

# MUSE: Multi-View Contrastive Learning for Heterophilic Graphs

Mengyi Yuan  
School of Data Science and  
Engineering  
East China Normal University  
Shanghai, China  
mengyiyuan@stu.ecnu.edu.cn

Minjie Chen  
School of Data Science and  
Engineering  
East China Normal University  
Shanghai, China  
minjiechen@stu.ecnu.edu.cn

Xiang Li\*  
School of Data Science and  
Engineering  
East China Normal University  
Shanghai, China  
xiangli@dase.ecnu.edu.cn

## ABSTRACT

In recent years, self-supervised learning has emerged as a promising approach in addressing the issues of label dependency and poor generalization performance in traditional GNNs. However, existing self-supervised methods have limited effectiveness on heterophilic graphs, due to the homophily assumption that results in similar node representations for connected nodes. In this work, we propose a multi-view contrastive learning model for heterophilic graphs, namely, MUSE. Specifically, we construct two views to capture the information of the ego node and its neighborhood by GNNs enhanced with contrastive learning, respectively. Then we integrate the information from these two views to fuse the node representations. Fusion contrast is utilized to enhance the effectiveness of fused node representations. Further, considering that the influence of neighboring contextual information on information fusion may vary across different ego nodes, we employ an information fusion controller to model the diversity of node-neighborhood similarity at both the local and global levels. Finally, an alternating training scheme is adopted to ensure that unsupervised node representation learning and information fusion controller can mutually reinforce each other. We conduct extensive experiments to evaluate the performance of MUSE on 9 benchmark datasets. Our results show the effectiveness of MUSE on both node classification and clustering tasks. We provide our data and codes at <https://anonymous.4open.science/r/MUSE-BD4B>.

## CCS CONCEPTS

• **Computer systems organization** → **Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability.

## KEYWORDS

Graph neural networks, representation learning, contrastive learning

## ACM Reference Format:

Mengyi Yuan, Minjie Chen, and Xiang Li. 2018. MUSE: Multi-View Contrastive Learning for Heterophilic Graphs. In *Proceedings of Make sure to*

\*Corresponding author.

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](mailto:permissions@acm.org).

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY

© 2018 Association for Computing Machinery.  
ACM ISBN 978-1-4503-XXXX-X/18/06...\$15.00  
<https://doi.org/XXXXXXX.XXXXXXX>

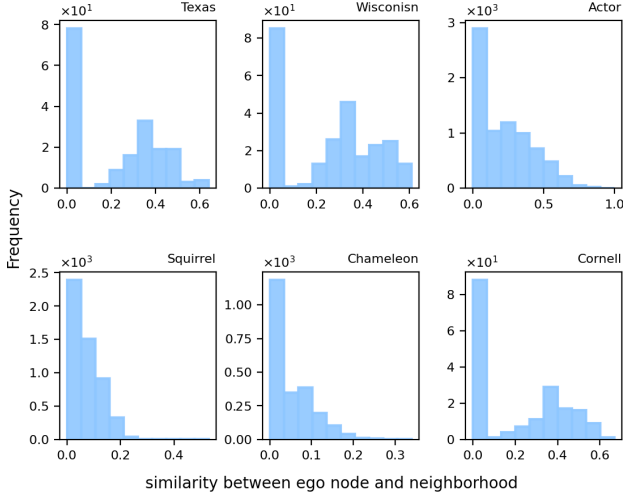
enter the correct conference title from your rights confirmation email (Conference acronym 'XX). ACM, New York, NY, USA, 10 pages. <https://doi.org/XXXXXXX.XXXXXXX>

## 1 INTRODUCTION

Graph neural networks (GNNs) [8, 15, 33, 37] have shown remarkable success in learning representations of graph-structured data. By iteratively aggregating information from a node's neighbors, GNNs map high-dimensional node representations into low-dimensional ones while preserving the useful information in the graph, which can be further applied to a variety of downstream tasks, such as node classification [8, 15, 39] and node clustering [35, 43].

Traditional GNNs mainly adopt a semi-supervised approach, which still depends on labeled data and poses several problems, like label sparsity and poor transferring capacity [2, 28]. Recent works have shown that combining self-supervised learning with GNNs can improve the robustness and generalization ability of the model [12, 24, 25, 34], since graph self-supervised learning can capture the underlying semantic and structure of the graph in an unsupervised manner. However, existing graph self-supervised methods are mainly based on the homophily assumption, leading to similar node representations between ego node and its neighboring nodes. In real-life scenarios, the assumption of homophily often does not hold, and the performance of these methods will significantly deteriorate. We conduct similarity analysis between the ego nodes and their neighborhoods on six heterophilic graphs, as shown in Figure 1. Evidently, there are a large number of extreme values at a low similarity level in each graph. This suggests that numerous nodes are dissimilar to their neighborhoods in real-world graphs.

Recently, most existing unsupervised learning methods [5, 31, 38] on heterophilic graphs distinguish the information of the ego node and its neighborhood. However, some of them emphasize the information from neighborhood at the expense of retaining ego node information, resulting in even worse performance on certain datasets than MLP [31, 38]. Some methods heavily rely on features of ego nodes to ensure their expressive power after multi-layer propagation in GNNs and fail to utilize the contextual information of neighboring nodes [5]. These methods also perform poorly on certain datasets. We attribute the poor generalization of these methods to the fact that they overlook the difference and diversity among nodes in a graph, which can also be observed from Figure 1. From the perspective of an individual graph, there exists a large discrepancy on the similarity between different ego nodes and their neighborhoods, which reflects the local diversity among nodes in the graph. We also observe that the global distribution of similarity among different graphs varies greatly. To cope with these two types of diversities, we first divide the sources of information for



**Figure 1: The distribution of similarity between ego node and its neighborhood in real-world graphs.**

nodes in a graph into two types: (1) features of ego node; (2) interaction between ego node and its neighborhood. Then a question naturally arises: *How to effectively fuse ego node information with neighboring contextual information to generate node representations?* On the one hand, information fusion needs to take into account the local difference of similarities between different ego nodes and their neighborhoods. On the other hand, effective representation fusion should reflect the global diversity across different graphs.

To address the aforementioned challenges, we propose a **M**ulti-view **c**ontra**S**tive learning method for **h**eterophilic graphs, namely, **MUSE**. Our model consists of two major components: *unsupervised node representation learning* and *information fusion controller*. For the former, we construct two views, namely, *semantic view* and *contextual view*, based on which we capture relevant information with GNNs and employ contrastive learning to learn node representations in each view that are invariant to perturbations, respectively. After that, we fuse the embeddings learned from the two views to generate cross-view node representations with an information fusion controller. This enables each node to adaptively integrate useful information from its neighborhood. To enhance the effectiveness of fused node representations, we further introduce contrastive learning to drive them to be perturbation invariant. For the information fusion controller, it determines the fusion weight by taking into account the diversity of node-neighborhood similarity at both the local and global levels, thereby leading to personalized information fusion. On the one hand, we consider the local diversity by measuring the similarity between embeddings generated from both semantic and contextual views. On the other hand, we handle the global diversity by constraining the distribution of similarity across the entire graph. Considering that the two components mutually reinforce each other, we adopt an alternating training scheme to optimize them simultaneously. Finally, our main contributions can be summarized as follows:

- We propose MUSE, a novel contrastive learning model for heterophilic graphs. To our best knowledge, MUSE is the first heterophilous graph contrastive learning method that considers both local and global node similarity diversity.
- We design an effective information fusion controller to model the diversity of node-neighborhood similarity, leading to personalized node representation fusion.
- We conduct extensive experiments on 9 benchmark datasets to evaluate the performance of MUSE. Our results show its superiority over other state-of-the-art competitors.

## 2 RELATED WORK

### 2.1 GNNs with heterophily

Existing graph neural network methods on heterophilic graphs can mainly be divided into two categories. One is to capture information from distant nodes [1, 17, 18, 23, 29]. For example, MixHop [1] concatenates information from multi-hop neighbours at each GNN layer. Geom-GCN [23] discovers potential neighbors in a continuous latent space. Both the neighbors in the original graph latent space are aggregated during the message passing. WRGAT [29] captures the information from distant nodes by defining the type and weight of edges in the entire graph to reconstruct a computation graph.

The other is to adaptively aggregate useful information from the neighborhood by refining the GNN architecture [3, 6, 20, 40, 44]. For example, H2GCN [44] excludes the self-loop connection and adopts a non-mixing operation in the GNN layer to emphasize the features of ego node. GGCN [40] learns a weighted combination of the prior layer node representations, with signed weight denoting different neighbors at every layer of GNN. ACM [20] uses high-pass, low-pass and identity filter to aggregate neighbor information by different channels.

