Multi-View Teacher with Curriculum Data Fusion for Robust Unsupervised Domain Adaptation

Yuhao Tang[1†], Junyu Luo[2†], Ling Yang[1], Xiao Luo[3], Wentao Zhang[4], and Bin Cui[1,5§]
[1] School of CS & Key Laboratory of High Confidence Software Technologies (MOE), Peking University
[2] School of CS, Peking University [3] University of California Los Angeles
[4] Center for Machine Learning Research, Peking University
[5] Institute of Computational Social Science, Peking University (Qingdao), China
Email: {tangyuhao, wentao.zhang, bin.cui}@pku.edu.cn {luojunyu, yangling}@stu.pku.edu.cn, xiaoluo@cs.ucla.edu

2024 IEEE 40th International Conference on Data Engineering (ICDE) | 979-8-3503-1715-2/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICDE60146.2024.00204

***Abstract*—Graph Neural Networks (GNNs) have emerged as an effective tool for graph classification, yet their reliance on extensive labeled data poses a significant challenge, especially when such labels are scarce. To address this challenge, this paper presents a novel framework, denoted as Multi-View Teacher with Curriculum Data Fusion (MTDF). MTDF achieves robust unsupervised domain adaptation in both the model and data perspectives. On the one hand, MTDF utilizes a multi-teacher framework with diverse update strategies for robust adaptation. Moreover, it employs a complementary perspective consistency model from local implicit representation and global explicit graph structure. On the other hand, MTDF generates source-mimicry data at the target domain to serve as a bridge to overcome the challenge of domain shift. MTDF achieves stable unsupervised domain adaptation through bi-directional processes from the perspective of both the model and the data. We have conducted comprehensive experimental evaluations across multiple real-world datasets with a range of baseline methods to demonstrate the superior performance of our proposed method.**

***Index Terms*—Graph Neural Network, Unsupervised Domain Adaptation**

I. Introduction

Graph Neural Networks (GNNs) have achieved great success in a wide range of domains, including molecular structures [1], [2], social networks [3], [4] traffic networks [5]–[8], industry [9] and relational databases [10], [11]. Among them, graph classification aims to predict the labels of entire graphs [12]–[14]. Most of these strategies adopt the message-passing paradigm [15], followed by a readout operator to summarize the node-level representations to graph-level representations for downstream classifications. However, real-world situations frequently face Out-of-Distribution (OOD) challenges, which occur when the data distribution during inference differs from what has been experienced during training [16]–[18]. The fundamental problem is exacerbated by the scarcity of labeled data in these new areas of application, making supervised adaptation impractical [19], [20].

Unsupervised Domain Adaptation (UDA) aims to use labeled data from a source domain to perform tasks in an unlabeled target domain [21], [22]. In UDA, approaches have demonstrated proficiency, particularly for Euclidean data, *e.g.*, Mean-Teacher [23] apply a student-teacher paradigm to maintain prediction consistency between teacher model (an exponential moving average of student model) and student model. However, when it comes to graph data, the training process encounters difficulties and may even deteriorate the model's performance, as illustrated in Figure 1a. We attribute this to two under-explored dilemmas.

(1) *How to overcome the label scarcity on the target domain?* The scarcity of labeled data in the target domain is a persistent barrier in UDA for graphs. While pseudo-labels show potential in bridging this gap [24], [25], their vulnerability to noise makes them less reliable. Figure 1b illustrates a common issue for models that train from the source domain and apply to the target domain. Typically, such models generate much less high-confidence predictions for the target domain compared to the source domain. These noisy pseudo-labels can introduce harmful self-reinforcing errors, which can propagate through the edges during the training process, ultimately resulting in poor domain adaptation performance.

(2) *How to effectively reduce the domain discrepancy between source and target domains?* Unlike Euclidean data, graphs can vary significantly in structure and scale, leading to differences in statistical properties of the source and target graphs. As demonstrated in Figure 1c, embeddings obtained from data of different domains with the same model exhibit significant differences. Traditional domain adaptation approaches rely on aligning feature distributions [26]. However, these methods struggle due to the complexity of graphs when accounting for node features, edge connectivity, and higher-order substructures. Consequently, carving out a method that considers the complex interplay among these attributes to bridge the domain discrepancy is a critical avenue in unsupervised graph domain adaptation.

In addressing the aforementioned issues, we introduce a new framework, Multi-View Teacher with Curriculum Data Fusion (**MTDF**). This approach robustly tackles unsupervised graph domain adaptation from both model and data perspectives. On the one hand, MTDF utilizes different updating strategies for teacher models to achieve robust domain adaptation and stable optimization, including the booster teacher and the stabilizer teacher. Furthermore, to fully extract topological

† equal contribution. § corresponding author.

2375-026X/24/$31.00 ©2024 IEEE
DOI 10.1109/ICDE60146.2024.00204

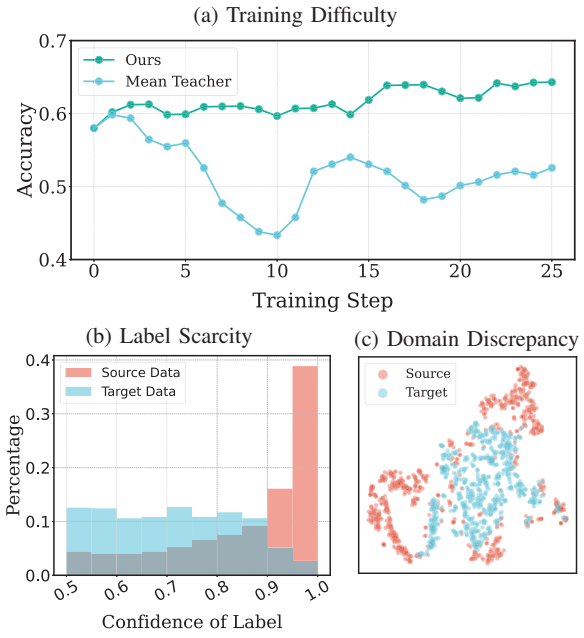


Fig. 1: Motivation of MTDF, all from experiments on NCI1 dataset. **(a)** The training difficulty of UDA, which caused by label scarcity and domain discrepancy. MTDF employs multi-teacher and curriculum data fusion strategy to stabilize the adaptation process. **(b)** Target label scarcity, illustrated by the histogram of confidence score of the source-trained model. MTDF mines the target domain from multi-view to extract as much information as possible from limited resources. **(c)** Domain discrepancy, illustrated by the t-SNE visualization demonstrates. MTDF utilize curriculum data fusion strategy to gradually achieve domain adaptation. Best viewed zoomed in and in color.

information under label scarcity, MTDF uses information from complementary perspectives in the target domain. One branch is the message-passing branch, which focuses on implicit local graph representations. The other is the graph motif branch, which pays attention to the explicit topological information of graphs, yielding explicit global graph structure representations. Through consistency learning, these branches interact and serve as mutual teachers, generating more robust graph representations. On the other hand, to address the domain discrepancy, MTDF incorporates curriculum data fusion to bridge the source and the target domains and achieve progressive domain adaptation. MTDF gradually introduces structural information of the target domain through a curriculum learning approach, starting from the simple structure of the source domain and progressively increasing the difficulty, making the entire domain adaptation process more stable. The main contributions of this paper can be summarized as follows:

- **New Perspective.** To tackle the issues of label scarcity and domain discrepancy, we propose a new framework that performs unsupervised domain adaptation bidirectionally from both model and data perspectives, simultaneously achieving a stable adaptation process.
- **New Method.** Specifically, to address the scarcity of labeled data, we utilize a stable optimization guided by teachers with various update strategies, further reinforced by robust graph learning, integrating consistency learning from multiple complementary perspectives. To address domain discrepancy, we employ data fusion through curriculum learning, achieving a gradual and stable adaptation process.
- **State-of-the-art Experimental Performance.** Experiments conducted on several graph classification benchmark datasets indicate that MTDF outperforms a variety of established baselines by a significant margin. Furthermore, we perform extensive ablation studies and visualization for further analysis.

II. Related Works

A. Graph Classification

Graph classification has emerged as an important task across several domains, such as social network analysis [27]–[33], bioinformatics [34], industry [35], and cheminformatics [36], [37], where the objective is to predict global properties of entire graphs. While traditional graph classification approaches heavily relied on graph kernels [38], [39] techniques that measure the similarity between graphs by mapping them into a high-dimensional feature space, recent advancements have pivoted towards Graph Neural Networks (GNNs) due to their superior capability in learning graph representations [15], [40]–[42]. These modern GNN-based methods leverage a message-passing paradigm to update node embeddings and employ various pooling techniques to synthesize these embeddings into a graph-level representation suitable for classification tasks [43]. Moreover, graph motif networks have further advanced GNNs by leveraging recurring subgraph patterns to enrich graph representations for classification [44]–[46]. Nevertheless, the scarcity of labeled data in target domains necessitates unsupervised domain adaptation to leverage auxiliary domain knowledge. Addressing this challenge, our work capitalizes on the use of limited pseudo-labeled data by consistency learning from graph classification techniques with different structures.

B. Unsupervised Domain Adaptation

Unsupervised domain adaptation (UDA) [47], [48] has emerged as a crucial strategy for transferring knowledge between labeled source data and unlabeled target domains [24], [49]–[51], with extensive applications in fields like computer vision [21], [25]. Prevailing methodologies revolve around domain alignment and discrimination learning [52]–[56]. Domain alignment techniques have evolved from early endeavors that leveraged statistical measure [57]–[59] to reduce domain discrepancies to contemporary adversarial learning models that harness gradient reversal layers and classifier collisions to mitigate the impact of domain shifts [60]. Discrimination learning, injected with insights from semi-supervised learning [61], [62], increasingly utilizes self-training to enhance

performance on target domains [24], [25]. Methods such as pseudo-labeling [63], where models generate labels by themselves, coupled with techniques that focus on ensuring those labels are reliable and that the model's uncertainty is reduced, are now commonly used to improve the ability of classifiers to correctly identify data in the target domain [64], [65]. While UDA has led to significant strides in computer vision, including areas such as encompassing image classification [21], semantic segmentation [25], and image retrieval [66], its exploration in graph data domains is in the early stages. There, the adaptation challenges are inherently distinct owing to the structured nature of graph data. Initial studies have tackled node classification within graphs [67], yet comprehensive solutions in graph classification have remained limited. Our work introduces an unsupervised domain adaptive framework for graph classification, which achieves effective domain alignment on the graph space. Expanding on this research landscape, Wu et al. [68] propose the Denoising and Nuclear-Norm Wasserstein Adaptation Network (DNAN), which introduces a unique methodology utilizing the Nuclear-norm Wasserstein discrepancy to address domain alignment and class differentiation challenges in graph data.

