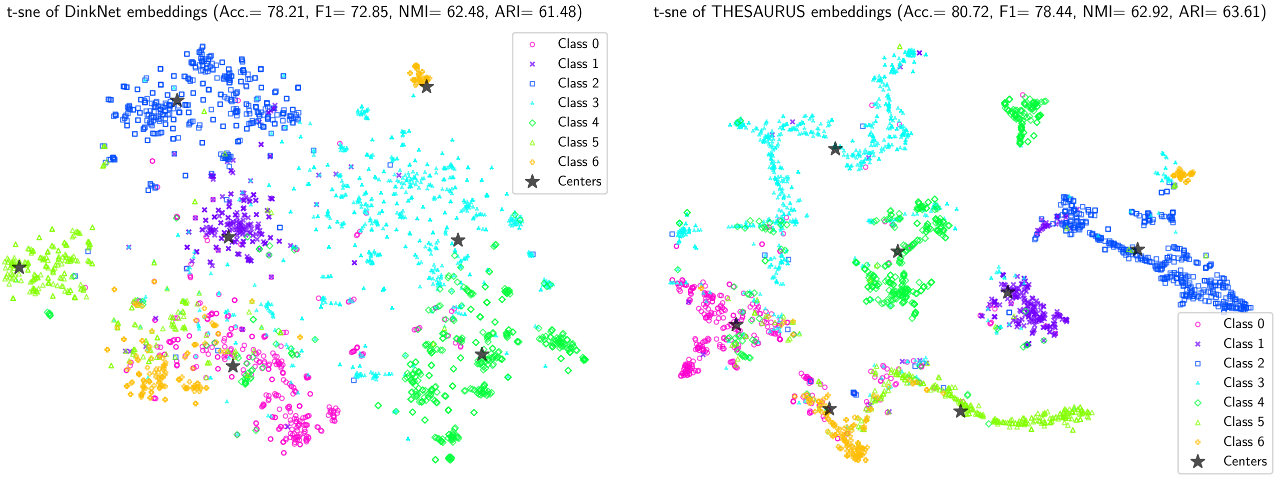
All t-SNE reduction experiments are performed using `sklearn.manifold.TSNE(n_components=2, learning_rate="auto", random_state=2050)` (Pedregosa et al. 2011) and `Matplotlib` (Hunter 2007). These visualizations clearly illustrate that the representation space developed by THESAURUS offers superior cluster separability over Dink-Net, characterized by larger inter-cluster distances and smaller intra-cluster distances.

D Reproducibility Checklist

1. This paper

   (a) includes a conceptual outline and/or pseudocode description of AI methods introduced (**yes**/partial/no/NA)

   (b) Clearly delineates statements that are opinions, hypothesis, and speculation from objective facts and results (**yes**/no)



(a) Dink-Net on Cora

(b) THESAURUS on Cora

Figure 5: The visualization of Dink-Net and THESAURUS on Cora, expanded from Fig. 3.

(c) Provides well marked pedagogical references for less-familiare readers to gain background necessary to replicate the paper (**yes**/no)

