Gaussian Mixture Model for Graph Domain Adaptation

Mengzhu Wang[1], Wenhao Ren[1], Yu Zhang[1], Yanlong Fan[1], Dianxi Shi[2]†, Luoxi Jing[3], Nan Yin[4]†
[1]Hebei University of Technology,
[2]Intelligent Game and Decision Lab (IGDL),
[3]Peking University,
[4]Hong Kong University of Science and Technology
dreamkily@gamil.com, renwh0620@gmail.com, morii5614@gmail.com,
ylong828@126.com, dxshi@nudt.edu.cn, jingluoxi@stu.pku.edu.cn,nanyin@ust.hk

Abstract

Unsupervised domain adaptation (UDA) has been widely studied with the goal of transferring knowledge from a label-rich source domain to a related but unlabeled target domain. Most UDA techniques achieve this by reducing the feature discrepancies between the two domains to learn domain-invariant feature representations. While domain-invariant feature representations can reduce the differences between the source and target domains, excessively simplifying these differences may cause the model to overlook important domain-specific features, resulting in a decline in transfer learning effectiveness. To address this issue, we propose a novel Gaussian Mixture Model for graph domain adaptation (GMM). This model effectively reduces the distributional bias by modeling the distribution differences on a graph structure. GMM leverages the local structural information of the graph and the clustering capability of the Gaussian mixture model to automatically learn the latent mapping relationships. To the best of our knowledge, this is the first work to introduce a Gaussian mixture model into UDA. Extensive experimental results on four standard benchmarks demonstrate that the proposed GMM algorithm outperforms state-of-the-art unsupervised domain adaptation methods in terms of performance.

1 Introduction

The problem of labeling scarcity has long been a challenge in deep learning, as collecting large amounts of labeled data is often expensive, time-consuming, and even infeasible [Pan and Yang, 2009; Zhang *et al.*, 2023]. Unsupervised domain adaptation (UDA) has emerged as an effective solution, which leverages knowledge from a source domain with abundant labeled data and transfers it to a target domain to compensate for the lack of labeled data in the target domain. While source and target domains typically share similar semantics, they differ in data distributions [Zhang *et al.*, 2023; Ganin and Lempitsky, 2015; Du *et al.*, 2024]. In this case, the knowledge from the source domain may not be directly applicable to the target task, leading to poor transfer performance.

To address the distribution discrepancy between the different domains, UDA methods often reduce the feature distribution gap to learn domain-invariant [Long *et al.*, 2017; Wang *et al.*, 2025; Wang *et al.*, 2024c; Wang *et al.*, 2024b; Wang *et al.*, 2024d] feature representations. This approach can effectively improve the performance of models in the target domain, even with a lack of labeled data, by utilizing the labeled data from the source domain. However, despite achieving some success, traditional UDA [Li *et al.*, 2021a; Li *et al.*, 2024] methods still face challenges due to significant differences between the two domains. These challenges include overly simplifying domain differences and overlooking domain-specific features. Therefore, how to better preserve the feature information of the target domain while reducing the distribution discrepancy has become a key research focus.

In recent years, UDA methods [Liu *et al.*, 2024; Shi *et al.*, 2024; Zhang *et al.*, 2024b] have begun to explore incorporating additional prior knowledge, structured information, and graph models to further enhance domain adaptation performance. For example, GCAN [Ma *et al.*, 2019] proposes an end-to-end graph convolutional adversarial network that jointly models data structures, domain labels, and class labels within a unified network framework to achieve UDA. AdaGCN [Dai *et al.*, 2022] develops a novel and principled framework for graph transfer learning, effectively combining adversarial domain adaptation and graph convolution techniques to improve transfer learning efficiency. StruRW [Liu *et al.*, 2023] estimates pseudo-node labels on the target graph to compute edge probabilities between different classes, and uses these probabilities to guide the bootstrapping of neighbors in the source graph's Graph Neural Network (GNN) computation, effectively reducing conditional shift between neighborhoods. However, these methods generally assume homogeneous clusters and rely on Euclidean distances between points in a given feature space for learning, which somewhat limits the efficiency of representation learning and clustering. The homogeneous clustering assumption overlooks the inherent complexity of the data, especially in domain adaptation tasks where there can be significant differences in data distributions across domains. In such cases, relying solely on Euclidean distances may fail to capture the complex relationships between data points, leading to sub-

†Corresponding Author

(a) Raw Data (b) Clusters

Figure 1: Illustration of our motivation. (a) The original data space distributions of the source and target domains not only fail to align but also cannot be well clustered. (b) Through the GMM model, we can cluster data points of the same class together.

optimal clustering results and poor performance in cross-domain transfer, making it difficult to achieve desirable generalization performance.

To handle this issue, we propose the use of the Gaussian Mixture Model (GMM) for graph domain adaptation. The GMM, with its ability to model data distributions as a mixture of multiple Gaussian components, offers a more flexible approach compared to traditional clustering methods. It can effectively capture the heterogeneity of clusters, thus allowing for a more accurate representation of complex, multi-modal distributions inherent in real-world data (see Figure. 1). Additionally, when applied to graph-based domain adaptation, the GMM can leverage the graph structure to encode relationships between data points more efficiently, allowing for improved domain alignment and more robust feature representations. By integrating GMM into the UDA framework, we can better model the domain shift and reduce the risk of negative transfer. The proposed approach not only enhances clustering performance but also facilitates smoother domain adaptation, enabling more reliable and scalable solutions in tasks such as cross-domain classification and data analysis. Our main contributions can summarized as follows:

- We propose a novel unified framework for graph domain adaptation, called GMM, which is capable of handling both heterogeneous graph structures and domain distribution shifts simultaneously. The framework effectively integrates the power of Gaussian mixture models for probabilistic distribution modeling and graph neural networks for structural representation learning.
- Incorporating graph structural information into the adaptation process, maintaining consistency in relational dependencies.
- We conduct extensive experiments on real-world information networks to verify the effectiveness of our model, which demonstrates its superior performance compared with state-of-the-art baselines, impressive label efficiency, and good model robustness against distribution discrepancy.

2 Related Work

2.1 Unsupervised Domain Adaptation

Current UDA techniques concentrate on acquiring feature representations that are domain-invariant. UDA techniques [Long *et al.*, a; Long *et al.*, b] have as one goal reducing the disparity between several domains. To quantify the difference between the source and target domains, early methods use combined MMD and MK-MMD. Additionally, various well-designed discrepancies and higher-order statistics are used [Zellinger *et al.*, 2017]. The intra-class and inter-class domain discrepancies introduced by CAN [Zellinger *et al.*, 2017] are used to leverage category information for domain alignment. An alternative approach involves creating an adversarial optimization goal for a domain discriminator and using adversarial learning to acquire domain-invariant representations. Through the use of a progressively disappearing bridge mechanism, GVB-GD [Cui *et al.*, 2020] encourages adversarial domain adaptation. Using several adversarial discriminators, GSDA [Hu *et al.*, 2020] implements hierarchical domain alignment. In contrast to STAR [Lu *et al.*, 2020], which samples classifiers from Gaussian distributions without additional parameters, MCD [Saito *et al.*, 2018] maximizes the discrepancy between two classifiers.