### 2.2 Unsupervised graph representation learning

Unsupervised graph representation learning methods can be divided into two types: generation-based methods and contrast-based methods.

Generation-based methods reconstruct the graph data from the perspectives of feature and structure of the graph, and use the input data as the supervision signal. Classic generation-based methods on graph include GAE [16], VGAE [16], SIG-VAE [11] which focus on reconstructing the structure information of the graph and MGAE [36], GALA [22] which put emphasis on the reconstruction of the feature information of the graph.

Contrast-based methods construct representations under different views and maximize their agreement. According to the scale of contrast, existing methods come into three sub-categories: Node-to-node contrast, graph-to-graph contrast, and node-to-graph contrast. For example, GRACE [45] pulls the representations of the same node closer under different augmentations and pushes away the representations of other nodes. GraphCL [42] brings the graph-level representations closer under different views to ensure perturbation invariance. DGI [34] and MVGRL [12] maximize the mutual information between node-level representations and graph-level representations, aiming to capture both local and global information.

However, most of these methods are based on the homophily assumption. Recent works have moved the emphasis to unsupervised scenarios on heterophilic graphs. Based on graph contrastive learning, HGRL [5] improves the node representations on heterophilic graphs by preserving the node original features and capturing the non-local neighbors. GREET [19] discriminates homophilic edges from heterophilic edges and uses low-pass and high-pass filters to capture the corresponding information. Based on generation methods, DSSL [38] decouples the diverse patterns in local neighborhood distribution to capture both homophilic and heterophilic information. NWR-GAE [31] emphasizes the role of the topological structure in the graphs and reconstructs the neighborhoods based on the local structure and features.

### 3 PRELIMINARIES

In this section, we introduce the notations used in this paper and a brief background of GNNs.

**Notations** Let  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$  be an undirected, unweighted graph, where  $\mathcal{V} = \{v_1, \dots, v_N\}$  is the set of nodes and  $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$  is the set of edges.  $\mathbf{X} \in \mathbb{R}^{N \times F}$  is the feature matrix where the  $i$ -th row  $\mathbf{x}_i$  is the  $F$ -dimensional feature vector of node  $v_i$ .  $\mathbf{A} \in \mathbb{R}^{N \times N}$  denotes the binary adjacent matrix with  $\mathbf{A}_{ij} = 1$  if  $e_{i,j} \in \mathcal{E}$  and  $\mathbf{A}_{ij} = 0$  otherwise. The neighboring set of node  $v$  is denoted as  $\mathcal{N}(v)$ .

**Graph Neural Networks** GNNs adopt a message passing mechanism, where the representation of each node  $v \in \mathcal{V}$  is updated by aggregating messages from its local neighboring nodes, and then combining the aggregated messages with the node's own representation. Generally, given a GNN model  $f(\cdot)$ , message passing in the  $l$ -th layer can be divided into two operations: one is to aggregate information from a node's neighbors while the other is to update a node's representation. Given a node  $v_i$ , these two operations are formulated as:

$$m_i^{(l)} = \text{AGGREGATE}^{(l)}\{h_j^{(l-1)}, \forall v_j \in \mathcal{N}(v_i)\}, \quad (1)$$

$$h_i^{(l)} = \text{COMBINE}^{(l)}\{h_i^{(l-1)}, m_i^{(l)}\}, \quad (2)$$

where  $m_i^{(l)}$  and  $h_i^{(l)}$  denote the message vector and representation of node  $v_i$  in the  $l$ -th layer, respectively.  $\text{AGGREGATE}^{(l)}(\cdot)$  and  $\text{COMBINE}^{(l)}(\cdot)$  are two functions in each GNN layer.

### 4 METHODOLOGIES

In this section, we introduce the model design of our proposed MUSE method which is illustrated in Figure 2. Our approach consists of two components: unsupervised node representation learning and information fusion controller. MUSE first constructs two views to capture the semantic and contextual information of nodes in a graph with GNN. Semantic and contextual contrast are served as supervision signals to learn relevant node representations invariant to perturbations. MUSE then sends the information captured under two views into the information fusion controller to model the diversity of similarity between two kinds of information and generate the fused node representations in a node-specific way. Fusion contrast is employed to enhance the effectiveness of the fused node representations. Since the two components in MUSE have a mutually reinforcing effect, an alternating training scheme is adopted to optimize these two components simultaneously.

#### 4.1 View Construction

In heterophilic graphs, ego node and its neighborhood often exhibit high degree of heterophily due to differences in features and structure. To address this issue, we adopt a node-specific approach and construct two views: the semantic and the contextual view, for ego node and its neighborhood, respectively. The semantic view describes the nodes with their intrinsic properties, while the contextual view characterizes nodes based on their local neighborhoods.

**4.1.1 Semantic View.** Under semantic view, nodes that represent similar features in the graph are considered similar. Semantic-level contrast is employed as a form of supervision signal, aiming to encourage the learned representations of nodes with similar features to be consistent. For the set of initial features of nodes in the graph, we employ a perturbation  $\tau_\alpha$  to generate a new set of features as positive samples:

$$\{\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \dots, \tilde{\mathbf{x}}_N\} \sim \tau_\alpha(\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}), \quad (3)$$

where  $\tilde{\mathbf{x}}_i$  is the augmented sample of  $\mathbf{x}_i$ .

We apply perturbations by altering only the features of the ego nodes while keeping the structure of graph unchanged. Specifically, we randomly mask the initial node features in different dimensionality with a probability of  $p_s$ . We sample a binary vector  $\mathbf{m} \in \mathbb{R}^{1 \times F}$  from the Bernoulli distribution with a probability of  $(1 - p_s)$ , i.e.,  $m_i \sim \text{Bernoulli}(1 - p_s)$ ,  $i \in \{1, \dots, F\}$ , and perform element-wise multiplication with the features of each node:

$$\tilde{\mathbf{x}}_i = \mathbf{x}_i \circ \mathbf{m}, \quad (4)$$

The initial features are fed into the GNN encoder  $f(\cdot) : \mathbb{R}^{N \times F} \rightarrow \mathbb{R}^{N \times F'}$  to capture the semantic information of each node, where  $F'$  is the embedding dimensionality. To independently encode each node without taking into account the information from neighboring nodes, we construct a unit matrix  $\mathbf{I} \in \mathbb{R}^{N \times N}$  as the adjacency matrix and feed it into  $f(\cdot)$  along with initial features. The semantic representations obtained are denoted as  $\mathbf{H}^s \in \mathbb{R}^{N \times F'}$ :

$$\mathbf{H}^s = f_\omega(\mathbf{X}, \mathbf{I}), \quad (5)$$

where  $\omega$  denotes the parameters of GNN encoder.

Given a node  $v_i$ , the semantic representation of  $\mathbf{x}_i$ , denoted by  $\mathbf{h}_i^s$ , is the anchor sample. The semantic representation of the corresponding augmentation sample  $\tilde{\mathbf{x}}_i$ , denoted by  $\tilde{\mathbf{h}}_i^s$ , is treated as a positive sample of  $\mathbf{h}_i^s$ . The node representations of other nodes in the graph and their corresponding augmentations are considered as the negative samples of  $\mathbf{h}_i^s$ .

We construct the semantic-level contrastive loss based on the normalized temperature-scaled cross entropy loss (NT-Xent) [7]. Firstly, a non-linear projection  $g(\cdot) : \mathbb{R}^{N \times F'} \rightarrow \mathbb{R}^{N \times F_p}$  is used to map the node representations to a latent space where contrastive loss is applied.  $F_p$  is the dimensionality of projected representations. The non-linear projection head  $g$  is verified to be able to remove the information related to the downstream tasks from the node representations [7]. Then we measure the similarity between pairs of samples in this latent space. Measuring metric between sample pair  $(\mathbf{h}_i, \mathbf{h}_j)$  is defined as  $\theta(\mathbf{h}_i, \mathbf{h}_j) = s(g(\mathbf{h}_i, \mathbf{h}_j))$ , where  $s$  denotes cosine similarity and  $g$  is implemented with a two-layer MLP. Given a positive node pair  $(\mathbf{h}_i^s, \tilde{\mathbf{h}}_i^s)$ , the pairwise semantic contrastive loss