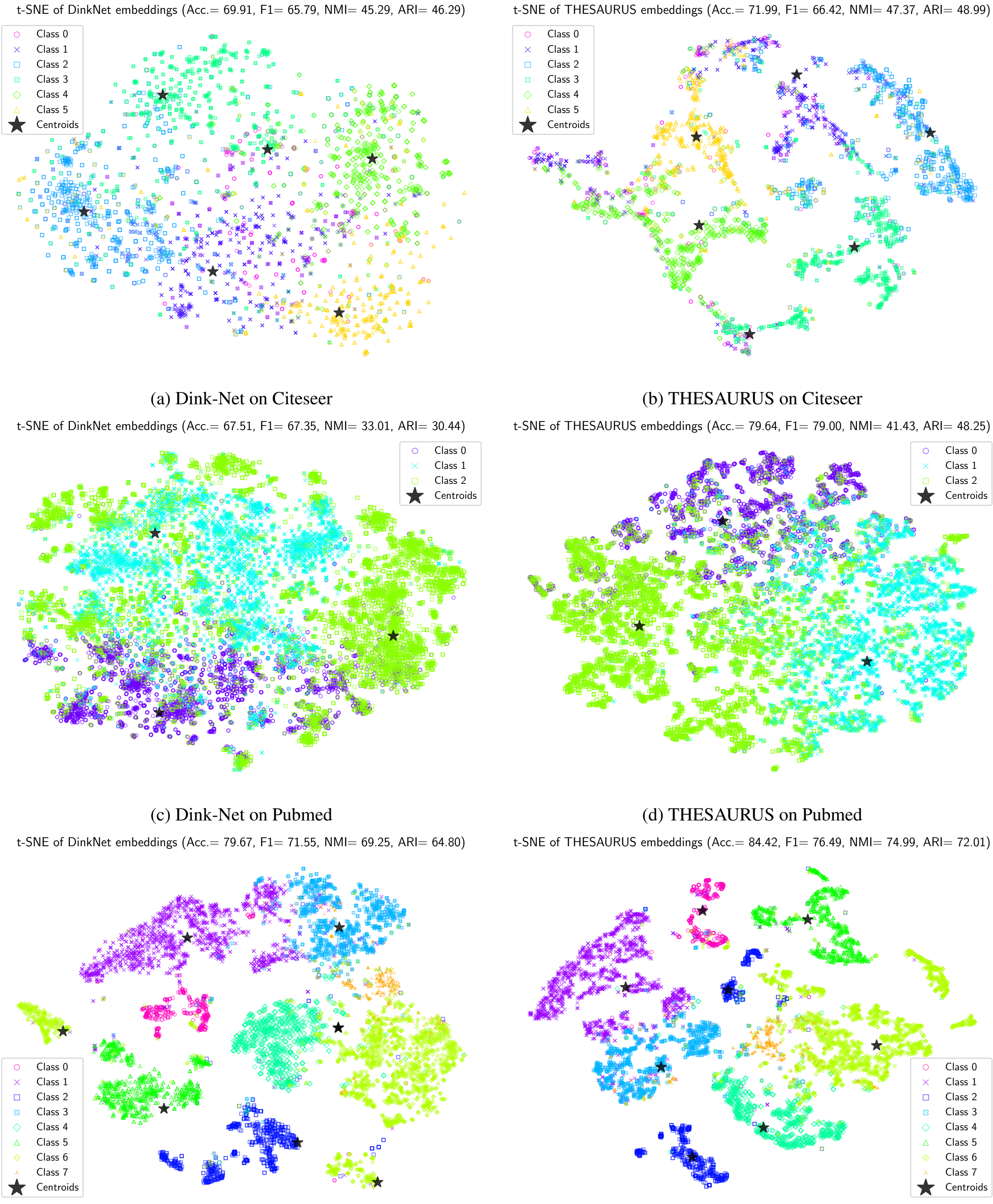
2. Does this paper make theoretical contributions? (yes/**no**)
   If yes, please complete the list below.
   (a) All assumptions and restrictions are stated clearly and formally. (yes/partial/no)
   (b) All novel claims are stated formally (e.g., in theorem statements). (yes/partial/no)
   (c) Proofs of all novel claims are included. (yes/partial/no)
   (d) Proof sketches or intuitions are given for complex and/or novel results. (yes/partial/no)
   (e) Appropriate citations to theoretical tools used are given. (yes/partial/no)
   (f) All theoretical claims are demonstrated empirically to hold. (yes/partial/no/NA)
   (g) All experimental code used to eliminate or disprove claims is included. (yes/no/NA)
3. Does this paper rely on one or more datasets? (**yes**/no)
   If yes, please complete the list below.
   (a) A motivation is given for why the experiments are conducted on the selected datasets (**yes**/partial/no/NA)
   (b) All novel datasets introduced in this paper are included in a data appendix. (yes/partial/no/**NA**)
   (c) All novel datasets introduced in this paper will be made publicly available upon publication of the paper with a license that allows free usage for research purposes. (yes/partial/no/**NA**)
   (d) All datasets drawn from the existing literature (potentially including authors' own previously published work) are accompanied by appropriate citations. (**yes**/no/NA)
   (e) All datasets drawn from the existing literature (potentially including authors' own previously published work) are publicly available. (**yes**/partial/no/NA)
