X-GOAL: Multiplex Heterogeneous Graph Prototypical Contrastive Learning

Baoyu Jing
baoyuj2@illinois.edu
University of Illinois at Urbana-Champaign

Shengyu Feng
shengyuf@andrew.cmu.edu
Language Technology Institute
Carnegie Mellon University

Yuejia Xiang
yuejiaxiang@tencent.com
Platform and Content Group
Tencent

Xi Chen
jasonxchen@tencent.com
Platform and Content Group
Tencent

Yu Chen
andyyuchen@tencent.com
Platform and Content Group
Tencent

Hanghang Tong
htong@illinois.edu
University of Illinois at Urbana-Champaign

ABSTRACT

Graphs are powerful representations for relations among objects, which have attracted plenty of attention in both academia and industry. A fundamental challenge for graph learning is how to train an effective Graph Neural Network (GNN) encoder without labels, which are expensive and time consuming to obtain. Contrastive Learning (CL) is one of the most popular paradigms to address this challenge, which trains GNNs by discriminating positive and negative node pairs. Despite the success of recent CL methods, there are still two under-explored problems. Firstly, how to reduce the semantic error introduced by random topology based data augmentations. Traditional CL defines positive and negative node pairs via the node-level topological proximity, which is solely based on the graph topology regardless of the semantic information of node attributes, and thus some semantically similar nodes could be wrongly treated as negative pairs. Secondly, how to effectively model the multiplexity of the real-world graphs, where nodes are connected by various relations and each relation could form a homogeneous graph layer. To solve these problems, we propose a novel multiplex heterogeneous graph prototypical contrastive leaning (X-GOAL) framework to extract node embeddings. X-GOAL is comprised of two components: the GOAL framework, which learns node embeddings for each homogeneous graph layer, and an alignment regularization, which jointly models different layers by aligning layer-specific node embeddings. Specifically, the GOAL framework captures the node-level information by a succinct graph transformation technique, and captures the cluster-level information by pulling nodes within the same semantic cluster closer in the embedding space. The alignment regularization aligns embeddings across layers at both node level and cluster level. We evaluate the proposed X-GOAL on a variety of real-world datasets and downstream tasks to demonstrate the effectiveness of the X-GOAL framework.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
CIKM '22, October 17–21, 2022, Atlanta, GA, USA.
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9236-5/22/10…$15.00
https://doi.org/10.1145/3511808.3557490

CCS CONCEPTS

• **Information systems** → **Data mining**; • **Computing methodologies** → **Unsupervised learning**; • **Networks**;

KEYWORDS

Prototypical Contrastive Learning, Multiplex Heterogeneous Graphs

ACM Reference Format:
Baoyu Jing, Shengyu Feng, Yuejia Xiang, Xi Chen, Yu Chen, and Hanghang Tong. 2022. X-GOAL: Multiplex Heterogeneous Graph Prototypical Contrastive Learning. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM '22), October 17–21, 2022, Atlanta, GA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3511808.3557490

1 INTRODUCTION

Graphs are powerful representations of formalisms and have been widely used to model relations among various objects [13, 23, 50, 65, 66, 74, 75], such as the citation relation and the same-author relation among papers. One of the primary challenges for graph representation learning is how to effectively encode nodes into informative embeddings such that they can be easily used in downstream tasks for extracting useful knowledge [13]. Traditional methods, such as Graph Convolutional Network (GCN) [23], leverage human labels to train the graph encoders. However, human labeling is usually time-consuming and expensive, and the labels might be unavailable in practice [6, 29, 60, 72, 73]. Self-supervised learning [29, 60], which aims to train graph encoders without external labels, has thus attracted plenty of attention in both academia and industry.

One of the predominant self-supervised learning paradigms in recent years is Contrastive Learning (CL), which aims to learn an effective Graph Neural Network (GNN) encoder such that positive node pairs will be pulled together and negative node pairs will be pushed apart in the embedding space [60]. Early methods, such as DeepWalk [42] and node2vec [12], sample positive node pairs based on their local proximity in graphs. Recent methods rely on graph transformation or augmentation [60] to generate positive pairs and negative pairs, such as random permutation [16, 18, 53], structure based augmentation [14, 67], sampling based augmentation [17, 45] as well as adaptive augmentation [76].

Albeit the success of these methods, they define positive and negative node pairs based upon the node-level information (or local

topological proximity) but have not fully explored the cluster-level (or semantic cluster/prototype) information. For example, in an academic graph, two papers about different sub-areas in graph learning (e.g., social network analysis and drug discovery) might not topologically close to each other since they do not have a direct citation relation or same-author relation. Without considering their semantic information such as the keywords and topics, these two papers could be treated as a negative pair by most of the existing methods. Such a practice will inevitably induce semantic errors to node embeddings, which will have a negative impact on the performance of machine learning models on downstream tasks such as classification and clustering. To address this problem, inspired by [27], we introduce a graph prototypical contrastive learning (GOAL) framework to simultaneously capture both node-level and cluster-level information. At the node level, GOAL trains an encoder by distinguishing positive and negative node pairs, which are sampled by a succinct graph transformation technique. At the cluster level, GOAL employs a clustering algorithm to obtain the semantic clusters/prototypes and it pulls nodes within the same cluster closer to each other in the embedding space.

Furthermore, most of the aforementioned methods ignore the multiplexity [18, 39] of the real-world graphs, where nodes are connected by multiple types of relations and each relation formulates a layer of the multiplex heterogeneous graph. For example, in an academic graph, papers are connected via the same authors or the citation relation; in an entertainment graph, movies are linked through the shared directors or actors/actresses; in a product graph, items have relations such as also-bought and also-view. Different layers could convey different and complementary information. Thus jointly considering them could produce more informative embeddings than separately treating different layers and then applying average pooling over them to obtain the final embeddings [18, 39]. Most of the prior deep learning methods use attention mechanism [4, 18, 30, 31, 38, 58] to combine embeddings from different layers. However, attention modules usually require extra tasks or loss functions to train, such as node classification [58] and concensus loss [38]. Besides, some attention modules are complex which require significant amount of extra efforts to design and tune, such as the hierarchical structures [58] and complex within-layer and cross-layer interactions [31]. Different from the prior methods, we propose an alternative nimble alignment regularization to jointly model and propagate information across different layers by aligning the layer-specific embeddings without extra neural network modules, and the final node embeddings are obtained by simply average pooling over these layer-specific embeddings. The key assumption of the alignment regularization is that layer-specific embeddings of the same node should be close to each other in the embedding space and they should also be semantically similar. We also theoretically prove that the proposed alignment regularization could effectively maximize the mutual information across layers.

We comprehensively evaluate X-GOAL on a variety of real-world attributed multiplex heterogeneous graphs. The experimental results show that the embeddings learned by GOAL and X-GOAL could outperform state-of-the-art methods of homogeneous graphs and multiplex heterogeneous graphs on various downstream tasks.

The main contributions are summarized as follows:

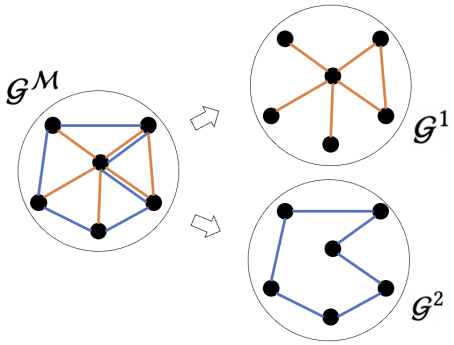


Figure 1: Illustration of the multiplex heterogeneous graph $\mathcal{G}^{\mathcal{M}}$, which can be decomposed into homogeneous graph layers $\mathcal{G}^1$ and $\mathcal{G}^2$ according to the types of relations. Different colors represent different relations.

- **Method.** We propose a novel X-GOAL framework to learn node embeddings for multiplex heterogeneous graphs, which is comprised of a GOAL framework for each single layer and an alignment regularization to propagate information across different layers. GOAL reduces semantic errors, and the alignment regularization is nimbler than attention modules for combining layer-specific node embeddings.
- **Theoretical Analysis.** We theoretically prove that the proposed alignment regularization can effectively maximize the mutual information across layers.
- **Empirical Evaluation.** We comprehensively evaluate the proposed methods on various real-world datasets and downstream tasks. The experimental results show that GOAL and X-GOAL outperform the state-of-the-art methods for homogeneous and multiplex heterogeneous graphs respectively.

2 PRELIMINARY

Definition 2.1 (Attributed Multiplex Heterogeneous Graph). An attributed multiplex heterogeneous graph with V layers and N nodes is denoted as $\mathcal{G}^{\mathcal{M}} = \{\mathcal{G}^v\}_{v=1}^{V}$, where $\mathcal{G}^v(\mathbf{A}^v, \mathbf{X})$ is the v-th homogeneous graph layer, $\mathbf{A}^v \in \mathbb{R}^{N\times N}$ and $\mathbf{X} \in \mathbb{R}^{N\times d_x}$ is the adjacency matrix and the attribute matrix, and d_x is the dimension of attributes. An illustration is shown in Figure 1.

Problem Statement. The task is to learn an encoder $\mathcal{E}$ for $\mathcal{G}^{\mathcal{M}}$, which maps the node attribute matrix $\mathbf{X} \in \mathbb{R}^{N\times d_x}$ to node embedding matrix $\mathbf{H}^{\mathcal{M}} \in \mathbb{R}^{N\times d}$ without external labels, where N is the number of nodes, d_x and d are the dimension sizes.