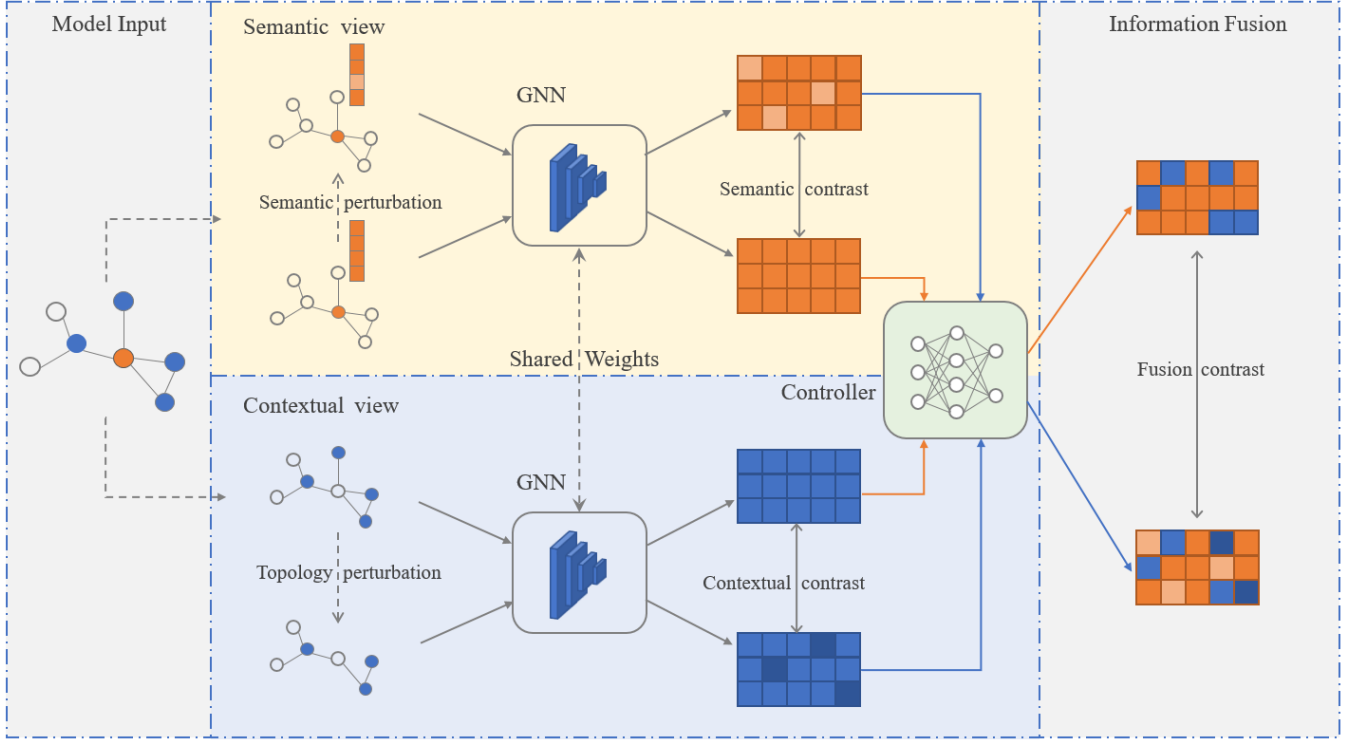


Figure 2: The overall framework of the proposed MUSE method.

is formulated as:

$$\ell_s(\mathbf{h}_i^s, \tilde{\mathbf{h}}_i^s) = -\log \frac{e^{\theta(\mathbf{h}_i^s, \tilde{\mathbf{h}}_i^s)/\tau}}{\sum_{v_j \in \mathcal{V} \setminus v_i} e^{\theta(\mathbf{h}_i^s, \mathbf{h}_j^s)/\tau} + \sum_{v_j \in \mathcal{V}} e^{\theta(\mathbf{h}_i^s, \tilde{\mathbf{h}}_j^s)/\tau}}, \quad (6)$$

Similarly, take  $\tilde{\mathbf{h}}_i^s$  as the anchor sample,  $\mathbf{h}_i^s$  is the positive sample of  $\tilde{\mathbf{h}}_i^s$ , and the negative samples remain unchanged. Correspondingly, pairwise contrastive loss between  $(\tilde{\mathbf{h}}_i^s, \mathbf{h}_i^s)$  is  $\ell(\tilde{\mathbf{h}}_i^s, \mathbf{h}_i^s)$ , and we can obtain the semantic contrastive loss  $\mathcal{L}_s$  as:

$$\mathcal{L}_s = \frac{1}{2N} \sum_{i=1}^N \left[ \ell(\mathbf{h}_i^s, \tilde{\mathbf{h}}_i^s) + \ell(\tilde{\mathbf{h}}_i^s, \mathbf{h}_i^s) \right], \quad (7)$$

**4.1.2 Contextual View.** Contextual view is constructed to capture the information from the interaction between ego node and its neighborhood. Different from the semantic view, here we focus on the contextual information provided by neighboring nodes rather than the features of the ego node.

Under the contextual view, nodes with similar local neighborhoods are considered to be similar. Context-level contrast is further employed to keep the representations of nodes with similar contexts to be consistent. When constructing positive samples under the contextual view, it should be ensured that the semantic information of nodes remains unchanged. Therefore, we introduce perturbations  $\tau_\beta$  to the topology of the neighboring nodes while preserving their semantic information:

$$\tilde{\mathbf{A}} \sim \tau_\beta(\mathbf{A}), \quad (8)$$

In practice, we randomly drop some edges in the graph with a probability of  $p_c$ . This is equivalent to removing a small portion of nodes from the neighborhood of each ego node to alter the contextual information. Specifically, we sample a binary masking matrix  $\mathbf{E} \in \{0, 1\}^{N \times N}$  from the Bernoulli distribution with a probability of  $(1 - p_c)$ , i.e.,  $e_{ij} \sim \text{Bernoulli}(1 - p_c)$ ,  $i, j \in \{1, \dots, N\}$ , and perform element-wise multiplication with the adjacent matrix:

$$\tilde{\mathbf{A}} = \mathbf{A} \circ \mathbf{E}, \quad (9)$$

To capture the contextual information of each node, we use feature matrix and adjacency matrix as the input of GNN encoder  $f(\cdot)$  and obtain the contextual representations denoted as  $\mathbf{H}^c \in \mathbb{R}^{N \times F'}$  by:

$$\mathbf{H}^c = f_\omega(\mathbf{X}, \mathbf{A}).$$

Similar as the contrastive loss under the semantic view, we construct the positive sample pair  $(\mathbf{h}_i^c, \tilde{\mathbf{h}}_i^c)$  and the corresponding negative sample pairs. The contextual contrastive loss is defined as:

$$\mathcal{L}_c = \frac{1}{2N} \sum_{i=1}^N \left[ \ell(\mathbf{h}_i^c, \tilde{\mathbf{h}}_i^c) + \ell(\tilde{\mathbf{h}}_i^c, \mathbf{h}_i^c) \right]. \quad (10)$$

## 4.2 Cross-view Node Representation Fusion

As previously discussed, we divide the source of information for a node in the graph into two types: (1) the node's own features, (2) the information brought by the interaction between the ego node and its neighborhood. To further enhance node representation learning in a graph, we fuse node embeddings learned from both the semantic view and the contextual view. We consider that the

contextual information plays a complementary role for the semantic information of the ego node. This fusion process not only captures the intrinsic properties of an individual ego node, but also includes the features and structural information of its neighborhood, thereby providing a more holistic node representation.

However, in practice, the amount of contextual information that each ego node needs varies, so we need to perform node-specific cross-view embedding fusion. Specifically, given a node  $v_i$ , we represent its fused representation as follows :

$$h_i^f = h_i^s + \lambda_i h_i^c, \quad (11)$$

where  $\lambda_i$  is a personalized weight which we will introduce in the next section.

Considering that the fused node representations should still remain invariant to perturbations, we introduce the fusion contrast to the unsupervised node representation learning. Given  $\mathbf{h}_i^f = \mathbf{h}_i^s + \lambda_i \mathbf{h}_i^c$  as an anchor sample, we consider  $\widetilde{\mathbf{h}}_i^f = \widetilde{\mathbf{h}}_i^s + \lambda_i \widetilde{\mathbf{h}}_i^c$  as the positive sample. We minimize the distance between the fused node representations before and after perturbations to ensure the invariance to perturbations. The cross-view fusion contrastive loss is defined as:

$$\mathcal{L}_f = \frac{1}{2N} \sum_{i=1}^N \left[ \ell(\mathbf{h}_i^f, \widetilde{\mathbf{h}}_i^f) + \ell(\widetilde{\mathbf{h}}_i^f, \mathbf{h}_i^f) \right], \quad (12)$$

To sum up, the unsupervised node representation loss can be obtained as:

$$\mathcal{L}_{contrast} = \mathcal{L}_s + \beta_1 \mathcal{L}_c + \beta_2 \mathcal{L}_f, \quad (13)$$

where  $\beta_1$  and  $\beta_2$  are weight factors for adjusting the importance of different components.