   (f) All datasets that are not publicly available are described in detail, with explanation why publicly available alternatives are not scientifically satisficing. (yes/partial/no/**NA**)
4. Does this paper include computational experiments? (**yes**/no)
   If yes, please complete the list below.
   (a) Any code required for pre-processing data is included in the appendix. (**yes**/partial/no).
   (b) All source code required for conducting and analyzing the experiments is included in a code appendix. (yes/**partial**/no)
   (c) All source code required for conducting and analyzing the experiments will be made publicly available upon publication of the paper with a license that allows free usage for research purposes. (**yes**/partial/no)
   (d) All source code implementing new methods have comments detailing the implementation, with references to the paper where each step comes from (**yes**/partial/no)
   (e) If an algorithm depends on randomness, then the method used for setting seeds is described in a way sufficient to allow replication of results. (**yes**/partial/no/NA)
   (f) This paper specifies the computing infrastructure used for running experiments (hardware and software), including GPU/CPU models; amount of memory; operating system; names and versions of relevant software libraries and frameworks. (**yes**/partial/no)
   (g) This paper formally describes evaluation metrics used and explains the motivation for choosing these metrics. (**yes**/partial/no)
   (h) This paper states the number of algorithm runs used to compute each reported result. (**yes**/no)
   (i) Analysis of experiments goes beyond single-dimensional summaries of performance (e.g., average;

