MIRAGE: MUTUAL INFORMATION-GUIDED RECONSTRUCTION AND ADAPTATION FOR GRAPH DOMAIN ADAPTATION WITH EDGE INCOMPLETENESS

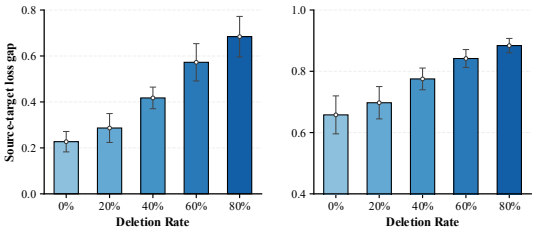
Anonymous authors
Paper under double-blind review

ABSTRACT

Graph Domain Adaptation (GDA) aims to transfer knowledge from a labeled source graph to an unlabeled target graph. Existing GDA methods typically rely on fully observed graph topologies for representation learning and cross-domain alignment. However, real-world graph topologies can be incomplete, with edges unobserved due to corruption, adversarial perturbations, or privacy constraints. In this paper, we formulate this scenario as *Edge-Incomplete Graph Domain Adaptation* (EIGDA), where domain shift coexists with structural incompleteness in source and target graphs. Moreover, we show that EIGDA poses a key challenge: missing edges compromise available neighborhood information and can further exacerbate the source-target performance gap. To address this issue, we propose **MIRAGE** (**M**utual **I**nformation-guided **R**econstruction and **A**daptation for **G**raph domain adaptation with **E**dge incompleteness), a novel algorithm that leverages mutual information maximization to guide transfer-aware edge selection, thereby allowing graph reconstruction to preserve local information while supporting cross-domain adaptation. Experiments on multiple benchmark datasets with varying missing-edge ratios show that MIRAGE consistently outperforms competitive baselines under structurally incomplete EIGDA settings.

1 INTRODUCTION

Graph Domain Adaptation (GDA) aims to transfer knowledge from a labeled source graph to an unlabeled target graph, thereby reducing annotation costs in label-scarce scenarios (Wu et al., 2020). Existing GDA methods typically learn node representations by propagating information over the input graph topology, and then align the source and target domains in the learned representation space (You et al., 2023; Wu et al., 2023; Shen et al., 2020). Their transferability is therefore closely tied to the structural signals encoded in the input graphs. However, real-world graphs are often incomplete due to noise, corruption, adversarial attacks, privacy constraints, or error-prone data collection (Chen et al., 2020; Kurant et al., 2010; Franceschini et al., 2014), leading to missing edges and incomplete connectivity information. For example, large online social networks are often only partially collected through graph crawling, leaving part of their connectivity unobserved (Kurant et al., 2010), while citation databases may omit citation links due to database errors (Franceschini et al., 2014).

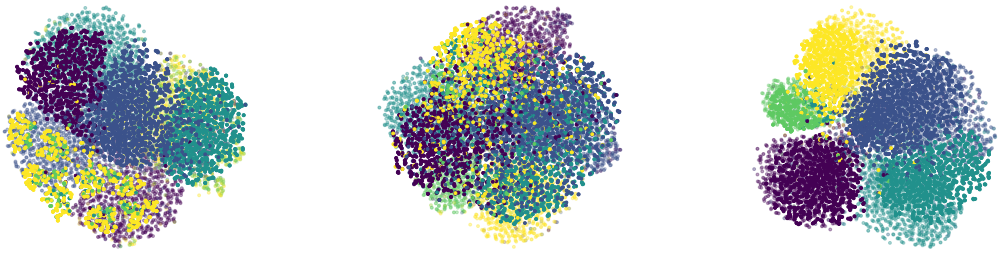


(a) Airport (b) Citation

Figure 1: Empirical analysis of structural incompleteness. The x-axis shows different edge deletion rates, and the y-axis reports the source-target performance gap. Error bars denote standard deviations over multiple runs.

To bridge this gap, we study *Edge-Incomplete Graph Domain Adaptation* (EIGDA), where source and target graphs are only partially observed due to missing edges. Critically, missing edges compromise the structural information available for representation learning, thereby weakening the capacity of node embeddings to capture local connectivity patterns. This poses a key challenge for EIGDA:

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) GDA on incomplete graphs (b) Reconstruction + GDA (c) MIRAGE

Figure 2: t-SNE visualization of learned node embeddings on Citation (ACMv9→DBLPv7) under the 80% edge deletion rate. We compare (a) conventional GDA on incomplete graphs, (b) independent graph reconstruction, implemented with VGAE (Kipf & Welling, 2016), followed by GDA, and (c) MIRAGE. Each color denotes a node class, and lighter/darker shades denote source/target nodes within the same class. Compared with the reconstruction + GDA pipeline, MIRAGE produces clearer class clusters with better source-target overlap within each class, indicating the benefit of transfer-aware reconstruction.

transferable correspondences between source and target nodes become harder to capture, and therefore existing GDA models cannot effectively mitigate the domain gap in EIGDA. To examine this effect, we progressively remove edges from both source and target graphs and track the source-target performance gap[1] of GRADE (Wu et al., 2023). As shown in Fig. 1, the performance gap consistently increases as the edge deletion rate becomes larger, indicating the impact of structural incompleteness on domain adaptation.

To address this impact, a straightforward strategy is to first recover the incomplete source and target graphs individually and then apply conventional GDA methods. However, existing graph recovery methods only rely on signals in a single domain (Kipf & Welling, 2016; Franceschi et al., 2019; Yu et al., 2020; Zhao et al., 2021; Jin et al., 2020), producing locally plausible topologies without considering its effect on cross-domain adaptation. Under domain shift, such independently recovered structures may reinforce domain-specific structural patterns, limiting their effectiveness in aligning source and target representations, as shown in Fig. 2(a,b).

To fill the gap, we propose Mutual Information-guided Reconstruction and Adaptation for Graph domain adaptation with Edge incompleteness (MIRAGE), which learns candidate-edge probabilities by jointly considering local structural preservation and source-to-target alignment. Specifically, MIRAGE constructs candidate edges from the observed graphs, learns a probability for each candidate edge, and adaptively selects edges to form the reconstructed graphs. The reconstruction is guided by two complementary objectives. Intra-domain mutual information encourages the reconstructed graphs to preserve available local neighborhood information, while cross-domain mutual information encourages them to align source and target representations at both the feature and structure levels to mitigate domain shift.

Our main contributions are summarized as follows:

- We investigate EIGDA, where source and target graphs have missing egdes, and empirically show that edge incompleteness can amplify the source-target performance gap in GDA, revealing a challenge beyond conventional domain shift.
- We propose **MIRAGE**, a mutual information-guided reconstruction algorithm for EIGDA that reconstructs the source and target graphs through learnable candidate-edge probabilities and adaptive edge selection. The reconstruction is jointly guided by intra-domain mutual information for local structural preservation and cross-domain mutual information for source-target alignment.
- Extensive experiments on multiple benchmark datasets under different edge deletion rates demonstrate that MIRAGE consistently outperforms graph reconstruction baselines and state-of-the-art GDA methods, especially under severe edge incompleteness.

[1] See Appendix A for details of the source-target performance gap metric.

2 Related Work

Graph Domain Adaptation. Graph domain adaptation aims to transfer knowledge from a labeled source graph to an unlabeled target graph under distribution shift (Wu et al., 2020). Early methods mainly learn transferable representations through adversarial alignment (Zhang et al., 2019; Shen et al., 2020; Wu et al., 2020; Zhang et al., 2021; Dai et al., 2022).

Recent studies address GDA from increasingly diverse perspectives. Methods for structural and distribution shifts exploit subtree discrepancy, spectral properties, structural reweighting, hierarchy, pairwise alignment, propagation, smoothness, causal structure, adaptive distribution alignment, and homophily-aware structure modeling (Wu et al., 2023; You et al., 2023; Pang et al., 2023; Liu et al., 2023; Shi et al., 2023; Liu et al., 2024c;a; Chen et al., 2025; Luo et al., 2025; Yang et al., 2025; Chen et al., 2026a;b; Wen et al., 2026; Fang et al., 2026a; 2025b;a). Other studies explore generative, probabilistic, and diffusion-based adaptation (Cai et al., 2024; Yin et al., 2025; Wang et al., 2025; Zhang et al., 2026), as well as class-imbalanced and prototype-based adaptation (Ma et al., 2025b;a). GDA has also been extended to source-free (Mao et al., 2024; Luo et al., 2024; Zhang et al., 2024; Luo et al., 2026), multi-source-free (Zhang & He, 2025a), open-set (Yin et al., 2024; Zhang & He, 2025b), semi-supervised (Qiao et al., 2023; 2024), multi-source (Lin et al., 2025; Shen et al., 2026), and test-time settings (Jin et al., 2023). Benchmarking studies further provide systematic evaluations of existing GDA methods (Liu et al., 2024b). In contrast to these directions, our work focuses on GDA when both source and target graphs have incomplete observed structures, where missing edges limit the neighborhood information available for cross-domain transfer.

Graph Structure Learning. Graph structure learning studies how to infer, refine, or reconstruct graph structures for downstream graph learning. Existing methods learn latent or improved graph structures through bilevel optimization (Franceschi et al., 2019), iterative graph refinement (Chen et al., 2020), topology denoising and robust structure learning (Jin et al., 2020; Sun et al., 2022), probabilistic edge modeling (Kipf & Welling, 2016), self-supervised structure learning (Fatemi et al., 2021), graph revision (Yu et al., 2020), and graph data augmentation (Zhao et al., 2021). Survey studies further summarize the progress and challenges of graph structure learning (Zhu et al., 2021). These methods show that modifying the observed topology can improve graph representation learning when the input graph is noisy, sparse, or partially observed.