2.2 Graph Neural Networks (GNN)

Graph Neural Networks (GNNs) are a natural extension of convolutional neural networks in non-Euclidean spaces, designed to process graph-structured data. GNNs can capture complex relationships and structured information, making them suitable for applications in social networks, molecular graphs, traffic networks, and more. The concept of GNNs was first introduced in [Gori *et al.*, 2005; Scarselli *et al.*, 2008; Wang *et al.*, 2024a; Wang *et al.*, 2025] as a trainable recursive message-passing model, where node representations are updated based on information passed between nodes, thereby learning the structural features of the graph. Subsequent research enhanced the model's expressive power by introducing gating mechanisms [Li *et al.*, 2015; Sukhbaatar *et al.*, 2016], improving its performance on complex graph structures. Spectral graph convolutional networks, proposed in [Bruna *et al.*, 2013; Henaff *et al.*, 2015], aim

to capture global information by learning spectral multipliers of the graph Laplacian. However, these methods come with high computational costs. To address this issue, [Defferrard *et al.*, 2016] introduced a method that learns polynomials of the Laplacian to replace the computation of eigenvectors, improving computational efficiency and reducing complexity, allowing GNNs to handle larger-scale graph data. Overall, the continuous optimization and innovation of GNNs have made them a powerful tool for processing graph data. They are widely applied in modeling and analyzing various complex systems. As research continues, the capabilities of GNNs will further improve, offering efficient solutions for a broader range of domains.

Our works is based on the spectral perspective line. The proposed model exploits the GCN to operate on a dense-connected instance graph so that data structure information can be jointly complemented with domain adaptation in a unified deep network.

3 Method

In unsupervised domain adaptation (UDA), we are given n_s labeled samples $\{(x_S^{(i)}, y_S^{(i)})\}_{i=1}^{n_s}$ from the source domain D_S, where $x_S^{(i)} \in \mathcal{X}_S$ and $y_S^{(i)} \in \mathcal{Y}_S$, with $\mathcal{X}_S$ and $\mathcal{Y}_S$ representing the source data space and label space, respectively. Additionally, we are also provided with n_t unlabeled target samples $\{(x_T^{(i)})\}_{i=1}^{n_t}$, where $x_T^{(i)} \in \mathcal{X}_T$, and $\mathcal{X}_T$ denotes the target data space. It is assumed that $\mathcal{X}_S$ and $\mathcal{X}_T$ are different but related. Our goal is to use the model trained on the source domain to predict the unlabeled samples in the target domain.

The proposed graph domain adaptation framework is shown in Figure 2. In the UDA task, when domain shift occurs, the label prediction function f is trained by minimizing the overall objective function, as detailed in Eq. 1:

$$\begin{aligned}\mathcal{L}\left(\mathcal{X}_S, \mathcal{Y}_S, \mathcal{X}_T\right) = \mathcal{L}_C\left(\mathcal{X}_S, \mathcal{Y}_S\right) + \lambda \mathcal{L}_{DA}\left(\mathcal{X}_S, \mathcal{X}_T\right) \\ + \gamma \mathcal{L}_{GAA}\left(\mathcal{X}_S, \mathcal{Y}_S, \mathcal{X}_T\right)\end{aligned} \tag{1}$$

The $\mathcal{L}_C\left(\mathcal{X}_S, \mathcal{Y}_S\right)$ denoted the classification loss. $\mathcal{L}_{DA}\left(\mathcal{X}_S, \mathcal{X}_T\right)$ and $\mathcal{L}_{GAA}\left(\mathcal{X}_S, \mathcal{Y}_S, \mathcal{X}_T\right)$ present the domain alignment loss and graph-aware alignment loss, respectively. The details are introduced as follows.

3.1 Domain Alignment

We use the domain adversarial similarity loss as the domain alignment loss, as shown in Eq. 2. In this process, we introduce an additional domain classifier D to distinguish whether the features output by the feature extractor G come from the source or target domain, while the goal of G is to make it impossible for D to differentiate between them. This two-player minimax game is expected to reach an equilibrium, where the features extracted by G are domain-invariant.

$$\begin{aligned}\mathcal{L}_{DA}\left(\mathcal{X}_S, \mathcal{X}_T\right) = \mathbb{E}_{x \in D_S}[\log(1 - D(G(x)))] \\ + \mathbb{E}_{x \in D_T}[\log(D(G(x)))]\end{aligned} \tag{2}$$

3.2 Graph-aware Alignment

Existing domain alignment mechanisms typically focus on aligning the global statistical information between the source and target domains, often neglecting the structural information within mini-batch samples. Traditional methods mainly align domains by comparing global features, which may fail to capture subtle relationships, thus limiting the performance of UDA tasks. In UDA, the local relationships and structural information between samples are often more critical than simple global statistics, particularly when faced with complex cross-domain transfers. Ignoring these details can lead to performance degradation. Increasing research has demonstrated that the structural information of data plays a crucial role in UDA tasks. Many advanced methods have successfully incorporated structural information modeling, significantly improving UDA performance. These methods not only focus on global statistical features but also emphasize modeling local data structures, leading to a better understanding and capturing of the complex relationships between the source and target domains [Ma *et al.*, 2019; Zhang *et al.*, 2024a].

To further enhance alignment performance in UDA, especially during deep learning-based training, we propose a novel graph-aware alignment mechanism. This mechanism effectively models the structural information of mini-batch samples from both the source and target domains by introducing a graph structure. Specifically, we use the graph structure to represent the similarity between samples and apply techniques like graph convolution to strengthen structural alignment between domains. This approach enables a finer-grained alignment process, allowing for better capture of the subtle differences between the source and target domains. The method not only improves domain alignment accuracy but also enhances the model's generalization ability in the target domain, particularly when there are significant data distribution differences between the domains, thus reducing negative transfer and improving task performance. Moreover, with the graph-aware alignment mechanism, the model can intelligently adjust and optimize when transferring across different domains, further expanding the application scope of UDA methods. This structural alignment strategy not only enhances the performance of traditional methods in complex tasks but also provides a new approach for broader cross-domain learning challenges. In particular, when data distributions are drastically different, this mechanism can reduce information loss during alignment through fine-grained structural alignment, further boosting the effectiveness and robustness of transfer learning.

