HOGDA: Boosting Semi-supervised Graph Domain Adaptation via High-Order Structure-Guided Adaptive Feature Alignment

Anonymous Authors

ABSTRACT

Semi-supervised graph domain adaptation, as a subfield of graph transfer learning, seeks to precisely annotate unlabeled target graph nodes by leveraging transferable features acquired from the limited labeled source nodes. However, most existing studies often directly utilize GCNs-based feature extractors to capture domain-invariant node features, while neglecting the issue that GCNs are insufficient in collecting complex structure information in graph. Considering the importance of graph structure information in encoding the complex relationship among nodes and edges, this paper aims to utilize such powerful information to assist graph transfer learning. To achieve this goal, we develop an novel framework called HOGDA. Concretely, HOGDA introduces a high-order structure information mixing module to effectively assist the feature extractor in capturing transferable node features. Moreover, to achieve fine-grained feature distributions alignment, the AWDA strategy is proposed to dynamically adjust the node weight during adversarial domain adaptation process, effectively boosting the model's transfer ability. Furthermore, to mitigate the overfitting phenomenon caused by limited source labeled nodes, we also design a TNC strategy to guide the unlabeled nodes to achieve discriminative clustering. Extensive experimental results show that our HOGDA outperforms the state-of-the-art methods on various transfer tasks.

KEYWORDS

Graph Transfer Learning, Adversarial Domain Adaptation, High-Order Moment, Node Clustering

1 INTRODUCTION

In multimodal applications, graphs are often used to model the correlation between different modal data. Among them, graph node classification techniques play a crucial role in analyzing nodes from different modalities. However, due to the distribution shift problem, well-trained models suffer severe performance degradation when applied straight to new domains, limiting the large-scale implementation of deep models in actual applications. Graph transfer learning (GTL) [6, 24] has been proposed as a paradigm to address such a problem by transferring some invariant features from a labeled source graph to an unlabeled target graph, greatly improving the model's generalization ability on the target graph.

ACM MM, 2024, Melbourne, Australia

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

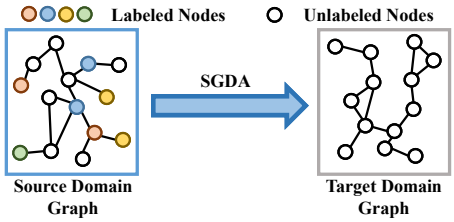


Figure 1: Illustration of the semi-supervised graph domain adaptation (SGDA) task.

Most existing studies [7, 18, 28] on GTL tends to focus on unsupervised domain adaptation, assuming that all nodes in the source graph are labeled, while overlooking the fact that this ideal scenario is not common in real-world applications, as annotating the entire source graph is a time-consuming task, especially for large-scale graphs. Therefore, in this paper, we focus on a more practical application scenario known as semi-supervised graph domain adaptation (SGDA) [24], where the source graph contains only a limited number of labeled nodes, as shown in Figure 1. The most crucial challenge for SGDA is to effectively leverage the transferable features learned from the label-scarce source graph to precisely annotate nodes in the target graph.

Unlike images and time series data, graph data usually contains rich structure information that encodes complex relationships among nodes and edges. Most existing GTL models [6, 24, 36] usually adopt graph convolutional network (GCN)-based feature extractors to learn domain-invariant node features. However, recent studies [5, 21, 40] have demonstrated that GCNs are insufficient in capturing the sophisticated structure information in graph, which may potentially affect the transfer of domain-invariant knowledge and consequently limit the model's generalization capability.

To address this problem, inspired by the effectiveness of high-order moment features in characterizing the data structure [9], we propose an novel SGDA framework named **HOGDA** that employs a High-order Structure Information Mixing (**HSIM**) module to effectively capture graph structure information. In order to further explain our motivation, we plot a point cloud (sampled from three Gaussian distributions) and visualize the moment features of different orders in Figure 2. As can be seen, the structure of the point cloud can be captured more accurately by using high-order moment features than by using low-order features. To this end, HSIM module seek to employ multi-view structure information to assist the feature extractor in extracting more discriminative domain-invariant node features.

Recent studies [4, 19, 42] on TL have demonstrated that different samples have different levels of transferability. However, many existing GTL methods usually assign equal weight to different nodes during adversarial domain adaptation process, ignoring the fact that hard-to-transfer nodes may harm the learning of transferable node

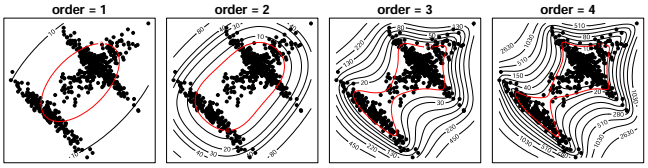


Figure 2: 240 data points sampled from three Gaussian distributions. And the level sets denotes the moment features with different orders. In comparison to low-order features, high-order moment features can more accurately capture the underlying structure of the point cloud.

features and even cause negative transfer. To remedy this issue, we propose a Adaptive Weighted Domain Alignment (**AWDA**) strategy. Specifically, AWDA adaptively estimates the node transferability by jointly leveraging entropy information from the classifier and discriminator, and dynamically adjusts the weight of node based on its transferability. By prioritizing easy-to-transfer nodes with higher weights during domain alignment process, the model can learn more domain-invariant features, thereby significantly improving its transfer ability on the target domain.

Furthermore, in the SGDA scenarios, due to the label scarcity of source graph, well-trained model on only a few labeled source nodes is prone to overfitting. Consequently, it may make ambiguous or even incorrect predictions for certain target graph nodes located near the decision boundaries or far from their corresponding class centers. To alleviate this overfitting phenomenon, we devise a Trust-aware Node Clustering (**TNC**) strategy to enhance the model's generalization performance. Specially, TNC aims to guide the discriminative clustering of unlabeled nodes by minimizing the discrepancy between the current cluster assignment distribution and the ideal cluster distribution, effectively promoting the alignment of category distributions across domains.

The following are the primary contributions of this paper:
(1) A novel HSIM module is devised to assist the feature extractor in capturing the graph structure information.
(2) A node re-weighted adversarial adaption strategy named AWDA is proposed to facilitate the alignment of feature distributions.
(3) To remedy overfitting issue, a simple but effective TNC strategy is introduced to guide the clustering of unlabeled nodes.
(4) Experimental results on various transfer tasks demonstrate the superiority of our HOGDA over the state-of-the-art methods.

2 RELATED WORKS

Graph Transfer Learning (GTL). GTL has attracted substantial attention as a promising solution for effectively alleviating the burden of collecting labeled data for novel tasks. A series of early studies commonly utilize available source labeled nodes to build a pre-train graph model for different but related tasks in the target domain [13, 23, 26]. However, due to the presence of distribution shift, this training paradigm inevitably causes the model to suffer severe performance degradation in the target domain. To solve this problem, recent studies have shifted their focus towards domain adaptation [7, 18, 28]. These works aims to boost the model's generalization ability by transferring some invariant features from a label-rich source domain to a label-scarce target domain. Methods for achieving domain adaptation can be roughly divided into two categories: (1) Extracting transferable features by minimizing the statistical metrics between two domains [10, 29]; (2) Leveraging adversarial training to enforce domain confusion to capture domain-invariant features [6, 24, 28, 36, 41].

Semi-supervised Learning on Graphs. Semi-supervised learning on graphs aims to tackles the node classification task by utilizing only a small fraction of labeled nodes. Early works, such as GCN [16], GraphSAGE [12] and GAT [34], typically employ the message passing paradigm to capture discriminative node features [12, 16, 34]. In recent studies, researchers have investigated a variety of techniques, such as adversarial training [14, 38], data augmentation [35], continuous graph [37], and meta-learning [25] to further boost the model's generalization performance.