median) to include measures of variation, confidence, or other distributional information. (**yes**/no)

(j) The significance of any improvement or decrease in performance is judged using appropriate statistical tests (e.g., Wilcoxon signed-rank). (yes/**partial**/no)

(k) This paper lists all final (hyper-)parameters used for each model/algorithm in the paper's experiments. (**yes**/partial/no/NA)

(l) This paper states the number and range of values tried per (hyper-) parameter during development of the paper, along with the criterion used for selecting the final parameter setting. (**yes**/partial/no/NA)



t-SNE of DinkNet embeddings (Acc.= 69.91, F1= 65.79, NMI= 45.29, ARI= 46.29)

t-SNE of THESAURUS embeddings (Acc.= 71.99, F1= 66.42, NMI= 47.37, ARI= 48.99)

(a) Dink-Net on Citeseer

(b) THESAURUS on Citeseer

t-SNE of DinkNet embeddings (Acc.= 67.51, F1= 67.35, NMI= 33.01, ARI= 30.44)

t-SNE of THESAURUS embeddings (Acc.= 79.64, F1= 79.00, NMI= 41.43, ARI= 48.25)

(c) Dink-Net on Pubmed

(d) THESAURUS on Pubmed

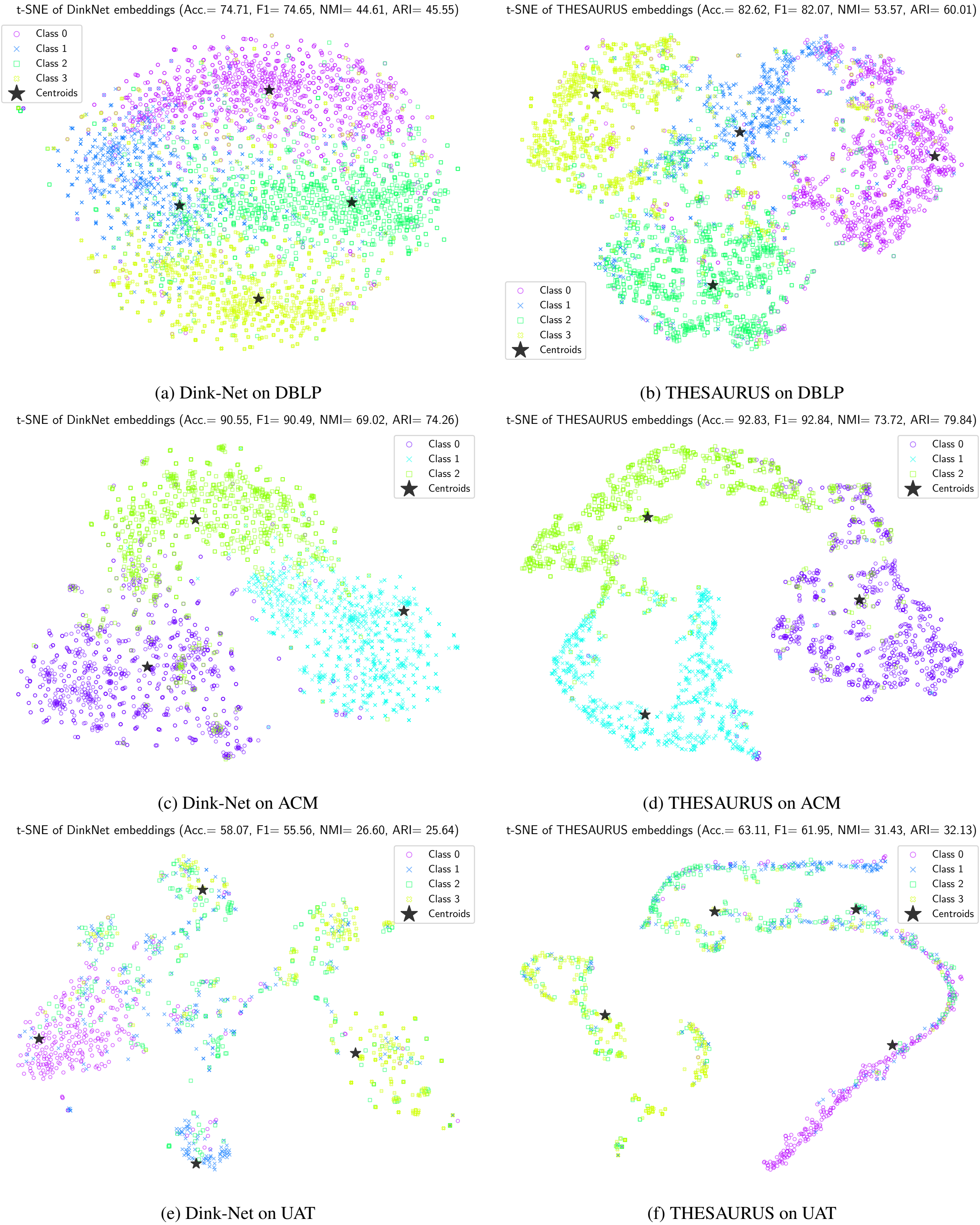
t-SNE of DinkNet embeddings (Acc.= 79.67, F1= 71.55, NMI= 69.25, ARI= 64.80)

t-SNE of THESAURUS embeddings (Acc.= 84.42, F1= 76.49, NMI= 74.99, ARI= 72.01)

(e) Dink-Net on Amazon-Photo

(f) THESAURUS on Amazon-Photo

Figure 6: The visualization comparison between Dink-Net and THESAURUS on Citeseer, Pubmed, and Amazon-photo



(a) Dink-Net on DBLP

(b) THESAURUS on DBLP

(c) Dink-Net on ACM

(d) THESAURUS on ACM

(e) Dink-Net on UAT

(f) THESAURUS on UAT

Figure 7: The visualization of Dink-Net and THESAURUS on DBLP, ACM, and UAT. Only one run is shown.