C. Unsupervised Learning

Unsupervised learning in graph data offers a compelling strategy to learning representations without labeled data, primarily by capturing the intrinsic structure and features of the graph. Building on Deep Graph Infomax (DGI) [69], which maximizes mutual information between node representations and a high-level summary of the entire graph, InfoGraph [70] enhances this method by focusing on whole-graph embeddings. While DGI is agnostic to graph labeling and effective for node embeddings, InfoGraph recursively applies the mutual information framework to derive embeddings conducive to graph classification tasks, extending the utility of this unsupervised learning strategy from local node features to global graph structures. Subsequent methods such as [71] take inspiration from natural language processing models like Doc2Vec [72] aim to learn latent document-like representations of entire graphs. Graph data autoencoders, including Graph Autoencoder (GAE) [73]and Variational Graph Autoencoder (VGAE) [74], employ reconstruction objectives, where they strive to build embeddings that can reconstruct the original graph structure, yielding embeddings that pack topological information. Graph Contrastive Learning [75], [76] promotes learning by distinguishing between different augmentations of the graph, thereby learning invariant features robust to perturbations. These unsupervised techniques facilitate the derivation of rich, informative features that could potentially be transferable across domains, setting the stage for improved unsupervised domain adaptation in graph models.

III. Preliminaries

A. Problem Definition

In the context of graph-based learning, a graph is formalized as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ denotes the set of nodes and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ represents the set of edges linking the nodes in $\mathcal{V}$. Accompanying the graph structure, a node feature matrix $X \in \mathbb{R}^{|\mathcal{V}| \times d}$ contains the attribute information, with each row $x_v \in \mathbb{R}^d$ being the feature representation for node $v \in \mathcal{V}$, and where d indicates the dimension of node features. The unsupervised domain adaptation task aims to transfer the knowledge between a labeled source domain $\mathcal{D}_{so} = \{(\mathcal{G}_i^{so}, y_i^{so})\}_{i=1}^{N_{so}}$, comprising N_{so} graph examples $\mathcal{G}_i^{so}$ with corresponding labels y_i^{so}, and an unlabeled target domain $\mathcal{D}_{ta} = \{\mathcal{G}_j^{ta}\}_{j=1}^{N_{ta}}$, which contains N_{ta} graph examples $\mathcal{G}_j^{ta}$ without label. Here, $\mathcal{D}_{so}$ and $\mathcal{D}_{ta}$ are sampled from an identical label space $\mathcal{Y} = \{1, 2, \ldots, C\}$, yet exhibit differential data distributions within the graph domain indicative of a domain shift. Our objective is to train predictive models that perform well on $\mathcal{D}_{ta}$ from a model trained by labeled data in $\mathcal{D}_{so}$, despite the lack of labels in the target domain and the presence of distributional differences between the domains. All these notations are summarized in Table I.

TABLE I: Frequently used notations

Notation	Meaning
$\mathcal{G} = (\mathcal{V}, \mathcal{E})$	A graph $\mathcal{G}$ with vertex set $\mathcal{V}$ and edge set $\mathcal{E}$
X	The feature matrix of node
$\mathcal{D}_{so} = \{(\mathcal{G}_i^{so}, y_i^{so})\}_{i=1}^{N_{so}}$	Labeled source dataset with N_{so} graphs
$\mathcal{D}_{ta} = \{\mathcal{G}_j^{ta}\}_{j=1}^{N_{ta}}$	Unlabeled target dataset with N_{ta} graphs
$\Phi(\cdot)$	The GNN classifier
$\mathcal{N}(v)$	The set of neighbors of node v
$\mathcal{M}_k$	The motifs
W	The Graphon Transformation Matrix, estimating the pairwise edge formation probabilities of a dataset.
A	The adjacency matrix of a graph.
δ, η	The scaling factors of Graphon Transformation Matrix in data fusion.
β, γ	The scaling factors of different losses.

B. Graph Neural Networks

Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with node features X, the representation of node v at the l-th layer of our GNN, denoted as $h_v^{(l)}$, is recursively computed as follows. The neighborhood aggregation stage at layer l synthesizes the feature information from v's immediate neighbors, formally:

$$a_v^{(l)} = \text{AGG}^{(l)} \left(\left\{ h_u^{(l-1)} \mid u \in \mathcal{N}(v) \right\} \right), \tag{1}$$

where $\mathcal{N}(v)$ represents the set of neighbors of node v, and $h_u^{(l-1)}$ stands for the representation of neighboring node u from the prior iteration $(l-1)$. $\text{AGG}^{(l)}$ is the neighborhood aggregation function applied at layer l.

Next, an update mechanism combines the aggregated neighborhood features with the node's existing features to produce a new representation:

$$h_v^{(l)} = \text{UPD}^{(l)} \left(h_v^{(l-1)}, a_v^{(l)} \right), \tag{2}$$

with $\text{UPD}^{(l)}$ being the update function at iteration l, which include non-linear transformations. After L iterations of message-passing, the graph is represented by pooling the node features from the last layer:

$$z_{\mathcal{G}} = \text{READOUT}\left(\left\{h_v^{(L)} \mid v \in \mathcal{V}\right\}\right), \tag{3}$$

where READOUT is a permutation invariant function, such as sum, mean, or a graph-level pooling operation, to obtain an entire graph's representation, $z_{\mathcal{G}}$.

The classification of the graph is conducted using a multi-layer perceptron (MLP) with a softmax activation to convert the pooled graph representation into a probability distribution over labels:

$$\hat{\mathbf{y}}_{\mathcal{G}} = \Phi(\mathcal{G}) = \text{HEAD}\left(z_{\mathcal{G}}\right), \tag{4}$$

where Φ represents the graph neural network, HEAD signifies the MLP classifier, and $\hat{\mathbf{y}}_{\mathcal{G}}$ is the predicted label distribution for graph $\mathcal{G}$.

During training on source domain graphs, the GNN's parameters are optimized by minimizing the cross-entropy loss between the predicted and ground true labels:

$$\mathcal{L}_{\text{source}} = -\frac{1}{N_{so}} \sum_{i=1}^{N_{so}} \log\left(\hat{\mathbf{y}}_{\mathcal{G}_i^{so}}[y_i^{so}]\right), \tag{5}$$

where y_i^{so} are the ground truth labels of source graphs, and $\hat{\mathbf{y}}_{\mathcal{G}_i^{so}}$ are the corresponding predictions from the GNN.

IV. Methodology

A. Framework Overview

To address the challenge of domain discrepancy in unsupervised graph domain adaptation, our proposed Multi-View Teacher with Curriculum Data Fusion (MTDF) framework takes a comprehensive approach, illustrated in Figure 2. As a foundational step, we use the graph neural network (GNN) trained on source data to generate pseudo-labels for the graphs in the target domain. These initial pseudo-labels can be used for subsequent adaptation steps.

The heart of MTDF lies a multi-teacher framework, designed to ensure effective learning and stable progress. (See Section IV-B) The first teacher, the booster, is designed to encourage adaptability in the learning process. The second teacher, the stabilizer, is assigned the task of maintaining a steady process throughout training. Together, they guide the learning process, allowing for adaptation to the target domain.

Following this, the multi-view framework comes into play. (See Section IV-C) The motif-based network dives into the high-level structural patterns within the source domain data to align features more effectively. By concentrating on these explicit graph topologies, this branch supports the GNN backbone by consistency learning, enabling it to utilize the structural information between the domains.

Finally, we introduce curriculum data fusion, a gradual blending process of source and target domain information. (See Section IV-D) Starting simple and growing more complex, this process reflects the concept of a learning curriculum, integrating structural knowledge from both domains. By incrementally exposing the model to the target structures, we facilitate a stable and effective domain adaptation.

B. Multi Teacher Method

Capitalizing on the Mean Teacher paradigm [23], we adapt this semi-supervised learning framework to unsupervised domain adaptation. Our multi-teacher (MuT) method employs two teacher models, *Booster Teacher* (BT) and *stabilizer Teacher* (ST), which have the same architecture and initialization as the pre-trained GNN model. Then, we fuse their predictions to provide a unified guide for the student model, which enhances the model's robustness against domain shift. Let Φ_θ be our student model with parameters θ and $\Phi_{\theta^{\text{BT}}}$ represent the BT model with parameters θ^{BT}. The parameters of BT at iteration $i+1$ are updated using the exponential moving average (EMA) of the student's parameters:

$$\theta_{i+1}^{\text{BT}} = \alpha\theta_i^{\text{BT}} + (1-\alpha)\theta_i\,, \tag{6}$$

where α is a smoothing coefficient that controls the update rate, which is set to a constant value of 0.99 following the previous practice [23]. In addition to this, we keep track of the ST model with parameters θ^{ST}. At each epoch, we perform *temporal-wise archiving* where we evaluate the BT's performance on the source dataset and update θ^{ST} if the BT surpasses the accuracy of the ST:

$$\theta^{\text{ST}} = \text{argmax}_{\theta^{\text{BT}}}(\text{Acc}(\theta^{\text{BT}}, \mathcal{D}_{so}))\,, \tag{7}$$

where Acc denotes the accuracy function.

Instead of using the multi-teacher models to supervise the student model independently, we unify their predictions before providing feedback to the student. Specifically, for each target graph instance $\mathcal{G}_j^{ta}$, we average the probabilistic outputs of the BT and ST to derive a single supervisory signal:

$$\hat{\mathbf{y}}_{\mathcal{G}_j^{ta}}^{\text{MuT}} = \frac{1}{2}(\hat{\mathbf{y}}_{\mathcal{G}_j^{ta}}^{\text{BT}} + \hat{\mathbf{y}}_{\mathcal{G}_j^{ta}}^{\text{ST}})\,, \tag{8}$$

where $\hat{\mathbf{y}}_{\mathcal{G}_j^{ta}}^{\text{BT}}$ and $\hat{\mathbf{y}}_{\mathcal{G}_j^{ta}}^{\text{ST}}$ are the soft prediction labels from the BT and ST, respectively.

The MuT consistency loss is now represented as the divergence between the student's prediction and this fused supervisory signal:

$$\mathcal{L}_{\text{MuT}}^{(j)} = D_{\text{KL}}\left(\hat{\mathbf{y}}_{\mathcal{G}_j^{ta}}^{\text{MuT}} \parallel \hat{\mathbf{y}}_{\mathcal{G}_j^{ta}}\right), \tag{9}$$

where $\hat{\mathbf{y}}_{\mathcal{G}_j^{ta}}$ corresponds to the student's prediction for target graph $\mathcal{G}_j^{ta}$. The $\mathcal{L}_{\text{MuT}}$ is then the mean of the consistency losses across all target domain graph instances:

$$\mathcal{L}_{\text{MuT}} = \frac{1}{N_{ta}} \sum_{j=1}^{N_{ta}} \mathcal{L}_{\text{MuT}}^{(j)}\,. \tag{10}$$

By feeding the student model with a fused supervisory signal from the two teacher models, the MuT methodology harmonizes their individual predictive strengths. This integration of teacher model outputs is expected to yield a more stable target for the student to learn from, enhancing its performance across the domain shift and ensuring more stable adaptation in the target domain.