In graph domain adaptation, we first use a Data Structure Analyzer (DSA) network to compute structure scores for each mini-batch sample. Then, we combine these scores with the features extracted from the samples through a Convolutional Neural Network (CNN) to construct densely connected instance graphs. Next, we apply a Graph Convolutional Network (GCN) to these instance graphs to learn features that effectively capture data structure information. Before detailing our method, we briefly review the GCN model introduced in [Kipf and Welling, 2016; Ma *et al.*, 2019]. The core goal of GCN is to learn layer-wise propagation operations that can be directly applied to graph structures. Specifically, given an undirected graph with m nodes and edges connecting them, along with an adjacency matrix $\mathbf{A} \in R^{m \times m}$, the

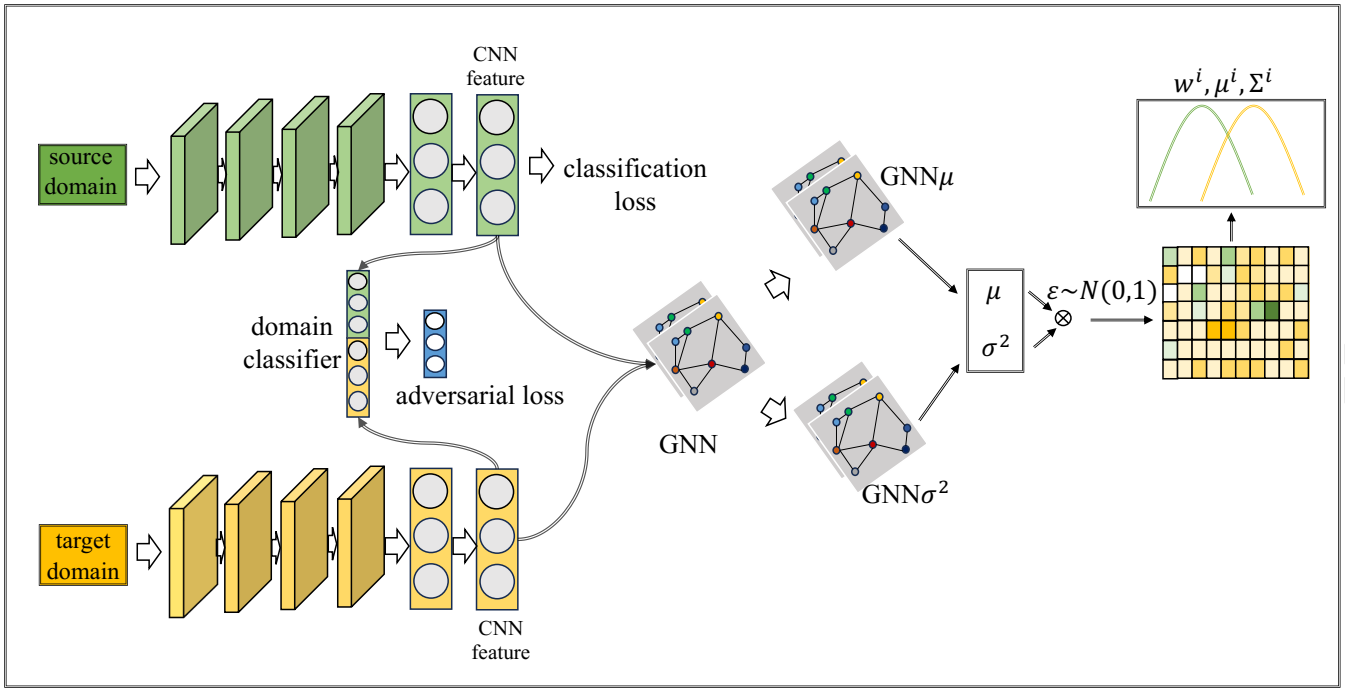

Figure 2: Overview of the GMM framework. In structure-aware alignment, the Data Structure Analyzer (DSA) network generates structure scores that encode source data structure information, while features are extracted from the samples using Convolutional Neural Networks (CNN). These structure scores and CNN features are then combined to construct densely connected instance graphs for further processing by the Graph Convolutional Network (GCN).

linear transformation of graph convolution can be represented as the product of the graph signal $\mathbf{X} \in R^{k \times m}$ and the filter $\mathbf{W} \in R^{k \times c}$:

$$\mathbf{Z} = \hat{\mathbf{D}}^{-\frac{1}{2}} \hat{\mathbf{A}} \hat{\mathbf{D}}^{-\frac{1}{2}} \mathbf{X}^T \mathbf{W} \tag{3}$$

where $\hat{\mathbf{A}} = \mathbf{A} + \mathbf{I}$, $\mathbf{I}$ denotes the identity matrix, and $\hat{\mathbf{D}}_{ii} = \sum_j \hat{\mathbf{A}} ij$. In this formula, the output is a matrix $\mathbf{Z}$ of size $c \times m$. It is worth noting that the GCN can be constructed by stacking multiple graph convolutional layers, as shown in Eq. 3, with a non-linear operation applied after each layer.

Next, we will explain how to construct densely-connected instance graphs for the GCN, and how to derive the graph signal $\mathbf{X}$ and adjacency matrix $\mathbf{A}$ as shown in Eq. 3. Each node in the instance graph corresponds to the feature of a sample, which is extracted by a standard convolutional network. Thus, the graph signal $\mathbf{X}$ can be obtained as follows:

$$\mathbf{X} = CNN\left(\mathbf{X}_b\right) \tag{4}$$

where $\mathbf{X}_b$ represent a mini-batch of samples. To construct the adjacency matrix $\hat{\mathbf{A}}$, we feed the same mini-batch samples into a Data Structure Analyzer (DSA) network to generate structure scores, and then construct the adjacency matrix $\hat{\mathbf{A}}$ based on these scores, as follows:

$$\hat{\mathbf{A}} = \mathbf{X}\mathbf{X}^T, \tag{5}$$

where $\mathbf{X} \in R^{b \times d}$, b is the batch size, and d is the dimension of the structure scores. Then we can obtain the latent variable $\mathbf{H}$:

$$\mathbf{H} = \delta\left(\mathbf{Z}\right) \tag{6}$$

Then, the mean vector $\boldsymbol{\mu}$ and the variance vector $\boldsymbol{\sigma}$ are as follows:

$$\begin{aligned} \boldsymbol{\mu} &= \delta\left(\hat{\mathbf{D}}^{-1/2} \hat{\mathbf{A}} \hat{\mathbf{D}}^{-1/2} \mathbf{H}\mathbf{W}_1\right) \\ \boldsymbol{\sigma} &= \delta\left(\hat{\mathbf{D}}^{-1/2} \hat{\mathbf{A}} \hat{\mathbf{D}}^{-1/2} \mathbf{H}\mathbf{W}_2\right) \end{aligned} \tag{7}$$