3 METHODOLOGY

We present the X-GOAL framework for multiplex heterogeneous graphs $\mathcal{G}^{\mathcal{M}}$, which is comprised of a GOAL framework and an alignment regularization. In Section 3.1, we present the GOAL framework, which simultaneously captures the node-level and the cluster-level information for each layer $\mathcal{G} = (\mathbf{A}, \mathbf{X})$ of $\mathcal{G}^{\mathcal{M}}$. In Section 3.2, we introduce a novel alignment regularization to align node embeddings across layers at both node and cluster level. In section 3.3, we provide theoretical analysis of the alignment regularization.

3.1 The GOAL Framework

The node-level graph topology based transformation techniques might contain semantic errors since they ignore the hidden semantics and will inevitably pair two semantically similar but topologically far nodes as a negative pair. To solve this issue, we introduce a GOAL framework for each homogeneous graph layer[1] $\mathcal{G} = (\mathbf{A}, \mathbf{X})$ to capture both node-level and cluster-level information. An illustration of GOAL is shown in Figure 2. Given a homogeneous graph $\mathcal{G}$ and an encoder $\mathcal{E}$, GOAL alternatively performs semantic clustering and parameter updating. In the semantic clustering step, a clustering algorithm C is applied over the embeddings $\mathbf{H}$ to obtain the hidden semantic clusters. In the parameter updating step, GOAL updates the parameters of $\mathcal{E}$ by the loss $\mathcal{L}$ given in Equation (4), which pulls topologically similar nodes closer and nodes within the same semantic cluster closer by the node-level loss and the cluster-level loss respectively.

A - Node-Level Loss. To capture the node-level information, we propose a graph transformation technique $\mathcal{T} = \{\mathcal{T}^+, \mathcal{T}^-\}$, where $\mathcal{T}^+$ and $\mathcal{T}^-$ denote positive and negative transformations, along with a contrastive loss similar to InfoNCE [36].

Given an original homogeneous graph $\mathcal{G} = (\mathbf{A}, \mathbf{X})$, the positive transformation $\mathcal{T}^+$ applies the dropout operation [48] over $\mathbf{A}$ and $\mathbf{X}$ with a pre-defined probability $p_{drop} \in (0, 1)$. We choose the dropout operation rather than the masking operation since the dropout re-scales the outputs by $\frac{1}{1-p_{drop}}$ during training, which improves the training results. The negative transformation $\mathcal{T}^-$ is the random shuffle of the rows for $\mathbf{X}$ [53]. The transformed positive and negative graphs are denoted by $\mathcal{G}^+ = \mathcal{T}^+(\mathcal{G})$ and $\mathcal{G}^- = \mathcal{T}^-(\mathcal{G})$, respectively. The node embedding matrices of $\mathcal{G}$, $\mathcal{G}^+$ and $\mathcal{G}^-$ are thus $\mathbf{H} = \mathcal{E}(\mathcal{G})$, $\mathbf{H}^+ = \mathcal{E}(\mathcal{G}^+)$ and $\mathbf{H}^- = \mathcal{E}(\mathcal{G}^-)$.

We define the node-level contrastive loss as:

$$\mathcal{L}_{\mathcal{N}} = -\frac{1}{N}\sum_{n=1}^{N} \log \frac{\mathbf{e}^{cos(\mathbf{h}_n, \mathbf{h}_n^+)}}{\mathbf{e}^{cos(\mathbf{h}_n, \mathbf{h}_n^+)} + \mathbf{e}^{cos(\mathbf{h}_n, \mathbf{h}_n^-)}} \tag{1}$$

where $cos(,)$ denotes the cosine similarity, $\mathbf{h}_n$, $\mathbf{h}_n^+$ and $\mathbf{h}_n^-$ are the n-th rows of $\mathbf{H}$, $\mathbf{H}^+$ and $\mathbf{H}^-$.

B - Cluster-Level Loss. We use a clustering algorithm C to obtain the semantic clusters of nodes $\{\mathbf{c}_k\}_{k=1}^K$, where $\mathbf{c}_k \in \mathbb{R}^d$ is the cluster center, K and d are the number of clusters and the dimension of embedding space. We capture the cluster-level semantic information to reduce the semantic errors by pulling nodes within the same cluster closer to their assigned cluster center. For clarity, the derivations of the cluster-level loss are provided in Appendix.

We define the probability of $\mathbf{h}_n$ belongs to the cluster k by:

$$p(k|\mathbf{h}_n) = \frac{\mathbf{e}^{(\mathbf{c}_k^T \cdot \mathbf{h}_n/\tau)}}{\sum_{k'=1}^{K} \mathbf{e}^{(\mathbf{c}_{k'}^T \cdot \mathbf{h}_n/\tau)}} \tag{2}$$

where $\tau > 0$ is the temperature parameter to re-scale the values.

The cluster-level loss is defined as the negative log-likelihood of the assigned cluster k_n for $\mathbf{h}_n$:

$$\mathcal{L}_C = -\frac{1}{N}\sum_{n=1}^{N} \log \frac{\mathbf{e}^{(\mathbf{c}_{k_n}^T \cdot \mathbf{h}_n/\tau)}}{\sum_{k=1}^{K} \mathbf{e}^{(\mathbf{c}_k^T \cdot \mathbf{h}_n/\tau)}} \tag{3}$$

where $k_n \in [1, \ldots, K]$ is the cluster index assigned to the n-th node.

[1]For clarity, we drop the script v of $\mathcal{G}^v$, $\mathbf{A}^v$ and $\mathbf{H}^v$ for this subsection.

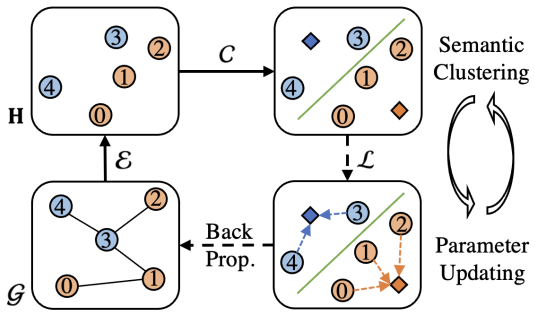


Figure 2: Illustration of GOAL. $\mathcal{E}$ and C are the encoder and clustering algorithm. $\mathcal{G}$ is a homogeneous graph layer and H is the embedding matrix. $\mathcal{L}$ is given in Equation (4). The circles and diamonds denote nodes and cluster centers. Blue and orange denote different hidden semantics. The green line is the cluster boundary. "Back Prop." means back propagation. The node-level topology based negative sampling treats the semantic similar node 0 and 2 as a negative pair. The cluster-level loss reduces semantic error by pulling node 0 and 2 closer to their cluster center.

C - Overall Loss. Combing the node-level loss in Equation (1) and the cluster-level loss in Equation (3), we have:

$$\mathcal{L} = \lambda_{\mathcal{N}}\mathcal{L}_{\mathcal{N}} + \lambda_C\mathcal{L}_C \tag{4}$$

where $\lambda_{\mathcal{N}}$ and λ_C are tunable hyper-parameters.

3.2 Alignment Regularization

Real-world graphs are often multiplex in nature, which can be decomposed into multiple homogeneous graph layers $\mathcal{G}^{\mathcal{M}} = \{\mathcal{G}^v\}_{v=1}^V$. The simplest way to extract the embedding of a node $\mathbf{x}_n$ in $\mathcal{G}^{\mathcal{M}}$ is separately extracting the embedding $\{\mathbf{h}_n^v\}_{v=1}^V$ from different layers and then combing them via average pooling. However, it has been empirically proven that jointly modeling different layers could usually produce better embeddings for downstream tasks [18]. Most prior studies use attention modules to jointly learn embeddings from different layers, which are clumsy as they usually require extra efforts to design and train [18, 30, 39, 58]. Alternatively, we propose a nimble alignment regularization to jointly learn embeddings by aligning the layer-specific $\{\mathbf{h}_n^v\}_{v=1}^V$ without introducing extra neural network modules, and the final node embedding of $\mathbf{x}_n$ is obtained by simply averaging the layer-specific embeddings $\mathbf{h}_n^{\mathcal{M}} = \frac{1}{V}\sum_{v=1}^V \mathbf{h}_n^v$. The underlying assumption of the alignment is that $\mathbf{h}_n^v$ should be close to and reflect the semantics of $\{\mathbf{h}_n^{v'}\}_{v' \neq v}^V$. The proposed alignment regularization is comprised of both node-level and cluster-level alignments.

Given $\mathcal{G}^{\mathcal{M}} = \{\mathcal{G}^v\}_{v=1}^V$ with encoders $\{\mathcal{E}^v\}_{v=1}^V$, we first apply GOAL to each layer $\mathcal{G}^v$ and obtain the original and negative node embeddings $\{\mathbf{H}^v\}_{v=1}^V$ and $\{\mathbf{H}^{v-}\}_{v=1}^V$, as well as the cluster centers $\{\mathbf{C}^v\}_{v=1}^V$, where $\mathbf{C}^v \in \mathbb{R}^{K^v \times d}$ is the concatenation of the cluster centers for the v-th layer, K^v is the number of clusters for the v-the layer. The node-level alignment is applied over $\{\mathbf{H}^v\}_{v=1}^V$ and $\{\mathbf{H}^{v-}\}_{v=1}^V$. The cluster-level alignment is used on $\{\mathbf{C}^v\}_{v=1}^V$ and $\{\mathbf{H}^v\}_{v=1}^V$.