3 Problem Setup

Definition 1 (***Edge-Incomplete Graph Domain Adaptation***)**.** Let $G^S = (V^S, E^S, A^S, X^S, Y^S)$ and $G^T = (V^T, E^T, A^T, X^T)$ denote the underlying source and target graphs, respectively. V^S and V^T denote the node sets (with N^S nodes and N^T nodes, respectively), and E^S and E^T denote the undirected edge sets. $A^S \in \mathbb{R}^{N^S \times N^S}$ and $A^T \in \mathbb{R}^{N^T \times N^T}$ denote the corresponding adjacency matrices. $A^S_{ij} = 1$ means the node i and node j are connected, and $A^S_{ij} = 0$ indicates no connection between them. The same is for A^T_{ij}. $X^S \in \mathbb{R}^{N^S \times m}$ and $X^T \in \mathbb{R}^{N^T \times m}$ are the node feature matrices, where m denotes the feature dimension. $Y^S \in \mathbb{R}^{N^S \times C}$ is the source label matrix with C classes. In Edge-Incomplete Graph Domain Adaptation (EIGDA), we denote $\hat{G}^S = (V^S, \hat{E}^S, \hat{A}^S, X^S, Y^S)$ as the incomplete source graph and $\hat{G}^T = (V^T, \hat{E}^T, \hat{A}^T, X^T)$ as the incomplete target graph, where $\hat{E}^S \subseteq E^S$ and $\hat{E}^T \subseteq E^T$ are the observed edge sets, and $\hat{A}^S$ and $\hat{A}^T$ are their corresponding adjacency matrices. We use $\hat{A}^S_{ij}$ and $\hat{A}^T_{ij}$ to denote the observed adjacency between nodes i and j in the source and target graphs, respectively. After reconstruction, MIRAGE obtains $\tilde{G}^S = (V^S, \tilde{E}^S, \tilde{A}^S, X^S, Y^S)$ and $\tilde{G}^T = (V^T, \tilde{E}^T, \tilde{A}^T, X^T)$, where $\tilde{E}^S$, $\tilde{E}^T$ and $\tilde{A}^S$, $\tilde{A}^T$ denote the reconstructed edge sets and adjacency matrices. $\tilde{A}^S_{ij}$ and $\tilde{A}^T_{ij}$ denote the corresponding adjacency between nodes i and j in the source and target graphs, respectively.

4 Methodology

An overview of our algorithm, MIRAGE, is shown in Fig. 3. Under mutual information guidance, MIRAGE reconstructs the incomplete source and target graphs in a transfer-aware manner, thus enhancing cross-domain alignment.

4.1 Shared Encoder

MIRAGE employs a graph encoder f_θ, parameterized by θ and shared between the source and target domains. Given the observed incomplete graphs, the encoder produces node embeddings $H^S = f_\theta(X^S, \hat{A}^S)$ and $H^T = f_\theta(X^T, \hat{A}^T)$, where $H^S = [h_1^S, \ldots, h_{N^S}^S]^\top \in \mathbb{R}^{N^S \times d}$ and $H^T = [h_1^T, \ldots, h_{N^T}^T]^\top \in \mathbb{R}^{N^T \times d}$. h_i^S and h_i^T denote the embeddings of node i in the source and target graphs, respectively, and d is the embedding dimension.

4.2 Mutual Information-guided Edge Reconstruction

MI-guided reconstruction. Mutual information maximization has been widely used to learn informative graph representations by preserving dependencies between inputs and embeddings (Velickovic et al., 2019; Hjelm et al., 2019; Peng et al., 2020). In EIGDA, we adapt this idea from representation learning to edge reconstruction: intra-domain MI preserves available local neighborhood information, while cross-domain MI introduces transfer-aware signals between source and target graphs.

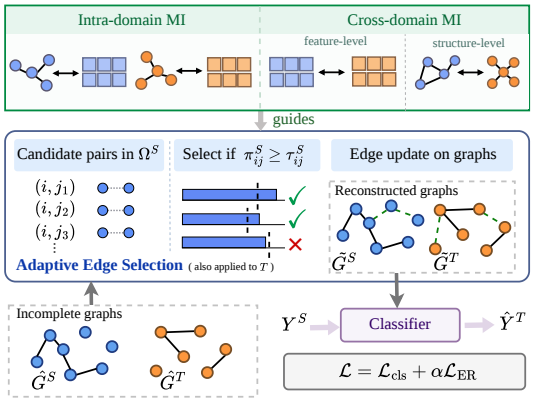


Figure 3: Overview of MIRAGE. MIRAGE reconstructs incomplete source and target graphs through adaptive edge selection. Intra-domain MI preserves local neighborhood information, while cross-domain MI guides edge selection toward source-to-target alignment at the feature level and structure level. The reconstructed graphs are then used for target prediction.

Candidate edge selection. For each node, we retrieve its K most similar nodes based on cosine similarity in the raw feature space. The source and target candidate edge sets are constructed as $\Omega^S = \{(i,j) \mid j \in \text{TopK}_x^S(i)\} \cup \hat{E}^S$ and $\Omega^T = \{(i,j) \mid j \in \text{TopK}_x^T(i)\} \cup \hat{E}^T$, where $\text{TopK}_x^S(i)$ and $\text{TopK}_x^T(i)$ denote the sets of K nodes with the highest cosine similarity to node i in the source and target feature spaces, respectively. Since the graphs are undirected, each candidate set is symmetrized and each unordered node pair is considered only once. Including the observed edges ensures that all available edges are retained as reconstruction candidates.

For each source candidate pair $(i,j) \in \Omega^S$, MIRAGE associates a trainable edge logit z_{ij}^S and computes the selection probability $\pi_{ij}^S = \sigma(z_{ij}^S)$, where $\sigma(\cdot)$ is the sigmoid function. Similarly, for each target candidate pair $(i,j) \in \Omega^T$, we define $\pi_{ij}^T = \sigma(z_{ij}^T)$. Instead of using a fixed global threshold, MIRAGE computes a pair-specific dynamic threshold from node embeddings: $\tau_{ij}^S = \sigma((h_i^S)^\top h_j^S)$ and $\tau_{ij}^T = \sigma((h_i^T)^\top h_j^T)$.

Because the same reconstruction rule is applied to both domains, we state it below for the source graph. For each $(i,j) \in \Omega^S$,

$$\tilde{A}_{ij}^S = \max\left\{\mathbb{I}\left(\pi_{ij}^S \geq \tau_{ij}^S\right), \mathbb{I}\left((i,j) \in \hat{E}^S\right)\right\}. \tag{1}$$

For pairs outside Ω^S, we set $\tilde{A}_{ij}^S = 0$. The target-domain adjacency $\tilde{A}_{ij}^T$ is defined analogously, with S replaced by T. The resulting edge sets are $\tilde{E}^S = \{(i,j) \mid \tilde{A}_{ij}^S = 1\}$ and $\tilde{E}^T = \{(i,j) \mid \tilde{A}_{ij}^T = 1\}$. This rule selects high-confidence candidate edges while preserving all observed edges.

Edge reconstruction loss. Because the true missing edges are unavailable during training, we define the reconstruction objective with respect to the observed incomplete adjacency over the candidate set. For the source graph, we use a binary cross-entropy reconstruction loss:

$$\mathcal{L}_{\text{edge}}^S = -\frac{1}{|\Omega^S|} \sum_{(i,j) \in \Omega^S} \left[\hat{A}_{ij}^S \log \pi_{ij}^S + (1 - \hat{A}_{ij}^S) \log(1 - \pi_{ij}^S)\right]. \tag{2}$$

The target reconstruction loss $\mathcal{L}_{\text{edge}}^T$ is defined analogously with S replaced by T.

4.2.1 INTRA-DOMAIN MUTUAL INFORMATION

The intra-domain mutual information encourages node representations to preserve local information available in each observed incomplete graph. Let $\hat{\mathcal{N}}_i^S = \{j \mid (i,j) \in \hat{E}^S\}$ and $\hat{\mathcal{N}}_i^T = \{j \mid (i,j) \in \hat{E}^T\}$ denote the observed neighbor sets of node i in the source and target graphs, respectively. Following the local mutual-information modeling in GMI (Peng et al., 2020), we define the overall intra-domain mutual information as

$$I_{\text{intra}} = \sum_{i=1}^{N^S} \sum_{j \in \hat{\mathcal{N}}_i^S} I(h_i^S; x_j^S) + \sum_{i=1}^{N^T} \sum_{j \in \hat{\mathcal{N}}_i^T} I(h_i^T; x_j^T), \tag{3}$$

where $I(\cdot;\cdot)$ denotes mutual information, estimated using a Jensen–Shannon estimator as detailed in Appendix B. Maximizing I_{intra} encourages node embeddings to retain information from observed local neighborhoods, thereby providing structural guidance for candidate-edge selection.

4.2.2 CROSS-DOMAIN MUTUAL INFORMATION

Unlike graph infomax objectives that operate only within a single graph, our cross-domain mutual information is designed to provide transfer-aware guidance for reconstruction. We instantiate cross-domain mutual information at both the feature and structure levels. We first obtain source and target feature-level graph summaries using mean pooling, given by $c_h^S = \frac{1}{N^S}\sum_{i=1}^{N^S} h_i^S$ and $c_h^T = \frac{1}{N^T}\sum_{i=1}^{N^T} h_i^T$. To incorporate candidate-edge information, we define the reconstructed-neighborhood representations as $r_i^S = \sum_{j:(i,j)\in\Omega^S} \pi_{ij}^S h_j^S$ and $r_i^T = \sum_{j:(i,j)\in\Omega^T} \pi_{ij}^T h_j^T$, with the corresponding domain-level summaries $c_r^S = \frac{1}{N^S}\sum_{i=1}^{N^S} r_i^S$ and $c_r^T = \frac{1}{N^T}\sum_{i=1}^{N^T} r_i^T$.

We denote the feature-level and structure-level cross-domain mutual information by I_{cf} and I_{cs}:

$$I_{\text{cf}} = \sum_{i=1}^{N^S} I(h_i^S; c_h^T) + \sum_{i=1}^{N^T} I(h_i^T; c_h^S), \quad I_{\text{cs}} = \sum_{i=1}^{N^S} I(r_i^S; c_r^T) + \sum_{i=1}^{N^T} I(r_i^T; c_r^S),$$

where the subscripts cf and cs denote cross-feature and cross-structure. The feature-level term encourages node embeddings to capture transferable domain-level information, whereas the structure-level term evaluates cross-domain mutual information through the candidate-edge-weighted neighborhood representations. Importantly, the structure-level term directly depends on π_{ij}^S and π_{ij}^T, thereby providing transfer-aware feedback to the edge-selection probabilities used for reconstruction.

We combine the two terms to define cross-domain mutual information:

$$I_{\text{cross}} = \lambda_{\text{cf}} I_{\text{cf}} + \lambda_{\text{cs}} I_{\text{cs}}, \tag{4}$$

where λ_{cf} and λ_{cs} control the contributions of the cross-feature and cross-structure dependencies, respectively. Maximizing I_{cross} encourages representations derived from the reconstructed neighborhoods to remain transferable across the source and target domains.