Node embeddings $\mathbf{Z}$ can be obtained by sampling from the posterior distribution $\mathcal{N} \sim (\boldsymbol{\mu}, \boldsymbol{\sigma})$. In deep learning models, sampling operations are typically non-differentiable, which prevents direct backpropagation through gradient descent, causing training difficulties. To address this issue, we introduce a random variable ε that follows a standard normal distribution, and transform the sampling process into a differentiable operation. Specifically, the sampling operation is transformed into the following form: $\mathbf{Y} = \boldsymbol{\varepsilon} * \boldsymbol{\sigma} + \boldsymbol{\mu}$, where $*$ denotes the dot product. This approach makes the sampling process linear and differentiable, allowing the model to perform backpropagation during training. This not only ensures the operability of the sampling process, but also effectively avoids training issues caused by non-differentiability in traditional sampling methods, significantly improving the model's training efficiency and stability.

Based on the above discussion, we can formalize the inference process achieved through the GCN encoder as follows:

$$
\begin{aligned}
q\left(\mathbf{Y} \mid \mathbf{X}, \mathbf{A}\right) &= \prod_{i=1}^{n} q\left(\mathbf{Y}^i \mid \mathbf{X}, \mathbf{A}\right), \\
q\left(\mathbf{Y}^i \mid \mathbf{X}, \mathbf{A}\right) &= \mathcal{N}\left(\mathbf{Y}^i \mid \boldsymbol{\mu}^i, \operatorname{diag}\left(\boldsymbol{\sigma}^i\right)^2\right)
\end{aligned} \tag{8}
$$

where $\mathbf{Z}^i$ is the i-th elements of $\mathbf{Z}$, $\boldsymbol{\mu}$ and σ are the i-th row of $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$, respectively.

An inner product between latent variables is adopted as the decoder to reconstruct the input graph's topological structure, More specifically,

$$
\hat{\mathbf{A}} = \delta\left(\mathbf{Z}\left(\mathbf{Z}\right)^T\right) \tag{9}
$$

where $\hat{\mathbf{A}}$ is the reconstructed adjacency matrix. δ is the logistic sigmoid function. We define $\mathbf{A}^{ij}$ are the elements of $\mathbf{A}$. The generative process can be formalized as,

$$
\begin{aligned}
p\left(\mathbf{A} \mid \mathbf{Y}\right) &= \prod_{i=1}^{n}\prod_{j=1}^{n} p\left(\mathbf{A}^{ij} \mid \mathbf{Y}^i, \mathbf{Y}^j\right), \\
p\left(A^{ij} = 1 \mid \mathbf{Y}^i, \mathbf{Y}^j\right) &= \delta\left(\left(\mathbf{Y}^i\right)^T \mathbf{Y}^j\right).
\end{aligned} \tag{10}
$$

The initial variational lower bound of GMM in our model consists of reconstruction loss and the KL divergence between the approximate posterior distribution $q\left(\mathbf{Y} \mid \mathbf{X}, \mathbf{A}\right)$ and the initialized prior distribution $P\left(\mathbf{Y}\right)$ i.e. single Gaussian distribution.

$$
\mathcal{L}_{GAA} = \mathbb{E}_{q(\mathbf{Y}|\mathbf{X},\mathbf{A})}\left[p\left(\mathbf{A} \mid \mathbf{Y}\right)\right] - \mathrm{KL}\left[q\left(\mathbf{Y} \mid \mathbf{X}, \mathbf{A}\right) \| p\left(\mathbf{Y}\right)\right]. \tag{11}
$$

3.3 Gaussian Mixture Model

The Gaussian Mixture Model (GMM) is a parametric probability density function, represented as a weighted sum of multiple Gaussian component densities. It describes the overall distribution of data by combining multiple Gaussian distributions and is a widely used clustering algorithm in both academic research and industry. Due to its ability to effectively model complex data distributions, we use GMM to partition the feature representations of all classes into different clusters. The formalization of GMM in UDA is described as follows:

$$
p(\mathbf{x} \mid \lambda) = \sum_{i=1}^{M} w^i g\left(\mathbf{x} \mid \boldsymbol{\mu}^{i,GMM}, \boldsymbol{\Sigma}^{i,GMM}\right) \tag{12}
$$

where $\mathbf{x}$ represents the input feature vector of the data, and w^i denotes the mixture weights for each domain, satisfying $w^i \geq 0$ and $\sum_{i=1}^{M} w_t^i = 1$. $\boldsymbol{\mu}_t^{i,GMM}$ and $\Sigma_t^{i,GMM}$ represent the mean vector and covariance matrix, respectively, of the i-th Gaussian component in the two domains. $\lambda = \left(\boldsymbol{\mu}^{i,GMM}, \boldsymbol{\Sigma}^{i,GMM}\right)$, where M is the total number of Gaussian components. $g\left(\mathbf{x} \mid \boldsymbol{\mu}^{i,GMM}, \boldsymbol{\Sigma}^{i,GMM}\right)$ denotes the Gaussian distribution density of the i-th component, where each component is a D-dimensional Gaussian function.

$$
\begin{aligned}
&g\left(\mathbf{x} \mid \boldsymbol{\mu}^{i,GMM}, \boldsymbol{\Sigma}^{i,GMM}\right) \\
&= \frac{1}{(2\pi)^{D/2}\left|\Sigma^{i,GMM}\right|^{1/2}} \\
&\times \exp\left\{-\frac{1}{2}\left(\mathbf{x} - \boldsymbol{\mu}^{i,GMM}\right)'\left(\boldsymbol{\Sigma}^{i,GMM}\right)^{-1}\left(\mathbf{x} - \boldsymbol{\mu}^{i,GMM}\right)\right\}
\end{aligned} \tag{13}
$$

Furthermore, to estimate the parameters of the GMM, we use the Expectation-Maximization (EM) algorithm [Moon, 1996]. The GMM generates M clusters, denoted as $K^i, i \in 0, 1, \ldots, M-1$, where each cluster represents a set of snapshots exhibiting similar structural and attribute characteristics.

$$
K^i = \mathrm{GMM}(z), \quad i \in \{0, 1, \ldots, M-1\}, \tag{14}
$$

The GMM can be accurately represented by a series of parameters, including the mean vector μ^{GMM}, the covariance matrix Σ^{GMM}, and the weights w of the two domains. We update the prior distribution by adjusting the parameters of the GMM to achieve joint iterative training. This approach enables information sharing across multiple domains and optimizes the model parameters in each iteration, allowing it to better adapt to the data distribution of different domains and improve domain adaptation performance.