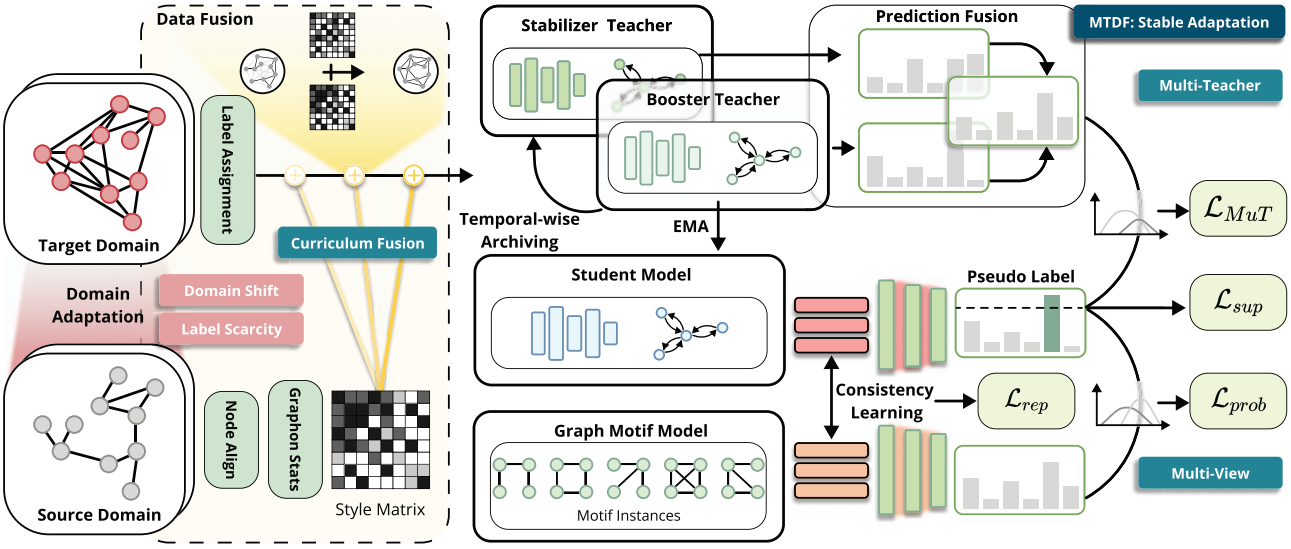


Fig. 2: Overview of MTDF. For stable adaptation in UDA, MTDF employs a multi-teacher framework with *booster* and *stabilizer* teachers. Moreover, MTDF leverages consistency learning for multi-view information from the graph motif model and message-passing model, which contains explicit topology information and implicit relation representation, respectively. We also utilize curriculum cross-domain data fusion mechanisms to bridge the target domain with the source domain, achieving effective domain adaptation.

C. Motif Network Model with Consistency Learning

1) Motif GNN Architecture: Motif-based Graph Neural Networks (MGNNs) [45], [77] capitalize on higher-order structural properties by introducing motif-based substructures to the model. Figure 3 briefly illustrates how MGNN works.

We denote the set of motif instances as $\mathcal{M}_k$, for $k \in \{1, \ldots, K\}$, where K is the number of distinct motifs considered in the analysis. We first construct the motif-based adjacency matrix $A_{\mathcal{M}_k}$ by using the following equation:

$$(A_{\mathcal{M}_k})_{ij} = \sum_{m \in \mathcal{M}_k} \mathbb{I}((x_i, x_j) \in m)\,, \tag{11}$$

where $\mathbb{I}$ represents an indicator function, *i.e.*, $\mathbb{I}(x) = 1$ if the statement x holds true, and $\mathbb{I}(x) = 0$ if it does not. Therefore, $A_{\mathcal{M}_k}$ counts the number of links between each pair of nodes through $\mathcal{M}_k$. Following the previous work [45], [77], the MGNN process can be captured by the simplified equation:

$$h_v^{\mathcal{M}} = \text{MGNN}\left(\mathcal{G}, \mathcal{M}_k, A_{\mathcal{M}_k}, L\right)\,, \tag{12}$$

where MGNN$(\cdot)$ represents the integration of motif-specific features through aggregation, distillation, and pooling across L layers to achieve the final node feature embedding $h_v^{\mathcal{M}}$, illustrated in Figure 3.

Similar to GNN's training, we use the READOUT function and MLP classifier for predicting label distribution.

$$z_{\mathcal{G}}^{\mathcal{M}} = \text{HEAD}\left(\text{READOUT}\left(\left\{h_v^{\mathcal{M}} \mid v \in \mathcal{V}\right\}\right)\right)\,, \tag{13}$$

and optimize parameters by minimizing the cross-entropy loss through the source dataset:

$$\mathcal{L}_{\text{source\_motif}} = -\frac{1}{N_{so}} \sum_{i=1}^{N_{so}} \log\left(\hat{\mathbf{y}}_{\mathcal{G}_i^{so}}^{\mathcal{M}}[\mathbf{y}_i^{so}]\right)\,. \tag{14}$$

MGNN distinguishes itself from traditional GNNs like GCN by integrating motif-based structural insights, capturing complex node relationships and higher-order graph properties. This approach enriches node representations and enhances model adaptability, offering a more detailed and effective methodology for graph unsupervised domain adaptation.

2) Consistency Learning for Domain Adaptation: In our approach, graph-level consistency learning is seamlessly integrated into our MGNN framework to improve UDA performance. This strategy strives to unify the global structural feature distributions of the source and target domain graphs into domain-invariant representations.

To achieve this, we formulate a probabilistic consistency loss, $\mathcal{L}_{\text{prob}}$, to encourage similarity between the predicted label distributions of the target graphs generated by the MGNN, $\hat{\mathbf{y}}_{\mathcal{G}}^{\mathcal{M}}$, and those predicted by the pre-trained GNN, $\hat{\mathbf{y}}_{\mathcal{G}}$. This loss enforces that the probability vectors produced by both models for the same graph should be similar, thus preserving label consistency across models:

$$\mathcal{L}_{\text{prob}} = -\frac{1}{N_{ta}} \sum_{i=1}^{N_{ta}} \log\left(\hat{\mathbf{y}}_{\mathcal{G}_i^{ta}}^{\mathcal{M}}[\hat{\mathbf{y}}_{\mathcal{G}_i^{ta}}]\right)\,. \tag{15}$$

Moreover, we define an additional consistency loss, $\mathcal{L}_{\text{rep}}$, that minimizes the distance between the graph-level representations from the MGNN, $z_{\mathcal{G}}^{\mathcal{M}}$, and the corresponding representations from the original GNN, $z_{\mathcal{G}}$:

$$\mathcal{L}_{\text{rep}} = \frac{1}{N_{ta}} \sum_{i=1}^{N_{ta}} \| z_{\mathcal{G}_i^{ta}}^{\mathcal{M}} - z_{\mathcal{G}_i^{ta}} \|_2^2\,, \tag{16}$$

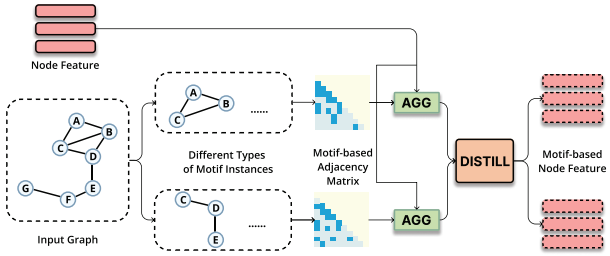


Fig. 3: Illustration of Motif-based Graph Neural Networks (MGNN). MGNN captures and integrates motif-specific features through motif-specialized aggregation, distillation, and average pooling, enabling effective modeling of higher-order structural properties in graph data.

where $\|\cdot\|_2$ represents the L2 norm. $\mathcal{L}_{\text{rep}}$ aligns the global graph representations learned by MGNN with those of the original GNN, thereby facilitating the transfer of knowledge while mitigating overfitting to motif-specific features in complex graph structures.

By uniting these two facets of consistency, *i.e.*, probabilistic and representational, we arrive at a composite motif-consistency loss, denoted as $\mathcal{L}_{\text{con}}$, which embodies both coherent predictions and aligned graph-level representations:

$$\mathcal{L}_{\text{con}} = \mathcal{L}_{\text{rep}} + \mathcal{L}_{\text{prob}} \,. \tag{17}$$

This combined loss aligns both label distributions and graph-level representations between source and target domains. This alignment, achieved through probabilistic and representational consistency losses, ensures coherent predictions and stable graph feature representations, enhancing unsupervised domain adaptation performance.

D. Cross-Domain Data Fusion with Curriculum Learning

Building upon the framework of GNNs and the unsupervised domain adaptation, we propose a novel methodology to fuse structural patterns across domains, enriching the training process with a curriculum learning strategy. Inspired by [78], [79], we create and utilize a graphon transformation matrix (GTM), a novel representation capturing the structure of a dataset, for cross-domain data fusion.

Formally, we define the GTM for domain $\mathcal{D}$ as $W_{\mathcal{D}} \in [0,1]^{|\mathcal{V}|\times|\mathcal{V}|}$, constructed by first ordering nodes based on their degree and then estimating the pairwise edge formation probabilities. The GTM serves as a high-level abstraction of domain-specific topological profiles, reflecting the likelihood of connectivity patterns among node pairs.

To achieve cross-domain data fusion, we introduce a dynamic fusion mechanism with a curriculum schedule $\lambda(t) \in [0,1]$, where t denotes the training time. This parameter makes the mixed GTM initially biased towards the source GTM $W_{\mathcal{D}_{so}}$, then gradually shifted to incorporate the target GTM $W_{\mathcal{D}_{ta}}$ as training progresses. The mixed GTM $\widetilde{W}(t)$ is computed through a convex combination:

$$\widetilde{W}(t) = \lambda(t)\eta W_{\mathcal{D}_{so}} + (1-\lambda(t))\delta W_{\mathcal{D}_{ta}} \,, \tag{18}$$

where $\lambda(t)$ is designed to decay from 1 to 0 following a linear schedule, and η, δ are the scaling factors for two GTMs, thus applying the curriculum learning paradigm to the domain adaptation process.