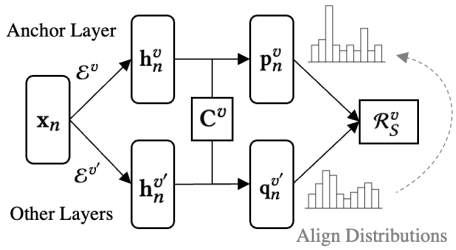


Figure 3: Cluster-level alignment. $\mathbf{x}_n$ is the node attribute. $\mathbf{h}_n^v$ and $\mathbf{h}_n^{v'}$ are the layer-specific embeddings. $\mathbf{C}^v$ is the anchor cluster center matrix. $\mathbf{p}_n^v$ and $\mathbf{q}_n^{v'}$ are the anchor and recovered semantic distributions. $\mathcal{R}_C^v$ is given in Equation (6).

A - Node-Level Alignment. For a node $\mathbf{x}_n$, its embedding $\mathbf{h}_n^v$ should be close to embeddings $\{\mathbf{h}_n^{v'}\}_{v'\neq v}^V$ and far away from the negative embedding $\mathbf{h}_n^{v-}$. Analogous to Equation (1), we define the node-level alignment regularization as:

$$\mathcal{R}_{\mathcal{N}} = -\frac{1}{Z}\sum_{n=1}^{N}\sum_{v=1}^{V}\sum_{v'\neq v}^{V}\log\frac{\mathbf{e}^{cos(\mathbf{h}_n^v,\mathbf{h}_n^{v'})}}{\mathbf{e}^{cos(\mathbf{h}_n^v,\mathbf{h}_n^{v'})}+\mathbf{e}^{cos(\mathbf{h}_n^v,\mathbf{h}_n^{v-})}} \tag{5}$$

where $Z = NV(V-1)$ is the normalization factor.

B - Cluster-Level Alignment. Similar to the node-level loss in Equation (1), the node-level alignment in Equation (5) could also introduce semantic errors since $\mathbf{h}_n^{v-}$ might be topologically far from but semantically similar to $\mathbf{h}_n^v$. To reduce the semantic error, we also align the layer-specific embeddings $\{\mathbf{h}_n^v\}_{v=1}^V$ at the cluster level.

Let the v-th layer be the anchor layer and its semantic cluster centers $\mathbf{C}^v \in \mathbb{R}^{K^v\times d}$ as the anchor cluster centers. For a node $\mathbf{x}_n$, we call its layer-specific embedding $\mathbf{h}_n^v$ as the anchor embedding, and its semantic distribution $\mathbf{p}_n^v \in \mathbb{R}^{K^v}$ as the anchor semantics, which is obtained via Equation (2) based on $\mathbf{h}_n^v$ and $\mathbf{C}^v$. Our key idea of the cluster-level alignment is to recover the anchor semantics $\mathbf{p}_n^v$ from embeddings $\{\mathbf{h}_n^{v'}\}_{v'\neq v}^V$ of other layers based on $\mathbf{C}^v$.

Our idea can be justified from two perspectives. Firstly, $\{\mathbf{h}_n^v\}_{v=1}^V$ reflect information of $\mathbf{x}_n$ from different aspects, if we can recover the anchor semantics $\mathbf{p}_n^v$ from the embedding $\mathbf{h}_n^{v'}$ of another layer $v' \neq v$, then it indicates that $\mathbf{h}_n^v$ and $\mathbf{h}_n^{v'}$ share hidden semantics to a certain degree. Secondly, it is impractical to directly align $\mathbf{p}_n^v$ and $\mathbf{p}_n^{v'}$, since their dimensions might be different $K^v \neq K^{v'}$, and even if $K^v = K^{v'}$, the cluster center vectors $\mathbf{C}^v$ and $\mathbf{C}^{v'}$ are distributed at different positions in the embedding space.

An illustration of the cluster-level alignment is presented in Figure 3. Given a node $\mathbf{x}_n$, on the anchor layer v, we have the anchor cluster centers $\mathbf{C}^v$, the anchor embedding $\mathbf{h}_n^v$, and the anchor semantic distribution $\mathbf{p}_n^v$. Next, we use the embedding $\mathbf{h}_n^{v'}$ from the layer $v' \neq v$ to obtain the recovered semantic distribution $\mathbf{q}_n^{v'}$ based on $\mathbf{C}^v$ via Equation (2). Then we align the semantics of $\mathbf{h}_n^v$ and $\mathbf{h}_n^{v'}$ by minimizing the KL-divergence of $\mathbf{p}_n^v$ and $\mathbf{q}_n^{v'}$:

$$\mathcal{R}_C^v = \frac{1}{N(V-1)}\sum_{n=1}^{N}\sum_{v'\neq v}^{V} KL(\mathbf{p}_n^v||\mathbf{q}_n^{v'}) \tag{6}$$

where $\mathbf{p}_n^v$ is treated as the ground-truth and the gradients are not allowed to pass through $\mathbf{p}_n^v$ during training.

Finally, we alternatively use all V layers as anchor layers and use the averaged KL-divergence as the final semantic regularization:

$$\mathcal{R}_S = \frac{1}{V}\sum_{v=1}^{V}\mathcal{R}_S^v \tag{7}$$

C - Overall Loss. By combining the node-level and cluster-level regularization losses, we have:

$$\mathcal{R} = \mu_{\mathcal{N}}\mathcal{R}_{\mathcal{N}} + \mu_C\mathcal{R}_C \tag{8}$$

where $\mu_{\mathcal{N}}$ and μ_C are tunable hyper-parameters.

The final training objective of the X-GOAL framework is the combination of the contrastive loss $\mathcal{L}$ in Equation (4) and the alignment regularization $\mathcal{R}$ in Equation (8):

$$\mathcal{L}_X = \sum_{v=1}^{V}\mathcal{L}^v + \mathcal{R} \tag{9}$$

where $\mathcal{L}^v$ is the loss of layer v

3.3 Theoretical Analysis

We provide theoretical analysis for the proposed regularization alignments. In Theorem 3.1, we prove that the node-level alignment maximizes the mutual information of embeddings $X^v \in \{\mathbf{h}_n^v\}_{n=1}^N$ of the anchor layer v and embeddings $X^{v'} \in \{\mathbf{h}_n^{v'}\}_{n=1}^N$ of another layer v'. In Theorem 3.2, we prove that the cluster-level alignment maximizes the mutual information of semantic cluster assignments $C^v \in [1,\cdots,K^v]$ for embeddings $\{\mathbf{h}_n^v\}_{n=1}^N$ of the anchor layer v and embeddings $H^{v'} \in \{\mathbf{h}_n^{v'}\}_{n=1}^N$ of the layer v'.

Theorem 3.1 (Maximization of MI of Embeddings from Different Layers). *Let $H^v \in \{\mathbf{h}_n^v\}_{n=1}^N$ and $H^{v'} \in \{\mathbf{h}_n^{v'}\}_{n=1}^N$ be the random variables for node embeddings of the v-th and v'-th layers, then the node-level alignment maximizes $I(H^v;H^{v'})$.*

Proof. According to [36, 43], the following inequality holds:

$$I(X;Y) \geq \mathbb{E}[\frac{1}{K_1}\sum_{i=1}^{K_1}\log\frac{\mathbf{e}^{f(x_i,y_i)}}{\frac{1}{K_2}\sum_{j=1}^{K_2}\mathbf{e}^{f(x_i,y_j)}}] \tag{10}$$

Let $K_1 = 1$, $K_2 = 2$, $f() = cos()$, $x_1 = \mathbf{h}_n^v$, $y_1 = \mathbf{h}_n^{v'}$, $y_2 = \mathbf{h}_n^{-v}$, then:

$$I(H^v;H^{v'}) \geq \mathbb{E}[\log\frac{\mathbf{e}^{cos(\mathbf{h}_n^v,\mathbf{h}_n^{v'})}}{\mathbf{e}^{cos(\mathbf{h}_n^v,\mathbf{h}_n^{v'})}+\mathbf{e}^{cos(\mathbf{h}_n^v,\mathbf{h}_n^{v-})}}] \tag{11}$$

The expectation $\mathbb{E}$ is taken over all the N nodes, and all the pairs of V layers, and thus we have:

$$I(H^v;H^{v'}) \geq \frac{1}{Z}\sum_{n=1}^{N}\sum_{v=1}^{V}\sum_{v'\neq v}^{V}\log\frac{\mathbf{e}^{cos(\mathbf{h}_n^v,\mathbf{h}_n^{v'})}}{\mathbf{e}^{cos(\mathbf{h}_n^v,\mathbf{h}_n^{v'})}+\mathbf{e}^{cos(\mathbf{h}_n^v,\mathbf{h}_n^{v-})}} \tag{12}$$

where $Z = NV(V-1)$ is the normalization factor, and the right side is $\mathcal{R}_{\mathcal{N}}$ in Equation (7). □

Theorem 3.2 (Maximization of MI between Embeddings and Semantic Cluster Assignments). *Let $C^v \in [1,\cdots,K^v]$ be the random variable for cluster assignments for $\{\mathbf{h}_n^v\}_{n=1}^N$ of the anchor layer v, and $H^{v'} \in \{\mathbf{h}_n^{v'}\}_{n=1}^N$ be the random variable for node embeddings of the v'-th layer, then the cluster-level alignment maximizes the mutual information of C^v and $H^{v'}$: $I(C^v;H^{v'})$.*