3 METHODOLOGY

3.1 Problem Formulation

Source Domain Graph: Let $\mathcal{G}^s = (\mathcal{V}^{s,l}, \mathcal{V}^{s,u}, A^s, X^s, Y^{s,l})$ be the source graph, where $\mathcal{V}^{s,l}$ denotes the labeled node set, and $\mathcal{V}^{s,u}$ denotes the remaining unlabeled node set in $\mathcal{G}_s$. The adjacency matrix $A^s \in \mathbb{R}^{N^s \times N^s}$ represents the connectivity of nodes in $\mathcal{G}^s$, where $N^s = |\mathcal{V}^{s,l}| + |\mathcal{V}^{s,u}|$ denotes the total number of nodes. If there exists an edge between nodes n_i and n_j, the corresponding element A^s_{ij} is assigned a value of 1; otherwise, it is set to 0. $Y^{s,l} \in \mathbb{R}^{|\mathcal{V}^{s,l}| \times C}$ indicates the label matrix of $\mathcal{V}^{s,l}$, where C is the number of node classes. If a node $n^s_i \in \mathcal{V}^{s,l}$ belongs to the c-th class, $y^s_{i,c} = 1$; otherwise, $y^s_{i,c} = 0$. $X^s \in \mathbb{R}^{N^s \times e}$ represents an attribute matrix, where e is the dimension of node attributes. In the SGDA setting, $|\mathcal{V}^{s,l}|$ is much smaller than $|\mathcal{V}^{s,u}|$.

Target Domain Graph: The target graph, denoted as $\mathcal{G}^t = (\mathcal{V}^t, A^t, X^t)$, is a completely unlabeled graph with an unlabeled node set $\mathcal{V}^t$. Similarly, the adjacency matrix $A^t \in \mathbb{R}^{N^t \times N^t}$ indicates the connections between nodes in $\mathcal{G}_t$, and the node attribute matrix $X^t \in \mathbb{R}^{N^t \times e}$ stores the attribute information for each target node. Here, $N^t = |\mathcal{V}^t|$ denotes the number of nodes in $\mathcal{G}_t$.

Semi-Supervised Graph Domain Adaptation (SGDA): Given a partially labeled source graph $\mathcal{G}^s$ and an unlabeled target graph $\mathcal{G}^t$, the key challenge in SGDA is how to accurately annotate target graph nodes by leveraging the transferable knowledge learned from the limited source labeled nodes.

3.2 Network Architecture

The architecture of our HOGDA model is composed of four components: a GCN-based feature extractor $\mathcal{F}$, a high-order structure information mixing (**HSIM**) module $\mathcal{H}$, a domain discriminator $\mathcal{D}$, and a node classifier C, as shown in Figure 3.

For brevity, we omit the domain-specific notation to describe the data flow through our model. Mathematically, given an input graph $\mathcal{G} = (\mathcal{V}, A, X)$, the node features extracted by $\mathcal{F}$ is denoted as $Z = \mathcal{F}(\mathcal{G}) \in \mathbb{R}^{|\mathcal{V}| \times e}$, and it is subsequently fed into the HSIM module to capture the corresponding high-order structure features $H = \mathcal{H}(Z) \in \mathbb{R}^{|\mathcal{V}| \times e}$, where e is the feature dimension and $|\mathcal{V}|$ denotes the number of nodes in $\mathcal{G}$. Then, the node features Z are concatenated with their corresponding structure features H and

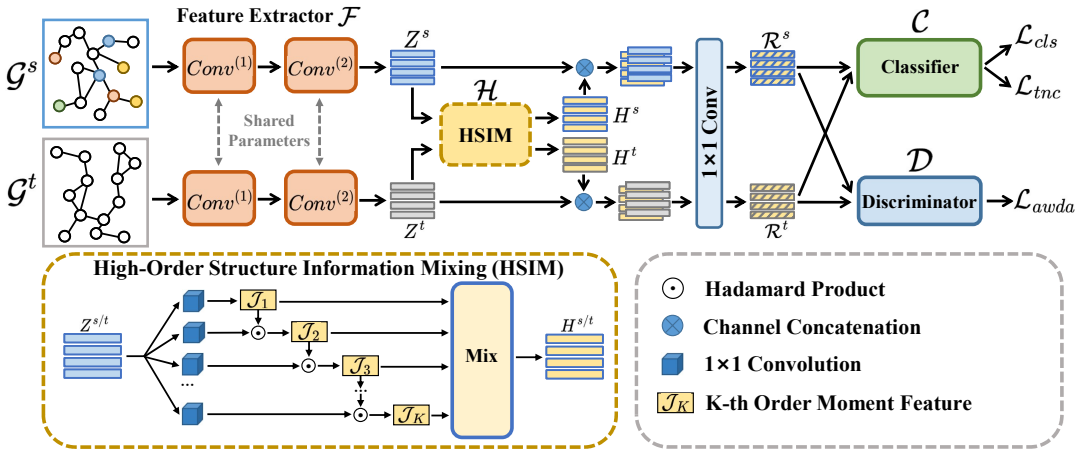


Figure 3: Global overview of the proposed HOGDA model.

passed through a learnable 1×1 convolution filter $\mathcal{W}^{1\times1}$ to obtain the mixed node features $\mathcal{R} \in \mathbb{R}^{|\mathcal{V}|\times e}$. These mixed features are then passed into the classifier C for the final classification prediction $C(\mathcal{R}) \in \mathbb{R}^{|\mathcal{V}|\times C}$. The domain discriminator $\mathcal{D}$ is trained to distinguish between the source and target domains, while the feature extractor $\mathcal{F}$ is optimized to confuse $\mathcal{D}$ in order to capture domain-invariant node features Z.

To better model the adjacency relationships among nodes in graph $\mathcal{G}$, we calculate the positive point-wise mutual information (PPIM) between nodes following [24, 36]. Concretely, for a given graph $\mathcal{G} = (\mathcal{V}, A, X)$, we employ random walk to sample a set of paths on A and construct a frequency matrix Ψ. Here, each entry Ψ_{ij} represents the occurrence count of node n_j within a predefined window in the context of node n_i. Then the PPIM matrix $\mathbb{P}$ is computed as:

$$\mathbb{P}_{ij} = \frac{\Psi_{ij}}{\sum_{i,j}\Psi_{ij}},\ \mathbb{P}_{i,*} = \frac{\sum_j \Psi_{ij}}{\sum_{i,j}\Psi_{ij}},\ \mathbb{P}_{*,j} = \frac{\sum_i \Psi_{ij}}{\sum_{i,j}\Psi_{ij}},$$
$$P_{ij} = \max\{\log(\frac{\mathbb{P}_{ij}}{\mathbb{P}_{i,*}\times\mathbb{P}_{*,j}}), 0\}, \tag{1}$$

where P_{ij} denotes the positive mutual information between nodes n_i and n_j, which quantifies the topological proximity between nodes. A higher value of P_{ij} indicates a strong connection between n_i and n_j, while a value of $P_{ij} = 0$ indicates the absence of such a connection. Then, the output of the l-th GCN layer $Conv^{(l)}(\cdot)$ is denoted as:

$$Z^{(l)} = Conv^{(l)}(P, Z^{(l-1)}) = \sigma(D^{-\frac{1}{2}}\widetilde{P}D^{-\frac{1}{2}}Z^{(l-1)}W^{(l)}), \tag{2}$$

where $\sigma(\cdot)$ is an activation function, and D is the diagonal degree matrix of P (i.e., $D_{ii} = \sum_j \widetilde{P}_{ij}$). Moreover, $\widetilde{P} = P + I$, where I is an identity matrix. $W^{(l)}$ denotes the learnable parameters of the l-th layer, and $Z^{(0)} = X$. Note that the feature extractor $\mathcal{F}$ consists of sequentially stacked L layers of GCN $Conv^{(l)}(l = 1, 2, \cdots, L)$.