4 Experiments

4.1 Datasets

Office-31 [Saenko *et al.*, 2010] is a widely used cross-domain dataset, particularly applied in office environments, consisting of 31 classes of images from three distinct domains: Amazon (A), Webcam (W), and DSLR (D). The dataset is imbalanced in terms of the number of images, with 2,817 images from Amazon, 795 images from Webcam, and 498 images from DSLR.

Office-Home [Venkateswara *et al.*, 2017] is a dataset comprising 15,500 photos from both home and business settings, organized into 65 categories. The dataset consists of four distinct domains: Product (Pr), Clipart (Cl), Real-World (Rw), and Art (Ar). These categories include real-world camera captures, product photos, clipart images, and artistic renderings, respectively.

VisDA-2017 [Peng *et al.*, 2017] contains over 280,000 images across 12 categories and presents a challenging benchmark for unsupervised domain adaptation (UDA). In our experiments, we use the validation set as the target domain and the training set as the source domain.

DomainNet [Peng *et al.*, 2019] is a large-scale domain adaptation dataset with over 590,000 images spanning six domains: Painting (pnt), Infograph (inf), Real (rel), Quickdraw (qdr), Clipart (clp), and Sketch (skt). Each domain contains items from 345 categories and is the largest image benchmark for domain adaptation currently available.

4.2 Implementation Details

We use PyTorch to implement our technique, and the backbone network for all datasets is ResNet [He *et al.*, 2016],

Methods	D→A	W→A	A→W	D→W	W→D	A→D	Avg.
ResNet-50	62.5	60.7	68.4	96.7	99.3	68.9	76.1
DDC	62.2	61.5	75.6	96.0	98.2	76.5	78.3
DAN	63.6	62.8	80.5	97.1	99.6	78.6	80.4
DANN	68.2	67.4	82.0	96.9	99.1	79.7	82.2
SymNet	73.4	71.6	88.6	98.2	99.7	85.3	86.2
CyCADA	72.8	71.4	89.5	97.9	99.8	87.7	86.5
CDAN	71.0	69.3	94.1	98.6	**100.0**	92.9	87.7
TADA	72.9	73.0	94.3	98.7	99.8	91.6	88.4
BSP	73.6	72.6	93.3	98.2	**100.0**	93.0	88.5
MCD	74.1	73.5	95.7	99.3	**100.0**	96.4	89.8
T^2SA	78.2	78.5	94.6	97.2	99.8	92.4	90.1
GMM (Ours)	**83.5**	**86.7**	**98.9**	**100.0**	**100.0**	**98.2**	**94.6**

Table 1: Recognition accuracy (%) using the Office-31 dataset.

Method	$Ar\rightarrow Cl$	$Ar\rightarrow Pr$	$Ar\rightarrow Rw$	$Cl\rightarrow Ar$	$Cl\rightarrow Pr$	$Cl\rightarrow Rw$	$Pr\rightarrow Ar$	$Pr\rightarrow Cl$	$Pr\rightarrow Rw$	$Rw\rightarrow Ar$	$Rw\rightarrow Cl$	$Rw\rightarrow Pr$	Avg.
ResNet-50	34.9	50.0	58.0	37.4	41.9	46.2	38.5	31.2	60.4	53.9	41.2	59.9	46.1
CDAN	50.7	70.6	76.0	57.6	70.0	70.0	57.4	50.9	77.3	70.9	56.7	81.6	65.8
BSP	52.0	68.6	76.1	58.0	70.3	70.2	58.6	50.2	77.6	72.2	59.3	81.9	66.3
SAFN	52.0	71.7	76.3	64.2	69.9	71.9	63.7	51.4	77.1	70.9	57.1	81.5	67.3
TADA	53.1	72.3	77.2	59.1	71.2	72.1	59.7	53.1	78.4	72.4	60.0	82.9	67.6
SymNet	47.7	72.9	78.5	64.2	71.3	74.2	64.2	48.8	79.5	74.5	52.6	82.7	67.6
T^2SA	61.0	78.5	83.1	71.4	80.1	79.9	70.3	60.1	83.5	75.6	62.6	86.6	74.4
GMM (Ours)	**65.7**	**79.3**	**85.7**	**73.4**	**79.2**	**82.5**	**73.9**	**63.4**	**85.6**	**82.9**	**70.4**	**93.5**	**78.0**

Table 2: Recognition accuracy (%) using the Office-Home dataset.

Methods	Synthetic → Real
DAN	53.0
DANN	57.4
SimNet	69.6
CDAN	70.0
MCD	73.7
DSAN	75.1
T^2SA	83.2
CDAN+InterBN	**88.3**

Table 3: Recognition accuracy (%) using the VisDA-2017 dataset.

which has been pre-trained on ImageNet [Russakovsky *et al.*, 2015]. We used Pytorch [Paszke *et al.*, 2019] to implement all of the experiments. Our Stochastic Gradient Descent (SGD) algorithm is used with momentum set to 0.9 and weight decay set to 0.001. For model optimization, we follow the learning rate annealing technique described in [Ganin *et al.*, 2016]. In this study, we specifically specify $\lambda = 0.5, \gamma = 1.0$ and perform a sensitivity analysis to find out how hyper-parameter selection affects the results. We present the average accuracy from three random trials for each task. We compare with the recent popular methods,e.g.,ResNet-50 [He *et al.*, 2016],DDC [Tzeng *et al.*, 2014],DDC [Tzeng *et al.*, 2014],DAN [Long *et al.*, 2015],DANN [Ganin *et al.*, 2016],SimNet [Zhang *et al.*, 2019],CyCADA [Hoffman *et al.*, 2018],CDAN [Long *et al.*, 2018],TADA [Hung *et al.*, 2023],BSP [Chen *et al.*, 2019],MCD [Saito *et al.*, 2018], T^2DA [Li *et al.*, 2021c], SAFN [Xu *et al.*, 2019],ADDA [Tzeng *et al.*, 2017], MIMTFL [Braytee *et al.*, 2022], MDD [Li *et al.*, 2020], SCDA [Li *et al.*, 2021b], CDAN+InterBN [Wang *et al.*, 2021].