Combining adjacency reconstruction, intra-domain MI, and cross-domain MI gives the edge reconstruction objective

$$\mathcal{L}_{\text{ER}} = \mathcal{L}_{\text{edge}}^S + \mathcal{L}_{\text{edge}}^T - \lambda_{\text{intra}} I_{\text{intra}} - I_{\text{cross}}. \tag{5}$$

Minimizing Eq. (5) jointly encourages consistency with the observed graph structures, preservation of local neighborhood information, and transfer-aware edge selection.

4.3 TARGET NODE CLASSIFICATION

After reconstruction, the source and target graphs $\tilde{G}^S$ and $\tilde{G}^T$ are used for node classification. Let g_ϕ denote the node classifier parameterized by ϕ. The corresponding class prediction matrices are $\hat{Y}^S = g_\phi(\tilde{G}^S) \in [0,1]^{N^S \times C}$ and $\hat{Y}^T = g_\phi(\tilde{G}^T) \in [0,1]^{N^T \times C}$, where $\hat{Y}_{ic}^S$ and $\hat{Y}_{ic}^T$ denote the predicted probabilities that node i belongs to class c in the source and target graphs, respectively.

Using the source labels Y^S, the supervised classification loss is

$$\mathcal{L}_{\text{cls}} = -\frac{1}{N^S} \sum_{i=1}^{N^S} \sum_{c=1}^{C} Y_{ic}^S \log \hat{Y}_{ic}^S. \tag{6}$$

The overall training objective is

$$\mathcal{L} = \mathcal{L}_{\text{cls}} + \alpha \mathcal{L}_{\text{ER}}, \tag{7}$$

where α balances source-label supervision and mutual information-guided edge reconstruction.

5 EXPERIMENT

5.1 DATASETS

We evaluate MIRAGE on three real-world benchmark datasets: Blog (Shen et al., 2020), Citation (Zhang et al., 2021), and Airport (Ribeiro et al., 2017). **Blog** consists of two social-network domains, Blog1 and Blog2; **Citation** contains three citation-network domains, ACMv9, DBLPv7, and Citationv1; and **Airport** includes three air-traffic domains, Brazil, USA, and Europe. We further evaluate MIRAGE on a 10-class subset of OGBN-MAG (Wang et al., 2020; Hu et al., 2020), following the dataset construction in Fang et al. (2026b). Detailed experiment results are reported in Appendix K. For all datasets, each domain is treated as an individual graph, and domain adaptation is performed between different domains. Detailed dataset statistics are summarized in Table 6. Details of the compared baselines are provided in Appendix C.

5.2 EXPERIMENTAL SETUP

We evaluate all methods on target-domain node classification under the EIGDA setting. In each transfer task, node labels are available only in the source graph, while the target graph is treated as unlabeled during training. Target labels are used only for final evaluation. Following prior studies on graph learning with incomplete structures (Franceschi et al., 2019; Chen et al., 2020; Ren & Wu, 2026), we simulate structural incompleteness by randomly removing edges from both source and target graphs via Bernoulli sampling, with deletion rates in $\{0\%, 20\%, 40\%, 60\%, 80\%\}$. Node features and labels are kept unchanged across different deletion rates, so that the effect of missing graph connectivity can be isolated.

We compare MIRAGE with both graph structure reconstruction baselines and GDA baselines. For VGAE-based baselines, we first reconstruct the source and target graph structures independently using VGAE (Kipf & Welling, 2016). A GNN classifier, either GCN (Kipf & Welling, 2017) or GAT (Velickovic et al., 2018), is then trained on the reconstructed source graph with source labels and directly applied to the reconstructed target graph for inference. For task-driven structure learning baselines, including GRCN (Yu et al., 2020) and IDGL (Chen et al., 2020), we adopt a source-supervised transfer protocol. Since these methods rely on downstream supervision to learn graph structures, they are trained only on the incomplete source graph using source labels, and the learned structure refinement model and classifier are then applied to the incomplete target graph. No target labels or cross-domain adaptation objectives are used for these baselines. For GDA baselines, we follow their original unsupervised domain adaptation settings, where adaptation is performed between the incomplete source and target graphs without explicit graph reconstruction.

Following standard practice, we report classification accuracy (Acc) and macro-averaged F1 score (Macro-F1) on the target domain. All results are averaged over three random runs with different edge-deletion seeds and model initialization seeds.

5.3 RESULTS AND ANALYSIS

We evaluate MIRAGE on three domain adaptation tasks under increasing graph sparsity (controlled by edge deletion rates from 0% to 80%). The tasks include: Blog (Blog1 $\rightarrow$ Blog2), Citation (ACMv9 $\rightarrow$ DBLPv7), and Airport (Brazil $\rightarrow$ USA). As shown in Tables 1, 2, and 3, MIRAGE consistently achieves the best performance in the vast majority of settings across all datasets and deletion rates.

Compared with non-GDA methods, which only perform structure reconstruction, MIRAGE shows significantly better performance, especially under high edge deletion rates.

Compared with GDA baselines, MIRAGE also achieves superior performance, particularly when the graph becomes sparse. This suggests that existing GDA methods are sensitive to incomplete graph structures, as they rely on observed topology for effective domain alignment. Furthermore, as the

Table 1: Node classification performance on Blog (Blog1→Blog2) under different edge deletion rates.

Methods	Delete 0%		Delete 20%		Delete 40%		Delete 60%		Delete 80%	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
GCN+VGAE	0.334	0.305	0.323	0.296	0.307	0.288	0.278	0.265	0.255	0.249
GAT+VGAE	0.259	0.215	0.253	0.183	0.238	0.201	0.187	0.112	0.180	0.094
GRCN	0.279	0.238	0.247	0.210	0.215	0.191	0.202	0.180	0.192	0.157
IDGL	0.284	0.251	0.265	0.243	0.237	0.200	0.208	0.174	0.186	0.140
UDAGCN	0.167	0.048	0.167	0.048	0.167	0.048	0.167	0.048	0.167	0.048
ACDNE	0.250	0.249	0.238	0.198	0.217	0.167	0.195	0.135	0.182	0.124
ASN	0.225	0.159	0.214	0.151	0.210	0.143	0.203	0.139	0.194	0.136
GRADE	0.240	0.168	0.231	0.157	0.221	0.152	0.232	0.148	0.220	0.139
SpecReg	0.183	0.092	0.181	0.085	0.179	0.079	0.177	0.068	0.175	0.063
PairAlign	0.223	0.154	0.209	0.173	0.214	0.141	0.216	0.144	0.219	0.125
StruRW	0.414	0.405	0.405	0.392	0.396	0.384	0.388	0.377	0.371	0.359
HGDA	0.436	0.412	0.423	0.418	0.407	0.384	0.381	0.352	0.356	0.320
MIRAGE	**0.538**	**0.504**	**0.524**	**0.481**	**0.519**	**0.477**	**0.510**	**0.461**	**0.494**	**0.442**

Table 2: Node classification performance on Citation (ACMv9→DBLPv7) under different edge deletion rates.

Methods	Delete 0%		Delete 20%		Delete 40%		Delete 60%		Delete 80%	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
GCN+VGAE	0.589	0.490	0.575	0.486	0.561	0.482	0.538	0.471	0.513	0.456
GAT+VGAE	0.587	0.468	0.575	0.459	0.564	0.450	0.542	0.441	0.519	0.421
GRCN	0.592	0.575	0.572	0.549	0.550	0.533	0.528	0.491	0.492	0.477
IDGL	0.601	0.589	0.584	0.542	0.551	0.515	0.527	0.484	0.508	0.460
UDAGCN	0.678	0.553	0.625	0.507	0.599	0.489	0.574	0.471	0.492	0.426
ACDNE	0.665	0.604	0.613	0.580	0.591	0.533	0.532	0.501	0.509	0.466
ASN	0.663	0.629	0.621	0.595	0.606	0.545	0.556	0.514	0.521	0.487
GRADE	**0.683**	**0.650**	0.632	0.609	0.608	0.552	0.553	0.514	0.507	0.455
SpecReg	0.682	0.553	0.619	0.527	0.597	0.496	0.571	0.481	0.512	0.467
PairAlign	0.609	0.489	0.589	0.503	0.546	0.485	0.517	0.470	0.492	0.426
StruRW	0.609	0.579	0.576	0.542	0.547	0.520	0.516	0.483	0.489	0.465
HGDA	0.656	0.573	0.616	0.573	0.589	0.553	0.547	0.522	0.510	0.471
MIRAGE	0.682	0.646	**0.663**	**0.624**	**0.633**	**0.597**	**0.618**	**0.560**	**0.596**	**0.536**

Table 3: Node classification performance on Airport (Brazil→USA) under different edge deletion rates.

Methods	Delete 0%		Delete 20%		Delete 40%		Delete 60%		Delete 80%	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
GCN+VGAE	0.428	0.389	0.399	0.344	0.392	0.331	0.373	0.298	0.352	0.287
GAT+VGAE	0.380	0.327	0.351	0.282	0.322	0.240	0.291	0.199	0.291	0.190
GRCN	0.408	0.350	0.388	0.325	0.343	0.315	0.307	0.280	0.284	0.257
IDGL	0.387	0.342	0.366	0.318	0.374	0.301	0.311	0.298	0.290	0.267
UDAGCN	0.249	0.099	0.249	0.099	0.249	0.099	0.249	0.099	0.249	0.099
ACDNE	0.267	0.186	0.265	0.183	0.266	0.184	0.263	0.181	0.264	0.183
ASN	0.448	0.435	0.416	0.405	0.401	0.398	0.397	0.365	0.386	0.353
GRADE	0.431	0.392	0.395	0.347	0.383	0.331	0.375	0.324	0.361	0.317
SpecReg	0.327	0.275	0.317	0.263	0.307	0.251	0.292	0.241	0.282	0.239
PairAlign	0.423	0.423	0.395	0.376	0.376	0.361	0.351	0.342	0.335	0.316
StruRW	0.373	0.353	0.358	0.342	0.339	0.312	0.315	0.301	0.299	0.284
HGDA	0.461	0.432	0.445	0.413	0.411	0.398	0.396	0.375	0.384	0.377
MIRAGE	**0.558**	**0.516**	**0.556**	**0.505**	**0.552**	**0.496**	**0.548**	**0.485**	**0.539**	**0.481**

edge deletion rate increases, the performance of all baselines degrades, while MIRAGE remains relatively stable. These results show that MIRAGE can better preserve adaptation performance as the observed graph structure becomes increasingly incomplete.