Given a target graph $\mathcal{G}^{ta}$ with its adjacency matrix A^{ta}, we synthesize an adjacency matrix $\widehat{A}^{ta}$ that merges $\widetilde{W}(t)$ with A^{ta}. This is achieved by using the mixed GTM to adjust the existing adjacency matrix, producing edge probabilities for the transformed target graph. The updated equation for obtaining the edge probability between nodes u and v in $\widehat{A}^{ta}$ is:

$$\widehat{A}^{ta}_{uv}(t) = (1-(\eta-\delta)\lambda(t)-\delta)A^{ta}_{uv} + \widetilde{W}_{uv}(t) \,, \tag{19}$$

where $\widehat{A}^{ta}_{uv}(t)$ represents the probability of an edge between nodes u and v in the synthesized adjacency matrix at training time t. Edges in the newly synthesized graph are then sampled based on these probabilities:

$$p(\widehat{A}^{ta}_{uv} = 1 \mid \widehat{A}^{ta}(t)) = \widehat{A}^{ta}_{uv}(t) \,. \tag{20}$$

Using this approach, we ensure that the inherent structure of the target graph is respected while still allowing the curriculum-driven GTM mixing strategy to introduce domain-adapted structural differences over time.

To continue with the subsequent loss computation and optimization, we use the fused probability matrix $\widehat{A}^{ta}(t)$ to generate the fused graphs $\widehat{\mathcal{G}}^{ta}_j(t)$.

We first generate the confidence set $\mathcal{C}$ for the pseudo-label learning as:

$$\mathcal{C} = \{\widehat{\mathcal{G}}^{ta}_j | c = \arg\max_{c'} p^{ta}_j \,,\, s^{ta}_j > \tau\} \,, \tag{21}$$

where s^{ta}_j is the confidence score, which is the maximum of the logits. Moreover, τ is the confidence score threshold, which is set to 0.90 following [80].

Then we have the pseudo-label classification loss on the target domain as:

$$\mathcal{L}_{\text{cls}}(t) = -\frac{1}{|\mathcal{C}|} \sum_{\mathcal{G}^{ta}_j \in \mathcal{G}} \log\left(p^{ta}_j\left[\hat{\mathbf{y}}_{\widehat{\mathcal{G}}^{ta}_j(t)}\right]\right) \,, \tag{22}$$

where $\hat{\mathbf{y}}_{\widehat{\mathcal{G}}^{ta}_j(t)}$ is the pseudo-label of sample $\widehat{\mathcal{G}}^{ta}_j$, which is generated from the class with maximum logits in the confident set $\mathcal{C}$. $\mathcal{L}_{\text{cls}}$ stands for the curriculum-driven classification loss reflecting the updated methodology. This approach aligns with the unsupervised domain adaptation objective, focusing on balancing learning from the original target graph structure and the domain-adaptive GTM representation.

E. Summarization

Our approach exploits the underlying graph structure by leveraging a classification loss (L_{cls}), a feature space matching loss (L_{MuT}), and a consistency loss (L_{con}). Consequently, the composite loss function is formalized as:

$$L_{total} = L_{cls} + \beta L_{MuT} + \gamma L_{con} \,, \tag{23}$$

where β and γ are hyperparameters controlling the significance of each loss component. Our unsupervised domain adaption

Algorithm 1 Optimization Algorithm of MTDF

Input: Source dataset $\mathcal{D}^{so}$, target dataset $\mathcal{D}^{ta}$,
Output: GNN-based classifier $\Phi(\cdot)$

Pre-train $\Phi(\cdot)$ using $\mathcal{D}^{so}$;
Train Motif-GNN using source graphs by minimizing Eq. 14;
Booster teacher model $\Phi_{BT}(\cdot) \leftarrow \Phi(\cdot)$;
Stabilizer teacher model $\Phi_{ST}(\cdot) \leftarrow \Phi(\cdot)$;
Generate confident target graphs with pseudo-labels;
for epoch = 1, 2, $\cdots$ **do**
  for each batch **do**
    Sample a mini-batch of target graphs;
    Calculate the loss objective using Eq. 23;
    Update parameters of $\Phi(\cdot)$ by back-propagation;
  end for
  Update $\Phi_{BT}(\cdot)$ using Eq. 6;
  if $\Phi_{BT}(\cdot)$ is better than $\Phi_{ST}(\cdot)$ on $\mathcal{D}^{so}$ **then**
    $\Phi_{ST}(\cdot) \leftarrow \Phi_{BT}(\cdot)$;
  end if
end for

framework is detailed in Algorithm 1 for an iterative optimization procedure.

The computing complexity of the adaptation in MTDF primarily relies on two types of networks. For given graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, d is the feature dimension. $||A_{\mathcal{M}}||_0$ is the number of nonzeros in all motif-based adjacency matrixes, which is linearly related to K. L_1 and L_2 denote the layer number of GCN and MGNN respectively. The GCN takes $\mathcal{O}(L_1|\mathcal{E}|d + L_1|\mathcal{V}|d^2)$ computational time while the MGNN takes $\mathcal{O}(L_2||A_{\mathcal{M}}||_0 d + L_2|V|d^2)$. As a result, the complexity of our MTDF is proportional to $|\mathcal{V}|$, $|\mathcal{E}|$, and $||A_{\mathcal{M}}||_0$.

V. Experiments

A. Experimental Settings

1) Datasets: In our experiments, we employ a variety of real-world datasets, from cheminformatics to social networks, for unsupervised graph domain adaptation. Notably, we use both splitting datasets and crossing datasets to evaluate the efficacy of our approach.

Specifically, we leverage the NCI1 dataset [81], [82], a cornerstone in cheminformatics research, for screening anti-cancer compounds, particularly focusing on lung cancer. The DD dataset [83] aids in classifying protein structures into enzymes and non-enzymes. The TWITTER-Real-Graph-Partial dataset [84], utilized for sentiment analysis of tweets, underscores its relevance in social network analysis. The PTC dataset [85] facilitates toxicology prediction in chemical compounds and is segmented into different domains based on rodent sex and species. Its importance in safety and regulatory assessments cannot be overstated, as it helps in evaluating chemical safety and environmental impact. Additionally, we incorporate the BZR, COX2, and DHFR datasets [86], [87], along with their MD variants. These datasets are segmented into different domains based on chemical spaces and are utilized for classifying biochemical ligands and inhibitors.

TABLE II: Statistics of the datasets.

Datasets		Graphs	Avg. Nodes	Avg. Edges
NCI1		4110	29.87	32.30
TWITTER-Real-Graph-Partial		144033	4.03	4.98
DD		1178	284.32	715.66
PTC	PTC_FM	349	14.11	14.48
	PTC_FR	351	14.56	15.00
	PTC_MM	336	13.97	14.32
	PTC_MR	344	14.29	14.69
BZR	BZR	405	35.75	38.36
	BZR_MD	306	21.30	225.06
COX2	COX2	467	41.22	43.45
	COX2_MD	303	26.28	335.12
DHFR	DHFR	756	42.43	44.54
	DHFR_MD	393	23.87	283.02

The statistics of these datasets are shown in Table II. In our experimental setup, we perform unsupervised domain adaptation across these datasets by considering permutations of source and target domains, such as PTC_FM→PTC_FR and BZR→BZR_MD. As we focus on classification tasks, our evaluation metric is the classification accuracy on the target domain, which is crucial for assessing the effectiveness of our domain adaptation approach.

2) Domains: The domain adaptation experiments on the TWITTER-Real-Graph-Partial, DD, and NCI1 datasets are designed to analyze the performance of MTDF in splitting datasets. To achieve this, all datasets are partitioned into four discrete groups, $T0, T1, T2$, and $T3$ for Twitter, $D0, D1, D2$, and $D3$ for DD, and $N0, N1, N2$, and $N3$ for NCI1, with the groupings reflecting incremental ranges of graph density. Graph density is defined as the equation:

$$\Psi(\mathcal{V}, \mathcal{E}) = \frac{2|\mathcal{E}|}{|\mathcal{V}|(|\mathcal{V}| - 1)}, \tag{24}$$

where $|\mathcal{E}|$ represents the number of edges and $|\mathcal{V}|$ represents the number of nodes. We incorporated domain shift by dividing the single dataset through the density to create different domains for experiments.

For the PTC, BZR/BZR_MD, COX2/COX2_MD, and DHFR/DHFR_MD datasets, the domain adaptation trials are conducted following a multi-dataset approach. Each labeled group within a dataset (*e.g.*, PTC_FM, PTC_FR, PTC_MM, and PTC_MR) serves as an isolated domain. Our approach investigates the transferability of models across these domains, which involve shifts in chemical structure spaces or biological properties. The pre-trained models initially tuned on the source domain (*e.g.*, PTC_FM) are adapted to the target domain (*e.g.*, PTC_MM), utilizing both labeled source domain data and unlabeled target domain data during the adaptation period.

3) Baselines: We compare **MTDF** to an extensive range of state-of-the-art baselines. These baselines are detailed below:

TABLE III: The classification results (in %) on NCI1 (source→target). N0, N1, N2, and N3 are split by the graph density.

Methods	N0→N1	N0→N2	N0→N3	N1→N0	N1→N2	N1→N3	N2→N0	N2→N1	N2→N3	N3→N0	N3→N1	N3→N2	Avg.
GIN	66.0	60.6	50.3	68.0	68.4	69.9	61.0	65.6	73.1	48.3	59.4	62.9	62.8
GCN	55.8	59.1	54.0	73.3	65.0	70.7	73.5	60.7	70.2	67.8	54.5	55.1	63.3
GAT	63.4	60.0	41.7	70.1	68.2	70.1	73.2	63.1	69.3	56.6	56.3	60.5	62.7
SAGE	54.9	55.8	50.1	74.5	59.7	66.0	76.2	59.7	71.7	70.6	57.2	64.8	63.4
MeanTeacher	54.9	45.2	51.6	73.8	45.2	50.7	73.3	54.9	50.2	72.8	55.8	47.1	56.3
InfoGraph	66.5	61.0	57.6	62.7	64.6	64.1	75.7	62.6	67.1	69.9	60.7	50.2	63.6
DANN	64.1	58.7	45.6	76.2	69.8	63.6	71.3	**70.9**	70.0	70.4	58.3	**67.5**	65.5
ToAlign	65.5	61.7	47.1	73.3	**69.9**	59.7	71.4	69.9	69.9	68.0	59.2	63.1	64.9
UDAGCN	59.8	53.9	53.9	75.8	62.5	68.2	71.3	62.1	68.4	57.7	52.9	58.8	62.1
Ours	**67.5**	**70.9**	**71.8**	**76.7**	65.0	**73.1**	**77.2**	62.5	**74.3**	**75.9**	**61.0**	57.8	**69.5**