4.3 Results

We present our results in Table. 1 to Table. 4. Our GMM method significantly enhances the performance of the compared state-of-the-art methods, achieving average accuracy improvements of 4.5%, 4.4%, 5.1%, and 9.0% on the Office-31, Office-Home, VisDA-2017, and DomainNet datasets, respectively. Notably, we observe substantial accuracy gains on some of the most challenging tasks, such as in the Office-31 dataset, where D→A improved from 78.2% to 83.5% and W→A improved from 78.5% to 86.7%. On the largest dataset, DomainNet, our GMM method also demonstrates impressive performance with a 9.0% increase compared to CDAN+InterBN. Importantly, across all four benchmark datasets, our proposed method outperforms all state-of-the-art approaches. Additionally, the absence of adversarial networks in our method further underscores its effectiveness.

4.4 Empirical Analysis

Distribution Discrepancy. The $\mathcal{A}$-distance [Ben-David *et al.*, 2010] is a widely used metric for measuring the distance between two distributions, with a larger $\mathcal{A}$-distance indicating a greater distinction between the source and target domains. However, the complexity of directly computing the $\mathcal{A}$-distance prompts us to use the proxy $\mathcal{A}$-distance, $\hat{d}_{\mathcal{A}}$, defined as $\hat{d}_{\mathcal{A}} = 2(1 - 2\varepsilon)$. In our analysis, we compute

ADDA	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	11.2	24.1	3.2	41.9	30.7	22.2
inf	19.1	-	16.4	3.2	26.9	14.6	16.0
pnt	31.2	9.5	-	8.4	39.1	25.4	22.7
qdr	15.7	2.6	5.4	-	9.9	11.9	9.1
rel	39.5	14.5	29.1	12.1	-	25.7	24.2
skt	35.3	8.9	25.2	14.9	37.6	-	25.4
Avg.	28.2	9.3	20.1	8.4	31.1	21.7	19.8

DANN	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	15.5	34.8	9.5	50.8	41.4	30.4
inf	31.8	-	30.2	3.8	44.8	25.7	27.3
pnt	39.6	15.1	-	5.5	54.6	35.1	30.0
qdr	11.8	2.0	4.4	-	9.8	8.4	7.3
rel	47.5	17.9	47.0	6.3	-	37.3	31.2
skt	47.9	13.9	34.5	10.4	46.8	-	30.7
Avg.	35.7	12.9	30.2	7.1	41.4	29.6	26.1

MIMTFL	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	15.1	35.6	10.7	51.5	43.1	31.2
inf	32.1	-	31.0	2.9	48.5	31.0	29.1
pnt	40.1	14.7	-	4.2	55.4	36.8	30.2
qdr	18.8	3.1	5.0	-	16.0	13.8	11.3
rel	48.5	19.0	47.6	5.8	-	39.4	32.1
skt	51.7	16.5	40.3	12.3	53.5	-	34.9
Avg.	38.2	13.7	31.9	7.2	45.0	32.8	28.1

ResNet-101	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	19.3	37.5	11.1	52.2	41.0	32.2
inf	30.2	-	31.2	3.6	44.0	27.9	27.4
pnt	39.6	18.7	-	4.9	54.5	36.3	30.8
qdr	7.0	0.9	1.4	-	4.1	8.3	4.3
rel	48.4	22.2	49.4	6.4	-	38.8	33.0
skt	46.9	15.4	37.0	10.9	47.0	-	31.4
Avg.	34.4	15.3	31.3	7.4	40.4	30.5	26.6

CDAN	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	20.4	36.6	9.0	50.7	42.3	31.8
inf	27.5	-	25.7	1.8	34.7	20.1	22.0
pnt	42.6	20.0	-	2.5	55.6	38.5	31.8
qdr	21.0	4.5	8.1	-	14.3	15.7	12.7
rel	51.9	23.3	50.4	5.4	-	41.4	34.5
skt	50.8	20.3	43.0	2.9	50.8	-	33.6
Avg.	38.8	17.7	32.8	4.3	41.2	31.6	27.7

MDD	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	20.5	40.7	6.2	52.5	42.1	32.4
inf	33.0	-	33.8	2.6	46.2	24.5	28.0
pnt	43.7	20.4	-	2.8	51.2	41.7	32.0
qdr	18.4	3.0	8.1	-	12.9	11.8	10.8
rel	52.8	21.6	47.8	4.2	-	41.2	33.5
skt	54.3	17.5	43.1	5.7	54.2	-	35.0
Avg.	40.4	16.6	34.7	4.3	43.4	32.3	28.6

SCDA	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	18.6	39.3	5.1	55.0	44.1	32.4
inf	29.6	-	34.0	1.4	46.3	25.4	27.3
pnt	44.1	19.0	-	2.6	56.2	42.0	32.8
qdr	30.0	4.9	15.0	-	25.4	19.8	19.0
rel	54.0	22.5	51.9	2.3	-	42.5	34.6
skt	55.6	18.5	44.7	6.4	53.2	-	35.7
Avg.	42.6	16.7	37.0	3.6	47.2	34.8	30.3

CDAN +InterBN	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	21.4	41.8	9.5	51.3	45.7	33.9
inf	28.9	-	26.7	4.5	37.5	21.9	23.9
pnt	45.6	22.4	-	4.8	57.2	42.4	34.5
qdr	29.0	4.8	13.4	-	21.5	20.3	17.8
rel	56.4	23.5	54.7	4.8	-	43.3	36.5
skt	59.2	22.2	48.5	11.7	57.3	-	39.8
Avg.	43.8	18.9	37.0	7.1	45.0	31.1	30.5

GMM	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	23.4	45.7	18.9	56.7	45.5	**38.1**
inf	31.8	-	34.5	6.9	49.7	32.5	**31.1**
pnt	46.7	24.8	-	7.3	58.4	42.5	**35.9**
qdr	34.3	7.9	21.5	-	30.5	23.1	**23.5**
rel	55.7	24.8	56.4	10.5	-	45.3	**38.5**
skt	61.2	24.2	50.5	16.7	59.7	-	**42.5**
Avg.	**45.9**	**21.0**	**41.7**	**12.1**	**51.0**	**37.8**	**39.5**

Table 4: DomainNet accuracy (%) table for UDA (ResNet-101), each sub-table is arranged such that the column-wise domains represent the source domains, while the row-wise domains are chosen as the target domains.