Table 1: Statistics of the datasets

Graphs	# Nodes	Views	# Edges	# Attributes	# Labeled Data	# Classes
ACM	3,025	Paper-Subject-Paper (PSP) Paper-Author-Paper (PAP)	2,210,761 29,281	1,830 (Paper Abstract)	600	3
IMDB	3,550	Movie-Actor-Movie (MAM) Movie-Director-Movie (MDM)	66,428 13,788	1,007 (Movie plot)	300	3
DBLP	7,907	Paper-Author-Paper (PAP) Paper-Paper-Paper (PPP) Paper-Author-Term-Author-Paper (PATAP)	144,783 90,145 57,137,515	2,000 (Paper Abstract)	80	4
Amazon	7,621	Item-AlsoView-Item (IVI) Item-AlsoBought-Item (IBI) Item-BoughtTogether-Item (IOI)	266,237 1,104,257 16,305	2,000 (Item description)	80	4

Proof. In the cluster-level alignment, the anchor distribution $\mathbf{p}_n^v$ is regarded as the ground-truth for the n-th node, and $\mathbf{q}_n^{v'} = f(\mathbf{h}_n^{v'})$ is the recovered distribution from the v'-th layer, where $f()$ is a K^v dimensional function defined by Equation (2). Specifically,

$$f(\mathbf{h}_n^{v'})[k] = p(k|\mathbf{h}_n^{v'}) = \frac{\mathbf{e}^{(\mathbf{c}_k^T \cdot \mathbf{h}_n^{v'}/\tau)}}{\sum_{k'=1}^{K^v} \mathbf{e}^{(\mathbf{c}_{k'}^T \cdot \mathbf{h}_n^{v'}/\tau)}} \tag{13}$$

where $\{\mathbf{c}_k\}_{k=1}^{K^v}$ is the set of cluster centers for the v-th layer.

Since $\mathbf{p}_n^v$ is the ground-truth, and thus its entropy $H(\mathbf{p}_n^v)$ is a constant. As a result, the KL divergence in Equation (6) is equivalent to cross-entropy $H(\mathbf{p}_n^v, \mathbf{q}_n^{v'}) = KL(\mathbf{p}_n^v||\mathbf{q}_n^{v'}) + H(\mathbf{p}_n^v)$. Therefore, minimizing the KL-divergence will minimize $H(\mathbf{p}_n^v, \mathbf{q}_n^{v'})$.

On the other hand, according to [33, 44], we have the following variational lower bound for $I(C^v; H^{v'})$:

$$I(C^v; H^{v'}) \geq \mathbb{E}[\log \frac{\mathbf{e}^{g(\mathbf{h}_n^{v'}, k)}}{\sum_{k'=1}^{K^v} \mathbf{e}^{g(\mathbf{h}_n^{v'}, k')}}] \tag{14}$$

where $g()$ is any function of $\mathbf{h}_n^{v'}$ and k.

In our case, we let

$$g(\mathbf{h}_n^{v'}, k) = \frac{1}{\tau} \mathbf{c}_k^T \cdot \mathbf{h}_n^{v'} \tag{15}$$

where $\mathbf{c}_k$ is the k-th semantic cluster center of the v-th layer, and τ is the temperature parameter.

As a result, we have

$$\frac{\mathbf{e}^{g(\mathbf{h}_n^{v'}, k)}}{\sum_{k'=1}^{K^v} \mathbf{e}^{g(\mathbf{h}_n^{v'}, k')}} = f[\mathbf{h}_n^{v'}][k] = \mathbf{q}_n^{v'}[k] \tag{16}$$

The expectation $\mathbb{E}$ is taken over the ground-truth distribution of the cluster assignments for the anchor layer v:

$$p_{gt}(\mathbf{h}_n^{v'}, k) = p_{gt}(\mathbf{h}_n^{v'}) p_{gt}(k|\mathbf{h}_n^{v'}) = \frac{1}{N} \mathbf{p}_n^v[k] \tag{17}$$

where $p_{gt}(k|\mathbf{h}_n^{v'}) = \mathbf{p}_n^v[k]$ is the ground-truth semantic distribution for $\mathbf{h}_n^{v'}$ on the anchor layer v, which is different from the recovered distribution $p(k|\mathbf{h}_n^{v'}) = \mathbf{q}_n^{v'}[k]$ shown in Equation (13).

Therefore, we have

$$I(C^v; H^{v'}) \geq \frac{1}{Z} \sum_{n=1}^{N} \sum_{k=1}^{K^v} \mathbf{p}_n^v[k] \log \mathbf{q}_n^{v'}[k] = -\frac{1}{Z} \sum_{n=1}^{N} H(\mathbf{p}_n^v, \mathbf{q}_n^{v'}) \tag{18}$$

where $Z = NK^v$ is the normalization factor.

Thus, minimizing $H(\mathbf{p}_n^v, \mathbf{q}_n^{v'})$ will maximize $I(C^v; H^{v'})$. □

4 EXPERIMENTS

4.1 Experimental Setups

Datasets. We use publicly available multiplex heterogeneous graph datasets [18, 39]: ACM, IMDB, DBLP and Amazon to evaluate the proposed methods. The statistics is summarized in Table 1.

Comparison Methods. We compare with methods for *(1) attributed graphs*, including methods disregarding node attributes: DeepWalk [42] and node2vec [12], and methods considering attributes: GCN [23], GAT [52], DGI [53], ANRL [71], CAN [34], DGCN [77], HDI[18], GCA [76] and GraphCL [67]; *(2) attributed multiplex heterogeneous graphs*, including methods disregarding node attributes: CMNA [5], MNE [68], and methods considering attributes: mGCN [31], HAN [58], MvAGC [28], DMGI, $\text{DMGI}_{\text{attn}}$ [39] and HDMI [18].

Evaluation Metrics. Following [18], we first extract embeddings from the trained encoder. Then we train downstream models with the extracted embeddings, and evaluate models' performance on the following tasks: *(1) a supervised task*: node classification; *(2) unsupervised tasks*: node clustering and similarity search. For the node classification task, we train a logistic regression model and evaluate its performance with Macro-F1 (MaF1) and Micro-F1 (MiF1). For the node clustering task, we train the K-means algorithm and evaluate it with Normalized Mutual Information (NMI). For the similarity search task, we first calculate the cosine similarity for each pair of nodes, and for each node, we compute the rate of the nodes to have the same label within its 5 most similar nodes (Sim@5).

Implementation Details. We use the one layer 1st-order GCN [23] with tangent activation as the encoder $\mathcal{E}^v = \tanh(\mathbf{A}^v\mathbf{X}\mathbf{W} + \mathbf{X}\mathbf{W}' + \mathbf{b})$. We set dimension $d = 128$ and $p_{drop} = 0.5$. The models are implemented by PyTorch [40] and trained on NVIDIA Tesla V-100 GPU. During training, we first warm up the encoders by training them with the node-level losses $\mathcal{L}_{\mathcal{N}}$ and $\mathcal{R}_{\mathcal{N}}$. Then we apply the overall loss $\mathcal{L}_{\mathcal{X}}$ with the learning rate of 0.005 for IMDB and 0.001 for other datasets. We use K-means as the clustering algorithm, and the semantic clustering step is performed every 5 epochs of parameter updating. We adopt early stopping with the patience of 100 to prevent overfitting.

4.2 Overall Performance

X-GOAL on Multiplex Heterogeneous Graphs. The overall performance for all of the methods is presented in Tables 2-3, where the upper and middle parts are the methods for homogeneous graphs and multiplex heterogeneous graphs respectively. "OOM"

Table 2: Overall performance of X-GOAL on the supervised task: node classification.

Dataset	ACM		IMDB		DBLP		Amazon	
Metric	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1
DeepWalk	0.739	0.748	0.532	0.550	0.533	0.537	0.663	0.671
node2vec	0.741	0.749	0.533	0.550	0.543	0.547	0.662	0.669
GCN/GAT	0.869	0.870	0.603	0.611	0.734	0.717	0.646	0.649
DGI	0.881	0.881	0.598	0.606	0.723	0.720	0.403	0.418
ANRL	0.819	0.820	0.573	0.576	0.770	0.699	0.692	0.690
CAN	0.590	0.636	0.577	0.588	0.702	0.694	0.498	0.499
DGCN	0.888	0.888	0.582	0.592	0.707	0.698	0.478	0.509
GraphCL	0.884	0.883	0.619	0.623	0.814	0.806	0.461	0.472
GCA	0.798	0.797	0.523	0.533	OOM	OOM	0.408	0.398
HDI	0.901	0.900	0.634	0.638	0.814	0.800	0.804	0.806
CMNA	0.782	0.788	0.549	0.566	0.566	0.561	0.657	0.665
MNE	0.792	0.797	0.552	0.574	0.566	0.562	0.556	0.567
mGCN	0.858	0.860	0.623	0.630	0.725	0.713	0.660	0.661
HAN	0.878	0.879	0.599	0.607	0.716	0.708	0.501	0.509
DMGI	0.898	0.898	0.648	0.648	0.771	0.766	0.746	0.748
$DMGI_{attn}$	0.887	0.887	0.602	0.606	0.778	0.770	0.758	0.758
MvAGC	0.778	0.791	0.598	0.615	0.509	0.542	0.395	0.414
HDMI	0.901	0.901	0.650	0.658	0.820	0.811	0.808	0.812
X-GOAL	**0.922**	**0.921**	**0.661**	**0.663**	**0.830**	**0.819**	**0.858**	**0.857**

Table 3: Overall performance of X-GOAL on the unsupervised tasks: node clustering and similarity search.