### 4.3 Information Fusion Controller

Considering the diversity across different nodes, we propose an information fusion controller  $\psi$  to model the diversity of similarity between the information deprived from two views and allow each node to adaptively integrate two kinds of information. Since node embeddings learned from the two views have a direct impact on the information fusion, we first filter the noisy features in  $h^s$  and  $h^c$ . Specifically, for node  $v_i$ , we have:

$$w_i^s = \varphi(h_i^s; \phi_1), w_i^c = \varphi(h_i^c; \phi_2), \quad (14)$$

where we implement the filter  $\varphi$  with a one-layer MLP and the output of filter is  $w^s \in R^{N \times F_g}$ ,  $w^c \in R^{N \times F_g}$ .

Given a node  $v_i$ , except the node's semantic properties and the contextual characteristics of its neighbors, the information fusion controller further takes into account the degree centrality. This measures the position of a node within the structure of the graph, which has been evidenced to benefit the performance of GNN on heterophilic graphs [21]. The degree centrality is formulated as:

$$d_i = \sum_{j=1}^N A_{ij}, \quad (15)$$

We calculate the personalized weight factor  $\lambda_i$  for each node  $v_i$  with a two-layer MLP by:

$$\lambda_i = \psi(w_i^s, w_i^c, d_i; \phi_3), \quad (16)$$

As shown in Figure 1, from the node level, there typically exists difference in the similarity between different ego nodes and their neighborhoods. When a node's semantic representation and contextual representation share high similarity, we consider that the contextual information provided by its neighboring nodes heavily overlaps with the node's own information. Therefore, this node needs less contextual information from neighborhood. From the graph level, the distribution of similarity between ego nodes and their neighborhoods in the whole graph varies across different graphs. Therefore, we impose some constraints on the overall distribution of  $\lambda$  with two regularization terms. Specifically, we constrain the average  $\lambda$  value for each individual graph to be a pre-set hyper-parameter. Additionally, considering that the overall distribution of all nodes in each individual graph may vary significantly, we use  $L_2$  norm to restrict the magnitude of this diversity. Therefore, we have the following objective function:

$$\mathcal{L}_\phi = \sum_{i=1}^N \lambda_i s(h_i^s, h_i^c) + \alpha_1 \|\lambda\|_2 + \alpha_2 \left| \frac{1}{N} \sum_{i=1}^N \lambda_i - \epsilon \right|, \quad (17)$$

where  $s(\cdot)$  denotes cosine similarity, coefficients  $\alpha_1$  and  $\alpha_2$  represent the weight factors in the objective,  $\epsilon$  is a hyper-parameter that controls the average  $\lambda$  in the graph,  $\phi$  denotes all the parameters in information fusion controller.

### 4.4 Overall Framework

**4.4.1 Model Training.** From the discussion above, we can obtain the overall optimization objective of MUSE as:

$$\mathcal{L}_{total} = \mathcal{L}_{contrast} + \mathcal{L}_\phi. \quad (18)$$

MUSE can be divided into two parts: unsupervised node representation learning and information fusion controller.

In the first part, we apply attribute-level perturbation to the initial input under the semantic view, and feed it into GNN to obtain corresponding node representations  $\mathbf{H}^s$  and  $\widetilde{\mathbf{H}}^s$ . Similarly, topology-level perturbation is applied to the structure of the graph and we can obtain the contextual node representations, denoted as  $\mathbf{H}^c$  and  $\widetilde{\mathbf{H}}^c$ . Furthermore, we fuse the node representations as  $\mathbf{H}^f = \mathbf{H}^s + \lambda \mathbf{H}^c$  to integrate information from two views.

In the second part, the semantic node representations  $\mathbf{H}^s$  and contextual node representations  $\mathbf{H}^c$  obtained in the first part is sent into an information fusion controller  $\psi$  to model the diversity of two kinds of information. The personalized weight factor  $\lambda$  learned by the information fusion controller fuses the node representations in a node-specific way.

**4.4.2 Optimization Strategy.** The two components in our model are coupled with each other. On the one hand, unsupervised node representation learning relies on the information fusion controller to control the integration of information from semantic and contextual views, and further generate node representations. On the other hand, the information fusion controller models the diversity of node-neighborhood similarity with node representations derived from different views. Therefore, we adopt an alternating training strategy to ensure the effectiveness of both components simultaneously. Specifically, during training, we first fix the information fusion controller  $\psi_\phi(\cdot)$  and train the encoder  $f_\omega(\cdot)$  and projector  $g_\mu(\cdot)$  by back-propagating  $\mathcal{L}_{contrast}$  to learn node representations.

Then, encoder  $f_\omega(\cdot)$  and projector  $g_\mu(\cdot)$  are fixed. Semantic and contextual node representations generated by the encoder  $f_\omega(\cdot)$  are sent into the information fusion controller  $\psi_\phi(\cdot)$ . We compute the controller loss  $\mathcal{L}_\phi$  to optimize the information fusion controller  $\psi_\phi(\cdot)$ . Here,  $\omega$  represents the parameters of encoder  $f(\cdot)$ ,  $\mu$  represents the parameters of projector  $g(\cdot)$ ,  $\phi$  denotes the parameters of information fusion controller  $\psi(\cdot)$ . It is worth noting that at the first epoch,  $\lambda$  used to update the GNN parameters via  $\mathcal{L}_{contrast}$  is obtained from the information fusion controller that is randomly initialized.

**4.4.3 Time Complexity Analysis.** We next analyze the time complexity of the two main components in our model. In unsupervised node representation learning, the time complexities of GNN encoder and projection head are  $O(F|\mathcal{E}|L + NFF'L)$  and  $O(NF'F_pL_p)$ , where  $|\mathcal{E}|$  is the number of edges,  $L$  and  $L_p$  are the layers of GNN and projection head, respectively,  $F$  is the dimensionality of initial features,  $F'$  is the dimensionality of final node representations,  $F_p$  is the dimensionality of projected representations in projection head and  $N$  is the number of nodes in a graph. Further, for the information fusion controller, the time complexities of the noise filter and computing  $\lambda$  in Equation 16 are  $O(NF'F_g)$  and  $O(NF_g)$ , respectively, where  $F_g$  is the output dimensionality of the filter and  $F_g, F' \ll N$ .

## 5 EXPERIMENTS

### 5.1 Experimental Settings

**5.1.1 Datasets.** We evaluate our model on nine real-world datasets, including three homophilic datasets (Cora, CiteSeer, PubMed [27]) and six heterophilic datasets (Cornell, Texas, Wisconsin, Chameleon, Squirrel, Actor [23]). Among them, Cora, Citeseer, Pubmed are citation networks. Cornell, Texas, and Wisconsin are school department webpage networks. Chameleon and Squirrel are Wikipedia networks. Actor is an actor co-occurrence network in Wiki pages. Details of these datasets are summarized in the Table 1.

**Table 1: Datasets statistics.**

Datasets	Node	Edges	Features	Classes
Cornell	183	295	1,703	5
Texas	183	309	1,703	5
Wisconsin	251	499	1,703	5
Chameleon	2,277	36,051	2325	5
Squirrel	5,201	216,933	2089	5
Actor	7,600	29,926	932	5
Cora	2708	10,556	1,433	7
CiteSeer	3,327	9,104	3,703	6
PubMed	19,717	88,648	500	3

**5.1.2 Baselines.** (1) For node classification as the downstream task, we compare our model with four groups of baseline methods: supervised learning methods (i.e., GCN [15], GAT [33], and MLP), supervised methods specially designed for heterophilic graphs (i.e., WRGAT [29], H2GCN [44]), contrast-based unsupervised learning methods designed for homophilic graphs (i.e., DGI [34], GMI [24],

MVGRL [12], BGRL [32], GRACE [45]) and unsupervised learning methods designed for heterophilic graphs (i.e., DSSL [38], NWR-GAE [31], HGRL [5], GREET [19]).

(2) For node clustering as the downstream task, we compare our model with four groups of baseline methods: traditional unsupervised clustering methods (i.e., AE [13], node2vec [9], struc2vec [26], LINE [30]), attributed graph clustering methods (i.e., GAE(VGAE) [16], GraphSAGE [10], SDCN [4]), contrast-based unsupervised methods designed for homophilic graphs (i.e., MVGRL [12], GRACE [45], BGRL [32]) and unsupervised methods designed for heterophilic graphs (i.e., DSSL [38], HGRL [5]).