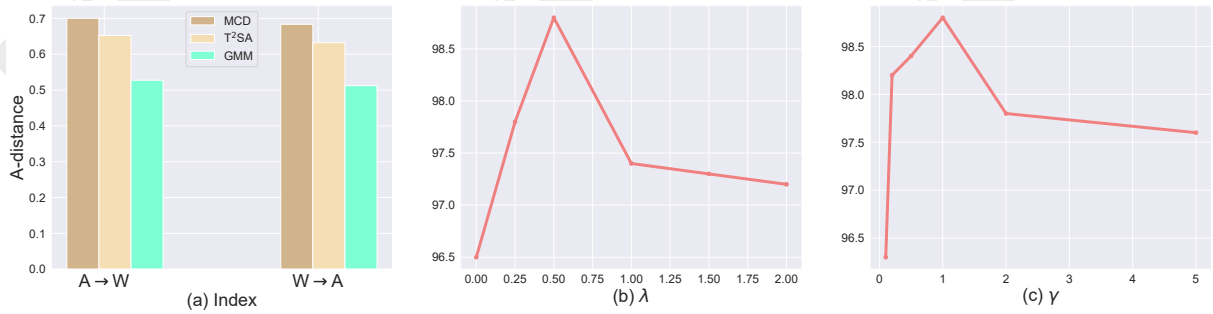

Figure 3: (a): $\mathcal{A}$-distance of different methods, (b): Sensitivity of λ, (c): Sensitivity of γ

this proxy $\mathcal{A}$-distance for various models, including ResNet, CDAN, T^{2}SA, and GMM, specifically focusing on the feature representations for the A $\rightarrow$ W task in Office-31. The results, shown in Figure 3(a), reveal that GMM has the lowest proxy $\mathcal{A}$-distance among the evaluated methods.

Hyper-parameter Sensitivity. To further validate the effectiveness of these hyper-parameter settings, we conducted additional experiments under various conditions. Specifically, we tested the performance of GMM on multiple datasets with different characteristics, including varying sizes, noise levels, and data distributions. The results consistently showed that the optimal hyper-parameter combination of $\lambda = 0.5$ and $\gamma = 1.0$ maintained superior performance across these diverse scenarios. Furthermore, we compared GMM with several state-of-the-art methods under the same experimental settings. Overall, these findings suggest that GMM is a reliable and efficient approach for the given task, and the identified optimal hyper-parameter settings can be applied broadly to achieve consistent and high-quality results.

5 Conclusions

In this paper, we propose a novel method called Gaussian mixture model for graph domain adaptation. Our approach leverages the flexibility of GMM to model multi-modal data distributions, effectively capturing the inherent complexity and heterogeneity of real-world data. By integrating GMM into the UDA framework, we enhance clustering performance and improve domain alignment, thereby reducing the risk of negative transfer and improving the robustness and scalability of domain adaptation.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grants No. 62406100, Tianjin Natural Science Foundation under Grants No. 24JCQNJC00320, Beijing Postdoctoral Research Foundation.

References

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

[Braytee *et al.*, 2022] Ali Braytee, Mohamad Naji, and Paul J Kennedy. Unsupervised domain-adaptation-based tensor feature learning with structure preservation. *TAI*, 3(3):370–380, 2022.

[Bruna *et al.*, 2013] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv*, 2013.

[Chen *et al.*, 2019] Xinyang Chen, Sinan Wang, Mingsheng Long, and Jianmin Wang. Transferability vs. discriminability: Batch spectral penalization for adversarial domain adaptation. In *ICML*, pages 1081–1090. PMLR, 2019.

[Cui *et al.*, 2020] Shuhao Cui, Shuhui Wang, Junbao Zhuo, Chi Su, Qingming Huang, and Qi Tian. Gradually vanishing bridge for adversarial domain adaptation. In *CVPR*, pages 12455–12464, 2020.

[Dai *et al.*, 2022] Quanyu Dai, Xiao-Ming Wu, Jiaren Xiao, Xiao Shen, and Dan Wang. Graph transfer learning via adversarial domain adaptation with graph convolution. *TKDE*, 35(5):4908–4922, 2022.

[Defferrard *et al.*, 2016] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *NeurIPS*, 29, 2016.

[Du *et al.*, 2024] Zhekai Du, Xinyao Li, Fengling Li, Ke Lu, Lei Zhu, and Jingjing Li. Domain-agnostic mutual prompting for unsupervised domain adaptation. In *CVPR*, pages 23375–23384, 2024.

[Ganin and Lempitsky, 2015] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, pages 1180–1189, 2015.

[Ganin *et al.*, 2016] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario March, and Victor Lempitsky. Domain-adversarial training of neural networks. *JMLR*, 17(59):1–35, 2016.

[Gori *et al.*, 2005] Marco Gori, Gabriele Monfardini, and Franco Scarselli. A new model for learning in graph domains. In *IJCNN*, volume 2, pages 729–734, 2005.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[Henaff *et al.*, 2015] Mikael Henaff, Joan Bruna, and Yann LeCun. Deep convolutional networks on graph-structured data. *arXiv*, 2015.

[Hoffman *et al.*, 2018] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *ICML*, pages 1989–1998, 2018.

[Hu *et al.*, 2020] Lanqing Hu, Meina Kan, Shiguang Shan, and Xilin Chen. Unsupervised domain adaptation with hierarchical gradient synchronization. In *CVPR*, pages 4043–4052, 2020.

[Hung *et al.*, 2023] Chia-Chien Hung, Lukas Lange, and Jannik Strötgen. Tada: Efficient task-agnostic domain adaptation for transformers. *arXiv*, 2023.

[Kipf and Welling, 2016] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv*, 2016.

[Li *et al.*, 2015] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. *arXiv*, 2015.

[Li *et al.*, 2020] Jingjing Li, Erpeng Chen, Zhengming Ding, Lei Zhu, Ke Lu, and Heng Tao Shen. Maximum density divergence for domain adaptation. *TPAMI*, 43(11):3918–3930, 2020.

[Li *et al.*, 2021a] Shuang Li, Mixue Xie, Kaixiong Gong, Chi Harold Liu, Yulin Wang, and Wei Li. Transferable semantic augmentation for domain adaptation. In *CVPR*, pages 11516–11525, 2021.

[Li *et al.*, 2021b] Shuang Li, Mixue Xie, Fangrui Lv, Chi Harold Liu, Jian Liang, Chen Qin, and Wei Li. Semantic concentration for domain adaptation. In *ICCV*, pages 9102–9111, 2021.

[Li *et al.*, 2021c] Yunsheng Li, Lu Yuan, Yinpeng Chen, Pei Wang, and Nuno Vasconcelos. Dynamic transfer for multi-source domain adaptation. In *CVPR*, pages 10998–11007, 2021.

[Li *et al.*, 2024] Zijian Li, Ruichu Cai, Guangyi Chen, Boyang Sun, Zhifeng Hao, and Kun Zhang. Subspace identification for multi-source domain adaptation. *NeurIPS*, 36, 2024.

[Liu *et al.*, 2023] Shikun Liu, Tianchun Li, Yongbin Feng, Nhan Tran, Han Zhao, Qiang Qiu, and Pan Li. Structural re-weighting improves graph domain adaptation. In *ICML*, pages 21778–21793, 2023.