3.3 High-order Structure Information Mixing

As mentioned in Section 1, in contrast to images and time series data, graph data (*e.g.*, social network and academic network) typically encompasses abundant structure information that encodes intricate relationships among nodes and edges. However, most existing GTL models [6, 24, 36] typically employ GCN-based feature extractors to learn domain-invariant node features, neglecting the issue that GCNs are insufficient in collecting complex structure information in graph [5, 21, 40]. This limitation may potentially affect the transfer of domain-invariant knowledge and consequently limit the model's generalization capability.

To address this issue, motivated by the effectiveness of high-order moment information in capturing the structure of data (as shown in Figure 2), we introduce a high-order structure information mixing (**HSIM**) module to capture the graph structure information from multiple views. Specially, HSIM module seeks to leverage these multi-view structure information to assist the feature extractor $\mathcal{F}$ in learning more discriminative transferable node features.

Let $\mathcal{J}_k(Z)$ denote a k-th order moment feature, where $Z \in \mathbb{R}^{|\mathcal{V}|\times e}$ is the deep node features extracted by $\mathcal{F}$. Most current works commonly adopt Kronecker product-based approach to calculate high-order moment features of data [3, 11, 33]. However, considering the numerous nodes contained in the graph (e.g., *ACMv9* has over 9000 nodes), directly calculating the Kronecker product of node features is computationally expensive and time-consuming, making it unsuitable for GTL tasks.

To solve this problem, we utilize the Random Maclaurin factorization scheme [15] to achieve the efficient computation of high-order moment. Concretely, as depicted in Figure 3, we employ multiple 1×1 convolution kernels as several random projectors to estimate the k-th order moment features:

$$\mathcal{J}_k(Z) \approx \mathcal{B}_1(Z) \odot \mathcal{B}_2(Z) \odot \cdots \odot \mathcal{B}_k(Z) \in \mathbb{R}^{|\mathcal{V}|\times e} \tag{3}$$

where $\odot$ is the Hadamard product, and $\mathcal{B}_1, \mathcal{B}_2, \cdots, \mathcal{B}_k$ refer to k randomly initialized 1×1 convolution kernels. However, since the estimators generated by Random Maclaurin scheme are independently of the analyzed distributions [27], which may result in the estimated high-order moment features containing some non-informative high-order components (*i.e.*, noisy components).

To eliminate the impact of these noisy components, we choose to learn the parameters of these projectors (*i.e.*, the 1×1 convolution kernels) directly from the input data. Note that because the calculation of high-order moments involves a large number of Hadamard product operations, it may cause the estimated high-order features degrade into low-order features. To prevent this degradation phenomenon, we employ a recursive mechanism to progressively approximate the high-order moment features:

$$\mathcal{J}_k(Z) = \mathcal{J}_{k-1}(Z) \odot \mathcal{B}_k(Z). \tag{4}$$

Because different order statistics can capture the graph structure information from different views, we mix various order moment features to more comprehensively capture the graph structure features:

$$H = \mathcal{H}(Z) = (\sum_{k=1}^{K} \mathcal{J}_k(Z)) \in \mathbb{R}^{|\mathcal{V}| \times e} \tag{5}$$

To leverage such powerful structure information to assist the feature extractor $\mathcal{F}$ in capturing discriminative transferable features, we concatenate the multi-view structure features H with the deep node features Z extract by $\mathcal{F}$ to obtain the augmented node features $[H; Z] \in \mathbb{R}^{|\mathcal{V}| \times e \times 2}$. After that, to adaptively combine the advantages of the node features and the structure features, a learnable 1×1 convolution filter $\mathcal{W}^{1\times1}$ is utilized to integrate these features:

$$\mathcal{R} = \mathcal{W}^{1\times1}([H; Z]) \in \mathbb{R}^{|\mathcal{V}| \times e}. \tag{6}$$

Then the mixed node features $\mathcal{R}$ will be fed into the classifier C for the final prediction. Given the source labeled node set $\mathcal{V}^{s,l}$, the supervised classification loss on the source graph $\mathcal{G}^s$ can be formulated as:

$$\mathcal{L}_{cls} = \frac{1}{|\mathcal{V}^{s,l}|} \sum_{n_i^s \in \mathcal{V}^{s,l}} \mathcal{L}_{ce}(C(r_i^s), y_i^s). \tag{7}$$

where $\mathcal{L}_{ce}$ is the standard cross-entropy loss, and $r_i^s \in \mathbb{R}^e$ denotes the i-th node feature in node features matrix $\mathcal{R}^s$.

It is worth mentioning that the integration of graph structure information is beneficial in guiding unlabeled nodes to achieve discriminative clustering, thus facilitating the learning of fine-grained domain-invariant features, as verified in Figure 4.

3.4 Adaptive Weighted Domain Alignment

Transferability denotes the ability of sample feature to bridge the discrepancy across domains. It has recently been demonstrated that, in real scenarios, different samples have different levels of transferability [4, 19, 42]. Specially, some samples contain more transferable features, which we term easy-to-transfer samples, and generally have higher transferability. In contrast, hard-to-transfer samples are difficult for the model to capture their transferable features and generally exhibit lower transferability.

Adversarial training has been widely adopted by existing GTL models to extract domain-invariant node features. However, most existing methods [6, 24, 28, 41] typically assign equal weight to different nodes during adversarial domain adaption, ignoring the fact that hard-to-transfer nodes may harm the learning of domain-invariant features and even lead to negative transfer.

To address this issue, motivated by curriculum learning [2], we propose an Adaptive Weighted Domain Alignment (**AWDA**) strategy, which dynamically adjusts the weight of each node based on its transferability. Specially, **(1) During the early training stage**, the model aims to roughly align the marginal feature distributions through adversarial training. Therefore, at this stage, node transferability should primarily be estimated by the discriminator $\mathcal{D}$. The entropy $\mathcal{E}_D$ of discriminator output can be regarded as a good indicator to measure node transferability. At such stage, easy-to-transfer nodes generally have significantly higher entropy $\mathcal{E}_D$ than hard-to-transfer nodes. **(2) During the mid and late training stages**, the model primarily focuses on aligning the category distributions across domains. In such case, node transferability should mainly be estimated based on the entropy $\mathcal{E}_C$ of the classifier output. Concretely, at these stages, easy-to-transfer samples generally have relatively certain classification predictions (*i.e.*, low entropy $\mathcal{E}_C$) since they are close to the corresponding class centers. Hard-to-transfer samples scattered near the decision boundaries, typically have uncertain predictions (*i.e.*, high entropy $\mathcal{E}_C$) and are prone to misclassification.

Based on the above analysis, we design a mixed entropy-aware weighted mechanism $w(n_i)$ that combines both the classifier and the discriminator information to adaptively estimate the transferability for each node n_i:

$$w(n_i) = 1 + e^{-[(2-d_{\mathcal{A}})\mathcal{E}_C - d_{\mathcal{A}}\mathcal{E}_D]}, \tag{8}$$

where $d_{\mathcal{A}}$ denotes the $\mathcal{A}$-distance [1] $d_{\mathcal{A}} = 2(1 - \epsilon(f))$, which is used to measure distribution discrepancy across domains, where $\epsilon(f)$ is the test error of a binary kernel SVM classifier f trained to distinguish the source and target nodes.