Dataset	ACM		IMDB		DBLP		Amazon	
Metric	NMI	Sim@5	NMI	Sim@5	NMI	Sim@5	NMI	Sim@5
DeepWalk	0.310	0.710	0.117	0.490	0.348	0.629	0.083	0.726
node2vec	0.309	0.710	0.123	0.487	0.382	0.629	0.074	0.738
GCN/GAT	0.671	0.867	0.176	0.565	0.465	0.724	0.287	0.624
DGI	0.640	0.889	0.182	0.578	0.551	0.786	0.007	0.558
ANRL	0.515	0.814	0.163	0.527	0.332	0.720	0.166	0.763
CAN	0.504	0.836	0.074	0.544	0.323	0.792	0.001	0.537
DGCN	0.691	0.690	0.143	0.179	0.462	0.491	0.143	0.194
GraphCL	0.673	0.890	0.149	0.565	0.545	0.803	0.002	0.360
GCA	0.443	0.791	0.007	0.496	OOM	OOM	0.002	0.478
HDI	0.650	0.900	0.194	0.605	0.570	0.799	0.487	0.856
CMNA	0.498	0.363	0.152	0.069	0.420	0.511	0.070	0.435
MNE	0.545	0.791	0.013	0.482	0.136	0.711	0.001	0.395
mGCN	0.668	0.873	0.183	0.550	0.468	0.726	0.301	0.630
HAN	0.658	0.872	0.164	0.561	0.472	0.779	0.029	0.495
DMGI	0.687	0.898	0.196	0.605	0.409	0.766	0.425	0.816
$DMGI_{attn}$	0.702	0.901	0.185	0.586	0.554	0.798	0.412	0.825
MvAGC	0.665	0.824	0.219	0.525	0.281	0.437	0.082	0.237
HDMI	0.695	0.898	0.198	0.607	0.582	**0.809**	0.500	0.857
X-GOAL	**0.773**	**0.924**	**0.221**	**0.613**	**0.615**	**0.809**	**0.556**	**0.907**

means out-of-memory. Among all the baselines, HDMI has the best overall performance. The proposed X-GOAL further outperforms HDMI. The proposed X-GOAL has 0.023/0.019/0.041/0.021 average improvements over the second best scores on Macro-F1/Micro-F1/NMI/Sim@5. For Macro-F1 and Micro-F1 in Table 2, X-GOAL improves the most on the Amazon dataset (0.050/0.044). For NMI and Sim@5 in Table 3, X-GOAL improves the most on the ACM (0.071) and Amazon (0.050) dataset respectively. The superior overall performance of X-GOAL demonstrate that the proposed approach can effectively extract informative node embeddings for multiplex heterogeneous graph.

GOAL on Homogeneous Graph Layers. We compare the proposed GOAL framework with recent infomax-based methods (DGI and HDI) and graph augmentation based methods (GraphCL and GCA). The experimental results for each single homogeneous graph layer are presented in Tables 4-5. It is evident that GOAL significantly outperforms the baseline methods on all single homogeneous graph layers. On average, GOAL has 0.137/0.129/0.151/0.119 improvements on Macro-F1/Micro-F1/NMI/Sim@5. For node classification in Table 4, GOAL improves the most on the PATAP layer of DBLP: 0.514/0.459 on Macro-F1/Micro-F1. For node clustering and similarity search in Table 5, GOAL improves the most on the

Table 4: Overall performance of GOAL on each layer: node classification.

Dataset	ACM				IMDB				DBLP						Amazon					
View	PSP		PAP		MDM		MAM		PAP		PPP		PATAP		IVI		IBI		IOI	
Metric	MaF1	MiF1	MaF1	MiF1	MaF1	MiF1	MaF1	MiF1	MaF1	MiF1	MaF1	MiF1	MaF1	MiF1	MaF1	MiF1	MaF1	MiF1	MaF1	MiF1
DGI	0.663	0.668	0.855	0.853	0.573	0.586	0.558	0.564	0.804	0.796	0.728	0.717	0.240	0.272	0.380	0.388	0.386	0.410	0.569	0.574
GraphCL	0.649	0.658	0.833	0.824	0.551	0.566	0.554	0.562	0.806	0.779	0.678	0.675	0.236	0.286	0.290	0.305	0.335	0.348	0.506	0.516
GCA	0.645	0.656	0.748	0.749	0.534	0.537	0.489	0.500	0.716	0.710	0.679	0.665	OOM	OOM	0.300	0.312	0.289	0.304	0.532	0.526
HDI	0.742	0.744	0.889	0.888	0.626	0.631	0.600	0.606	0.812	0.803	0.751	0.745	0.241	0.284	0.581	0.583	0.524	0.529	0.796	0.799
GOAL	**0.833**	**0.836**	**0.908**	**0.908**	**0.649**	**0.653**	**0.653**	**0.652**	**0.817**	**0.804**	**0.765**	**0.755**	**0.755**	**0.745**	**0.849**	**0.848**	**0.850**	**0.848**	**0.851**	**0.851**

Table 5: Overall performance of GOAL on each layer: node clustering and similarity search.

Dataset	ACM				IMDB				DBLP						Amazon					
View	PSP		PAP		MDM		MAM		PAP		PPP		PATAP		IVI		IBI		IOI	
Metric	NMI	Sim@5	NMI	Sim@5	NMI	Sim@5	NMI	Sim@5	NMI	Sim@5	NMI	Sim@5	NMI	Sim@5	NMI	Sim@5	NMI	Sim@5	NMI	Sim@5
DGI	0.526	0.698	0.651	0.872	0.145	0.549	0.089	0.495	0.547	0.800	0.404	0.741	0.054	0.583	0.002	0.395	0.003	0.414	0.038	0.701
GraphCL	0.524	0.735	0.675	0.874	0.128	0.554	0.060	0.485	0.539	0.794	0.347	0.702	0.052	0.595	0.001	0.334	0.002	0.360	0.036	0.630
GCA	0.389	0.662	0.062	0.764	0.008	0.491	0.008	0.463	0.076	0.775	0.223	0.683	OOM	OOM	0.002	0.315	0.007	0.329	0.008	0.588
HDI	0.528	0.716	0.662	0.886	0.194	0.592	0.143	0.527	0.562	0.805	0.408	0.742	0.054	0.591	0.169	0.544	0.153	0.525	0.407	0.826
GOAL	**0.600**	**0.851**	**0.735**	**0.917**	**0.210**	**0.602**	**0.180**	**0.585**	**0.589**	**0.809**	**0.447**	**0.757**	**0.412**	**0.733**	**0.551**	**0.901**	**0.544**	**0.903**	**0.536**	**0.905**

Table 6: Ablation study of X-GOAL at the multiplex heterogeneous graph level.

Dataset	ACM				IMDB				DBLP				Amazon			
Metric	MaF1	MiF1	NMI	Sim@5	MaF1	MiF1	NMI	Sim@5	MaF1	MiF1	NMI	Sim@5	MaF1	MaF1	MiF1	Sim@5
X-GOAL	**0.922**	**0.921**	**0.773**	**0.924**	**0.661**	**0.663**	**0.221**	**0.613**	**0.830**	**0.819**	**0.615**	**0.809**	**0.858**	**0.857**	**0.556**	**0.907**
w/o $\mathcal{R}_S$	0.919	0.917	0.770	0.922	0.658	0.661	0.211	0.606	0.817	0.807	0.611	0.804	0.856	0.856	0.555	0.906
w/o $\mathcal{R}_N$, $\mathcal{R}_S$	0.893	0.893	0.724	0.912	0.651	0.658	0.194	0.606	0.803	0.791	0.590	0.801	0.835	0.834	0.506	0.904

IBI layer of Amazon: 0.391 on NMI and 0.378 on Sim@5. The superior performance of GOAL indicates that the proposed prototypical contrastive learning strategy is better than the infomax-based and graph augmentation based instance-wise contrastive learning strategies. We believe this is because prototypical contrasive learning could effectively reduce the semantic errors.

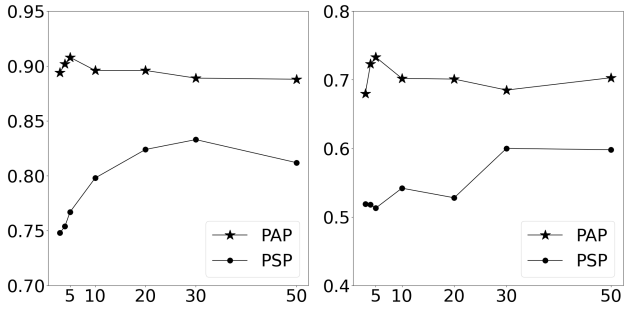
4.3 Ablation Study

Multiplex Heterogeneous Graph Level. In Table 6, we study the impact of the node-level and semantic-level alignments. The results in Table 6 indicate that both of the node-level alignment ($\mathcal{R}_N$) and the semantic-level alignment ($\mathcal{R}_S$) can improve the performance.