5.4 Ablation Study

We evaluate the contributions of different MIRAGE components using the following variants:

- **w/o cross-domain MI**: removes cross-domain MI, retaining intra-domain MI and edge reconstruction.
- **w/o intra-domain MI**: removes intra-domain MI while retaining cross-domain MI and edge reconstruction.
- **w/o MI**: removes all MI guidance, retaining only edge reconstruction.
- **Feat-only**: retains only the feature-level component of cross-domain MI, together with intra-domain MI and edge reconstruction.
- **Struct-only**: retains only the structure-level component of cross-domain MI, together with intra-domain MI and edge reconstruction.

Fig. 4 compares these variants under 60% and 80% edge deletion rates on ACMv9→DBLPv7 (A→D) and Blog1→Blog2 (B1→B2). Removing either cross-domain MI or intra-domain MI consistently reduces accuracy on both tasks, showing that both transfer-aware guidance and local structural information contribute to MIRAGE. Removing all MI guidance leads to a larger performance drop, indicating that edge reconstruction alone is insufficient. The Feat-only and Struct-only variants further show that using either feature-level or structure-level cross-domain MI alone does not recover the performance of the full model. In contrast, MIRAGE consistently achieves the highest accuracy across both tasks and deletion rates, showing the benefit of combining intra-domain MI with both feature-level and structure-level cross-domain dependencies.

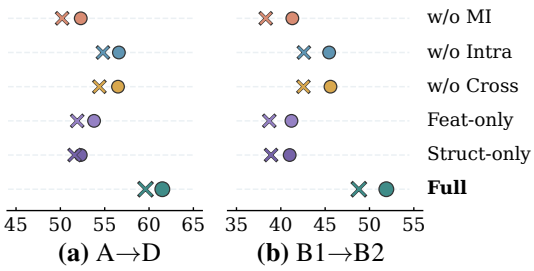


Figure 4: Ablation study results. The horizontal axis reports accuracy. • and × markers denote results at 60% and 80% deletion, respectively.

5.5 Evaluation with Correlated Missing Edges

Random edge deletion provides a controlled setting but assumes that missing edges are independently sampled. In practice, graph incompleteness can be structurally correlated. Motivated by BFS-based network collection (Kurant et al., 2010), we further evaluate MIRAGE under a BFS-based edge deletion setting on Blog1→Blog2.

Starting from a seed node in each domain, BFS visits nodes layer by layer, and we retain the discovered edges until the required number of observed edges is reached. The nodes and node features remain unchanged, and the procedure is applied separately to the source and target graphs. This produces structurally correlated missing edges concentrated outside the explored regions.

As shown in Table 4, MIRAGE achieves accuracies of 0.481, 0.473, and 0.462 at 40%, 60%, and 80% deletion rates, respectively, and consistently outperforms the compared baselines. The results show that MIRAGE remains effective when edge incompleteness follows a correlated pattern induced by graph traversal rather than independent deletion.

5.6 Comparison with Independent Reconstruction Baselines

To further compare with the independent reconstruction pipeline illustrated in Fig. 2(b), we enhance three representative GDA baselines, SpecReg (You et al., 2023), PairAlign (Liu et al., 2024c), and StruRW (Liu et al., 2023), with VGAE (Kipf & Welling, 2016). Specifically, VGAE is first applied independently to reconstruct the incomplete source and target graphs, after which each GDA method is

Table 4: Target classification accuracy under BFS-based edge deletion on Blog1→Blog2.

Method	40%	60%	80%
GRADE	0.239	0.228	0.226
SpecReg	0.205	0.183	0.176
PairAlign	0.213	0.208	0.202
GRCN	0.201	0.190	0.174
MIRAGE	**0.481**	**0.473**	**0.462**

Table 5: Classification accuracy of VGAE-enhanced GDA baselines and MIRAGE under 60% and 80% edge deletion rates.

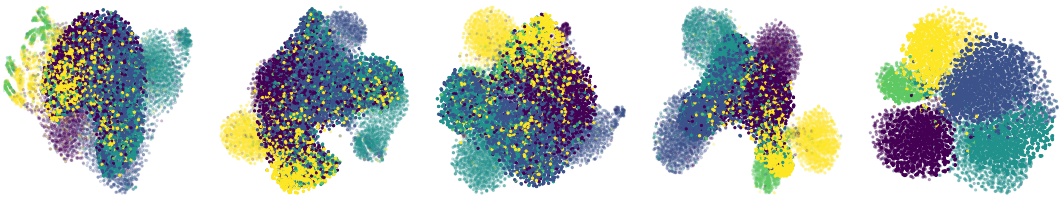
Method	B1→B2		A→D		B→U	
	60%	80%	60%	80%	60%	80%
SpecReg+VGAE	0.181	0.178	0.563	0.538	0.305	0.283
PairAlign+VGAE	0.217	0.204	0.510	0.494	0.331	0.308
StruRW+VGAE	0.366	0.347	0.526	0.512	0.318	0.299
MIRAGE	**0.510**	**0.494**	**0.618**	**0.596**	**0.548**	**0.539**

trained on the reconstructed graphs. We evaluate one representative task from each dataset under 60% and 80% edge deletion rates. For brevity, we use B1→B2, A→D, and B→U to denote Blog1→Blog2, ACMv9→DBLPv7, and Brazil→USA, respectively.

As shown in Table 5, MIRAGE achieves the highest accuracy in all six settings, outperforming the strongest VGAE-enhanced baseline by 5.5–23.1 percentage points. These results indicate that independently reconstructing the source and target graphs before adaptation is less effective than integrating transfer-aware reconstruction into the adaptation process.

6 Visualization

To further demonstrate the advantage of our proposed method, we visualize the embedding results under the 80% edge deletion rate on Citation (ACMv9→DBLPv7) generated by competitive methods UDAGCN (Wu et al., 2020), GRADE (Wu et al., 2023), SpecReg (You et al., 2023), and PairAlign (Liu et al., 2024c), which are shown in Fig. 5. Each color represents a distinct node class, while within each class, lighter and darker shades denote nodes from the source and target domains, respectively. MIRAGE achieves the best source-target overlap within each class when 80% of edges are missing. In other words, the representations learned by MIRAGE better align the source and target distributions. These results suggest that our transfer-aware reconstruction strategy improves domain adaptation under severe edge incompleteness.



(a) UDAGCN (b) GRADE (c) SpecReg (d) PairAlign (e) MIRAGE

Figure 5: t-SNE visualization of source and target node representations on Citation dataset.

7 Conclusion

We presented MIRAGE, a mutual information-guided reconstruction and adaptation algorithm for Edge-Incomplete Graph Domain Adaptation (EIGDA), where both source and target graphs suffer from missing edges. MIRAGE learns candidate-edge probabilities to reconstruct graph structures under both local-preservation and transfer-aware guidance. Experiments under different edge deletion rates show that MIRAGE consistently improves over both structure-reconstruction baselines and existing GDA methods, with particularly strong robustness under severe structural sparsity. These results highlight the importance of transfer-aware graph reconstruction for domain adaptation on incomplete graphs.

AI USE STATEMENT

In this work, we did not use generative AI tools for tasks requiring disclosure, including developing theoretical models or conceptual frameworks, formulating mathematical claims, designing the methodology or experiments, implementing the proposed method, or interpreting experimental results. The remaining required-disclosure tasks are not applicable to this work. Additionally, we used generative AI tools to edit the manuscript for language polishing and readability improvement. We have reviewed all AI-assisted edits to ensure technical correctness, consistency with the intended meaning, and accuracy of the final presentation. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

Ruichu Cai, Fengzhu Wu, Zijian Li, Pengfei Wei, Lingling Yi, and Kun Zhang. Graph domain adaptation: A generative view. *ACM Transactions on Knowledge Discovery from Data*, 2024.

Wei Chen, Guo Ye, Yakun Wang, Zhao Zhang, Libang Zhang, Daixin Wang, Zhiqiang Zhang, and Fuzhen Zhuang. Smoothness really matters: A simple yet effective approach for unsupervised graph domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025.

Wei Chen, Xingyu Guo, Shuang Li, Zhao Zhang, Yan Zhong, Fuzhen Zhuang, and DeQing Wang. Learning adaptive distribution alignment with neural characteristic function for graph domain adaptation. In *International Conference on Learning Representations*, 2026a.

Wei Chen, Xingyu Guo, Shuang Li, Yan Zhong, Zhao Zhang, Fuzhen Zhuang, Hongrui Liu, Libang Zhang, Guo Ye, and Huimei He. Learning structure-semantic evolution trajectories for graph domain adaptation. In *International Conference on Learning Representations*, 2026b.

Yu Chen, Lingfei Wu, and Mohammed J. Zaki. Iterative deep graph learning for graph neural networks: Better and robust node embeddings. In *Advances in Neural Information Processing Systems*, 2020.

Quanyu Dai, Xiao-Ming Wu, Jiaren Xiao, Xiao Shen, and Dan Wang. Graph transfer learning via adversarial domain adaptation with graph convolution. *IEEE Transactions on Knowledge and Data Engineering*, 2022.

Ruiyi Fang, Bingheng Li, Zhao Kang, Qiuhao Zeng, Nima Hosseini Dashtbayaz, Ruizhi Pu, Charles Ling, and Boyu Wang. On the benefits of attribute-driven graph domain adaptation. In *The Thirteenth International Conference on Learning Representations*, 2025a.

Ruiyi Fang, Bingheng Li, Jingyu Zhao, Ruizhi Pu, Qiuhao Zeng, Gezheng Xu, Charles Ling, and Boyu Wang. Homophily enhanced graph domain adaptation. In *Forty-second International Conference on Machine Learning*, 2025b.

Ruiyi Fang, Shuo Wang, Ruizhi Pu, Qiuhao Zeng, Hao Zheng, Ziyan Wang, Jiale Cai, Zhimin Mei, Song Tang, Charles Ling, and Boyu Wang. Graph domain adaptation via homophily-agnostic reconstructing structure. In *Fortieth AAAI Conference on Artificial Intelligence*, 2026a.