Concretely, **(1) during the early training stage**, the source and target domains exhibit significant discrepancy in the deep feature space, and the binary classifier f can almost perfectly distinguish between them (*i.e.*, $\epsilon(f) \to 0$ and $d_{\mathcal{A}} \to 2$). In this way, node transferability is primarily determined by the discriminator $\mathcal{D}$, *i.e.*, $w(n_i) \approx 1 + e^{d_{\mathcal{A}}\mathcal{E}_D}$. **(2) As training progresses**, when the marginal distributions of two domains almost coincide, the binary classifier h cannot distinguish between them, and thus $\epsilon(f) \to 0.5$ and $d_{\mathcal{A}} \to 0$. In such case, node transferability is mainly estimated by the classifier C, *i.e.*, $w(n_i) \approx 1 + e^{-(2-d_{\mathcal{A}})\mathcal{E}_C}$.

Notably, easy-to-transfer nodes usually contain more transferable features, while hard-to-transfer nodes tend to have fewer transferable features. To facilitate the fine-grained alignment of feature distributions and accelerate the learning of domain-invariant features, we encourage the model to pay more attention to easy-to-transfer nodes during adversarial domain adaptation process. Therefore, the node transferability weighted adversarial training loss $\mathcal{L}_{awda}$ can be defined as:

$$\mathcal{L}_{awda} = \frac{1}{N^s} \sum_{i=1}^{N^s} w(n_i^s) \log[\mathcal{D}(T(q_i^s))] + \frac{1}{N^t} \sum_{j=1}^{N^t} w(n_j^t) \log[1 - \mathcal{D}(T(q_j^t))]. \tag{9}$$

where $q_i = (r_i, C(r_i))$ is the joint variable of mixed node feature r_i and its corresponding classifier prediction $C(r_i)$. $T(\cdot)$ is a multi-linear map employed to promote the alignment of multi-modal category distributions as previous study [19].

In this way, the model will focus on the learning of easy-to-transfer samples to roughly align the two domains during the early training stage. As the distribution discrepancy decreases, hard-to-transfer samples get more attention in the adversarial alignment process and are gradually turned into easy-to-transfer ones.

3.5 Trust-aware Node Clustering

Due to the limited number of labeled nodes in $\mathcal{G}^s$, the model is likely to encounter overfitting issue if it simply relies on $\mathcal{L}_{cls}$ for optimization, which may severely degrades the model's generalization performance on $\mathcal{G}^t$. Existing studies usually adopt pseudo-labels strategy [24] or conditional entropy term [36] to guide the learning of unlabeled nodes to mitigate this overfitting phenomenon.

However, there is a concern that the pseudo-labels based strategy inevitably introduces noise into the model and minimizing the conditional entropy term may lead to degenerate clustering solutions (i.e., all unlabeled nodes are assigned to the same cluster), which severely affect the alignment of feature distributions.

To alleviate this overfitting issue, we devise a innovative Trust-aware Node Clustering (**TNC**) strategy to enhance the model's robustness. Considering that both the spatial prototype information and the classifier prediction information can estimate the cluster assignment of node from different views during training, TNC aims to adaptively combine these information to guide the discriminative clustering of unlabeled nodes.

Specially, we first utilizes source labeled nodes to approximate the class centers (i.e., prototypes) μ_c^s of the source domain:

$$\mu_c^s = \frac{1}{M}\sum_{i=1}^{|\mathcal{V}^{s,l}|} r_i^s \cdot \phi(y_i^s, c), \tag{10}$$

where $\phi(y_i^s, c) = 1$ if $y_i^s = c$, otherwise $\phi(y_i^s, c) = 0$. $c \in \{1, 2, \cdots, C\}$ is the class indicator and $M = \sum_{i=1}^{|\mathcal{V}_{s,l}|} \phi(y_i^s, c)$.

On one hand, the spatial prototype information can estimate cluster assignment for each node n_i by measuring the similarity between the node feature r_i and the class center μ_c^s:

$$\mathbb{S}(i,c) = \frac{\exp^{-\gamma(r_i,\mu_c^s)}}{\sum_{c'=1}^{C}\exp^{-\gamma(r_i,\mu_{c'}^s)}}, \tag{11}$$

where $\mathbb{S}(i,c)$ denotes the probability of assigning node n_i to the c-th cluster, and $\gamma(r_i, \mu_c^s)$ denotes the similarity between the node features r_i and the c-th class centers μ_c^s. In our experiment, we employ the Student's t-distribution based kernel strategy [32] as the similarity metric $\gamma(\cdot,\cdot)$, which can be defined as:

$$\gamma(r_i, \mu_c^s) = \frac{\exp((1+\frac{\|r_i-\mu_c^s\|^2}{\alpha}))^{-\frac{\alpha+1}{2}}}{\sum_{c'=1}^{C}\exp((1+\frac{\|r_i-\mu_{c'}^s\|^2}{\alpha}))^{-\frac{\alpha+1}{2}}}, \tag{12}$$

where α is the degree of freedom of the Student's t-distribution. In this work, α is set to 1 for all experiments.

One the other hand, the classifier C can also predict the cluster assignment for each input node n_i:

$$\mathbb{W}(i,c) = Pro(y_i = c|C(r_i)), \tag{13}$$

where $\mathbb{W}(i,c)$ denotes the probability of node n_i belonging to class c.

Although both $\mathbb{S}(i,c)$ and $\mathbb{W}(i,k)$ can measure the probability of assigning node n_i to cluster c, their confidence changes dynamically during training. To this end, we introduce a trustworthy weighted mechanism to dynamically adjust their importance:

$$\Omega(i,c) = \frac{d_{\mathcal{A}}\mathbb{S}(i,c) + (2-d_{\mathcal{A}})\mathbb{W}(i,c)}{\sum_{c'=1}^{C}(d_{\mathcal{A}}\mathbb{S}(i,c') + (2-d_{\mathcal{A}})\mathbb{M}(i,c'))}, \tag{14}$$

where $\Omega(i,c)$ can estimate the probability of node n_i belonging to the c-th cluster in a more robust manner, and $d_{\mathcal{A}}$ denotes the $\mathcal{A}$-distance between two domains. Specially, **during the early training stage**, as the classifier C is learned from scratch, $\mathbb{S}(i,c)$ is much more reliable than $\mathbb{W}(i,k)$. In such case, the clustering assignment of node is primarily estimated by the spatial prototype information(*i.e.*, $\epsilon(f) \to 0$ and $d_{\mathcal{A}} \to 2$). **During the mid and late training stages**, as the category distributions are gradually aligned (*i.e.*, $\epsilon(f) \to 0.5$ and $d_{\mathcal{A}} \to 0$), the classifier C can provide more precise estimations for cluster assignment. In such case, the term $\mathbb{W}(i,c)$ in $\Omega(i,c)$ becomes the main contributor.

Notably, the ideal clustering should satisfy these two conditions: **i)** The clustering assignment Ω of each node should be sufficiently certain; **ii)** Each cluster should contain some nodes (*i.e.*, degenerate solutions will not occur). Here, $\Omega \in \mathbb{R}^{|\mathcal{V}|\times C}$ is the clustering assignment matrix of all nodes, which can be viewed as a distribution.