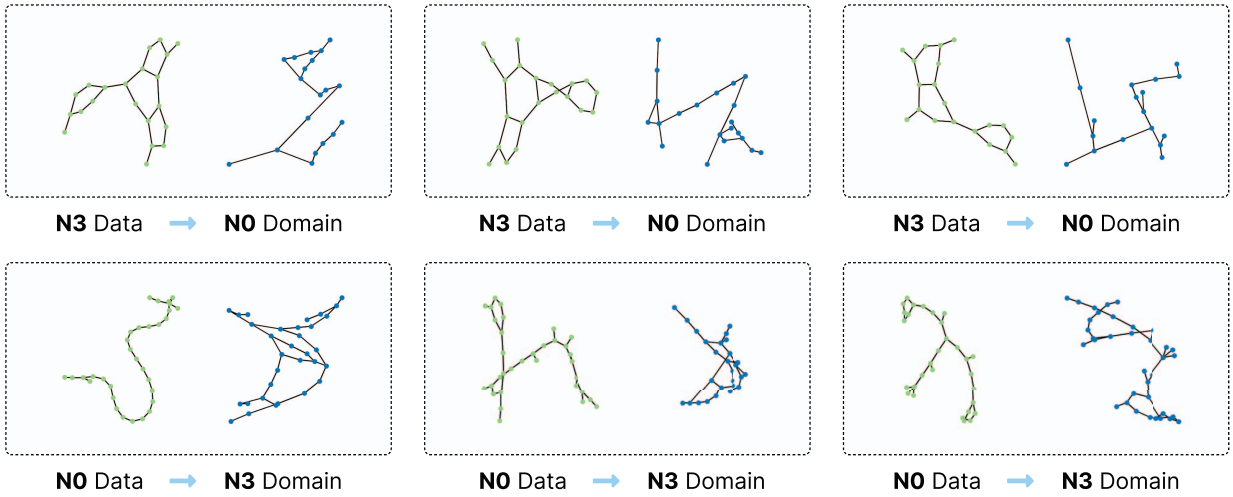


Fig. 4: Visualization of data fusion process on NCI1 dataset. N0 and N3 are subsets of NCI1, with N0 being sparser and N3 being denser. As $\mathbf{N3} \rightarrow \mathbf{N0}$, the graphs become sparser. As $\mathbf{N0} \rightarrow \mathbf{N3}$, the graphs become denser. The proposed curriculum data fusion strategy is shown to effectively merge the style of the source domain graphs while maintaining the semantics of the target domain graphs, thereby facilitating a more stable domain adaptation process.

- **GIN** [40]: This message passing neural network generalizes the powerful Weisfeiler-Lehman graph isomorphism test, and is designed to capture a wide variety of topological graph structures more accurately.
- **GCN** [15]: A widely-used graph neural network that utilizes a first-order approximation of spectral graph convolutions, GCN adeptly embeds structural information, along with node features, into its framework.
- **GAT** [41]: Graph Attention Networks leverage attention mechanisms to weigh neighbors differently, enabling flexible, localized learning on graphs without the need for costly spectral operations.
- **GraphSAGE** [42]: GraphSAGE innovates on inductive learning for large graphs by aggregating sampled neighborhood features, facilitating learning on unseen nodes during the training process.
- **Mean-Teacher** [23]: Mean-Teacher employs a student-teacher paradigm, improving semi-supervised learning by maintaining a consistency between the teacher (an averaged student model) and the student predictions.
- **InfoGraph** [70]: InfoGraph maximizes the mutual information between different graph scales to learn powerful graph representations in a semi-supervised GNN setting.
- **DANN** [60]: Domain-adversarial neural Networks enhance domain adaptation by learning features that serve the source task while remaining domain-neutral, leveraging a gradient reversal layer for aligning distributions from domains with minimal architecture changes.
- **ToAlign** [88]: Operating under the tutelage of task-induced prior knowledge, ToAlign executes a methodical feature decomposition and alignment between the source and target domains.
- **UDAGCN** [16]: UDAGCN combines graph convolutional networks with domain adaptation to minimize the distribution discrepancy between source and target domains for robust graph-based learning.

By comparing our approach to these methods, we can effectively measure the performance of our approach.

4) Implementation Details: The baseline methods are implemented by using PyTorch and PyTorch Geometric library,

TABLE IV: The classification results (in %) on TWITTER-Real-Graph-Partial (source→target). T0, T1, T2, and T3 are split by the graph density.

Methods	T0→T1	T0→T2	T0→T3	T1→T0	T1→T2	T1→T3	T2→T0	T2→T1	T2→T3	T3→T0	T3→T1	T3→T2	Avg.
GIN	59.7	62.8	60.4	64.2	62.2	61.3	61.7	63.2	61.0	62.3	61.8	62.4	61.9
GCN	62.0	62.9	59.7	64.1	63.4	59.8	64.2	62.8	60.5	62.7	61.4	62.7	62.2
GAT	60.6	63.2	60.0	63.1	61.6	59.8	63.5	61.6	59.5	**63.4**	62.1	63.7	61.9
SAGE	61.0	64.6	62.1	61.9	61.9	60.8	62.9	62.6	60.9	61.7	60.9	63.4	62.1
MeanTeacher	52.2	49.2	46.1	49.0	50.7	46.1	49.5	51.7	52.6	48.1	48.0	51.1	49.5
InfoGraph	63.9	65.1	61.6	**65.6**	65.0	59.2	64.3	63.3	60.8	63.3	62.4	63.3	63.2
DANN	58.4	60.0	58.0	59.0	59.4	57.4	57.7	58.1	58.4	58.2	57.9	60.4	58.6
ToAlign	58.6	59.5	55.5	57.7	58.1	56.1	56.3	57.2	57.8	57.7	57.6	60.2	57.7
UDAGCN	64.4	65.8	64.8	64.5	65.0	**64.6**	62.7	62.7	61.2	61.7	62.2	60.0	63.3
Ours	**64.5**	**66.4**	**65.1**	64.7	**65.2**	62.2	**64.9**	**63.5**	**63.2**	63.2	**63.4**	**64.4**	**64.2**

TABLE V: The classification results (in %) on PTC (source→target).

Methods	MR→MM	MR→FM	MR→FR	MM→MR	MM→FM	MM→FR	FM→MR	FM→MM	MR→FR	FR→MR	FR→MM	FR→FM	Avg.
GIN	61.8	64.3	57.7	56.5	45.7	53.5	37.7	42.6	44.5	59.4	66.2	54.3	53.7
GCN	63.2	62.9	66.2	55.1	45.7	67.6	62.3	54.4	64.8	58.0	60.3	52.0	59.4
GAT	60.0	46.0	70.7	57.1	46.3	65.9	54.5	53.8	53.2	69.0	65.9	51.7	57.8
SAGE	58.8	55.1	67.6	56.5	47.4	66.2	48.1	48.2	45.1	58.0	63.2	52.6	55.6
MeanTeacher	61.8	61.4	73.2	60.9	52.9	50.7	**65.2**	44.1	35.1	66.7	55.9	42.9	55.9
InfoGraph	63.2	60.0	66.2	59.4	48.6	67.6	55.1	56.4	64.8	63.8	69.1	54.3	60.7
DANN	**76.5**	64.3	69.0	63.8	55.7	73.2	50.7	48.5	66.2	71.0	70.6	52.9	63.5
ToAlign	73.5	45.7	67.6	**66.7**	54.3	67.6	58.0	50.0	67.6	71.0	**76.5**	55.7	62.9
UDAGCN	68.8	62.3	70.1	58.4	55.6	71.5	62.9	55.8	66.9	69.0	66.1	55.7	63.6
Ours	65.9	**64.8**	**76.9**	61.2	**56.0**	**73.9**	62.6	**56.8**	**69.0**	**71.1**	71.6	**56.0**	**65.5**

which are initiated with hyperparameters as the corresponding paper and fine-tune them to optimize performance. As for our MTDF, implementation details in both the source domain training stage and target domain adaptation stage are:

Model Training on the Source Domain: The foundation of our model begins with source domain training, where we employ a GCN encoder to learn initial graph representations. The GCN is configured with a hidden dimensionality of 128 and is structured into 2 layers, optimizing learning in a mini-batch with a size of 256. We utilize the Adam optimizer with a learning rate of 0.001. The training procedure consists of 100 epochs to ensure model convergence. Before domain adaptation, the graph motif model with a hidden dimensionality of 128 and 2 layers is initialized with a preliminary phase of 50 epochs, which focuses on capturing higher-order patterns to be later used for feature alignment purposes.

Domain Adaptation on the Target Domain: In the domain adaptation process, we first generate pseudo-labels from the model trained by source data. The threshold for generating pseudo labels during unsupervised training is firmly set to 0.9, ensuring a high level of confidence in the temporal labels assigned for the target domain data. We attribute a motif model loss weight of 0.3, embodying its significance in feature alignment through consistency learning. The integration of the multi-teacher is set at a loss weight of 0.2. This allows the model to play an effective role in stabilizing the training process and supervision during the adaptation process. We further incorporate data fusion regularization techniques, with the source fusion weight gradually decaying from 0.9 to 0 and the target fusion weight gradually increasing from 0 to 0.3, effectively combining features from both domains to help the model against domain discrepancies. The adaptation process is conducted at 10 epochs using the Adam optimizer with a learning rate of 0.001 and a batch size of 256.

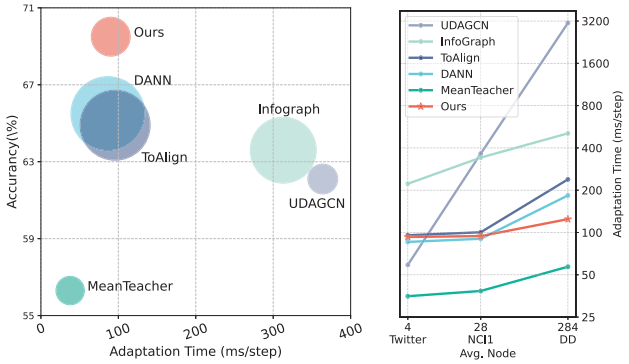
B. Performance Comparison

In this section, we evaluate various methods across several datasets, as shown in Tables III, V, and VII.

First, the experiments show the superior performance of domain adaptation methods over source-only GNN approaches in most cases, demonstrating that domain adaptation strategies are better for handling the situation of domain shifts.

Second, semi-supervised methods (*e.g.*, InfoGraph), generally surpass source-only methods. These approaches can leverage both labeled data from the source domain and unlabeled data from the target domain. However, these methods do not take domain shift into account, leading to poor performance.

Third, domain adaptation methods (*e.g.*, DANN) shows higher performance on unspervised domain adaptation tasks (*e.g.*, BM → B, MR → MM). However, these methods do not specifically consider the problems of non-European data, leading to worse performance sometimes (*e.g.*, in Table IV). In contrast, graph domain adaptation method (*i.e.*, UDAGCN) outperforms methods like DANN as it takes into account the graph structure, which is crucial for the underlying data representation. However, UDAGCN is tailored for node clas-



(a) Efficiency analysis (b) Scalability analysis

Fig. 5: (a) For efficiency analysis, we evaluate the model performance, adaptation speed, and model complexity on the NCI1 dataset. The radius of the circle represents the parameter volume of the method. (b) For scalability analysis, we evaluate the adaptation time of each batch across different sizes of graph and different methods. Both axes are in log scale.

sification tasks, whereas our focus is on graph classification, a domain where MTDF excels by considering the holistic graph features rather than just individual nodes.