Homogeneous Graph Layer Level. The results for different configurations of GOAL on the PAP layer of ACM are shown in Table 7. First, all of the warm-up, the semantic-level loss $\mathcal{L}_S$ and the node-level loss $\mathcal{L}_N$ are critical. Second, comparing GOAL (1st-order GCN with tanh activation) with other GCN variants, (1) with the same activation function, the 1st-order GCN perform better than the original GCN; (2) tanh is better than relu. We believe this is because the 1st-order GCN has a better capability for capturing the attribute information, and tanh provides a better normalization for the node embeddings. Finally, for the configurations of graph transformation, if we replace dropout with masking, the performance will drop. This is because dropout re-scales the outputs by $1/(1-p_{drop})$, which improves the performance. Besides, dropout on both attributes and adjacency matrix is important.

Table 7: Ablation study of GOAL on the PAP layer of ACM.

	MaF1	MiF1	NMI	Sim@5
GOAL	**0.908**	**0.908**	**0.735**	**0.917**
w/o warm-up	0.863	0.865	0.721	0.903
w/o $\mathcal{L}_S$	0.865	0.867	0.693	0.899
w/o $\mathcal{L}_N$	0.878	0.880	0.678	0.881
1st-ord. GCN (relu)	0.865	0.866	0.559	0.859
GCN (tanh)	0.881	0.881	0.486	0.886
GCN (relu)	0.831	0.831	0.410	0.837
dropout → masking	0.888	0.890	0.716	0.903
w/o attribute drop	0.843	0.845	0.568	0.869
w/o adj. matrix drop	0.888	0.888	0.715	0.903

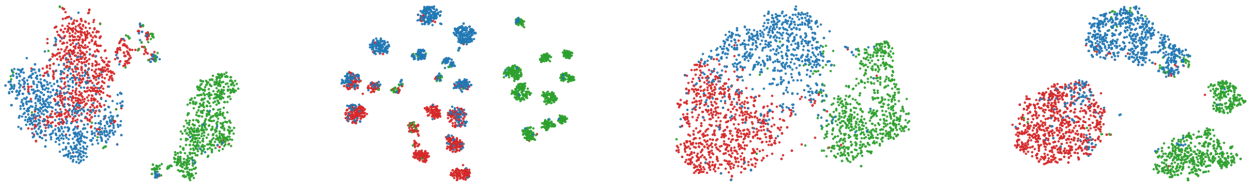


(a) Macro-F1 v.s. K **(b) NMI v.s. K**

Figure 4: The number of K on PSP and PAP of ACM

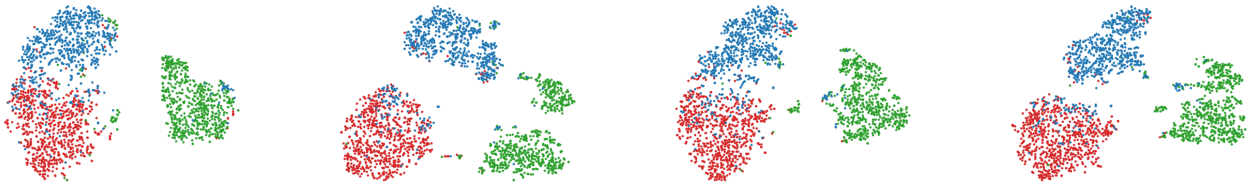
4.4 Number of Clusters

Figure 4 shows the Macro-F1 and NMI scores on the PSP and PAP layers of ACM w.r.t. the number of clusters $K \in [3, 4, 5, 10, 20, 30, 50]$. For PSP and PAP, the best Macro-F1 and NMI scores are obtained when $K = 30$ and $K = 5$. The number of ground-truth classes for ACM is 3, and the results in Figure 4 indicate that over-clustering is beneficial. We believe this is because there are many sub-clusters in the embedding space, which is consistent with the prior findings on image data [27].



(a) $\mathcal{L}_N$ on PSP (b) $\mathcal{L}_N + \mathcal{L}_C$ on PSP (c) $\mathcal{L}_N$ on PAP (d) $\mathcal{L}_N + \mathcal{L}_C$ on PAP

Figure 5: Visualization of the embeddings for the PAP and PSP layers of the ACM graph.



(a) $\mathcal{L}_N$ (b) $\mathcal{L}_N + \mathcal{L}_C$ (c) $\mathcal{L}_N + \mathcal{L}_C + \mathcal{R}_N$ (d) $\mathcal{L}_N + \mathcal{L}_C + \mathcal{R}_N + \mathcal{R}_C$

Figure 6: Visualization of the combined embeddings for the ACM graph.

4.5 Visualization

Homogeneous Graph Layer Level. The t-SNE [32] visualizations of the embeddings for PSP and PAP of ACM are presented in Figure 5. $\mathcal{L}_N$, $\mathcal{L}_C$, $\mathcal{R}_N$ and $\mathcal{R}_C$ are the node-level loss, cluster-level loss, node-level alignment and cluster-level alignment. The embeddings extracted by the full GOAL framework ($\mathcal{L}_N + \mathcal{L}_C$) are better separated than the node-level loss $\mathcal{L}_N$ only. For GOAL, the numbers of clusters for PSP and PAP are 30 and 5 since they have the best performance as shown in Figure 4.

Multiplex Heterogeneous Graph Level. The visualizations for the combined embeddings are shown in Figure 6. Embeddings in Figures 6a-6b are the average pooling of the layer-specific embeddings in Figure 5. Figure 6c and 6d are X-GOAL w/o cluster-level alignment and the full X-GOAL. Generally, the full X-GOAL best separates different clusters.

5 RELATED WORK

5.1 Contrastive Learning for Graphs

The goal of CL is to pull similar nodes into close positions and push dis-similar nodes far apart in the embedding space. Inspired by word2vec [35], early methods, such as DeepWalk [42] and node2vec [12] use random walks to sample positive pairs of nodes. LINE [50] and SDNE [56] determine the positive node pairs by their first and second-order structural proximity. Recent methods leverage graph transformation to generate node pairs. DGI [53], GMI [41], HDI [18] and CommDGI [69] obtain negative samples by randomly shuffling the node attributes. MVGRL [14] transforms graphs via techniques such as graph diffusion [24]. The objective of the above methods is to maximize the mutual information of the positive embedding pairs. GraphCL [67] uses various graph augmentations to obtain positive nodes. GCA [76] generates positive and negative pairs based on their importance. gCool [25] introduces graph communal contrastive learning. Ariel [8, 9] proposes a information regularized adversarial graph contrastive learning. These methods use the contrastive losses similar to InfoNCE [36].

For multiplex heterogeneous graphs, MNE [68], MVN2VEC [47] and GATNE [4] sample node pairs based on random walks. DMGI [39] and HDMI [18] use random attribute shuffling to sample negative nodes. HeCo [59] decides positive and negative pairs based on the connectivity between nodes. Above methods mainly rely on the topological structures to pair nodes, yet do not fully explore the semantic information, which could introduce semantic errors.

5.2 Deep Clustering and Contrastive Learning

Clustering algorithms [2, 62] can capture the semantic clusters of instances. DeepCluster [2] is one of the earliest works which use cluster assignments as "pseudo-labels" to update the parameters of the encoder. DEC [62] learns a mapping from the data space to a lower-dimensional feature space in which it iteratively optimizes a clustering objective. Inspired by these works, SwAV [3] and PCL [27] combine deep clustering with CL. SwAV compares the cluster assignments rather than the embeddings of two images. PCL is the closest to our work, which alternatively performs clustering to obtain the latent prototypes and train the encoder by contrasting positive and negative pairs of nodes and prototypes. However, PCL has some limitations compared with the proposed X-GOAL: it is designed for single view image data; it heavily relies on data augmentations and momentum contrast [15]; it has some complex assumptions over cluster distributions and embeddings.

5.3 Multiplex Heterogeneous Graph Neural Networks

The multiplex heterogeneous graph [4] considers multiple relations among nodes, and it is also known as multiplex graph [18, 39], multi-view graph [46], multi-layer graph [26] and multi-dimension graph [30]. MVE [46] and HAN [58] uses attention mechanisms to combine embeddings from different views. mGCN [31] models both within and across view interactions. VANE [11] uses adversarial training to improve the comprehensiveness and robustness of the embeddings. Multiplex graph neural networks have been used in many applications [7], such as time series [19], text summarization [21], temporal graphs [10], graph alignment [63], abstract reasoning [57], global poverty [22] and bipartite graphs [64].

5.4 Deep Graph Clustering

Graph clustering aims at discovering groups in graphs. SAE [51] and MGAE [55] first train a GNN, and then run a clustering algorithm over node embeddings to obtain the clusters. DAEGC [54] and SDCN [1] jointly optimize clustering algorithms and the graph reconstruction loss. AGC [70] adaptively finds the optimal order for graph filters based on the intrinsic clustering scores. M3S [49] uses clustering to enlarge the labeled data with pseudo labels. SDCN [1] proposes a structural deep clustering network to integrate the structural information into deep clustering. COIN [20] co-clusters two types of nodes in bipartite graphs. MvAGC [28] extends AGC [70] to multi-view settings. However, MvAGC is not neural network based methods which might not exploit the attribute and non-linearity information. Recent methods combine CL with clustering to further improve the performance. SCAGC [61] treats nodes within the same cluster as positive pairs. MCGC [37] combines CL with MvAGC [28], which treats each node with its neighbors as positive pairs. Different from SCAGC and MCGC, the proposed GOAL and X-GOAL capture the semantic information by treating a node with its corresponding cluster center as a positive pair.