To achieve this goal, we define a ideal clustering distribution $\Phi \in \mathbb{R}^{|\mathcal{V}|\times C}$ and encourage the current cluster assignment distribution Ω to approach the ideal distribution Φ by using the following objective function $\mathcal{I}$:

$$\begin{aligned}\mathcal{I} &= KL(\Phi||\Omega) + KL(\rho||u)\\ &= \left[\frac{1}{|\mathcal{V}|}\sum_{i=1}^{|\mathcal{V}|}\sum_{c=1}^{C}\Phi(i,c)\log\frac{\Phi(i,c)}{\Omega(i,c)}\right] + \left[\frac{1}{|\mathcal{V}|}\sum_{c=1}^{C}\rho_c\log\frac{\rho_c}{u_c}\right]\\ &= \frac{1}{|\mathcal{V}|}\sum_{i=1}^{|\mathcal{V}|}\sum_{c=1}^{C}\Phi(i,c)\log\frac{\Phi(i,c)}{\Omega(i,c)} + \Phi(i,c)\log\frac{\rho_c}{u_c},\end{aligned} \tag{15}$$

where KL represents the Kullback-Leiber divergence, u is the uniform prior, and ρ_c denotes the soft frequency of cluster assignments in the ideal distribution Φ:

$$\rho_c = \frac{1}{|\mathcal{V}|}\sum_i \Phi(i,c). \tag{16}$$

Specially, in Eq. 15, the first KL term denotes the discrepancy between the current cluster assignment Ω and the ideal target Φ. The second KL term is used to promote balanced cluster assignments in order to avoid degenerate solutions.

To estimate Φ, we employ iterative learning mechanism to optimize this objective function $\mathcal{I}$. Concretely, in each training iteration, assuming the network parameters θ are fixed, we can infer the variable Φ by solving the following optimization problem:

$$\begin{aligned}&\min_{\Phi}\frac{1}{|\mathcal{V}|}\sum_{i=1}^{|\mathcal{V}|}\sum_{c=1}^{C}\Phi(i,c)\log\frac{\Phi(i,c)}{\Omega(i,c)} + \Phi(i,c)\log\frac{\rho_c}{u_c},\\ &s.t.\sum_c\Phi(i,c) = 1.\end{aligned} \tag{17}$$

As this optimization problem can be effectively solved by utilizing several gradient-based algorithms [22] (such as Nesterov optimal and projected gradient descent methods), we can calculate the partial derivative of function $\mathcal{I}$ with respect to variable Φ as:

$$\frac{\partial \mathcal{I}}{\partial \Phi(i,c)} \propto \log\left(\frac{\Phi(i,c)\rho_c}{\Omega(i,c)}\right) + \frac{\Phi(i,c)}{\sum_{i'=1}^{|\mathcal{V}|}\Phi(i',c)} + 1, \tag{18}$$

Since the number of nodes $|\mathcal{V}|$ is typically very large, we can approximate the gradient in Eq. 18 by neglecting the second term. In this way, we obtain an approximate closed-form solution for Φ by setting the gradient to zero:

$$\Phi(i,c)^* = \frac{\Omega(i,c)/(\sum_{i'}\Omega(i',c))^{\frac{1}{2}}}{\sum_{c'}\Omega(i,c')/(\sum_{i'}\Omega(i',c'))^{\frac{1}{2}}}. \tag{19}$$

In TNC strategy, both source and target domain nodes $\mathcal{V}^{s,l} \cup \mathcal{V}^{s,u} \cup \mathcal{V}^{t}$ are used to compute the clustering assignment matrix Ω. Therefore, the loss function $\mathcal{L}_{tnc}$ of TNC strategy can be defined as:

$$\mathcal{L}_{tnc} = KL(\Phi^*||\Omega) + KL(\rho^*||u). \tag{20}$$

The reason we employ source labeled nodes $\mathcal{V}^{s,l}$ in TNC strategy is because they can effectively guide the discriminative clustering of unlabeled nodes towards the desired direction. Notably, our TNC strategy does not involve any pseudo-labels, which not only enhances the model's robustness, but also promotes the precise alignment of category distributions (see Figure 5 for further analysis).

3.6 Model Optimization

To sum up, the total loss function of HOGDA can be formulated as:

$$\min_{\mathcal{F},\mathcal{C},\mathcal{H},\mathcal{W}^{1\times1}} \max_{\mathcal{D}} \mathcal{L}_{cls} + \eta\mathcal{L}_{awda} + \beta\mathcal{L}_{tnc} \tag{21}$$

where hyper-parameters η and β are used to balance the contributions of the corresponding term.

4 EXPERIMENTS

4.1 Setup

Datasets. Our experiments encompass three real-world graphs[31]: *ACMv9* (**A**), *Citationv1* (**C**), and *DBLPv7* (**D**). In these graphs, every node corresponds to a paper, and the attribute of each paper is a sparse bag-of-words vector derived from its title. The edges in these graphs depict citation relationships among the papers. Considering that these graphs contain diverse sets of node attributes, we merge their attribute sets and resize the attribute dimension to 6775 following [24]. Each node is assigned a 5-class label, determined by its relevant research areas, including *Artificial Intelligence*, *Computer Vision*, *Database*, *Information Security*, and *Networking*. Six typical cross-domain tasks will be carried out in our experiments: **A→C**, **A→D**, **C→A**, **C→D**, **D→A** and **D→C**. Further settings and implementation can be found in the **Supplementary Materials**.
Compared Methods. We mainly compare our method with several SOTA **(1)** graph semi-supervised learning methods and **(2)** graph domain adaptation methods following the pioneering work [24]: **(1) GCN** [16], **GSAGE** [12], **GAT** [34], **GIN** [39], **(2) DANN** [8], **CDAN** [20], **UDA-GCN** [36], **AdaGCN** [6] and **SGDA** [24]. Note that **DANN**$_{GCN}$ and **CDAN**$_{GCN}$ are two variants that replace the MLP-based encoders with GCN-based feature extractors.
Evaluation Metrics. Following previous works [24, 28], we employ **Micro-F1** and **Macro-F1** as evaluation metrics. We repeat each experiment 5 times and record the average accuracy along with standard deviation. Additionally, to address the impact of randomness, we sample different label sets for each experiment.

4.2 Results and Discussion

To demonstrate the superiority of our HOGDA, we follow [24] to evaluate its performance in the challenging scenario, where only **5%** of the nodes in the source graph are labeled. Table 1 lists the classification results of different methods on the target graph.

We can observe that our model obtains the overall best results on all transfer tasks. Concretely, HOGDA significantly outperforms the SOTA competitor SGDA [24] by +8.1% and +10.3% on "Micro-F1" and "Macro-F1" respectively for the **C→A** task, indicating the advantage in extracting discriminative transferable features. Furthermore, HOGDA achieves substantial performance gains in some hard transfer scenarios, such as **D→A** and **C→A**, where the size of target domain is larger than source domain, implying the robustness of our model in face of some small-scale dataset scenarios. Notably, we find that most methods, especially adversarial training-based methods, perform poorly because they can only roughly align the marginal distributions and cannot effectively utilize unlabeled nodes. In contrast, our method addresses these limitations. Additionally, the results with a smaller fluctuation range imply the stability of our model, which further confirms the importance of adaptively guiding domain alignment and node clustering.

4.3 Ablation Study and Analysis

Due to page size limitation, more experiments and analysis are given in the **Supplementary Materials**.
1) Ablation Study: To analyze the contribution of each component in our model, we compare HOGDA and its 7 variants on different transfer tasks. Table 2 describes the variants of HOGDA, and Table 1 shows the results of ablation study.
Contribution of Each Component: The results in Table 1 reflect the following observations: **(1)** Due to the label scarcity in source domain, HOGDA-S (baseline) inevitably suffers from overfitting issues, resulting in poor generalization performance on all tasks. **(2)** Variants HOGDA-H, HOGDA-A and HOGDA-T greatly outperform HOGDA-S on all tasks, implying that incorporating high-order structure information, conducting node weighted domain alignment, and guiding discriminative clustering of unlabeled nodes all effectively promote the learning of domain-invariant node features.
Correlation of Our Strategies: The results in Table 1 show that combining different strategies can significantly boost the model's transfer ability, suggesting a distinct complementary relationship among the HSIM, AWDA, and TNC strategies.
2) Node Features Visualization: To showcase the superior transfer ability of our model, we utilize t-SNE [32] to visualize the node features on task **A→C** under the same 5% label rate setting, as depicted in Figure 4. As for the SOTA method SGDA, the category distributions are not well aligned and the decision boundary is not

Table 1: Transfer performance (%) on six tasks with a source graph label rate of 5% for semi-supervised graph domain adaptation.