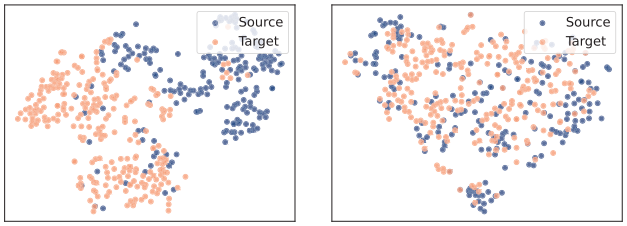
Fourth, our MTDF demonstrates improvements in both splitting datasets and crossing datasets scenarios with a large margin, particularly in the case where other methods perform poorly. There is an accuracy improvement of up to 4.0% across these datasets, as III, IV, V, VI, and VII shows.

The enhancements of MTDF can be summarized into three aspects: 1) The multi-teacher paradigm facilitates the stable adaptation process, allowing the model to learn from the unlabeled target data in data scarcity scenarios. 2) The multi-view framework, combining explicit global information and implicit connection information with consistency learning, enables more comprehensive modeling of graph data. 3) Data fusion with curriculum learning bridges the source and target domain. It helps the stable adaptation of the model by learning more about the source-style structure in the initial phase and gradually introducing target-style structural information.

C. Scalability and Efficiency Analysis

Under the setting in Section V-A4, we examine the scalability and efficiency of every method that has the adaptation process, as shown in Figure 5, in terms of model performance, time cost during adaptation, and model complexity. Our experiments were conducted on a test environment equipped with an RTX 3090 GPU, a 10-core Intel(R) Xeon(R) Platinum 8255C CPU @ 2.50GHz, and 30GB RAM. The MTDF can successfully balance time cost and model complexity to achieve the best performance.

In terms of adaptation time, MTDF takes 90.2ms per batch to adapt, which is competitive with other advanced methods such as DANN and ToAlign, which have adaptation times of 86.5ms and 95.8ms per batch, respectively. Despite having the fastest adaptation time, MeanTeacher does not reflect this efficiency advantage in its final accuracy. In contrast, our method not only maintains a low adaptation time but also achieves the highest accuracy of the evaluated methods, demonstrating superior performance and indicating a robust model that does not compromise quality for speed.



(a) Before data fusion (b) After data fusion

Fig. 6: T-SNE visualization on NCI1 dataset.

In terms of model complexity, our method is of 356K parameters, significantly less than InfoGraph's 1.02M, DANN's 1.28M, and ToAlign's 1.15M. This suggests that it is not the quantity but the quality of parameters that contribute to the superior accuracy of our approach. MTDF does not introduce excessive model complexity in achieving efficient unsupervised domain adaptation.

In evaluating scalability, the MTDF exhibited remarkable efficiency across datasets of varying sizes, as evidenced by the adaptation time cost depicted in Figure 5b. The datasets ranged from Twitter, with an average of 4 nodes, to DD, with an average of 284 nodes, representing a substantial increase in graph complexity. Notably, from Twitter to DD, the adaptation time for our method increased by a mere 32.6%, a modest uptick considering the substantial growth in dataset size. This contrasts sharply with other methods, which all experienced at least a 60.9% increase in adaptation time under the same conditions. This data underscores the MTDF's superior scalability, maintaining both efficiency and performance even as dataset complexity escalates, positioning it as a suitable solution for large-scale graph data applications.

D. Visualization

We demonstrate the effect of the data fusion process through visualization, as shown in Figure 4. We employ a data fusion strategy that not only preserves the essential topological information but also adapts the graph to the patterns in the source domain. This balance between maintaining graph integrity and domain adaptation is visualized on the NCI1 dataset.

We visualize the subsets with the lowest and highest edge densities, labeled N0 and N3 for NCI1. These represent the most challenging scenarios within our adaptation framework, as they have the largest discrepancies in edge density. Our visualizations show the bidirectional process from sparse graphs (N0) to dense graphs (N3) and vice versa. This demonstrates our method's capability to both retain the unique structure of the original graph and integrate features from another domain.

In summary, the MTDF adeptly fuses graphs while meticulously preserving their inherent structures and adapting to new domains, a dual capability that is effectively demonstrated in

TABLE VI: The classification results (in %) on DD (source→target). D0, D1, D2, and D3 are split by the graph density.

Methods	D0→D1	D0→D2	D0→D3	D1→D0	D1→D2	D1→D3	D2→D0	D2→D1	D2→D3	D3→D0	D3→D1	D3→D2	Avg.
GIN	82.6	58.3	61.2	93.0	64.4	62.9	52.5	61.0	66.1	69.8	59.7	59.3	65.9
GCN	80.4	60.0	63.6	93.2	57.6	61.0	91.5	79.6	67.8	63.2	63.4	59.5	70.1
GAT	80.3	59.0	62.9	94.9	62.7	62.7	93.2	79.6	76.3	60.2	60.0	56.3	70.7
SAGE	80.9	60.2	64.4	93.4	59.3	62.8	93.2	78.0	71.2	55.9	57.6	57.2	69.5
MeanTeacher	79.3	60.3	71.1	83.1	57.6	69.5	74.4	72.6	71.2	76.4	69.3	64.4	70.8
InfoGraph	77.8	59.2	70.9	86.9	59.2	67.6	76.1	71.0	69.3	74.3	75.9	57.6	70.5
DANN	**84.7**	67.8	71.0	88.1	**67.8**	65.9	84.6	74.6	74.4	80.8	76.8	57.7	74.5
ToAlign	81.4	**69.5**	74.4	86.4	66.1	69.3	81.2	72.8	74.7	74.4	70.7	59.3	73.3
UDAGCN	79.4	57.3	69.0	86.5	59.4	70.7	86.5	79.4	64.7	69.3	68.9	53.8	70.4
Ours	81.4	61.0	**74.6**	**96.6**	59.3	**74.6**	**94.9**	**79.7**	**76.3**	**94.7**	**79.9**	**66.1**	**78.2**

TABLE VII: The classification results (in %) on COX2,BZR and DHFR (source→target). B, BM, C, CM, D and, DM denote BZR, BZR_MD, COX2, COX2_MD, DHFR, and DHFR_MD datasets, respectively.

Methods	B→BM	BM→B	C→CM	CM→C	D→DM	DM→D	Avg.
GIN	54.3	79.1	45.9	61.7	57.0	67.9	61.0
GCN	46.8	79.0	57.4	76.6	51.5	68.4	63.3
GAT	49.4	79.5	52.6	68.1	49.3	67.1	61.0
SAGE	48.1	65.4	58.5	59.2	55.2	54.1	56.7
MeanTeacher	54.8	76.5	49.2	76.7	42.8	61.2	60.2
InfoGraph	56.5	77.8	57.4	72.3	55.4	55.4	62.5
DANN	54.8	**84.0**	52.6	67.0	63.4	58.2	63.3
ToAlign	54.9	79.0	57.6	76.6	55.4	62.9	64.4
UDAGCN	57.9	71.3	56.0	75.2	57.0	66.1	63.9
Ours	**58.1**	79.8	**59.0**	**77.0**	**64.6**	**70.2**	**68.1**

the NCI1 dataset visualizations. Figure 6 further substantiates the efficacy of our data fusion process. Before data fusion, the graph embeddings are distinctly clustered into two separate groups, indicative of the gap between source and target data. However, after applying our data fusion strategy, the embeddings coalesce, reflecting a significantly reduced discrepancy between the domains. This visual evidence underscores the MTDF's proficiency in narrowing the domain gap, thereby simplifying the adaptation process.

E. Ablation Study

In this section, we present ablation experiments to analyze the effectiveness of each component of our MTDF. The complete model consists of several techniques: the multi-teacher (MuT) framework with *booster* and *stabilizer*, the multi-view complementary framework with consistency learning, and cross-domain data fusion with curriculum learning. Our ablation study evaluates the effectiveness of each component by adding components to the baseline model.

Table VIII summarizes the results of the ablation study. Each row represents the performance of the method when a component is added to the baseline. Several observations can be made from the table:

1) Component Contributions: Each submodule contributes to the performance upon the baseline. The multi-teacher (line w/ MuT) significantly improves model stability by temporally combining *booster* and *stabilizer*. The multi-view framework introduces a graph motif branch (line w/ motif) to exploit structural information in the graph and facilitate the discrimination of feature representations. In addition, the inclusion of the data fusion process (line w/ Fusion-s, Fusion-t, and Fusion) bridges the source domain with the target domain and contributes to stable adaptation.

2) Comprehensive Integration: Our proposed method MTDF encapsulates all of the techniques mentioned above. Remarkably, MTDF consistently outperforms each method on almost all subtasks, demonstrating that these components work well when integrated. The superiority of our complete model validates the hypothesis that the combination of these techniques addresses domain shift and data scarcity, thus providing a more robust framework for graph UDA.

3) Data Fusion Insights: To fully validate the data fusion module, we study it with three variants, *i.e*., data fusion with source data only (line w/ Fusion-s), data fusion with target data only (line w/ Fusion-t), and data fusion with both source and target data (line w/ Fusion). All three variants contribute to the improvements over the baseline but also play different roles. In particular, Fusion-s outperforms Fusion-t, indicating that starting the model with a richer representation of the source domain is more important than adding the global target domain information during the adaptation process. Nevertheless, each fusion variant is critical, as their combination exploits the strengths of the data from both domains and achieves better performance than either one individually.

The overall results of the ablation study demonstrate not only the effectiveness of each technique but also that all techniques work together efficiently. Ultimately, the combination of these techniques in MTDF leads to state-of-the-art performance for graph UDA, as the results in Table VIII.

F. Hyper-parameter Sensitivity

In this section, we analyze the sensitivity of the hyperparameters on the serval datasets shown in Figure 7, 8, and 9. These hyperparameters include loss weights and fusion weights, none of which have a large effect on the results of the method when varied. The specific analysis is as follows.

1) Analysis to the loss weight of multi-teachers: The loss weight β helps to balance the training object in our multi-teacher mechanism. By varying the weight β from 0 to 1,

TABLE VIII: The ablation study classification results (in %) on NCI1 (source→target). N0 - N3 are split by the graph density.