6 CONCLUSION

In this paper, we introduce a novel X-GOAL framework for multiplex heterogeneous graphs, which is comprised of a GOAL framework for each homogeneous graph layer and an alignment regularization to jointly model different layers. The GOAL framework captures both node-level and cluster-level information. The alignment regularization is a nimble technique to jointly model and propagate information across different layers, which could maximize the mutual information of different layers. The experimental results on real-world multiplex heterogeneous graphs demonstrate the effectiveness of the proposed X-GOAL framework.

A DERIVATION OF SEMANTIC LEVEL LOSS

The node-level contrastive loss is usually noisy, which could introduce semantic errors by treating two semantic similar nodes as a negative pair. To tackle this issue, we use a clustering algorithm C (e.g. K-means) to obtain the semantic clusters of nodes, and we use the EM algorithm to update the parameters of $\mathcal{E}$ to pull node embeddings closer to their assigned clusters (or prototypes).

Following [27], we maximize the following log likelihood:

$$\sum_{n=1}^{N} \log p(\mathbf{h}_n|\mathbf{\Theta}, \mathbf{C}) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} p(\mathbf{h}_n, k|\mathbf{\Theta}, \mathbf{C}) \tag{19}$$

where $\mathbf{h}_n$ is the n-th row of $\mathbf{h}$, $\mathbf{\Theta}$ and $\mathbf{C}$ are the parameters of $\mathcal{E}$ and K-means algorithm C, $k \in [1, \cdots, K]$ is the cluster index, and K is the number of clusters. Directly optimizing this objective is impracticable since the cluster index is a latent variable.

The Evidence Lower Bound (ELBO) of Equation (19) is given by:

$$\text{ELBO} = \sum_{n=1}^{N} \sum_{k=1}^{K} Q(k|\mathbf{h}_n) \log \frac{p(\mathbf{h}_n, k|\mathbf{\Theta}, \mathbf{C})}{Q(k|\mathbf{h}_n)} \tag{20}$$

where $Q(k|\mathbf{h}_n) = p(k|\mathbf{h}_n, \mathbf{\Theta}, \mathbf{C})$ is the auxiliary function.

In the E-step, we fix $\mathbf{\Theta}$ and estimate the cluster centers $\hat{\mathbf{C}}$ and the cluster assignments $\hat{Q}(k|\mathbf{h}_n)$ by running the K-means algorithm over the embeddings of the original graph $\mathbf{H} = \mathcal{E}(\mathcal{G})$. If a node $\mathbf{h}_n$ belongs to the cluster k, then its auxiliary function is an indicator function satisfying $\hat{Q}(k|\mathbf{h}_n) = 1$, and $\hat{Q}(k'|\mathbf{h}_n) = 0$ for $\forall k' \neq k$.

In the M-step, based on $\hat{\mathbf{C}}$ and $\hat{Q}(k|\mathbf{h}_n)$ obtained in the E-step, we update $\mathbf{\Theta}$ by maximizing ELBO:

$$\begin{aligned} \text{ELBO} = &\sum_{n=1}^{N} \sum_{k=1}^{K} \hat{Q}(k|\mathbf{h}_n) \log p(\mathbf{h}_n, k|\mathbf{\Theta}, \hat{\mathbf{C}}) \\ &- \sum_{n=1}^{N} \sum_{k=1}^{K} \hat{Q}(k|\mathbf{h}_n) \log \hat{Q}(k|\mathbf{h}_n) \end{aligned} \tag{21}$$

Dropping the second term of the above equation, which is a constant, we will minimize the following loss function:

$$\mathcal{L}_{\mathcal{S}} = -\sum_{n=1}^{N} \sum_{k=1}^{K} \hat{Q}(k|\mathbf{h}_n) \log p(\mathbf{h}_n, k|\mathbf{\Theta}, \hat{\mathbf{C}}) \tag{22}$$

Assuming a uniform prior distribution over $\mathbf{h}_n$, we have:

$$p(\mathbf{h}_n, k|\mathbf{\Theta}, \hat{\mathbf{C}}) \propto p(k|\mathbf{h}_n, \mathbf{\Theta}, \hat{\mathbf{C}}) \tag{23}$$

We define $p(k|\mathbf{h}_n, \mathbf{\Theta}, \hat{\mathbf{C}})$ by:

$$p(k|\mathbf{h}_n, \mathbf{\Theta}, \hat{\mathbf{C}}) = \frac{\mathbf{e}^{(\hat{\mathbf{c}}_k^T \cdot \mathbf{h}_n/\tau)}}{\sum_{k'=1}^{K} \mathbf{e}^{(\hat{\mathbf{c}}_{k'}^T \cdot \mathbf{h}_n/\tau)}} \tag{24}$$

where $\mathbf{h}_n \in \mathbb{R}^d$ is the embedding of the node $\mathbf{x}_n$, $\hat{\mathbf{c}}_k \in \mathbb{R}^d$ is the vector of the k-th cluster center, τ is the temperature parameter.

Let's use k_n to denote the cluster assignment of $\mathbf{h}_n$, and normalize the loss by $\frac{1}{N}$, then Equation (22) can be rewritten as:

$$\mathcal{L}_{\mathcal{S}} = -\frac{1}{N} \sum_{n=1}^{N} \log \frac{\mathbf{e}^{(\mathbf{c}_{k_n}^T \cdot \mathbf{h}_n/\tau)}}{\sum_{k=1}^{K} \mathbf{e}^{(\mathbf{c}_k^T \cdot \mathbf{h}_n/\tau)}} \tag{25}$$

The above loss function captures the semantic similarities between nodes by pulling nodes within the same cluster closer to their assigned cluster center.

ACKNOWLEDGMENTS

BJ and HT are partially supported by NSF (1947135, 2134079 and 1939725), and NIFA (2020-67021-32799).

REFERENCES

[1] Deyu Bo, Xiao Wang, Chuan Shi, Meiqi Zhu, Emiao Lu, and Peng Cui. 2020. Structural deep clustering network. In *Proceedings of The Web Conference 2020*.

[2] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. 2018. Deep clustering for unsupervised learning of visual features. In *ECCV*.

[3] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. 2020. Unsupervised learning of visual features by contrasting cluster assignments. *NeurIPS* (2020).

[4] Yukuo Cen, Xu Zou, Jianwei Zhang, Hongxia Yang, Jingren Zhou, and Jie Tang. 2019. Representation learning for attributed multiplex heterogeneous network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1358–1368.

[5] Xiaokai Chu, Xinxin Fan, Di Yao, Zhihua Zhu, Jianhui Huang, and Jingping Bi. 2019. Cross-network embedding for multi-network alignment. In *The World Wide Web Conference*. 273–284.

[6] Boxin Du, Changhe Yuan, Robert Barton, Tal Neiman, and Hanghang Tong. 2021. Hypergraph Pre-training with Graph Neural Networks. *arXiv preprint arXiv:2105.10862* (2021).

[7] Boxin Du, Si Zhang, Yuchen Yan, and Hanghang Tong. 2021. New Frontiers of Multi-Network Mining: Recent Developments and Future Trend. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 4038–4039.

[8] Shengyu Feng, Baoyu Jing, Yada Zhu, and Hanghang Tong. 2022. Adversarial graph contrastive learning with information regularization. In *Proceedings of the ACM Web Conference 2022*. 1362–1371.

[9] Shengyu Feng, Baoyu Jing, Yada Zhu, and Hanghang Tong. 2022. ARIEL: Adversarial Graph Contrastive Learning. https://doi.org/10.48550/ARXIV.2208.06956

[10] Dongqi Fu, Liri Fang, Ross Maciejewski, Vetle I. Torvik, and Jingrui He. 2022. Meta-Learned Metrics over Multi-Evolution Temporal Graphs. In *KDD 2022*.

[11] Dongqi Fu, Zhe Xu, Bo Li, Hanghang Tong, and Jingrui He. 2020. A View-Adversarial Framework for Multi-View Network Embedding. In *CIKM*.

[12] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD*. 855–864.

[13] William L Hamilton, Rex Ying, and Jure Leskovec. 2017. Representation learning on graphs: Methods and applications. *arXiv preprint arXiv:1709.05584* (2017).

[14] Kaveh Hassani and Amir Hosein Khasahmadi. 2020. Contrastive Multi-View Representation Learning on Graphs. *arXiv preprint arXiv:2006.05582* (2020).

[15] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *CVPR*.

[16] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. 2019. Strategies for pre-training graph neural networks. *arXiv preprint arXiv:1905.12265* (2019).

[17] Yizhu Jiao, Yun Xiong, Jiawei Zhang, Yao Zhang, Tianqi Zhang, and Yangyong Zhu. 2020. Sub-Graph Contrast for Scalable Self-Supervised Graph Representation Learning. In *2020 IEEE International Conference on Data Mining (ICDM)*.

[18] Baoyu Jing, Chanyoung Park, and Hanghang Tong. 2021. Hdmi: High-order deep multiplex infomax. In *Proceedings of the Web Conference 2021*. 2414–2424.

[19] Baoyu Jing, Hanghang Tong, and Yada Zhu. 2021. Network of Tensor Time Series. In *The World Wide Web Conference*. https://doi.org/10.1145/3442381.3449969

[20] Baoyu Jing, Yuchen Yan, Yada Zhu, and Hanghang Tong. 2022. COIN: Co-Cluster Infomax for Bipartite Graphs. *arXiv preprint arXiv:2206.00006* (2022).