[Liu *et al.*, 2024] Meihan Liu, Zeyu Fang, Zhen Zhang, Ming Gu, Sheng Zhou, Xin Wang, and Jiajun Bu. Rethinking propagation for unsupervised graph domain adaptation. In *AAAI*, volume 38, pages 13963–13971, 2024.

[Long *et al.*, a] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I. Jordan. Learning transferable features with deep adaptation networks. In *ICML*, volume 37, pages 97–105.

[Long *et al.*, b] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I. Jordan. Deep transfer learning with joint adaptation networks. In *ICML*, volume 70, pages 2208–2217.

[Long *et al.*, 2015] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *ICML*, pages 97–105, 2015.

[Long *et al.*, 2017] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Deep transfer learning with joint adaptation networks. In *ICML*, pages 2208–2217, 2017.

[Long *et al.*, 2018] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. *NeurIPS*, 31, 2018.

[Lu *et al.*, 2020] Zhihe Lu, Yongxin Yang, Xiatian Zhu, Cong Liu, Yi-Zhe Song, and Tao Xiang. Stochastic classifiers for unsupervised domain adaptation. In *CVPR*, pages 9111–9120, 2020.

[Ma *et al.*, 2019] Xinhong Ma, Tianzhu Zhang, and Changsheng Xu. Gcan: Graph convolutional adversarial network for unsupervised domain adaptation. In *CVPR*, pages 8266–8276, 2019.

[Moon, 1996] Todd K Moon. The expectation-maximization algorithm. *IEEE Signal Processing Magazine*, 13(6):47–60, 1996.

[Pan and Yang, 2009] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *TKDE*, 22(10):1345–1359, 2009.

[Paszke *et al.*, 2019] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *NeurIPS*, 32, 2019.

[Peng *et al.*, 2017] Xingchao Peng, Ben Usman, Neela Kaushik, Judy Hoffman, Dequan Wang, and Kate Saenko. Visda: The visual domain adaptation challenge. *arXiv:1710.06924*, 2017.

[Peng *et al.*, 2019] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *ICCV*, pages 1406–1415, 2019.

[Russakovsky *et al.*, 2015] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 115:211–252, 2015.

[Saenko *et al.*, 2010] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *ECCV*, pages 213–226, 2010.

[Saito *et al.*, 2018] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *CVPR*, pages 3723–3732, 2018.

[Scarselli *et al.*, 2008] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2008.

[Shi *et al.*, 2024] Boshen Shi, Yongqing Wang, Fangda Guo, Bingbing Xu, Huawei Shen, and Xueqi Cheng. Graph domain adaptation: Challenges, progress and prospects. *arXiv*, 2024.

[Sukhbaatar *et al.*, 2016] Sainbayar Sukhbaatar, Rob Fergus, et al. Learning multiagent communication with backpropagation. *NeurIPS*, 29, 2016.

[Tzeng *et al.*, 2014] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv*, 2014.

[Tzeng *et al.*, 2017] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *CVPR*, pages 7167–7176, 2017.

[Venkateswara *et al.*, 2017] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *CVPR*, pages 5018–5027, 2017.

[Wang *et al.*, 2021] Mengzhu Wang, Wei Wang, Baopu Li, Xiang Zhang, Long Lan, Huibin Tan, Tianyi Liang, Wei Yu, and Zhigang Luo. Interbn: Channel fusion for adversarial unsupervised domain adaptation. In *ACM MM*, pages 3691–3700, 2021.

[Wang *et al.*, 2024a] Mengzhu Wang, Jiao Li, Houcheng Su, Nan Yin, Liang Yang, and Shen Li. Graphcl: Graph-based clustering for semi-supervised medical image segmentation. *arXiv preprint arXiv:2411.13147*, 2024.

[Wang *et al.*, 2024b] Mengzhu Wang, Junze Liu, Ge Luo, Shanshan Wang, Wei Wang, Long Lan, Ye Wang, and Feiping Nie. Smooth-guided implicit data augmentation for domain generalization. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

[Wang *et al.*, 2024c] Mengzhu Wang, Yuehua Liu, Jianlong Yuan, Shanshan Wang, Zhibin Wang, and Wei Wang. Inter-class and inter-domain semantic augmentation for domain generalization. *IEEE Transactions on Image Processing*, 33:1338–1347, 2024.

[Wang *et al.*, 2024d] Mengzhu Wang, Shanshan Wang, Xun Yang, Jianlong Yuan, and Wenju Zhang. Equity in unsupervised domain adaptation by nuclear norm maximization. *IEEE Transactions on Circuits and Systems for Video Technology*, 34(7):5533–5545, 2024.

[Wang *et al.*, 2025] Mengzhu Wang, Houcheng Su, Sijia Wang, Shanshan Wang, Nan Yin, Li Shen, Long Lan, Liang Yang, and Xiaochun Cao. Graph convolutional mixture-of-experts learner network for long-tailed domain generalization. *IEEE Transactions on Circuits and Systems for Video Technology*, 2025.

[Xu *et al.*, 2019] Ruijia Xu, Guanbin Li, Jihan Yang, and Liang Lin. Larger norm more transferable: An adaptive feature norm approach for unsupervised domain adaptation. In *ICCV*, pages 1426–1435, 2019.

[Zellinger *et al.*, 2017] Werner Zellinger, Thomas Grubinger, Edwin Lughofer, Thomas Natschläger, and Susanne Saminger-Platz. Central moment discrepancy (cmd) for domain-invariant representation learning. *ICLR*, 2017.

[Zhang *et al.*, 2019] Yabin Zhang, Hui Tang, Kui Jia, and Mingkui Tan. Domain-symmetric networks for adversarial domain adaptation. In *CVPR*, pages 5031–5040, 2019.

[Zhang *et al.*, 2023] Yixin Zhang, Zilei Wang, Junjie Li, Jiafan Zhuang, and Zihan Lin. Towards effective instance discrimination contrastive loss for unsupervised domain adaptation. In *ICCV*, pages 11388–11399, 2023.

[Zhang *et al.*, 2024a] Xinxun Zhang, Pengfei Jiao, Mengzhou Gao, Tianpeng Li, Yiming Wu, Huaming Wu, and Zhidong Zhao. Vggm: Variational graph gaussian mixture model for unsupervised change point detection in dynamic networks. *TIFS*, 2024.

[Zhang *et al.*, 2024b] Zhen Zhang, Meihan Liu, Anhui Wang, Hongyang Chen, Zhao Li, Jiajun Bu, and Bingsheng He. Collaborate to adapt: Source-free graph domain adaptation via bi-directional adaptation. In *WWW*, pages 664–675, 2024.