Methods	N0→N1	N0→N2	N0→N3	N1→N0	N1→N2	N1→N3	N2→N0	N2→N1	N2→N3	N3→N0	N3→N1	N3→N2	Avg.
Baseline	55.8	59.1	54.0	73.3	65.0	70.7	73.5	60.7	70.2	67.8	54.5	55.1	63.3
w/ MuT	62.6	57.3	63.1	76.2	64.3	72.1	74.5	55.6	72.8	74.8	57.5	58.7	65.8
w/ Motif	64.1	60.8	68.2	75.5	**67.0**	72.2	63.6	58.9	72.2	71.7	54.9	**59.3**	65.7
w/ Fusion-s	63.6	58.0	68.7	75.7	58.3	71.8	73.8	50.5	73.1	74.3	54.9	57.3	65.0
w/ Fusion-t	62.9	58.5	65.5	74.8	62.9	71.8	74.8	54.6	72.8	72.8	50.0	55.6	64.7
w/ Fusion	60.4	59.0	70.1	75.7	63.6	72.6	74.0	57.5	72.3	74.3	57.3	59.2	66.3
Ours	**67.5**	**70.9**	**71.8**	**76.7**	65.0	**73.1**	**77.2**	**62.5**	**74.3**	**75.9**	**61.0**	57.8	**69.5**

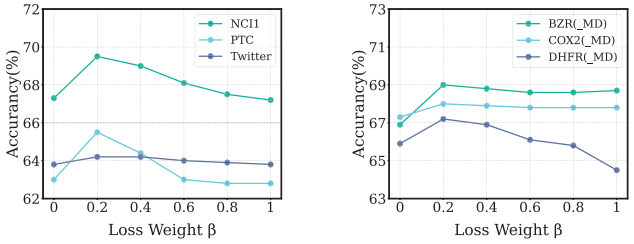


Fig. 7: Sensitivity analysis on loss weight of multi teachers.

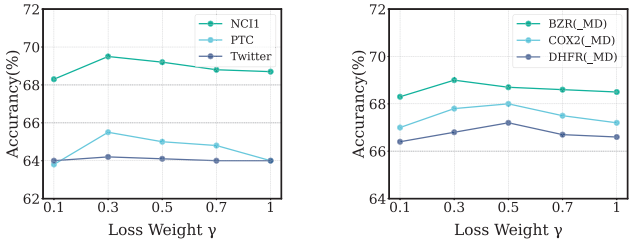
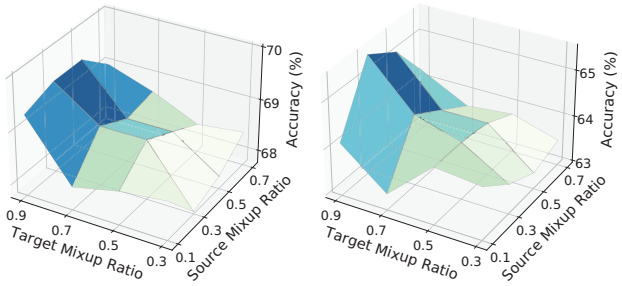


Fig. 8: Sensitivity analysis on loss weight of motif network.

as shown in Figure 7, we can see some influence on the adaptation performance. We find that setting the weight to 0.2 gives the best average accuracy across different datasets. A positive but low weight on the consistency loss seems to favor the stability and trustworthiness of the pseudo-labels by allowing the influence of the teachers to complement, rather than replace, the primary learning signals. This balance ensures that our model remains responsive to the dynamic nature of the unsupervised domain adaptation task.

2) Analysis to the loss weight of multi-view framework: The loss weight γ is used to balance the loss of consistency learning of the motif-based branch, which introduces the explicit structural information. Our empirical investigation of the weight ranges from 0.1 to 1, with the finding that the model generally performs better at a weight of 0.3, as shown in Figure 8. In addition, changing γ has little effect on performance, suggesting that the introduction of multi-view information itself, rather than the hyperparameter, has a more notable effect on model improvement.

3) Analysis of data fusion coefficient weights for source and target domains: We attentively explore the interaction between source and target data fusion weights. Our investigation includes combinations in the range of $\{0.3, 0.5, 0.7, 0.9\}\times$



(a) NCI1 dataset (b) PTC dataset

Fig. 9: Sensitivity analysis on data fusion weights.

$\{0.1, 0.3, 0.5, 0.7\}$, where we identify $(0.9, 0.3)$ as optimal, as shown in Figure 9. We infer that a higher source fusion weight provides the model with a smoother start, facilitating the integration of source domain knowledge. Conversely, the target domain benefits from a moderate fusion weight, avoiding the model from being overly biased towards them. This fusion method effectively exploits the structural similarities between domains while maintaining a balance between exploration and exploitation during the UDA process.

VI. Conclusion

In this work, we have investigated the challenging issue of unsupervised graph domain adaptation. Our novel framework, MTDF, features a multi-view teacher architecture, integrating with a multi-teacher-based learning scheme with different update strategies and a multi-view framework with a motif-based network for explicit structural features. Additionally, it adapts data fusion through the curriculum learning mechanism. Through thorough empirical validation across various datasets, our method not only sets a new state-of-the-art for counteracting domain shift and data scarcity but also paves the way for further research of graph UDA. The remarkable performance of MTDF also demonstrates the significant potential of graph domain adaptation in the scenarios lacking labeled target data.

Acknowledgment

We thank all reviewers for their constructive comments. This work was supported by the National Natural Science Foundation of China (No. U23B2048 and U22B2037) and the High-performance Computing Platform of Peking University.

REFERENCES

[1] C. Lu, Q. Liu, C. Wang, Z. Huang, P. Lin, and L. He, "Molecular property prediction: A multilevel quantum interactions modeling perspective," in *Proc. of Association for the Advancement of Artificial Intelligence*, 2019.

[2] S. Kim, D. Lee, S. Kang, S. Lee, and H. Yu, "Learning topology-specific experts for molecular property prediction," in *Proc. of Association for the Advancement of Artificial Intelligence*, 2023.

[3] S. Fang, K. Zhao, G. Li, and J. Xu Yu, "Community search: A meta-learning approach," in *Proc. of IEEE International Conference on Data Engineering (ICDE)*, 2023.

[4] Y. Liu, Z. Liu, X. Feng, and Z. Li, "Robust attributed network embedding preserving community information," in *Proc. of IEEE International Conference on Data Engineering (ICDE)*, 2022.

[5] D. A. Tedjopurnomo, Z. Bao, B. Zheng, F. M. Choudhury, and A. K. Qin, "A survey on modern deep neural network for traffic prediction: Trends, methods and challenges," in *Proc. of IEEE International Conference on Data Engineering (ICDE)*, 2023.

[6] H. Yu, X. Guo, X. Luo, W. Bian, and T. Zhang, "Construct trip graphs by using taxi trajectory data," *Data Science and Engineering*, 2023.

[7] J. Hu, C. Guo, B. Yang, and C. S. Jensen, "Stochastic weight completion for road networks using graph convolutional networks," in *Proc. of IEEE International Conference on Data Engineering (ICDE)*, 2019.

[8] T. Sun, J. Xu, and C. Hu, "An efficient algorithm of star subgraph queries on urban traffic knowledge graph," *Data Science and Engineering*, 2022.

[9] J. Zhang, Y. Cheng, and X. He, "Fault diagnosis of energy networks: A graph embedding learning approach," *IEEE Transactions on Instrumentation and Measurement*, 2022.

[10] M. Cvitkovic, "Supervised learning on relational databases with graph neural networks," *arXiv preprint arXiv:2002.02046*, 2020.

[11] S. Cai, K. Zheng, G. Chen, H. Jagadish, B. C. Ooi, and M. Zhang, "Arm-net: Adaptive relation modeling network for structured data," in *Proceedings of the 2021 International Conference on Management of Data*, 2021.

[12] M. Zhang, Z. Cui, M. Neumann, and Y. Chen, "An end-to-end deep learning architecture for graph classification," in *Proc. of Association for the Advancement of Artificial Intelligence*, 2018.

[13] Z. Ying, J. You, C. Morris, X. Ren, W. Hamilton, and J. Leskovec, "Hierarchical graph representation learning with differentiable pooling," in *Proc. of Neural Information Processing Systems*, 2018.

[14] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[15] M. Welling and T. N. Kipf, "Semi-supervised classification with graph convolutional networks," in *Proc. of International Conference on Learning Representations*, 2016.

[16] M. Wu, S. Pan, C. Zhou, X. Chang, and X. Zhu, "Unsupervised domain adaptive graph convolutional networks," in *Proc. of ACM Web Conference*, 2020.

[17] M. Lin, W. Li, D. Li, Y. Chen, G. Li, and S. Lu, "Multi-domain generalized graph meta learning," in *Proc. of Association for the Advancement of Artificial Intelligence*, 2023.

[18] Y. You, T. Chen, Z. Wang, and Y. Shen, "Graph domain adaptation via theory-grounded spectral regularization," in *Proc. of International Conference on Learning Representations*, 2022.

[19] Z. Hao, C. Lu, Z. Huang, H. Wang, Z. Hu, Q. Liu, E. Chen, and C. Lee, "Asgn: An active semi-supervised graph neural network for molecular property prediction," in *KDD*, 2020.

[20] S. Suresh, P. Li, C. Hao, and J. Neville, "Adversarial graph augmentation to improve graph contrastive learning," in *NeurIPS*, 2021.

[21] M. Long, Z. Cao, J. Wang, and M. I. Jordan, "Conditional adversarial domain adaptation," in *NeurIPS*, 2018.

[22] W. Wang, Y. Cao, J. Zhang, F. He, Z.-J. Zha, Y. Wen, and D. Tao, "Exploring sequence feature alignment for domain adaptive detection transformers," in *Proceedings of the 29th ACM International Conference on Multimedia*, 2021.

[23] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *Proc. of Neural Information Processing Systems*, 2017.

[24] J. Liang, D. Hu, and J. Feng, "Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation," in *Proc. of International Conference on Machine Learning*, 2020.

[25] Y. Zou, Z. Yu, B. Kumar, and J. Wang, "Unsupervised domain adaptation for semantic segmentation via class-balanced self-training," in *ECCV*, 2018.

[26] Z. Luo, J. Lian, H. Huang, H. Jin, and X. Xie, "Ada-gnn: Adapting to local patterns for improving graph neural networks," in *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, 2022.

[27] W. Ju, Z. Fang, Y. Gu, Z. Liu, Q. Long, Z. Qiao, Y. Qin, J. Shen, F. Sun, Z. Xiao, J. Yang, J. Yuan, Y. Zhao, Y. Wang, X. Luo, and M. Zhang, "A comprehensive survey on deep graph representation learning," *Neural Networks*, 2024.

[28] X. Ma, Z. Li, G. Song, and C. Shi, "Learning discrete adaptive receptive fields for graph convolutional networks," *Science China Information Sciences*, 2023.

[29] S. Xiao, D. Zhu, C. Tang, and Z. Huang, "Combining graph contrastive embedding and multi-head cross-attention transfer for cross-domain recommendation," *Data Science and Engineering*, 2023.

[30] S. Dai, Y. Yu, H. Fan, and J. Dong, "Spatio-temporal representation learning with social tie for personalized poi recommendation," *Data Science and Engineering*, 2022.