Ruiyi Fang, Jingyu Zhao, Shuo Wang, Ruizhi Pu, Bingheng Li, Jiale Cai, Zhihao Li, Zihao Jing, Jian Zhu, Song Tang, et al. Saga: Structural aggregation guided alignment with dynamic view and neighborhood order selection for multiview graph domain adaptation. In *International Conference on Learning Representations*, 2026b.

Bahare Fatemi, Layla El Asri, and Seyed Mehran Kazemi. SLAPS: self-supervision improves structure learning for graph neural networks. In *Advances in Neural Information Processing Systems*, 2021.

Luca Franceschi, Mathias Niepert, Massimiliano Pontil, and Xiao He. Learning discrete structures for graph neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

Fiorenzo Franceschini, Domenico A. Maisano, and Luca Mastrogiacomo. Scientific journal publishers and omitted citations in bibliometric databases: Any relationship? *Journal of Informetrics*, 2014.

R. Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Philip Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. In *7th International Conference on Learning Representations*, 2019.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. In *Advances in Neural Information Processing Systems*, 2020.

Wei Jin, Yao Ma, Xiaorui Liu, Xianfeng Tang, Suhang Wang, and Jiliang Tang. Graph structure learning for robust graph neural networks. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2020.

Wei Jin, Tong Zhao, Jiayuan Ding, Yozen Liu, Jiliang Tang, and Neil Shah. Empowering graph representation learning with test-time graph transformation. In *International Conference on Learning Representations*, 2023.

Thomas N Kipf and Max Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations*, 2017.

Maciej Kurant, Athina Markopoulou, and Patrick Thiran. On the bias of BFS (breadth first search). In *22nd International Teletraffic Congress*, 2010.

Chuanyun Lin, Xi Zhou, and Xiao Shen. Hierarchical graph contrastive domain adaptation for multi-source cross-network node classification. *Expert Systems with Applications*, 2025.

Meihan Liu, Zeyu Fang, Zhen Zhang, Ming Gu, Sheng Zhou, Xin Wang, and Jiajun Bu. Rethinking propagation for unsupervised graph domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024a.

Meihan Liu, Zhen Zhang, Jiachen Tang, Jiajun Bu, Bingsheng He, and Sheng Zhou. Revisiting, benchmarking and understanding unsupervised graph domain adaptation. In *Advances in Neural Information Processing Systems Datasets and Benchmarks Track*, 2024b.

Shikun Liu, Tianchun Li, Yongbin Feng, Nhan Tran, Han Zhao, Qiang Qiu, and Pan Li. Structural reweighting improves graph domain adaptation. In *International Conference on Machine Learning*, 2023.

Shikun Liu, Deyu Zou, Han Zhao, and Pan Li. Pairwise alignment improves graph domain adaptation. In *Forty-first International Conference on Machine Learning*, 2024c.

Junyu Luo, Zhiping Xiao, Yifan Wang, Xiao Luo, Jingyang Yuan, Wei Ju, Langechuan Liu, and Ming Zhang. Rank and align: Towards effective source-free graph domain adaptation. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, 2024.

Junyu Luo, Yuhao Tang, Yiwei Fu, Xiao Luo, Zhizhuo Kou, Zhiping Xiao, Wei Ju, Wentao Zhang, and Ming Zhang. Sparse causal discovery with generative intervention for unsupervised graph domain adaptation. In *Forty-second International Conference on Machine Learning*, 2025.

Junyu Luo, Haoyu Tao, Xiao Luo, Yusheng Zhao, Zhiping Xiao, Dailan He, Wei Ju, Chong Chen, Xian-Sheng Hua, and Ming Zhang. Robust cross supervision with target mining for source-free graph domain adaptation. *IEEE Transactions on Knowledge and Data Engineering*, 2026.

Xin Ma, Yifan Wang, Siyu Yi, Wei Ju, Junyu Luo, Yusheng Zhao, Xiao Luo, and Jiancheng Lv. Dual prototype-enhanced contrastive framework for class-imbalanced graph domain adaptation. In *Advances in Neural Information Processing Systems*, 2025a.

Xin Ma, Yifan Wang, Siyu Yi, Wei Ju, Bei Wu, Ziyue Qiao, Chenwei Tang, and Jiancheng Lv. PALA: class-imbalanced graph domain adaptation via prototype-anchored learning and alignment. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, 2025b.

Haitao Mao, Lun Du, Yujia Zheng, Qiang Fu, Zelin Li, Xu Chen, Shi Han, and Dongmei Zhang. Source free graph unsupervised domain adaptation. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, 2024.

Jinhui Pang, Zixuan Wang, Jiliang Tang, Mingyan Xiao, and Nan Yin. Sa-gda: Spectral augmentation for graph domain adaptation. In *Proceedings of the 31st ACM international conference on multimedia*, 2023.

Zhen Peng, Wenbing Huang, Minnan Luo, Qinghua Zheng, Yu Rong, Tingyang Xu, and Junzhou Huang. Graph representation learning via graphical mutual information maximization. In *WWW '20: The Web Conference 2020*, 2020.

Ziyue Qiao, Xiao Luo, Meng Xiao, Hao Dong, Yuanchun Zhou, and Hui Xiong. Semi-supervised domain adaptation in graph transfer learning. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, 2023.

Ziyue Qiao, Meng Xiao, Weiyu Guo, Xiao Luo, and Hui Xiong. Information filtering and interpolating for semi-supervised graph domain adaptation. *Pattern Recognition*, 2024.

Jianxin Ren and Weining Wu. Discrete structure augmentation for graph convolutional networks. In *Fortieth AAAI Conference on Artificial Intelligence*, 2026.

Leonardo Filipe Rodrigues Ribeiro, Pedro H. P. Saverese, and Daniel R. Figueiredo. *struc2vec*: Learning node representations from structural identity. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017.

Xiao Shen, Quanyu Dai, Fu-Lai Chung, Wei Lu, and Kup-Sze Choi. Adversarial deep network embedding for cross-network node classification. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020.

Xiao Shen, Chuanyun Lin, Zhihao Chen, Junwei Gong, and Xi Zhou. Prototypical multi-source graph contrastive domain adaptation. *Pattern Recognition*, 2026.

Boshen Shi, Yongqing Wang, Fangda Guo, Jiangli Shao, Huawei Shen, and Xueqi Cheng. Improving graph domain adaptation with network hierarchy. In *Proceedings of the 32nd ACM international conference on information and knowledge management*, 2023.

Qingyun Sun, Jianxin Li, Hao Peng, Jia Wu, Xingcheng Fu, Cheng Ji, and Philip S. Yu. Graph structure learning with variational information bottleneck. In *Thirty-Sixth AAAI Conference on Artificial Intelligence*, 2022.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *6th International Conference on Learning Representations*, 2018.

Petar Velickovic, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R. Devon Hjelm. Deep graph infomax. In *7th International Conference on Learning Representations*, 2019.

Kuansan Wang, Zhihong Shen, Chiyuan Huang, Chieh-Han Wu, Yuxiao Dong, and Anshul Kanakia. Microsoft academic graph: When experts are not enough. *Quantitative Science Studies*, 2020.

Mengzhu Wang, Wenhao Ren, Yu Zhang, Yanlong Fan, Dianxi Shi, Luoxi Jing, and Nan Yin. Gaussian mixture model for graph domain adaptation. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, 2025.

Hongwei Wen, Can Zhang, Haoyu He, Hanyuan Hang, and Minglong Lei. Progressive graph structure adjustment for homophily shift adaptation. In *Forty-third International Conference on Machine Learning*, 2026.

Jun Wu, Jingrui He, and Elizabeth A. Ainsworth. Non-iid transfer learning on graphs. In *Thirty-Seventh AAAI Conference on Artificial Intelligence*, 2023.

Man Wu, Shirui Pan, Chuan Zhou, Xiaojun Chang, and Xingquan Zhu. Unsupervised domain adaptive graph convolutional networks. In *WWW '20: The Web Conference 2020*, 2020.

Liang Yang, Xin Chen, Jiaming Zhuo, Di Jin, Chuan Wang, Xiaochun Cao, Zhen Wang, and Yuanfang Guo. Disentangled graph spectral domain adaptation. In *Forty-second International Conference on Machine Learning*, 2025.

Nan Yin, Mengzhu Wang, Zhenghan Chen, Li Shen, Huan Xiong, Bin Gu, and Xiao Luo. Dream: Dual structured exploration with mixup for open-set graph domain adaption. In *The Twelfth International Conference on Learning Representations*, 2024.

Nan Yin, Li Shen, Mengzhu Wang, Xinwang Liu, Chong Chen, and Xian-Sheng Hua. Dream: a dual variational framework for unsupervised graph domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

Yuning You, Tianlong Chen, Zhangyang Wang, and Yang Shen. Graph domain adaptation via theory-grounded spectral regularization. In *The Eleventh International Conference on Learning Representations*, 2023.

Donghan Yu, Ruohong Zhang, Zhengbao Jiang, Yuexin Wu, and Yiming Yang. Graph-revised convolutional network. In *Machine Learning and Knowledge Discovery in Databases - European Conference*, 2020.

Haodong Zhang, Tao Ren, Changhu Wang, Yifan Wang, Wei Ju, Huaizhi Tang, Junyu Luo, Zimo Wang, Ziyue Qiao, Xian-Sheng Hua, and Xiao Luo. Disco: Diffusion-guided unbiased discriminative learning for unsupervised graph domain adaptation. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2026.

Xiaowen Zhang, Yuntao Du, Rongbiao Xie, and Chongjun Wang. Adversarial separation network for cross-network node classification. In *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management*, 2021.

Yizhou Zhang, Guojie Song, Lun Du, Shuwen Yang, and Yilun Jin. DANE: domain adaptive network embedding. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, 2019.

Zhen Zhang and Bingsheng He. Aggregate to adapt: Node-centric aggregation for multi-source-free graph domain adaptation. In *Proceedings of the ACM on Web Conference 2025*, 2025a.

Zhen Zhang and Bingsheng He. Towards unsupervised open-set graph domain adaptation via dual reprogramming. In *Advances in Neural Information Processing Systems*, 2025b.

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 *Proceedings of the ACM Web Conference 2024*, 2024.