Methods	A→C		A→D		C→A		C→D		D→A		D→C	
	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1
MLP	41.3±1.15	35.8±0.72	42.8±0.88	36.3±0.77	39.4±0.57	33.7±0.58	43.7±0.69	36.7±0.55	37.3±0.32	30.8±0.37	39.4±0.99	32.8±0.99
GCN	54.4±1.52	52.0±1.62	56.9±2.33	53.4±2.81	54.1±1.40	52.3±1.98	58.9±0.99	54.5±1.55	50.1±2.14	48.0±3.28	56.0±1.24	51.9±1.49
GSAGE	49.3±2.18	46.4±2.06	51.8±1.35	47.4±1.62	46.8±2.56	45.0±2.78	51.7±1.95	48.1±1.97	41.7±2.17	37.4±4.59	45.4±2.11	39.3±3.45
GAT	55.1±3.22	50.8±1.45	55.3±2.52	51.8±2.60	50.0±1.20	45.6±2.36	55.4±2.73	49.2±2.59	44.8±2.74	38.3±4.84	50.4±3.35	42.0±4.46
GIN	64.6±2.47	56.0±2.73	60.0±2.09	51.3±3.99	57.1±1.19	54.4±2.57	62.0±1.05	56.8±1.40	51.9±2.00	45.4±2.16	60.2±3.05	53.0±2.10
DANN	44.3±2.03	39.3±1.86	44.0±1.42	38.7±1.47	41.8±1.95	37.6±1.24	45.5±0.71	39.6±1.55	37.8±3.66	33.2±2.23	41.7±2.32	35.6±2.55
CDAN	44.6±1.30	38.6±1.07	45.5±0.85	38.0±0.86	42.4±0.64	36.2±1.17	46.7±1.17	39.2±0.96	39.0±1.08	32.3±1.09	41.7±1.55	34.8±1.56
$DANN_{GCN}$	63.0±6.75	59.6±6.02	62.2±1.90	57.7±3.16	56.7±0.38	55.2±1.03	65.3±2.04	59.0±2.39	52.3±2.59	48.6±4.52	58.1±2.78	52.4±3.81
$CDAN_{GCN}$	70.3±0.84	66.5±0.66	65.0±1.00	61.3±0.96	56.3±1.78	53.6±2.70	65.2±2.19	58.8±2.38	53.0±1.34	48.7±3.51	59.0±1.52	53.3±1.99
UDA-GCN	72.4±2.75	65.2±6.51	68.0±6.38	64.3±7.12	62.9±0.33	62.2±1.44	71.4±2.56	67.5±2.25	55.8±3.50	52.4±2.68	65.2±4.41	60.7±6.84
AdaGCN	70.8±0.95	68.5±0.73	68.2±3.84	64.2±3.91	61.5±2.20	60.4±3.15	69.1±1.96	65.8±2.87	56.1±1.75	53.8±2.95	64.1±0.91	62.8±1.56
SGDA	75.6±0.57	71.4±0.82	69.2±0.73	64.7±2.36	66.3±0.68	62.3±0.96	72.9±1.26	68.9±1.83	60.6±0.86	56.0±0.90	73.2±0.59	69.3±1.01
HOGDA-S	55.8±1.76	53.6±1.84	54.2±2.11	44.9±2.04	58.2±1.52	48.9±1.94	57.0±1.03	46.3±1.60	49.8±2.33	41.0±3.46	55.9±1.41	45.2±1.72
HOGDA-H	73.5±0.51	67.1±0.79	67.4±0.82	63.8±1.27	66.0±0.76	62.7±0.85	67.3±0.98	62.2±1.37	57.9±0.81	55.3±0.85	70.6±0.52	68.4±0.94
HOGDA-A	76.7±0.42	71.2±0.65	71.8±0.59	68.1±1.02	69.1±0.66	64.0±0.94	72.6±0.92	68.7±1.01	63.4±0.76	59.6±0.81	74.8±0.53	71.5±0.90
HOGDA-T	74.9±0.42	70.8±0.71	69.0±0.49	64.5±0.92	65.8±0.50	62.1±0.83	70.6±0.89	66.6±1.17	62.9±0.65	57.2±0.81	72.7±0.46	69.0±0.73
HOGDA-HA	80.7±0.53	78.3±0.59	74.7±0.82	72.2±1.25	72.6±0.64	71.5±0.83	75.2±1.07	72.1±1.26	65.7±0.71	63.3±0.80	77.4±0.59	73.8±0.87
HOGDA-HT	80.3±0.62	77.5±0.50	74.2±0.57	71.6±1.09	72.2±0.53	71.1±0.87	74.8±1.05	71.7±1.61	64.9±0.63	62.4±0.75	78.5±0.49	74.0±0.76
HOGDA-AT	81.5±0.43	78.7±0.51	75.4±0.50	73.3±0.87	73.1±0.55	72.0±0.89	75.8±0.97	72.4±1.26	66.7±0.51	64.2±0.63	79.1±0.46	74.5±0.82
HOGDA	**82.4**±0.36	**79.2**±0.43	**76.5**±0.47	**73.8**±0.82	**74.4**±0.46	**72.6**±0.78	**77.1**±0.93	**72.9**±1.21	**68.2**±0.47	**65.1**±0.56	**80.3**±0.41	**75.0**±0.82

Table 2: Different variants of HOGDA.

Variant	$\mathcal{L}_{cls}$	HSIM	$\mathcal{L}_{awda}$	$\mathcal{L}_{tnc}$
HOGDA-S	✓			
HOGDA-H	✓	✓		
HOGDA-A	✓		✓	
HOGDA-T	✓			✓
HOGDA-HA	✓	✓	✓	
HOGDA-HT	✓	✓		✓
HOGDA-AT	✓		✓	✓
HOGDA	✓	✓	✓	✓