**5.1.3 Implementation Details.** We implement our model by PyTorch and optimize the model by Adam optimizer [14]. We utilize a two-layer GCN [15] as the GNN encoder, and conduct all the experiments following the standard linear evaluation scheme that is widely adopted [19, 34, 45]. In the training step, we train the model in an unsupervised learning manner to learn the node representations. In the evaluating step, the learned node representations are sent into the downstream tasks. We run the experiments with 10 random splits, and report mean classification accuracy with standard deviation. We set  $\alpha_2$  in Equation 17 to 1 and the embedding dimensionality of filter  $F_g$  to 30. We fine-tune the following hyper-parameters:  $lr \in \{0.0001, 0.0005, 0.001, 0.005, 0.01\}$ ,  $lr_{controller} \in \{0.0001, 0.0005, 0.001, 0.005, 0.01\}$ ,  $\beta_1 \in \{0.001, 0.01, 0.1, 1, 10, 100\}$ ,  $\beta_2 \in \{0.001, 0.01, 0.1, 1, 10, 100\}$ ,  $p_s \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$ ,  $p_c \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$ ,  $\epsilon \in \{0, 0.1, 0.2, \dots, 1\}$ ,  $\alpha_1 \in \{10^2, 10^4, 10^6\}$ ,  $dropout \in \{0.1, 0.2, 0.3, 0.4\}$ .

Two downstream tasks are employed to evaluate the effectiveness and generalizability of the learned node representations. (1) Node classification: For homophilic graphs, we adopt the public splits with 20 nodes per class for training, 500 nodes for validation and 1,000 nodes for testing [15, 41]. For heterophilic graphs, we adopt the commonly used training/validation/test split ratio of 48/32/20 as previous works [19, 23]. A linear model is trained on top of the frozen node representations, and test accuracy is adopted as the evaluation metric. Results for GCN, GAT, MLP, DGI, GMI, MVGRL, BGRL, GRACE are reported from [19]. For WRGAT, H2GCN, DSSL, NWR-GAE and GREET, results are derived from the original papers. For HGRL and results not reported in the original papers, we run their public code on standard splits while keeping other settings the same. (2) Node clustering: Frozen node representation is fed into a K-means clustering model and the number of clusters is set as the number of classes. We adopt three evaluation metrics: accuracy (ACC), normalized mutual information (NMI) and average rand index (ARI). We run the public code of DSSL and fine-tune its hyper-parameters to report the best results. Results for other baselines are directly derived from [5].

### 5.2 Experimental Results

**5.2.1 Node Classification Performance.** Table 2 summarizes the performance results of node classification on three homophilic datasets and six heterophilic datasets. From the table, we make the following observations:

(1) MLP achieves good results on four of six heterophilic graphs (Cornell, Texas, Wisconsin, Actor), indicating that node features only can play a vital role in node representation learning. On the

**Table 2: Results of node classification (in percent  $\pm$  standard deviation). \* indicates the results are derived from the original papers. The best and runner-up results are highlighted with bold and underline, respectively.**

Methods	Heterophilic						Homophilic			Average Performance		
	Cornell	Texas	Wisconsin	Chameleon	Squirrel	Actor	Cora	CiteSeer	PubMed	All	Hete.	Homo.
GCN	57.03 $\pm$ 3.30	60.00 $\pm$ 4.80	56.47 $\pm$ 6.55	59.63 $\pm$ 2.32	36.28 $\pm$ 1.52	30.83 $\pm$ 0.77	81.5*	70.3*	79.0*	59.00	50.04	76.93
GAT	59.46 $\pm$ 3.63	61.62 $\pm$ 3.78	54.71 $\pm$ 6.87	56.38 $\pm$ 2.19	32.09 $\pm$ 3.27	28.06 $\pm$ 1.48	83.0*	72.5*	79.0*	58.54	48.72	78.17
MLP	81.08 $\pm$ 7.93	81.62 $\pm$ 5.51	84.31 $\pm$ 3.40	46.91 $\pm$ 2.15	29.28 $\pm$ 1.33	35.66 $\pm$ 0.94	56.11 $\pm$ 0.34	56.91 $\pm$ 0.42	71.35 $\pm$ 0.05	60.36	59.81	61.46
WRGAT*	81.62 $\pm$ 3.90	83.62 $\pm$ 5.50	<u>86.98<math>\pm</math>3.78</u>	65.24 $\pm$ 0.87	48.85 $\pm$ 0.78	36.53 $\pm$ 0.77	<b>88.20<math>\pm</math>2.26</b>	<u>76.81<math>\pm</math>1.89</u>	<u>88.52<math>\pm</math>0.92</u>	<u>72.93</u>	<u>67.14</u>	<u>84.51</u>
H2GCN*	82.16 $\pm$ 4.80	84.86 $\pm$ 6.77	86.67 $\pm$ 4.69	59.39 $\pm$ 1.98	37.90 $\pm$ 2.02	35.86 $\pm$ 1.03	<u>87.81<math>\pm</math>1.35</u>	<b>77.07<math>\pm</math>1.64</b>	<b>89.59<math>\pm</math>0.33</b>	71.26	64.47	<b>84.82</b>
DGI	63.35 $\pm$ 4.61	60.59 $\pm$ 7.56	55.41 $\pm$ 5.96	39.95 $\pm$ 1.75	31.80 $\pm$ 0.77	29.82 $\pm$ 0.69	82.29 $\pm$ 0.56	71.49 $\pm$ 0.14	77.43 $\pm$ 0.84	56.90	46.82	77.07
GMI	54.76 $\pm$ 5.06	50.49 $\pm$ 2.21	45.98 $\pm$ 2.76	46.97 $\pm$ 3.43	30.11 $\pm$ 1.92	30.11 $\pm$ 1.92	82.51 $\pm$ 1.47	71.56 $\pm$ 0.56	79.83 $\pm$ 0.90	54.45	42.69	77.97
MVGRL	64.30 $\pm$ 5.43	62.38 $\pm$ 5.61	62.37 $\pm$ 4.32	51.07 $\pm$ 2.68	35.47 $\pm$ 1.29	30.02 $\pm$ 0.70	83.03 $\pm$ 0.27	72.75 $\pm$ 0.46	79.63 $\pm$ 0.38	60.11	50.94	78.47
BGRL	57.30 $\pm$ 5.51	59.19 $\pm$ 5.85	52.35 $\pm$ 4.12	47.46 $\pm$ 2.74	32.64 $\pm$ 0.78	29.86 $\pm$ 0.75	81.08 $\pm$ 0.17	71.59 $\pm$ 0.42	79.97 $\pm$ 0.36	56.83	46.47	77.55
GRACE	54.86 $\pm$ 6.95	57.57 $\pm$ 5.68	50.00 $\pm$ 5.83	48.05 $\pm$ 1.81	31.33 $\pm$ 1.22	29.01 $\pm$ 0.78	80.08 $\pm$ 0.53	71.41 $\pm$ 0.38	80.15 $\pm$ 0.34	55.83	45.14	77.21
DSSL	53.15 $\pm$ 1.28	62.11 $\pm$ 1.53	56.29 $\pm$ 4.42	48.74 $\pm$ 1.53	40.51 $\pm$ 0.38	28.36 $\pm$ 0.65	83.06 $\pm$ 0.53	73.51 $\pm$ 0.64	82.98 $\pm$ 0.49	58.75	48.19	79.85
NWR-GAE	58.64 $\pm$ 5.61	69.62 $\pm$ 6.66	68.23 $\pm$ 6.11	<u>72.04<math>\pm</math>2.59</u>	<b>64.81<math>\pm</math>1.83</b>	30.17 $\pm$ 0.17	83.62 $\pm$ 1.61	71.45 $\pm$ 2.41	83.44 $\pm$ 0.92	66.89	60.59	79.50
HGRL	79.46 $\pm$ 4.45	82.16 $\pm$ 6.00	86.28 $\pm$ 3.58	48.29 $\pm$ 1.64	35.79 $\pm$ 0.89	<u>36.97<math>\pm</math>0.98</u>	80.66 $\pm$ 0.43	68.56 $\pm$ 1.10	80.35 $\pm$ 0.58	66.50	61.49	76.52
GREET	<b>85.14<math>\pm</math>4.87</b>	<u>87.03<math>\pm</math>2.36</u>	84.90 $\pm$ 4.48	63.64 $\pm$ 1.26	42.29 $\pm$ 1.43	36.55 $\pm$ 1.01	83.81 $\pm$ 0.87	<u>73.08<math>\pm</math>0.84</u>	80.29 $\pm$ 1.00	70.75	66.59	79.06
MUSE	<u>82.16<math>\pm</math>3.42</u>	<b>89.73<math>\pm</math>2.79</b>	<b>88.24<math>\pm</math>3.20</b>	<b>72.37<math>\pm</math>2.21</b>	<u>54.19<math>\pm</math>3.04</u>	<b>38.55<math>\pm</math>1.34</b>	82.24 $\pm$ 0.41	71.14 $\pm$ 0.40	82.90 $\pm$ 0.59	<b>73.39</b>	<b>70.87</b>	78.76