Tong Zhao, Yozen Liu, Leonardo Neves, Oliver J. Woodford, Meng Jiang, and Neil Shah. Data augmentation for graph neural networks. In *Thirty-Fifth AAAI Conference on Artificial Intelligence*, 2021.

Yanqiao Zhu, Weizhi Xu, Jinghao Zhang, Yuanqi Du, Jieyu Zhang, Qiang Liu, Carl Yang, and Shu Wu. A survey on graph structure learning: Progress and opportunities. *arXiv preprint arXiv:2103.03036*, 2021.

A Details of the Empirical Analysis

To better understand how structural incompleteness affects graph domain adaptation, we conduct an empirical analysis before evaluating MIRAGE. This analysis is not intended to compare different methods, but to examine whether missing edges enlarge the performance discrepancy between source and target domains for a representative existing GDA model. We use GRADE (Wu et al., 2023) as the backbone and evaluate it on two transfer tasks, Airport (Ribeiro et al., 2017) with Brazil→USA and Citation, following the task setting used in UDAGCN (Wu et al., 2020), with ACMv9→DBLPv7. To simulate structural incompleteness, we randomly remove edges from both source and target graphs with deletion rates in $\{0\%, 20\%, 40\%, 60\%, 80\%\}$, while keeping node features and labels unchanged.

We measure the source-target performance discrepancy using the normalized source-target loss gap. Specifically, we first compute the source and target classification losses:

$$L_S = -\frac{1}{N^S}\sum_{i=1}^{N^S}\sum_{c=1}^{C} Y_{ic}^S \log \hat{Y}_{ic}^S, \qquad \hat{L}_T = -\frac{1}{N^T}\sum_{i=1}^{N^T}\sum_{c=1}^{C} Y_{ic}^T \log \hat{Y}_{ic}^T.$$

The normalized loss gap is then defined as

$$\Delta_L^{\mathrm{norm}} = \frac{\hat{L}_T - L_S}{\hat{L}_T + L_S + \epsilon},$$

where ϵ is a small constant for numerical stability. Larger values of Δ_L^{norm} indicate a larger discrepancy between source and target performance, and the normalization reduces the influence of absolute loss-scale differences across datasets and deletion rates. Target labels are used only to compute $\hat{L}_T$ for this empirical analysis and are never used for training, hyperparameter selection, edge reconstruction, or adaptation.

For each edge deletion rate, we train GRADE on the corresponding incomplete source and target graphs and report the average normalized loss gap over multiple runs. Error bars denote standard deviations. As shown in Fig. 1, the normalized loss gap increases as more edges are removed in both transfer tasks, suggesting that structural incompleteness can widen the source-target performance discrepancy of conventional GDA methods.

B MI Computation Details

Discriminator. Following GMI (Peng et al., 2020), we use bilinear discriminators to estimate the MI terms. Given two representations u and v, the discriminator computes

$$D_w(u, v) = \sigma(u^\top W v + b), \tag{8}$$

where W and b are trainable parameters, and $\sigma(\cdot)$ is the sigmoid function. For each positive pair, negative pairs are constructed by replacing one side of the pair with samples from the corresponding empirical distribution.

Intra-domain MI computation. The intra-domain terms are estimated using the Jensen–Shannon MI estimator adopted in GMI (Peng et al., 2020). For the source domain,

$$I(h_i^S; x_j^S) = -\mathrm{sp}\big(-D_w(h_i^S, x_j^S)\big) - \mathbb{E}_{x_j' \sim \tilde{\mathbb{P}}_{X^S}}\left[\mathrm{sp}\big(D_w(h_i^S, x_j')\big)\right], \tag{9}$$

where x_j' is a negative feature sampled from the empirical source feature distribution $\tilde{\mathbb{P}}_{X^S}$. The corresponding target-domain term is

$$I(h_i^T; x_j^T) = -\mathrm{sp}\big(-D_w(h_i^T, x_j^T)\big) - \mathbb{E}_{x_j' \sim \tilde{\mathbb{P}}_{X^T}}\left[\mathrm{sp}\big(D_w(h_i^T, x_j')\big)\right], \tag{10}$$

where $\mathrm{sp}(x) = \log(1 + e^x)$ denotes the softplus function.

Table 6: Statistics of the datasets used in our experiments.

Dataset	Domain	# Nodes	# Edges	# Labels
Blog	Blog1	2300	33471	6
	Blog2	2896	53836	6
Citation	ACMv9	9360	15602	5
	DBLPv7	5484	8130	5
	Citationv1	8935	15113	5
Airport	Brazil	131	1038	4
	USA	1190	13599	4
	Europe	399	5995	4
MAG	CN	25467	43683	10
	DE	17919	45528	10
	FR	12052	30915	10
	JP	12651	20747	10
	RU	3690	3539	10
	US	55392	289833	10

Cross-domain MI computation. The cross-domain MI terms are estimated using the same Jensen–Shannon principle. For the feature-level MI term, the two transfer directions are estimated as

$$\begin{aligned} I(h_i^S; c_h^T) = & -\mathrm{sp}\big(-D_w(h_i^S, c_h^T)\big) - \mathbb{E}_{c_h' \sim \tilde{\mathbb{P}}_{c_h^T}} \left[\mathrm{sp}\big(D_w(h_i^S, c_h')\big)\right], \\ I(h_i^T; c_h^S) = & -\mathrm{sp}\big(-D_w(h_i^T, c_h^S)\big) - \mathbb{E}_{c_h' \sim \tilde{\mathbb{P}}_{c_h^S}} \left[\mathrm{sp}\big(D_w(h_i^T, c_h')\big)\right]. \end{aligned} \tag{11}$$

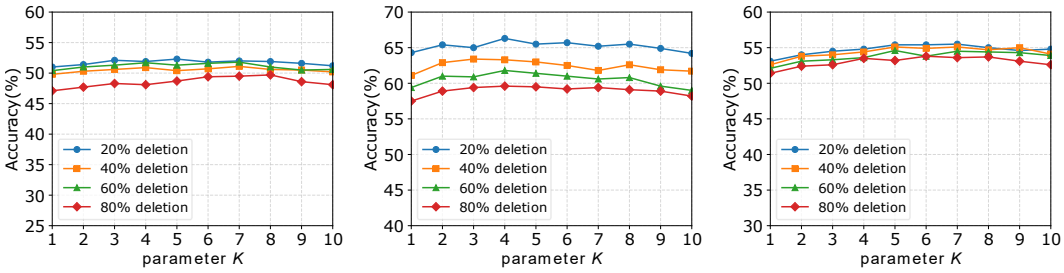
Similarly, for the structure-level MI term,

$$\begin{aligned} I(r_i^S; c_r^T) = & -\mathrm{sp}\big(-D_w(r_i^S, c_r^T)\big) - \mathbb{E}_{c_r' \sim \tilde{\mathbb{P}}_{c_r^T}} \left[\mathrm{sp}\big(D_w(r_i^S, c_r')\big)\right], \\ I(r_i^T; c_r^S) = & -\mathrm{sp}\big(-D_w(r_i^T, c_r^S)\big) - \mathbb{E}_{c_r' \sim \tilde{\mathbb{P}}_{c_r^S}} \left[\mathrm{sp}\big(D_w(r_i^T, c_r')\big)\right]. \end{aligned} \tag{12}$$

C Baseline Details

We compare MIRAGE with both structure-reconstruction baselines and state-of-the-art graph domain adaptation (GDA) methods. For non-GDA baselines, we consider two types of graph structure learning approaches. First, we combine standard graph neural networks with graph autoencoder-based structure reconstruction. Specifically, GCN (Kipf & Welling, 2017) performs node classification via first-order neighborhood aggregation, and GAT (Velickovic et al., 2018) introduces attention mechanisms to adaptively weight neighboring nodes. To handle incomplete graphs, both models are combined with VGAE (Kipf & Welling, 2016) to reconstruct missing edges before node classification. Second, we include task-driven graph structure learning methods. GRCN (Yu et al., 2020) revises graph structures by predicting missing edges and refining edge weights for node classification, while IDGL (Chen et al., 2020) jointly and iteratively learns graph structures and node embeddings.

For GDA baselines, we consider representative methods that address cross-domain transfer from different perspectives. UDAGCN (Wu et al., 2020) integrates adversarial training and dual GCN encoders to align node representations across domains. ACDNE (Shen et al., 2020) embeds nodes from different networks into a shared latent space through adversarial learning, while ASN (Zhang et al., 2021) separates domain-specific and domain-invariant features for cross-network classification. GRADE (Wu et al., 2023) models graph distribution shift via graph subtree discrepancy, and SpecReg (You et al., 2023) regularizes spectral graph properties while minimizing distribution discrepancy. PairAlign (Liu et al., 2024c) mitigates structural and label distribution shifts through pairwise alignment, StruRW (Liu et al., 2023) addresses structural shift by reweighting edges based on conditional structure distributions, and HGDA (Fang et al., 2025b) models homophily-aware representations across graph domains.



(a) Blog (b) Citation (c) Airport

Figure 6: Sensitivity analysis of the candidate size K under different edge deletion rates. Each curve corresponds to one deletion rate, and the x-axis denotes $K \in \{1, 2, \ldots, 10\}$.

D Dataset Details

We provide additional details of the datasets and evaluation protocol used in our experiments. We evaluate MIRAGE on three benchmark graph domain adaptation datasets, namely Blog (Shen et al., 2020), Citation (Zhang et al., 2021), and Airport (Ribeiro et al., 2017). Blog is a social network dataset consisting of two domains, Blog1 and Blog2, where nodes correspond to users or blog entities and edges represent social links or interactions. Citation is a citation network dataset containing three domains, ACMv9, DBLPv7, and Citationv1, where nodes represent papers and edges denote citation or citation-related relations. Airport is an air-traffic network dataset with three domains, Brazil, USA, and Europe, where nodes represent airports and edges represent air-traffic connections. For a concise main evaluation, we select one representative transfer task from each dataset: Blog1→Blog2 for Blog, ACMv9→DBLPv7 for Citation, and Brazil→USA for Airport.

We additionally evaluate MIRAGE on MAG (Fang et al., 2026b), for which we use a 10-class subset of OGBN-MAG (Wang et al., 2020; Hu et al., 2020). The MAG data are divided into six domains: CN, US, DE, FR, JP, and RU. Since the dataset provides both PAP and PP graph views, we use the PP view in our experiments. Detailed MAG experiments are reported in Appendix K.