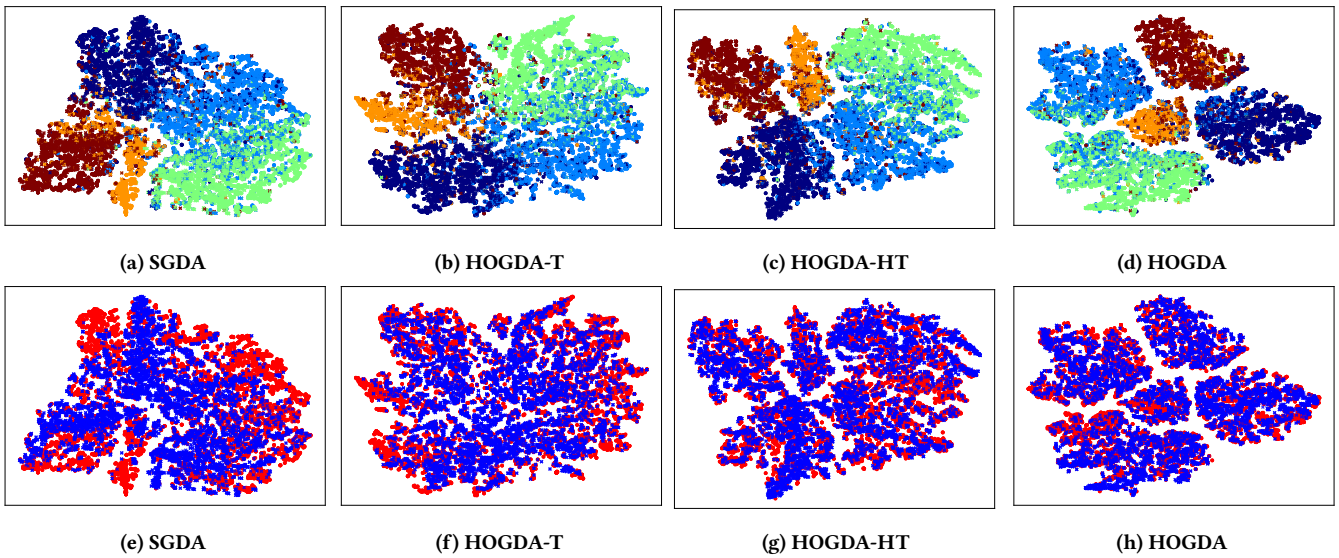
clear enough. We can see noticeable overlaps between different clusters, which increases the risk of misclassifying hard-to-transfer nodes. In contrast, our HOGDA precisely align the category distributions across domains and achieves exactly 5 clusters with clean decision boundaries, implying that our method can promote transferable node features become more discrimative.

3) Effect of HSIM: To demonstrate the effectiveness of HSIM, we conduct in-depth experiments from both quantitative and visual aspects: **(1)** As reported in Table 1, variants HOGDA-HA and HOGDA-HT greatly surpass HOGDA-A and HOGDA-T respectively, implying that the injection of structure information can facilitate the learning of transferable features. **(2)** As illustrated in Figure 4, compared to HOGDA-T, variant HOGDA-HT exhibits better intra-class compactness and inter-class separability in the feature space, indicating that the graph structure information can guide the discriminative clustering of unlabeled nodes, thereby promoting the learning of domain-invariant features.

4) Effect of TNC: To showcase the superiority of TNC strategy, we conduct a comparative analysis with existing nodes clustering strategies, including conditional entropy minimization strategy (**ENT**) [36] and the recently proposed posterior scores-based pseudo-labeling strategy (**PSPL**) [24]. We record the trend of Micro-F1 score during training on tasks **A→C** and **A→D**, respectively. The curves in Figure 5 reflect the following observations: **(1)** HOGDA-T exhibits smoother and faster convergence, leading to superior transfer performance, which implies that our TNC strategy can effectively guide the discriminative clustering of unlabeled nodes and promote the fine-grained alignment of category distributions. **(2)** Compared to HOGDA-S (baseline), HOGDA-S + ENT suffers from severe performance degradation, as ENT causes the unlabeled nodes to fall into a degenerate clustering solution. **(3)** PSPL strategy inevitably introduces some pseudo-label noise to the model during training, making it difficult to further improve the model's generalization ability.

5) Effect of Label Rate: We investigate the model's performance under different label scarcity settings on two typical transfer tasks: **A→C** and **A→D**. Specially, the source graph is assigned label rates of 1%, 5%, 7%, 9%, and 10% respectively, as depicted in Figure 6. We can find that HOGDA surpasses other competitors by a significant margin, even in the most challenging environment of 1% label rate, which indicating the robustness of HOGDA when facing different challenging transfer scenarios.

6) Effect of AWDA: To demonstrate the effectiveness of our AWDA strategy, we compare it with existing domain alignment strategies, including the standard adversarial domain alignment strategy (**AD**) [6], sliced Wasserstein distance-based alignment strategy (**SWD**)[17], class-conditional MMD strategy (**CMMD**) [30], and the recently proposed shifting-guided adversarial domain alignment strategy (**SAD**) [24]. We employ variant HOGDA-S as the baseline



(a) SGDA (b) HOGDA-T (c) HOGDA-HT (d) HOGDA

(e) SGDA (f) HOGDA-T (g) HOGDA-HT (h) HOGDA

Figure 4: The t-SNE visualization of node features learned by SGDA, HOGDA and its two variants on the A→C task (5 classes) with 5% label rate. In all subfigures, the marks • and × denotes the source domain nodes and target domain nodes, respectively. Fig 4(a-d) illustrate category distributions alignment (Different colors represents different classes). Fig 4(e-h) depict domain alignment (Red: Source domain; Blue: Target domain). *Best viewed in color.*

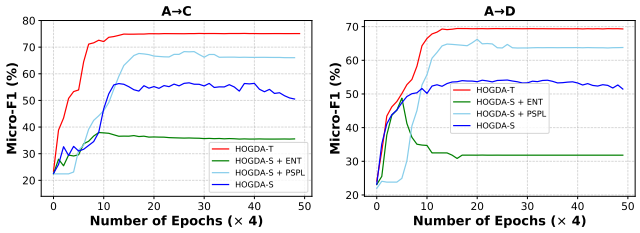


Figure 5: The trend of Micro-F1 during model training.

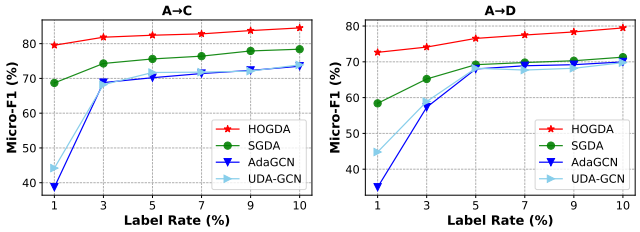


Figure 6: Transfer performance with different label rates.

and evaluate the performance gains brought by these strategies on **A→C** and **A→D** tasks, as shown in Figure 7.

We can find that our AWDA strategy significantly outperforms all compared strategies. This is because the compared strategies can only roughly align marginal distributions, while the proposed AWDA can align feature distributions at the class level, which facilitate the model to extract more fine-grained transferable features.

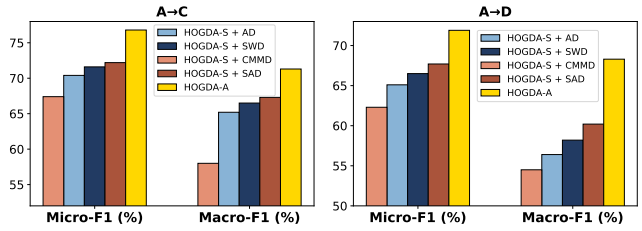


Figure 7: Comparison with different domain alignment strategies on A→C and A→D tasks.

5 CONCLUSION

In this paper, we propose a novel model called HOGDA for SGDA. Specially, we introduce a HSIM module to capture high-order structure information in graph in order to better assist the GCN-based feature extractor in learning transferable node features. Additionally, we propose a novel AWDA strategy to encourage the model to pay more attention to easy-to-transfer nodes during adversarial domain alignment, significantly boosting the model's generalization ability on the target domain. More importantly, to address the overfitting issue, a simple but effective TNC strategy is devised to guide the clustering of unlabeled nodes. Comprehensive experiments validate the superiority and stability of our HOGDA on various popular benchmarks.

REFERENCES

[1] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A theory of learning from different domains. *Machine learning* 79 (2010), 151–175.