**Table 3: Results of node clustering (in percent  $\pm$  standard deviation). The best and runner-up results are highlighted with bold and underline, respectively.**

Methods	Texas			Actor			Cornell			CiteSeer		
	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI
AE	50.49 $\pm$ 0.01	16.63 $\pm$ 0.01	14.60 $\pm$ 0.01	24.19 $\pm$ 0.11	0.97 $\pm$ 0.03	0.50 $\pm$ 0.04	52.19 $\pm$ 0.01	17.08 $\pm$ 0.01	17.41 $\pm$ 0.01	58.79 $\pm$ 0.19	30.91 $\pm$ 0.21	30.29 $\pm$ 0.23
node2vec	48.80 $\pm$ 1.93	2.58 $\pm$ 0.70	-1.62 $\pm$ 0.65	25.02 $\pm$ 0.04	0.09 $\pm$ 0.01	0.06 $\pm$ 0.02	50.98 $\pm$ 0.01	5.84 $\pm$ 0.01	0.18 $\pm$ 0.01	20.76 $\pm$ 0.27	0.35 $\pm$ 0.03	-0.01 $\pm$ 0.04
struc2vec	49.73 $\pm$ 0.01	18.61 $\pm$ 0.01	20.97 $\pm$ 0.01	22.49 $\pm$ 0.34	0.04 $\pm$ 0.01	-0.05 $\pm$ 0.05	32.68 $\pm$ 0.01	1.54 $\pm$ 0.01	-2.20 $\pm$ 0.01	21.22 $\pm$ 0.45	1.18 $\pm$ 0.08	0.17 $\pm$ 0.06
LINE	49.40 $\pm$ 2.08	16.90 $\pm$ 1.57	18.08 $\pm$ 1.06	22.70 $\pm$ 0.08	0.09 $\pm$ 0.01	0.11 $\pm$ 0.01	34.10 $\pm$ 0.77	2.85 $\pm$ 0.21	-1.54 $\pm$ 0.25	28.42 $\pm$ 0.88	8.49 $\pm$ 0.74	3.54 $\pm$ 0.56
GAE	42.02 $\pm$ 1.22	8.49 $\pm$ 1.31	10.83 $\pm$ 1.92	23.45 $\pm$ 0.04	0.18 $\pm$ 0.01	-0.04 $\pm$ 0.01	43.72 $\pm$ 1.25	5.11 $\pm$ 0.38	6.51 $\pm$ 1.74	48.37 $\pm$ 0.37	24.59 $\pm$ 0.22	19.50 $\pm$ 0.31
VGAE	50.27 $\pm$ 1.87	11.73 $\pm$ 0.95	21.51 $\pm$ 1.81	23.30 $\pm$ 0.22	0.21 $\pm$ 0.03	0.34 $\pm$ 0.05	43.39 $\pm$ 0.99	5.46 $\pm$ 0.46	3.97 $\pm$ 0.49	55.67 $\pm$ 0.13	32.45 $\pm$ 0.10	28.34 $\pm$ 0.13
GraphSAGE	56.83 $\pm$ 0.56	16.97 $\pm$ 1.93	23.50 $\pm$ 2.98	23.08 $\pm$ 0.29	0.58 $\pm$ 0.14	0.22 $\pm$ 0.07	44.70 $\pm$ 2.00	4.33 $\pm$ 0.93	5.64 $\pm$ 1.33	49.28 $\pm$ 1.18	22.97 $\pm$ 0.80	19.21 $\pm$ 1.33
SDCN	44.04 $\pm$ 0.56	14.24 $\pm$ 1.93	10.65 $\pm$ 2.98	23.67 $\pm$ 0.29	0.08 $\pm$ 0.14	-0.01 $\pm$ 0.07	36.94 $\pm$ 2.00	6.6 $\pm$ 0.93	3.38 $\pm$ 1.33	59.86 $\pm$ 1.18	30.37 $\pm$ 0.80	29.70 $\pm$ 1.33
MVGRL	<u>62.79<math>\pm</math>2.33</u>	25.66 $\pm$ 1.81	33.54 $\pm$ 4.59	28.58 $\pm$ 1.03	2.42 $\pm$ 0.51	2.80 $\pm$ 0.57	43.77 $\pm$ 3.03	8.38 $\pm$ 2.82	7.09 $\pm$ 2.95	45.67 $\pm$ 9.08	23.41 $\pm$ 7.73	19.92 $\pm$ 7.92
GRACE	56.99 $\pm$ 2.23	20.65 $\pm$ 1.02	29.54 $\pm$ 4.22	25.87 $\pm$ 0.45	0.56 $\pm$ 0.28	0.93 $\pm$ 0.38	43.55 $\pm$ 4.60	8.23 $\pm$ 1.16	6.43 $\pm$ 1.97	54.66 $\pm$ 5.41	31.70 $\pm$ 3.78	27.40 $\pm$ 5.60
BGRL	58.68 $\pm$ 1.80	21.95 $\pm$ 2.40	23.74 $\pm$ 2.29	28.20 $\pm$ 0.27	1.84 $\pm$ 0.19	2.35 $\pm$ 0.12	55.08 $\pm$ 1.68	7.98 $\pm$ 0.53	3.92 $\pm$ 1.03	<u>64.27<math>\pm</math>1.68</u>	<u>36.63<math>\pm</math>1.71</u>	<u>36.71<math>\pm</math>1.85</u>
DSSL	57.43 $\pm$ 3.51	18.60 $\pm$ 2.25	25.68 $\pm$ 3.73	26.15 $\pm$ 0.46	0.76 $\pm$ 0.10	1.27 $\pm$ 0.17	44.70 $\pm$ 2.44	7.14 $\pm$ 1.81	7.11 $\pm$ 2.86	54.32 $\pm$ 3.69	28.67 $\pm$ 2.73	26.59 $\pm$ 3.54
HGRL	61.97 $\pm$ 3.10	44.58 $\pm$ 2.14	37.05 $\pm$ 4.78	29.79 $\pm$ 1.11	3.80 $\pm$ 0.83	4.09 $\pm$ 1.16	60.56 $\pm$ 3.72	<u>44.61<math>\pm</math>3.32</u>	<u>35.65<math>\pm</math>5.24</u>	61.14 $\pm$ 1.49	34.06 $\pm$ 1.71	33.65 $\pm$ 2.10
MUSE	<b>74.86<math>\pm</math>2.79</b>	<b>45.53<math>\pm</math>4.22</b>	<b>49.80<math>\pm</math>5.71</b>	<b>32.43<math>\pm</math>0.60</b>	<b>8.57<math>\pm</math>0.60</b>	<b>6.31<math>\pm</math>0.23</b>	<b>71.59<math>\pm</math>1.82</b>	<b>46.49<math>\pm</math>2.36</b>	<b>49.53<math>\pm</math>2.43</b>	<b>69.11<math>\pm</math>1.19</b>	<b>43.50<math>\pm</math>0.84</b>	<b>44.05<math>\pm</math>1.17</b>

other hand, the basic GCN model performs relatively well on the other two heterophilic graphs (Chameleon and Squirrel), suggesting that nodes in these two graphs require more contextual information from neighboring nodes. This further shows the presence of global diversity among different graphs.

(2) MUSE outperforms traditional supervised models (like GCN, GAT and MLP) and traditional self-supervised GNN models (like DGI, GMI, MVGRL, BGRL and GRACE). This is because these methods are designed for homophilic graphs without considering the heterophily in the graphs.

(3) Existing unsupervised methods designed for heterophilic graphs exhibit a significant difference in performance across different datasets. DSSL and NWR-GAE perform even worse than MLP on four of the six heterophilic datasets (Cornell, Texas, Wisconsin and Actor), although performing well on the other two heterophilic

datasets. HGRL is the runner-up on Actor, but the accuracy on Chameleon is only 0.4829, while the best result is 0.7237 (MUSE). GREET ranks first on Cornell, but the accuracy score on Chameleon is 0.6364. We ascribe this instability in performance to the neglect of node diversity in these methods, which adopt a uniform processing strategy for all nodes and thus compromise the expressive power of the learned node representation. MUSE determines the amount of contextual information that each ego node needs in a node-specific way and shows superiority consistently on all six heterophilic datasets, ranking first on four of the datasets, and second on the other two.