The statistics of these datasets are summarized in Table 6. For all datasets, each domain is treated as an individual graph, and domain adaptation is performed between different domains. Following the standard unsupervised graph domain adaptation setting, we do not split nodes within each graph into separate training, validation, and test subsets. Instead, all labeled source nodes are used for source-supervised training, and the unlabeled target graph is used for unsupervised adaptation. This protocol follows common ractice in cross-network node classification and graph domain adaptation, where labeled source nodes and unlabeled target nodes are jointly used during model training, and performance is evaluated on the target graph (Zhang et al., 2021; Wu et al., 2020). To evaluate graph domain adaptation under structural incompleteness, we randomly remove edges from both source and target graphs with deletion rates in $\{0\%, 20\%, 40\%, 60\%, 80\%\}$. Node features and labels remain unchanged, and all methods are trained and evaluated on the same incomplete graphs at each deletion rate.

E Algorithm Description

Algorithm 1 summarizes the training procedure of MIRAGE, including candidate construction, MI-guided edge reconstruction, and source-supervised node classification.

F Hyperparameter Analysis

We analyze the sensitivity of MIRAGE to key hyperparameters, including the candidate size K, the mutual information coefficients λ_{intra}, λ_{cf}, λ_{cs}, and the reconstruction coefficient α. Unless otherwise specified, we vary one hyperparameter at a time while keeping the others fixed to their default values.

Algorithm 1 Training Procedure of MIRAGE

Require: Incomplete source graph $\hat{G}^S = (V^S, \hat{E}^S, \hat{A}^S, X^S, Y^S)$, incomplete target graph $\hat{G}^T = (V^T, \hat{E}^T, \hat{A}^T, X^T)$, candidate size K, training epochs T, and hyperparameters $\alpha, \lambda_{\text{intra}}, \lambda_{\text{cf}}, \lambda_{\text{cs}}$
Ensure: Target predictions $\hat{Y}^T$

```
Initialize encoder f_θ, classifier g_φ, and MI estimators
Construct candidate sets Ω^S = {(i,j) | j ∈ TopK_x^S(i)} ∪ Ê^S and Ω^T = {(i,j) | j ∈ TopK_x^T(i)} ∪ Ê^T, and symmetrize them
Initialize candidate-edge logits {z_ij^S}_{(i,j)∈Ω^S} and {z_ij^T}_{(i,j)∈Ω^T}
for epoch = 1 to T do
    Compute node embeddings H^S = f_θ(X^S, Â^S) and H^T = f_θ(X^T, Â^T)
    Compute edge probabilities π_ij^S = σ(z_ij^S) and π_ij^T = σ(z_ij^T)
    Compute dynamic thresholds τ_ij^S = σ((h_i^S)^⊤ h_j^S) and τ_ij^T = σ((h_i^T)^⊤ h_j^T)
    Construct Ã^S using Eq. 1 and construct Ã^T analogously
    Compute L_edge^S using Eq. 2 and compute L_edge^T analogously
    Compute intra-domain mutual information I_intra using Eq. 3
    Compute feature-level summaries c_h^S, c_h^T and reconstructed-neighborhood representations r_i^S, r_i^T
    Compute structure-level summaries c_r^S, c_r^T and cross-domain mutual information I_cross
    Compute reconstruction objective L_ER using Eq. 5
    Form reconstructed source graph G̃^S = (V^S, Ẽ^S, Ã^S, X^S, Y^S)
    Predict source labels Ŷ^S = g_φ(G̃^S)
    Compute L_cls using Eq. 6
    Compute overall objective L = L_cls + αL_ER
    Update θ, φ, candidate-edge logits, and MI-estimator parameters by minimizing L
end for
Form the final reconstructed target graph G̃^T = (V^T, Ẽ^T, Ã^T, X^T)
Predict target labels Ŷ^T = g_φ(G̃^T)
return Ŷ^T
```

F.1 Effect of Candidate Size K

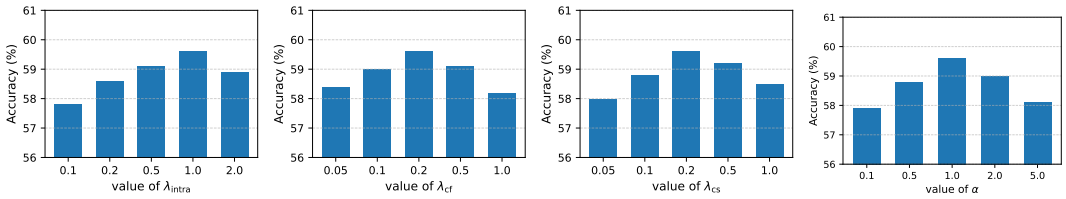
The candidate size K controls the number of feature-similar node pairs considered for edge selection. A small K may miss useful candidate edges, while a large K may introduce noisy candidates and increase the reconstruction cost. We evaluate $K \in \{1, 2, \ldots, 10\}$ on Airport (Brazil→USA), Blog (Blog1→Blog2), and Citation (ACMv9→DBLPv7) under four edge deletion rates: 20%, 40%, 60%, and 80%. Fig. 6 reports the sensitivity of MIRAGE to K on the three datasets.

As shown in Fig. 6, MIRAGE generally benefits from a moderate candidate size. When K is too small, the candidate set may fail to cover useful missing edges. When K becomes too large, additional noisy candidates may be introduced into edge selection. Overall, MIRAGE remains stable across a reasonable range of K, indicating that its performance is not overly sensitive to the exact candidate size.

F.2 Effect of Objective Coefficients

We further analyze the sensitivity of MIRAGE to the coefficients in the training objective on Citation (ACMv9→DBLPv7) under the 80% edge deletion rate. Specifically, λ_{intra} controls the strength of the intra-domain mutual information term, λ_{cf} and λ_{cs} control the strengths of the feature-level and structure-level cross-domain mutual information terms, respectively, and α balances the source-supervised classification loss and the MI-guided reconstruction objective in Eq. 7.

We conduct one-at-a-time sensitivity analysis by varying one coefficient while fixing the others to their default values. For λ_{intra}, we consider $\{0.1, 0.2, 0.5, 1.0, 2.0\}$. For λ_{cf} and λ_{cs}, we consider $\{0.05, 0.1, 0.2, 0.5, 1.0\}$. For α, we consider $\{0.1, 0.5, 1.0, 2.0, 5.0\}$. The results are shown in Fig. 7.



(a) λ_{intra} (b) λ_{cf} (c) λ_{cs} (d) α

Figure 7: Sensitivity analysis of the objective coefficients on Citation (ACMv9→DBLPv7) under the 80% edge deletion rate. Each coefficient is varied independently while the others are fixed to their default values.

Table 7: Dataset-level hyperparameter settings used in MIRAGE. The reported hyperparameters are used for all deletion rates $\{0\%, 20\%, 40\%, 60\%, 80\%\}$.

Dataset	K	λ_{cf}	λ_{cs}	Epochs
Airport	8	0.05	0.05	200
Blog	8	0.2	0.2	300
Citation	4	0.2	0.2	20

As shown in Fig. 7, MIRAGE achieves stable performance within a moderate range of coefficient values. Very small coefficients weaken the MI guidance for edge selection or the contribution of reconstruction, while overly large coefficients may dominate the training objective and reduce the effect of source-supervised classification. Overall, the results suggest that MIRAGE is not overly sensitive to the exact choice of these coefficients, provided that they are set within a reasonable range.

G Implementation Details

The experiments are implemented in PyTorch using an Intel(R) Xeon(R) Silver 4210R CPU @ 2.40GHz and an NVIDIA GeForce RTX A5000 24GB GPU. We train MIRAGE with the Adam optimizer, using a learning rate of 2×10^{-3}, a weight decay of 1×10^{-4}, and a hidden dimension of 512. All results are averaged over three independent runs. Unless otherwise specified, we set the trade-off coefficient α to 1.0 and the intra-domain mutual information coefficient λ_{intra} to 1.0. Other dataset-specific hyperparameters, including the candidate size K, λ_{cf}, λ_{cs}, and the number of training epochs, are summarized in Table 7.

Table 8: Efficiency-performance comparison under severe structural incompleteness. Training time and peak GPU memory usage are normalized with respect to UDAGCN on each dataset. Accuracy is reported under the 80% edge deletion rate. Lower normalized cost is better, while higher accuracy is better.

Dataset	Method	Training Time (w.r.t. UDAGCN)	Peak GPU Memory (w.r.t. UDAGCN)	Accuracy (%)
Blog (Blog1→Blog2)	UDAGCN	1.000	1.000	16.7
	GRADE	1.142	1.093	22.0
	SpecReg	1.209	1.176	17.5
	PairAlign	1.118	1.153	21.9
	MIRAGE	3.459	2.163	44.8
Citation (ACMv9→DBLPv7)	UDAGCN	1.000	1.000	49.2
	GRADE	1.196	1.102	50.7
	SpecReg	1.231	1.145	51.2
	PairAlign	1.219	1.342	49.2
	MIRAGE	3.372	2.524	59.6
Airport (Brazil→USA)	UDAGCN	1.000	1.000	24.9
	GRADE	1.113	1.019	36.1
	SpecReg	1.008	1.012	28.2
	PairAlign	1.107	1.115	33.5
	MIRAGE	3.286	2.309	54.9

Table 9: Mean and standard deviation of representative methods under the 80% edge deletion rate.

Dataset	Method	Acc	Macro-F1
Blog (Blog1→Blog2)	HGDA	35.6%±1.21%	32.0%±1.43%
Blog (Blog1→Blog2)	MIRAGE	49.4%±0.68%	44.2%±0.87%
Citation (ACMv9→DBLPv7)	ASN	52.1%±0.73%	48.7%±0.91%
Citation (ACMv9→DBLPv7)	MIRAGE	59.6%±0.54%	53.6%±0.69%
Airport (Brazil→USA)	HGDA	38.4%±1.21%	37.7%±1.43%
Airport (Brazil→USA)	MIRAGE	53.9%±0.83%	48.1%±1.07%

H Model Efficiency Evaluation

We further evaluate the efficiency-performance trade-off of MIRAGE under severe structural incompleteness. Following common practice, we report the normalized training time and peak GPU memory usage of each method with respect to UDAGCN on the same dataset. Specifically, UDAGCN is assigned a normalized cost of 1.0, and larger values indicate higher computational cost. The comparison is conducted under the 80% edge deletion rate, where structural information is highly incomplete and graph reconstruction becomes particularly important.