[31] Z.-Y. Li, M.-S. Chen, Y. Gao, and C.-D. Wang, "Signal contrastive enhanced graph collaborative filtering for recommendation," *Data Science and Engineering*, 2023.

[32] Y. Yuan, D. Ma, A. Zhang, and G. Wang, "Consistent subgraph matching over large graphs," in *Proc. of IEEE International Conference on Data Engineering (ICDE)*, 2022.

[33] B. Liu, F. Zhang, W. Zhang, X. Lin, and Y. Zhang, "Efficient community search with size constraint," in *Proc. of IEEE International Conference on Data Engineering (ICDE)*, 2021.

[34] K. M. Borgwardt, C. S. Ong, S. Schönauer, S. Vishwanathan, A. J. Smola, and H.-P. Kriegel, "Protein function prediction via graph kernels," *Bioinformatics*, 2005.

[35] J. Zhang, Y. Cheng, and X. He, "Fault diagnosis of energy networks based on improved spatial–temporal graph neural network with massive missing data," *IEEE Transactions on Automation Science and Engineering*, 2023.

[36] K. Hansen, F. Biegler, R. Ramakrishnan, W. Pronobis, O. A. Von Lilienfeld, K.-R. Müller, and A. Tkatchenko, "Machine learning predictions of molecular properties: Accurate many-body potentials and nonlocality in chemical space," *The journal of physical chemistry letters*, 2015.

[37] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *ICML*, 2017.

[38] K. M. Borgwardt and H.-P. Kriegel, "Shortest-path kernels on graphs," in *Proc. of IEEE International Conference on Data Mining*, 2005.

[39] P. Yanardag and S. Vishwanathan, "Deep graph kernels," in *KDD*, 2015.

[40] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" in *Proc. of International Conference on Learning Representations*, 2018.

[41] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proc. of International Conference on Learning Representations*, 2018.

[42] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. of Neural Information Processing Systems*, 2017.

[43] J. Baek, M. Kang, and S. J. Hwang, "Accurate learning of graph representations with graph multiset pooling," in *Proc. of International Conference on Learning Representations*, 2021.

[44] Z. Zhang, Q. Liu, H. Wang, C. Lu, and C. Lee, "Motif-based graph self-supervised learning for molecular property prediction," *Cornell University - arXiv,Cornell University - arXiv*, 2021.

[45] X. Chen, R. Cai, Y. Fang, M. Wu, Z. Li, and Z. Hao, "Motif graph neural network," *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[46] S. Huang, Y. Li, Z. Bao, and Z. Li, "Towards efficient motif-based graph partitioning: An adaptive sampling approach," in *Proc. of IEEE International Conference on Data Engineering (ICDE)*, 2021.

[47] W. Ju, S. Yi, Y. Wang, Q. Long, J. Luo, Z. Xiao, and M. Zhang, "A survey of data-efficient graph learning," *arXiv preprint arXiv:2402.00447*, 2024.

[48] Z. Yang, N. Liang, Z. Li, and S. Xie, "A convergence algorithm for graph co-regularized transfer learning," *Science China Information Sciences*, 2023.

[49] T. He, L. Shen, Y. Guo, G. Ding, and Z. Guo, "Secret: Self-consistent pseudo label refinement for unsupervised domain adaptive person re-identification," in *AAAI*, 2022.

[50] N. Yin, M. Wang, Z. Chen, L. Shen, H. Xiong, B. Gu, and X. Luo, "DREAM: Dual structured exploration with mixup for open-set graph domain adaption," in *Proc. of International Conference on Learning Representations*, 2024.

[51] J. Luo, J. Yang, X. Ye, X. Guo, and W. Zhao, "Fedskel: efficient federated learning on heterogeneous systems with skeleton gradients update," in *CIKM*, 2021.

[52] G. Wei, C. Lan, W. Zeng, and Z. Chen, "Metaalign: Coordinating domain alignment and classification for unsupervised domain adaptation," in *Proc. of Computer Vision and Pattern Recognition*, 2021.

[53] Y. Xu, B. Shi, T. Ma, B. Dong, H. Zhou, and Q. Zheng, "Cldg: Contrastive learning on dynamic graphs," in *Proc. of IEEE International Conference on Data Engineering (ICDE)*, 2023.

[54] L. Li, S. Luo, Y. Zhao, C. Shan, Z. Wang, and L. Qin, "Coclep: Contrastive learning-based semi-supervised community search," in *Proc. of IEEE International Conference on Data Engineering (ICDE)*, 2023.

[55] Z. Liu, C. Wang, Y. Lou, and H. Feng, "Fast unsupervised graph embedding via graph zoom learning," in *Proc. of IEEE International Conference on Data Engineering (ICDE)*, 2023.

[56] Z. Li, C. Huang, L. Xia, Y. Xu, and J. Pei, "Spatial-temporal hypergraph self-supervised learning for crime prediction," in *Proc. of IEEE International Conference on Data Engineering (ICDE)*, 2022.

[57] H. Yan, Y. Ding, P. Li, Q. Wang, Y. Xu, and W. Zuo, "Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation," in *Proc. of Computer Vision and Pattern Recognition*, 2017.

[58] W. Zellinger, T. Grubinger, E. Lughofer, T. Natschläger, and S. Saminger-Platz, "Central moment discrepancy (cmd) for domain-invariant representation learning," *Learning*, 2017.

[59] C.-Y. Lee, T. Batra, M. H. Baig, and D. Ulbricht, "Sliced wasserstein discrepancy for unsupervised domain adaptation," in *Proc. of Computer Vision and Pattern Recognition*, 2019.

[60] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *Journal of Machine Learning Research*, 2016.

[61] B. Hu, Z.-J. Zha, J. Liu, X. Zhu, and H. Xie, "Cluster and scatter: A multi-grained active semi-supervised learning framework for scalable person re-identification," in *Proc. of ACM International Conference on Multimedia*, 2021.

[62] J. E. van Engelen and H. H. Hoos, "A survey on semi-supervised learning," *Machine Learning*, 2020.

[63] D.-H. Lee *et al.*, "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," in *Proc. of International Conference on Machine Learning*, 2013.

[64] M. Chen, H. Xue, and D. Cai, "Domain adaptation for semantic segmentation with maximum squares loss," in *Proc. of Computer Vision and Pattern Recognition*, 2019.

[65] Y. Zou, Z. Yu, X. Liu, B. V. K. V. Kumar, and J. Wang, "Confidence regularized self-training," in *Proc. of International Conference on Computer Vision*, 2019.

[66] F. Ye, W. Luo, M. Dong, H. He, and W. Min, "Sar image retrieval based on unsupervised domain adaptation and clustering," *IEEE Geoscience and Remote Sensing Letters*, 2019.

[67] M. Wu, S. Pan, C. Zhou, X. Chang, and X. Zhu, "Unsupervised domain adaptive graph convolutional networks," in *WWW*, 2020.

[68] M. Wu and M. Rostami, "Graph harmony: Denoising and nuclear-norm wasserstein adaptation for enhanced domain transfer in graph-structured data," *arXiv e-prints*, 2023.

[69] P. Velickovic, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm, "Deep graph infomax." in *Proc. of International Conference on Learning Representations*, 2019.

[70] F.-Y. Sun, J. Hoffmann, V. Verma, and J. Tang, "Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization," in *Proc. of International Conference on Learning Representations*, 2020.

[71] A. Narayanan, M. Chandramohan, R. Venkatesan, L. Chen, Y. Liu, and S. Jaiswal, "graph2vec: Learning distributed representations of graphs," *arXiv preprint arXiv:1707.05005*, 2017.

[72] J. H. Lau and T. Baldwin, "An empirical evaluation of doc2vec with practical insights into document embedding generation," *arXiv preprint arXiv:1607.05368*, 2016.

[73] S. Pan, R. Hu, G. Long, J. Jiang, L. Yao, and C. Zhang, "Adversarially regularized graph autoencoder for graph embedding," *arXiv preprint arXiv:1802.04407*, 2018.

[74] T. N. Kipf and M. Welling, "Variational graph auto-encoders," *arXiv preprint arXiv:1611.07308*, 2016.

[75] S. Li, J. Zhou, T. Xu, D. Dou, and H. Xiong, "Geomgcl: geometric graph contrastive learning for molecular property prediction," in *AAAI*, 2022.

[76] Y. Yin, Q. Wang, S. Huang, H. Xiong, and X. Zhang, "Autogcl: Automated graph contrastive learning via learnable view generators," in *AAAI*, 2022.

[77] Z. Yu and H. Gao, "Molecular representation learning via heterogeneous motif graph neural networks," in *Proc. of International Conference on Machine Learning*, 2022.

[78] X. Han, Z. Jiang, N. Liu, and X. Hu, "G-mixup: Graph data augmentation for graph classification," in *Proc. of International Conference on Machine Learning*, 2022.

[79] Y. Wang, W. Wang, Y. Liang, Y. Cai, and B. Hooi, "Mixup for node and graph classification," in *WWW*, 2021.

[80] K. Sohn, D. Berthelot, N. Carlini, Z. Zhang, H. Zhang, C. A. Raffel, E. D. Cubuk, A. Kurakin, and C.-L. Li, "Fixmatch: Simplifying semi-supervised learning with consistency and confidence," in *Proc. of Neural Information Processing Systems*, 2020.

[81] N. Wale and G. Karypis, "Comparison of descriptor spaces for chemical compound retrieval and classification," in *Proc. of IEEE International Conference on Data Mining*, 2006.

[82] M. Togninalli, E. Ghisu, F. Llinares-López, B. Rieck, and K. Borgwardt, "Wasserstein weisfeiler-lehman graph kernels," in *NeurIPS*, 2019.

[83] P. D. Dobson and A. J. Doig, "Distinguishing enzyme structures from non-enzymes without alignments," *Journal of Molecular Biology*, vol. 330, no. 4, p. 771–783, Jul 2003.

[84] S. Pan, J. Wu, and X. Zhu, "Cogboost: Boosting for fast cost-sensitive graph classification," *IEEE Transactions on Knowledge and Data Engineering*, 2015.

[85] C. Helma, R. D. King, S. Kramer, and A. Srinivasan, "The Predictive Toxicology Challenge 2000–2001 ," *Bioinformatics*, 2001.

[86] J. J. Sutherland, L. A. O'brien, and D. F. Weaver, "Spline-fitting with a genetic algorithm: A method for developing classification structure-activity relationships," *Journal of Chemical Information and Computer Sciences*, 2003.

[87] N. Kriege and P. Mutzel, "Subgraph matching kernels for attributed graphs," *arXiv preprint arXiv:1206.6483*, 2012.

[88] G. Wei, C. Lan, W. Zeng, Z. Zhang, and Z. Chen, "Toalign: Task-oriented alignment for unsupervised domain adaptation," in *NeurIPS*, 2021.