(4) MUSE outperforms all the baselines in terms of average performance over all the datasets. On heterophilic graphs, the average performance of MUSE was higher than the runner-up which is a



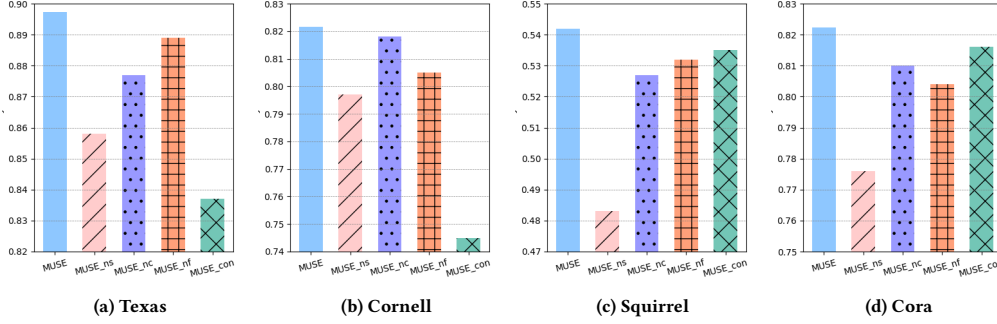


Figure 3: Ablation study

supervised method designed for heterophilic graphs, while there is a slight tradeoff in performance on homophilic graphs.

**5.2.2 Node Clustering Performance.** The node clustering results on three heterophilic datasets and one homophilic dataset are reported in Table 3. From Table 3, we have the following observation:

It is notable that our model outperforms all the competitors on both heterophilic and homophilic datasets. We achieve relative improvement up to 19.22% (ACC) on Texas than MVGRL. The improvement of ACC indicates that the node representation learned by our model can be correctly assigned to their respective clusters in the K-means clustering model. Relative improvement up to 125.53% (NMI) on Actor than HGRL reflects a high degree of similarity between the clustering results of the node representation and true labels. Moreover, the 54.28% (ARI) relative improvement on Actor than HGRL demonstrates excellent performance of our model in the clustering task even when considering the penalty for random clustering.

### 5.3 Ablation Study

In order to examine the effectiveness of each component of MUSE, we conduct ablation experiments on various variants of the model. We primarily validated the effectiveness of our model from two perspectives: (1) Effectiveness of the components in the node representation learning module, and (2) Effectiveness of the information fusion controller.

**5.3.1 Effectiveness of the Components in the Node Representation Learning Module.** The major components in the node representation learning module include: semantic contrast, contextual contrast, and fusion contrast. To show the importance of each component, we design model variants by removing different contrasts from our model. Specifically, we remove  $\mathcal{L}_s$  from Eq. 13 and call the variant **MUSE\_ns** (no semantic contrast); we remove  $\mathcal{L}_c$  and call the variant **MUSE\_nc** (no context contrast); we call the variant removing  $\mathcal{L}_r$  as **MUSE\_nf** (no fusion contrast). We conduct experiments on three heterophilic graphs (e.g., Texas, Cornell and Squirrel) and one homophilic graph (e.g., Cora), the results of node classification task are shown in Figure 3. From the figure, we can see that:

(1) MUSE outperforms MUSE\_ns and MUSE\_nc on these four datasets. This shows that both semantic contrast and contextual

contrast in the node representation learning module play a vital role. Further, the advantage of MUSE over MUSE\_nf shows that cross-view fusion contrast is also a necessity in enhancing the effectiveness of node representation learning.

(2) Compared with MUSE\_nc and MUSE\_nf, MUSE leads to a much larger performance gap than MUSE\_ns, especially on Squirrel and Cora. This highlights the importance of ego node’s features for node representations in both homophilic and heterophilic graphs.

**5.3.2 Effectiveness of the Information Fusion Controller.** The information fusion controller takes into account the difference in the local diversity of node-neighborhood similarity across different nodes and the global distribution of similarity in the whole graph. With the information fusion controller, we can combine the semantic and contextual information at the node level to leverage the diversity of nodes. To validate its effectiveness, we set the value of  $\lambda$  to 1 for each node and we call this variant as **MUSE\_con(controller)**. This variant takes the same amount of contextual information from the neighborhood for all nodes without considering the diversity among nodes. Figure 3 clearly indicates a decrease in performance of MUSE\_con compared with MUSE, especially on Texas and Cornell, suggesting the existence of diversity among nodes and the necessity of effectively combining semantic information and contextual information at the node level. Therefore, the information fusion controller has demonstrated a highly significant impact.

### 5.4 Parameter Analysis

In this section, we conduct experiments to investigate the impact of parameters in our model, including the pre-set hyper-parameter  $\epsilon$  and the weight factors of contrastive loss  $\beta_1, \beta_2$ .

**5.4.1 Analysis of Weight Factor  $\beta_1$  and  $\beta_2$ .** We study the sensitivity of our model with respect to the weight factors  $\beta_1$  and  $\beta_2$  in Equation 13 by varying the values from  $10^{-3}$  to  $10^2$  (in Figure 4a and 4b). A common phenomenon is that a too large  $\beta_1$  or  $\beta_2$  leads to an obvious performance degradation, which indicates that excessive information from neighbors may have an opposite effect. Specifically, the best choice of  $\beta_1$  and  $\beta_2$  for Texas and Cornell is 0.01 and 0.1, as the features of ego node plays a more important role in this two datasets. The best choice of  $\beta_1$  and  $\beta_2$  for Squirrel



is 1, indicating the importance of contextual information and fused information.

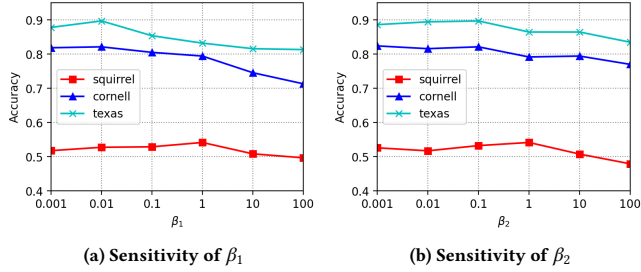


Figure 4: Sensitivity analysis on weight factor  $\beta_1, \beta_2$

**5.4.2 Analysis of hyper-parameter  $\epsilon$ .** To study the impact of hyper-parameter  $\epsilon$  on our model, we set  $\epsilon$  from 0 to 1 and show the node classification accuracy in Figure 5. We observe that the node classification accuracy is highly sensitive to the pre-set hyper-parameter on Squirrel and Chameleon. This suggests that our restriction on the global diversity of nodes in the graph is necessary. Specifically, performance on this two datasets is better with larger  $\epsilon$ , indicating that nodes in the graphs require more contextual information provided by neighboring nodes.

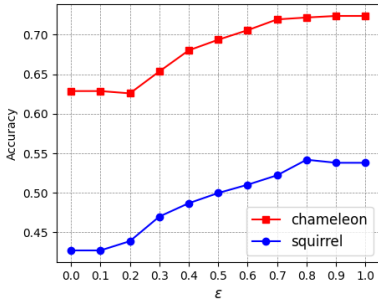


Figure 5: Sensitivity analysis on hyper-parameter  $\epsilon$

## 5.5 Generalizability

As previously discussed, labeled data is usually scarce in real-life scenarios. Therefore, we propose a self-supervised model on heterophilic graphs to alleviate the dependency on labels. By pre-training, we can obtain task-agnostic node representations and evaluate them on downstream tasks with a small quantity of labels, aiming to achieve superior performance. In this section, we further investigate the effectiveness of the frozen representations obtained by pre-training with fewer downstream labels for fine-tuning.

Except the commonly used training/validation/test partition of 48/32/20 adopted by most baselines on heterophilic graphs as in Section 5.2, we further follow the experimental settings in HGRL [5] and alter the data partition to 10/10/80. The performance on node classification task is given in Table 4, and the results of baselines

are derived from [5]. From the table, we can observe that, MUSE outperforms other competitors on Squirrel and Chameleon, and is the runner-up on Cora. This shows that MUSE still achieves impressive results on the downstream task despite the decrease in labeled data, particularly on heterophilic graphs, outperforming other unsupervised learning methods significantly. This further verifies that MUSE has strong generalization ability.