[21] Baoyu Jing, Zeyu You, Tao Yang, Wei Fan, and Hanghang Tong. 2021. Multiplex Graph Neural Network for Extractive Text Summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. 133–139.

[22] Muhammad Raza Khan and Joshua E Blumenstock. 2019. Multi-gcn: Graph convolutional networks for multi-view networks, with applications to global poverty. In *AAAI*, Vol. 33. 606–613.

[23] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[24] Johannes Klicpera, Stefan Weißenberger, and Stephan Günnemann. 2019. Diffusion improves graph learning. *NeurIPS* (2019).

[25] Bolian Li, Baoyu Jing, and Hanghang Tong. 2022. Graph Communal Contrastive Learning. In *Proceedings of the ACM Web Conference 2022*. 1203–1213.

[26] Jundong Li, Chen Chen, Hanghang Tong, and Huan Liu. 2018. Multi-layered network embedding. In *SDM*. 684–692.

[27] Junnan Li, Pan Zhou, Caiming Xiong, and Steven CH Hoi. 2021. Prototypical contrastive learning of unsupervised representations. *ICLR* (2021).

[28] Zhiping Lin and Zhao Kang. 2021. Graph filter-based multi-view attributed graph clustering. In *IJCAI*. 19–26.

[29] Yixin Liu, Shirui Pan, Ming Jin, Chuan Zhou, Feng Xia, and Philip S Yu. 2021. Graph self-supervised learning: A survey. *arXiv preprint arXiv:2103.00111* (2021).

[30] Yao Ma, Zhaochun Ren, Ziheng Jiang, Jiliang Tang, and Dawei Yin. 2018. Multi-dimensional network embedding with hierarchical structure. In *WSDM*. 387–395.

[31] Yao Ma, Suhang Wang, Chara C Aggarwal, Dawei Yin, and Jiliang Tang. 2019. Multi-dimensional graph convolutional networks. In *SDM*. 657–665.

[32] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, Nov (2008), 2579–2605.

[33] David McAllester and Karl Stratos. 2020. Formal limitations on the measurement of mutual information. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 875–884.

[34] Zaiqiao Meng, Shangsong Liang, Hongyan Bao, and Xiangliang Zhang. 2019. Co-embedding attributed networks. In *WSDM*.

[35] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

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

[37] Erlin Pan and Zhao Kang. 2021. Multi-view Contrastive Graph Clustering. *Advances in Neural Information Processing Systems* 34 (2021).

[38] Chanyoung Park, Jiawei Han, and Hwanjo Yu. 2020. Deep multiplex graph infomax: Attentive multiplex network embedding using global information. *Knowledge-Based Systems* 197 (2020), 105861.

[39] Chanyoung Park, Donghyun Kim, Jiawei Han, and Hwanjo Yu. 2020. Unsupervised Attributed Multiplex Network Embedding. In *AAAI*. 5371–5378.

[40] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems* 32 (2019), 8026–8037.

[41] Zhen Peng, Wenbing Huang, Minnan Luo, Qinghua Zheng, Yu Rong, Tingyang Xu, and Junzhou Huang. 2020. Graph Representation Learning via Graphical Mutual Information Maximization. In *Proceedings of The Web Conference 2020*.

[42] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD*. 701–710.

[43] Ben Poole, Sherjil Ozair, Aaron Van Den Oord, Alex Alemi, and George Tucker. 2019. On variational bounds of mutual information. In *ICML*.

[44] Zhenyue Qin, Dongwoo Kim, and Tom Gedeon. 2019. Rethinking softmax with cross-entropy: Neural network classifier as mutual information estimator. *arXiv preprint arXiv:1911.10688* (2019).

[45] Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. 2020. Gcc: Graph contrastive coding for graph neural network pre-training. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1150–1160.

[46] Meng Qu, Jian Tang, Jingbo Shang, Xiang Ren, Ming Zhang, and Jiawei Han. 2017. An attention-based collaboration framework for multi-view network representation learning. In *CIKM*.

[47] Yu Shi, Fangqiu Han, Xinwei He, Xinran He, Carl Yang, Jie Luo, and Jiawei Han. 2018. mvn2vec: Preservation and collaboration in multi-view network embedding. *arXiv preprint arXiv:1801.06597* (2018).

[48] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* 15, 1 (2014), 1929–1958.

[49] Ke Sun, Zhouchen Lin, and Zhanxing Zhu. 2020. Multi-stage self-supervised learning for graph convolutional networks on graphs with few labeled nodes. In *AAAI*, Vol. 34. 5892–5899.

[50] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*. 1067–1077.

[51] Fei Tian, Bin Gao, Qing Cui, Enhong Chen, and Tie-Yan Liu. 2014. Learning deep representations for graph clustering. In *AAAI*, Vol. 28.

[52] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph attention networks. *ICLR* (2018).

[53] Petar Veličković, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2019. Deep graph infomax. *ICLR* (2019).

[54] Chun Wang, Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, and Chengqi Zhang. 2019. Attributed graph clustering: A deep attentional embedding approach. *arXiv preprint arXiv:1906.06532* (2019).

[55] Chun Wang, Shirui Pan, Guodong Long, Xingquan Zhu, and Jing Jiang. 2017. Mgae: Marginalized graph autoencoder for graph clustering. In *CIKM*. 889–898.

[56] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. 1225–1234.

[57] Duo Wang, Mateja Jamnik, and Pietro Lio. 2020. Abstract Diagrammatic Reasoning with Multiplex Graph Networks. In *ICLR*.

[58] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous graph attention network. In *TheWebConf*.

[59] Xiao Wang, Nian Liu, Hui Han, and Chuan Shi. 2021. Self-supervised heterogeneous graph neural network with co-contrastive learning. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 1726–1736.

[60] Lirong Wu, Haitao Lin, Zhangyang Gao, Cheng Tan, Stan Li, et al. 2021. Self-supervised on Graphs: Contrastive, Generative, or Predictive. *arXiv preprint arXiv:2105.07342* (2021).

[61] Wei Xia, Quanxue Gao, Ming Yang, and Xinbo Gao. 2021. Self-supervised Contrastive Attributed Graph Clustering. *NeurIPS* (2021).

[62] Junyuan Xie, Ross Girshick, and Ali Farhadi. 2016. Unsupervised deep embedding for clustering analysis. In *ICML*.

[63] Hao Xiong, Junchi Yan, and Li Pan. 2021. Contrastive Multi-View Multiplex Network Embedding with Applications to Robust Network Alignment. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 1913–1923.

[64] Hansheng Xue, Luwei Yang, Vaibhav Rajan, Wen Jiang, Yi Wei, and Yu Lin. 2021. Multiplex Bipartite Network Embedding using Dual Hypergraph Convolutional Networks. In *Proceedings of the Web Conference 2021*. 1649–1660.

[65] Yuchen Yan, Lihui Liu, Yikun Ban, Baoyu Jing, and Hanghang Tong. 2021. Dynamic Knowledge Alignment. In *AAAI*.

[66] Yuchen Yan, Si Zhang, and Hanghang Tong. 2021. Bright: A bridging algorithm for network alignment. In *Proceedings of the Web Conference 2021*. 3907–3917.

[67] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. 2020. Graph contrastive learning with augmentations. *Advances in Neural Information Processing Systems* 33 (2020), 5812–5823.

[68] Hongming Zhang, Liwei Qiu, Lingling Yi, and Yangqiu Song. 2018. Scalable Multiplex Network Embedding.. In *IJCAI*, Vol. 18. 3082–3088.

[69] Tianqi Zhang, Yun Xiong, Jiawei Zhang, Yao Zhang, Yizhu Jiao, and Yangyong Zhu. 2020. CommDGI: community detection oriented deep graph infomax. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 1843–1852.

[70] Xiaotong Zhang, Han Liu, Qimai Li, and Xiao-Ming Wu. 2019. Attributed graph clustering via adaptive graph convolution. *arXiv preprint arXiv:1906.01210* (2019).

[71] Zhen Zhang, Hongxia Yang, Jiajun Bu, Sheng Zhou, Pinggang Yu, Jianwei Zhang, Martin Ester, and Can Wang. 2018. ANRL: Attributed Network Representation Learning via Deep Neural Networks.. In *IJCAI*, Vol. 18. 3155–3161.

[72] Lecheng Zheng, Dongqi Fu, and Jingrui He. 2021. Tackling oversmoothing of gnns with contrastive learning. *arXiv preprint arXiv:2110.13798* (2021).

[73] Lecheng Zheng, Yada Zhu, Jingrui He, and Jinjun Xiong. 2021. Heterogeneous Contrastive Learning. *arXiv preprint arXiv:2105.09401* (2021).

[74] Dawei Zhou, Lecheng Zheng, Jiawei Han, and Jingrui He. 2020. A data-driven graph generative model for temporal interaction networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 401–411.

[75] Dawei Zhou, Lecheng Zheng, Jiejun Xu, and Jingrui He. 2019. Misc-GAN: A multi-scale generative model for graphs. *Frontiers in big Data* 2 (2019), 3.

[76] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. 2021. Graph contrastive learning with adaptive augmentation. In *Proceedings of the Web Conference 2021*. 2069–2080.

[77] Chenyi Zhuang and Qiang Ma. 2018. Dual graph convolutional networks for graph-based semi-supervised classification. In *Proceedings of the 2018 World Wide Web Conference*. 499–508.