[2] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference*

on machine learning. 41–48.
[3] Victor M Calo, Pouria Behnoudfar, M Łoś, and M Paszyński. 2023. A Kronecker product linear-cost solver for the high-order generalized-α method for multi-dimensional hyperbolic systems. *Computers & Mathematics with Applications* 142 (2023), 257–267.
[4] Xinyang Chen, Sinan Wang, Mingsheng Long, and Jianmin Wang. 2019. Transferability vs. discriminability: Batch spectral penalization for adversarial domain adaptation. In *International conference on machine learning*. PMLR, 1081–1090.
[5] Luca Cosmo, Giorgia Minello, Michael M Bronstein, Emanuele Rodolà, Luca Rossi, and Andrea Torsello. 2021. Graph Kernel Neural Networks. (2021).
[6] Quanyu Dai, Xiao-Ming Wu, Jiaren Xiao, Xiao Shen, and Dan Wang. 2023. Graph Transfer Learning via Adversarial Domain Adaptation With Graph Convolution. *IEEE Transactions on Knowledge and Data Engineering* 35, 5 (2023), 4908–4922. https://doi.org/10.1109/TKDE.2022.3144250
[7] Zhengming Ding, Sheng Li, Ming Shao, and Yun Fu. 2018. Graph adaptive knowledge transfer for unsupervised domain adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 37–52.
[8] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario March, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *Journal of machine learning research* 17, 59 (2016), 1–35.
[9] Mengran Gou, Octavia Camps, and Mario Sznaier. 2017. mom: Mean of moments feature for person re-identification. In *Proceedings of the IEEE international conference on computer vision workshops*. 1294–1303.
[10] Gaoyang Guo, Chaokun Wang, Bencheng Yan, Yunkai Lou, Hao Feng, Junchao Zhu, Jun Chen, Fei He, and Philip Yu. 2022. Learning adaptive node embeddings across graphs. *IEEE Transactions on Knowledge and Data Engineering* (2022).
[11] Wolfgang Hackbusch, Boris N Khoromskij, and Eugene E Tyrtyshnikov. 2005. Hierarchical Kronecker tensor-product approximations. *Journal of Numerical Mathematics* 13, 2 (2005).
[12] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems* 30 (2017).
[13] Ziniu Hu, Yuxiao Dong, Kuansan Wang, Kai-Wei Chang, and Yizhou Sun. 2020. Gpt-gnn: Generative pre-training of graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1857–1867.
[14] Wei Jin, Yaxing Li, Han Xu, Yiqi Wang, Shuiwang Ji, Charu Aggarwal, and Jiliang Tang. 2021. Adversarial attacks and defenses on graphs. *ACM SIGKDD Explorations Newsletter* 22, 2 (2021), 19–34.
[15] Purushottam Kar and Harish Karnick. 2012. Random feature maps for dot product kernels. In *Artificial intelligence and statistics*. PMLR, 583–591.
[16] Thomas N Kipf and Max Welling. 2016. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations*.
[17] Chen-Yu Lee, Tanmay Batra, Mohammad Haris Baig, and Daniel Ulbricht. 2019. Sliced wasserstein discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10285–10295.
[18] Shikun Liu, Tianchun Li, Yongbin Feng, Nhan Tran, Han Zhao, Qiang Qiu, and Pan Li. 2023. Structural re-weighting improves graph domain adaptation. In *International Conference on Machine Learning*. PMLR, 21778–21793.
[19] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. 2018. Conditional adversarial domain adaptation. *Advances in neural information processing systems* 31 (2018).
[20] Mingsheng Long, ZHANGJIE CAO, Jianmin Wang, and Michael I Jordan. 2018. Conditional Adversarial Domain Adaptation. In *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.), Vol. 31. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2018/file/ab88b15733f543179858600245108dd8-Paper.pdf
[21] Qingqing Long, Yilun Jin, Yi Wu, and Guojie Song. 2021. Theoretically improving graph neural networks via anonymous walk graph kernels. In *Proceedings of the Web Conference 2021*. 1204–1214.
[22] Yurii Nesterov. 2013. *Introductory lectures on convex optimization: A basic course*. Vol. 87. Springer Science & Business Media.
[23] Ziyue Qiao, Yanjie Fu, Pengyang Wang, Meng Xiao, Zhiyuan Ning, Denghui Zhang, Yi Du, and Yuanchun Zhou. 2022. RPT: toward transferable model on heterogeneous researcher data via pre-training. *IEEE Transactions on Big Data* 9, 1 (2022), 186–199.
[24] Ziyue Qiao, Xiao Luo, Meng Xiao, Hao Dong, Yuanchun Zhou, and Hui Xiong. 2023. Semi-supervised domain adaptation in graph transfer learning. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*. 2279–2287.
[25] Ziyue Qiao, Pengyang Wang, Pengfei Wang, Zhiyuan Ning, Yanjie Fu, Yi Du, Yuanchun Zhou, Jianqiang Huang, Xian-Sheng Hua, and Hui Xiong. 2023. A Dual-Channel Semi-Supervised Learning Framework on Graphs via Knowledge Transfer and Meta-Learning. *ACM Transactions on the Web* (2023).
[26] 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.
[27] Ali Rahimi and Benjamin Recht. 2007. Random features for large-scale kernel machines. *Advances in neural information processing systems* 20 (2007).
[28] Xiao Shen, Quanyu Dai, Fu-lai Chung, Wei Lu, and Kup-Sze Choi. 2020. Adversarial deep network embedding for cross-network node classification. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 2991–2999.
[29] Xiao Shen, Quanyu Dai, Sitong Mao, Fu-lai Chung, and Kup-Sze Choi. 2020. Network together: Node classification via cross-network deep network embedding. *IEEE Transactions on Neural Networks and Learning Systems* 32, 5 (2020), 1935–1948.
[30] Xiao Shen, Quanyu Dai, Sitong Mao, Fu-Lai Chung, and Kup-Sze Choi. 2021. Network Together: Node Classification via Cross-Network Deep Network Embedding. *IEEE Transactions on Neural Networks and Learning Systems* 32, 5 (2021), 1935–1948. https://doi.org/10.1109/TNNLS.2020.2995483
[31] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. 990–998.
[32] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, 11 (2008).
[33] Charles F Van Loan and Nikos Pitsianis. 1993. *Approximation with Kronecker products*. Springer.
[34] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *International Conference on Learning Representations*.
[35] Yiwei Wang, Wei Wang, Yuxuan Liang, Yujun Cai, Juncheng Liu, and Bryan Hooi. 2020. Nodeaug: Semi-supervised node classification with data augmentation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 207–217.
[36] Man Wu, Shirui Pan, Chuan Zhou, Xiaojun Chang, and Xingquan Zhu. 2020. Unsupervised domain adaptive graph convolutional networks. In *Proceedings of The Web Conference 2020*. 1457–1467.
[37] Louis-Pascal Xhonneux, Meng Qu, and Jian Tang. 2020. Continuous graph neural networks. In *International Conference on Machine Learning*. PMLR, 10432–10441.
[38] Jiarong Xu, Yang Yang, Junru Chen, Xin Jiang, Chunping Wang, Jiangang Lu, and Yizhou Sun. 2022. Unsupervised adversarially robust representation learning on graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 4290–4298.
[39] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2018. How Powerful are Graph Neural Networks?. In *International Conference on Learning Representations*.
[40] Nan Yin, Li Shen, Mengzhu Wang, Long Lan, Zeyu Ma, Chong Chen, Xian-Sheng Hua, and Xiao Luo. 2023. Coco: A coupled contrastive framework for unsupervised domain adaptive graph classification. In *International Conference on Machine Learning*. PMLR, 40040–40053.
[41] Xiaowen Zhang, Yuntao Du, Rongbiao Xie, and Chongjun Wang. 2021. Adversarial separation network for cross-network node classification. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 2618–2626.
[42] Lin Zuo, Mengmeng Jing, Jingjing Li, Lei Zhu, Ke Lu, and Yang Yang. 2021. Challenging tough samples in unsupervised domain adaptation. *Pattern Recognition* 110 (2021), 107540.