Table 4: Results of node classification task with few labels

Data	Methods	Squirrel	Chameleon	Cora
X,A,Y	GCN	39.50±1.54	54.65±2.17	82.26±1.20
X,A,Y	H2GCN	41.18±0.81	54.02±1.56	81.38±1.16
X, A	MVGRL	33.49±0.84	42.34±2.11	<b>84.53±1.05</b>
X, A	GRACE	34.47±1.11	45.89±3.10	83.69±0.73
X, A	BGRL	31.50±0.57	45.54±1.94	83.01±0.71
X, A	HGRL	35.42±0.91	45.04±1.91	82.08±0.84
X, A	MUSE	<b>41.67±0.90</b>	<b>57.89±1.27</b>	<u>84.11±0.73</u>

## 5.6 Efficiency

In terms of efficiency, we evaluate the performance of our model against four state-of-the-art self-supervised learning baselines. Specifically, we focus on the training time during the representation pre-training stage and conduct experiment on Squirrel dataset to report the time cost for each epoch. The experimental results are shown in Table 5. It can be observed that MUSE requires significantly less training time per epoch compared to other three methods DSSL, NWR-GAE and GREET that also use GNN as the encoder. While HGRL is more efficient, it uses MLP as the encoder and performs worse than MUSE in all the node classification/clustering comparisons as shown in Tables 2 and 3. All these results show that our model is both effective and efficient.

Table 5: Comparison of efficiency on Squirrel

Methods	DSSL	NWR-GAE	HGRL	GREET	MUSE
Time	1.848s	1.104s	<b>0.11s</b>	1.136s	<u>0.255s</u>

## 6 CONCLUSION

In this paper, we propose a novel model named MUSE for unsupervised node representation learning on heterophilic graphs. We employ GNNs enhanced with contrastive learning to capture the information from semantic and contextual views and fuse the node representations with an information fusion controller. Fusion contrast is utilized to enhance the effectiveness of the fused node representations. The information fusion controller considers both the local diversity of similarity across different ego nodes and the global distribution of similarity in the whole graph. We train the two components with an alternating strategy to boost each other. Extensive experiments reveal the effectiveness and generalizability of our method.

## REFERENCES

- [1] Sami Abu-El-Hajja, Bryan Perozzi, Amol Kapoor, Nazanin Alipourfard, Kristina Lerman, Hrayr Harutyunyan, Greg Ver Steeg, and Aram Galstyan. 2019. Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *international conference on machine learning*. PMLR, 21–29.
- [2] Yoshua Bengio. 2012. Deep learning of representations for unsupervised and transfer learning. In *Proceedings of ICML workshop on unsupervised and transfer learning*. JMLR Workshop and Conference Proceedings, 17–36.
- [3] Deyu Bo, Xiao Wang, Chuan Shi, and Huawei Shen. 2021. Beyond low-frequency information in graph convolutional networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 3950–3957.
- [4] 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*. 1400–1410.
- [5] Jingfan Chen, Guanghui Zhu, Yifan Qi, Chunfeng Yuan, and Yihua Huang. 2022. Towards Self-supervised Learning on Graphs with Heterophily. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 201–211.
- [6] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. 2020. Simple and deep graph convolutional networks. In *International conference on machine learning*. PMLR, 1725–1735.
- [7] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*. PMLR, 1597–1607.
- [8] Johannes Gasteiger, Aleksandar Bojchevski, and Stephan Günnemann. 2018. Predict then propagate: Graph neural networks meet personalized pagerank. *arXiv preprint arXiv:1810.05997* (2018).
- [9] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. 855–864.
- [10] Will Hamilton, Zitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems* 30 (2017).
- [11] Arman Hasanzadeh, Ehsan Hajiramezani, Krishna Narayanan, Nick Duffield, Mingyuan Zhou, and Xiaoning Qian. 2019. Semi-implicit graph variational auto-encoders. *Advances in neural information processing systems* 32 (2019).
- [12] Kaveh Hassani and Amir Hosein Khasahmadi. 2020. Contrastive multi-view representation learning on graphs. In *International conference on machine learning*. PMLR, 4116–4126.
- [13] Geoffrey E Hinton and Ruslan R Salakhutdinov. 2006. Reducing the dimensionality of data with neural networks. *science* 313, 5786 (2006), 504–507.
- [14] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
- [15] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).
- [16] Thomas N Kipf and Max Welling. 2016. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308* (2016).
- [17] Xiang Li, Renyu Zhu, Yao Cheng, Caihua Shan, Siqiang Luo, Dongsheng Li, and Weining Qian. 2022. Finding global homophily in graph neural networks when meeting heterophily. In *International Conference on Machine Learning*. PMLR, 13242–13256.
- [18] Meng Liu, Zhengyang Wang, and Shuiwang Ji. 2021. Non-local graph neural networks. *IEEE transactions on pattern analysis and machine intelligence* 44, 12 (2021), 10270–10276.
- [19] Yixin Liu, Yizhen Zheng, Daokun Zhang, Vincent Lee, and Shirui Pan. 2022. Beyond Smoothing: Unsupervised Graph Representation Learning with Edge Heterophily Discriminating. *arXiv preprint arXiv:2211.14065* (2022).
- [20] Sitao Luan, Chenqing Hua, Qincheng Lu, Jiaqi Zhu, Mingde Zhao, Shuyuan Zhang, Xiao-Wen Chang, and Doina Precup. 2021. Is Heterophily A Real Nightmare For Graph Neural Networks To Do Node Classification? *arXiv preprint arXiv:2109.05641* (2021).
- [21] Yao Ma, Xiaorui Liu, Neil Shah, and Jiliang Tang. 2021. Is homophily a necessity for graph neural networks? *arXiv preprint arXiv:2106.06134* (2021).
- [22] Jiwoong Park, Minsik Lee, Hyung Jin Chang, Kyuewang Lee, and Jin Young Choi. 2019. Symmetric graph convolutional autoencoder for unsupervised graph representation learning. In *Proceedings of the IEEE/CVF international conference on computer vision*. 6519–6528.
- [23] Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. 2020. Geom-gcn: Geometric graph convolutional networks. *arXiv preprint arXiv:2002.05287* (2020).
- [24] 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*. 259–270.
- [25] 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.
- [26] Leonardo FR Ribeiro, Pedro HP Saverese, and Daniel R Figueiredo. 2017. struc2vec: Learning node representations from structural identity. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*. 385–394.
- [27] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. 2008. Collective classification in network data. *AI magazine* 29, 3 (2008), 93–93.
- [28] Fan-Yun Sun, Jordan Hoffmann, Vikas Verma, and Jian Tang. 2019. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. *arXiv preprint arXiv:1908.01000* (2019).
- [29] Susheel Suresh, Vinith Budde, Jennifer Neville, Pan Li, and Jianzhu Ma. 2021. Breaking the limit of graph neural networks by improving the assortativity of graphs with local mixing patterns. *arXiv preprint arXiv:2106.06586* (2021).
- [30] 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.
- [31] Mingyue Tang, Carl Yang, and Pan Li. 2022. Graph auto-encoder via neighborhood wasserstein reconstruction. *arXiv preprint arXiv:2202.09025* (2022).
- [32] Shantanu Thakoor, Corentin Tallec, Mohammad Gheshlaghi Azar, Mehdi Azabou, Eva L Dyer, Remi Munos, Petar Velicković, and Michal Valko. 2021. Large-scale representation learning on graphs via bootstrapping. *arXiv preprint arXiv:2102.06514* (2021).
- [33] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).
- [34] Petar Veličković, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2019. Deep graph infomax. *ICLR (Poster)* 2, 3 (2019), 4.
- [35] 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).
- [36] Chun Wang, Shirui Pan, Guodong Long, Xingquan Zhu, and Jing Jiang. 2017. Mgae: Marginalized graph autoencoder for graph clustering. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 889–898.
- [37] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying graph convolutional networks. In *International conference on machine learning*. PMLR, 6861–6871.
- [38] Teng Xiao, Zhengyu Chen, Zhimeng Guo, Zeyang Zhuang, and Suhang Wang. 2022. Decoupled Self-supervised Learning for Graphs. *Advances in Neural Information Processing Systems* 35 (2022), 620–634.
- [39] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. 2018. Representation learning on graphs with jumping knowledge networks. In *International conference on machine learning*. PMLR, 5453–5462.
- [40] Yujun Yan, Milad Hashemi, Kevin Swersky, Yaoqing Yang, and Danai Koutra. 2021. Two sides of the same coin: Heterophily and oversmoothing in graph convolutional neural networks. *arXiv preprint arXiv:2102.06462* (2021).
- [41] Zhilin Yang, William Cohen, and Ruslan Salakhutdinov. 2016. Revisiting semi-supervised learning with graph embeddings. In *International conference on machine learning*. PMLR, 40–48.
- [42] 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.
- [43] Xiaotong Zhang, Han Liu, Qimai Li, and Xiao-Ming Wu. 2019. Attributed graph clustering via adaptive graph convolution. *arXiv preprint arXiv:1906.01210* (2019).
- [44] Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. 2020. Beyond homophily in graph neural networks: Current limitations and effective designs. *Advances in Neural Information Processing Systems* 33 (2020), 7793–7804.
- [45] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. 2020. Deep graph contrastive representation learning. *arXiv preprint arXiv:2006.04131* (2020).