Table 8 summarizes the efficiency-performance comparison. MIRAGE requires more training time than conventional GDA baselines because it jointly performs edge reconstruction and domain adaptation with mutual information-guided objectives. Nevertheless, its peak GPU memory usage remains within a moderate range compared with the baselines. More importantly, MIRAGE achieves substantially higher accuracy under severe edge deletion across all datasets. These results indicate that the additional training cost mainly arises from the transfer-aware reconstruction process, while providing a favorable trade-off between computational cost and adaptation performance on incomplete graphs.

I Stability Analysis

We report the standard deviations of MIRAGE and representative strong baselines over three runs. The results in Table 9 show that MIRAGE maintains stable performance across different edge-deletion and model initialization seeds.

J Additional Transfer Tasks

To further examine whether the performance of MIRAGE is consistent across different source-target pairs, we report additional transfer-task results on Citation and Airport. While the main experiments use one representative task from each dataset for a concise comparison, this section evaluates two additional Citation tasks, DBLPv7→ACMv9 and Citationv1→ACMv9, and two additional Airport tasks, USA→Europe and Europe→USA. These tasks involve different transfer directions and graph scales, providing a broader evaluation of MIRAGE under EIGDA.

We follow the same experimental protocol as in the main experiments. Source labels are used for training, target labels are used only for final evaluation, and edges are randomly removed from both source and target graphs. To focus on challenging EIGDA scenarios, we report results under the 60% and 80% edge deletion rates. For clarity, we compare MIRAGE with representative structure-reconstruction and GDA baselines.

As shown in Tables 10–11, MIRAGE maintains strong performance across additional transfer directions under severe structural incompleteness. These results indicate that the benefit of transfer-aware edge reconstruction is not limited to the representative source-target pairs reported in the main experiments.

Table 10: Additional transfer-task performance on Citation under severe edge deletion rates. Best performance is highlighted in bold.

Transfer Task	Method	Delete 60%		Delete 80%	
		Acc	Macro-F1	Acc	Macro-F1
DBLPv7→ACMv9	GCN+VGAE	0.437	0.403	0.416	0.358
	GRCN	0.442	0.387	0.429	0.354
	UDAGCN	0.450	0.398	0.427	0.375
	GRADE	0.452	0.406	0.423	0.386
	PairAlign	0.483	0.417	0.456	0.388
	MIRAGE	**0.539**	**0.465**	**0.501**	**0.435**
Citationv1→ACMv9	GCN+VGAE	0.439	0.404	0.407	0.381
	GRCN	0.427	0.389	0.402	0.377
	UDAGCN	0.435	0.414	0.417	0.390
	GRADE	0.446	0.423	0.424	0.405
	PairAlign	0.481	0.458	0.442	0.411
	MIRAGE	**0.569**	**0.540**	**0.530**	**0.475**

Table 11: Additional transfer-task performance on Airport under severe edge deletion rates.

Transfer Task	Method	Delete 60%		Delete 80%	
		Acc	Macro-F1	Acc	Macro-F1
USA→Europe	GCN+VGAE	0.328	0.290	0.304	0.276
	GRCN	0.315	0.287	0.290	0.274
	UDAGCN	0.248	0.099	0.248	0.099
	GRADE	0.364	0.340	0.348	0.327
	PairAlign	0.412	0.391	0.385	0.364
	MIRAGE	**0.474**	**0.421**	**0.463**	**0.404**
Europe→USA	GCN+VGAE	0.351	0.322	0.334	0.307
	GRCN	0.368	0.349	0.335	0.314
	UDAGCN	0.249	0.099	0.249	0.099
	GRADE	0.439	0.408	0.417	0.381
	PairAlign	0.442	0.426	0.420	0.404
	MIRAGE	**0.528**	**0.482**	**0.514**	**0.482**

K Additional Experiments on MAG

Dataset. We further evaluate MIRAGE on MAG, a large-scale graph domain adaptation dataset introduced in prior multi-view graph domain adaptation work (Fang et al., 2026b). MAG is constructed as a subset of OGBN-MAG (Wang et al., 2020; Hu et al., 2020), where papers are partitioned into six country-specific subgraphs, including CN, US, DE, FR, JP, and RU. Each country subgraph corresponds to one domain, and the ten most frequent classes are selected for node classification. The original 10-class MAG dataset provides two graph views for each domain, namely the PAP view and the PP view. Since MIRAGE focuses on single-view graph domain adaptation under structural incompleteness, we conduct experiments on the PP view.

The full 20-class MAG setting exceeds the memory budget of our current single-GPU implementation. Therefore, we use the filtered 10-class MAG setting as an additional large-scale evaluation. Following the EIGDA protocol, source labels are used for training, target labels are used only for final evaluation, and edges are randomly removed from both source and target graphs with deletion rates in $\{0\%, 20\%, 40\%, 60\%, 80\%\}$. We evaluate two transfer tasks, CN→DE and JP→DE, and report the results in Tables 12 and 13, where the best performance is highlighted in bold.

As shown in Tables 12 and 13, MIRAGE achieves the best accuracy across all deletion rates on both MAG transfer tasks. On CN→DE, MIRAGE consistently outperforms the strongest baseline in both

Table 12: Node classification performance on MAG(CN→DE) under different edge deletion rates.

Method	Delete 0%		Delete 20%		Delete 40%		Delete 60%		Delete 80%	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
UDAGCN	0.381	0.351	0.390	0.362	0.396	0.370	0.404	0.378	0.409	0.382
GRADE	0.398	0.360	0.384	0.346	0.386	0.346	0.388	0.354	0.370	0.326
PairAlign	0.382	0.358	0.392	0.371	0.405	0.385	0.406	0.383	0.420	0.401
MIRAGE	**0.438**	**0.398**	**0.432**	**0.407**	**0.434**	**0.415**	**0.438**	**0.398**	**0.444**	**0.422**

Table 13: Node classification performance on MAG(JP→DE) under different edge deletion rates.

Method	Delete 0%		Delete 20%		Delete 40%		Delete 60%		Delete 80%	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
UDAGCN	0.413	0.384	0.426	0.400	0.430	0.403	0.437	0.409	0.438	0.412
GRADE	0.434	0.406	0.429	0.404	0.435	0.397	0.424	0.390	0.412	0.378
PairAlign	0.408	0.387	0.423	0.399	0.433	0.408	0.442	0.415	0.449	0.423
MIRAGE	**0.456**	**0.445**	**0.457**	**0.417**	**0.460**	**0.424**	**0.496**	**0.436**	**0.488**	**0.434**

Acc and Macro-F1, and obtains the best performance at the 80% deletion rate. On JP→DE, MIRAGE also achieves the highest Acc under all deletion rates and obtains the best Macro-F1 in most settings. These results indicate that transfer-aware reconstruction remains effective on larger graphs, even when the graph topology is substantially perturbed.

Interestingly, MAG exhibits a dataset-specific behavior that differs from the main benchmark datasets. Increasing the edge deletion rate does not lead to a clear monotonic degradation on MAG. Some methods, such as UDAGCN and PairAlign, even improve as the graph becomes sparser, while GRADE shows mild fluctuations or a slight decrease. This suggests that edge deletion on MAG may not simply remove useful structural information.

To better understand this behavior, we compute the edge homophily of the PP view in MAG. As shown in Table 14, the overall PP edge homophily is 0.642, indicating that a non-negligible fraction of PP edges connect papers from different classes. Therefore, random edge deletion may partly act as implicit graph sparsification on MAG: it reduces message passing over both redundant and cross-class connections, which can alleviate noisy propagation or over-smoothing on this large graph. This explains why increasing deletion rates do not necessarily cause the same performance degradation observed on the main datasets.

For this reason, the MAG results should be interpreted as an additional large-scale evaluation under a relatively noisy graph view, rather than as direct evidence that higher deletion rates always make EIGDA harder. Despite this dataset-specific trend, MIRAGE maintains strong performance across deletion rates, suggesting that transfer-aware reconstruction can still provide benefits in large-scale EIGDA settings.

L Complexity Analysis

We analyze the computational complexity of MIRAGE. Let N^S and N^T denote the numbers of source and target nodes, and let $|\hat{\mathcal{E}}^S|$ and $|\hat{\mathcal{E}}^T|$ denote the numbers of observed edges. We define $N = N^S + N^T$ and $\hat{E} = |\hat{\mathcal{E}}^S| + |\hat{\mathcal{E}}^T|$. Let d be the hidden dimension, m be the input feature dimension, K be the number of top-K candidate neighbors, and L be the number of GNN layers. Since the candidate set combines top-K feature-neighbor pairs with observed edges, the total number of candidate pairs is $M = |\Omega^S| + |\Omega^T| = O(KN + \hat{E})$.

Constructing candidate pairs by exact feature similarity requires $O(((N^S)^2 + (N^T)^2)m)$, which is a one-time preprocessing cost. In each training epoch, encoding the observed graphs with an L-layer sparse GNN takes $O(L(\hat{E}d + Nd^2))$. Adaptive edge selection over candidate pairs, including edge probabilities, dynamic thresholds, and reconstructed adjacency entries, takes $O(Md)$. The mutual

Table 14: PP edge homophily statistics of MAG domains. Edge homophily is computed as the fraction of PP edges connecting nodes with the same class label.

Domain	#Nodes	#PP Edges	#Same-label Edges	Edge Homophily
CN	25467	43683	25769	0.590
DE	17919	45528	28239	0.620
FR	12052	30915	20489	0.663
JP	12651	20747	13487	0.650
RU	3690	3539	2401	0.678
US	55392	289833	188297	0.650
Overall	–	434245	278682	0.642

information terms require $O(\hat{E}d)$ for intra-domain mutual information and $O(Md + Nd)$ for cross-domain mutual information. The node classifier on reconstructed graphs takes $O(L(Md + Nd^2))$, since reconstructed edges are selected from the candidate set. Therefore, the per-epoch complexity is

$$O\big(L(\hat{E}d + Nd^2) + Md + \hat{E}d + Nd + L(Md + Nd^2)\big),$$

which can be simplified as $O(L(Md + Nd^2))$. With T training epochs, the total time complexity is

$$O\big(((N^S)^2 + (N^T)^2)m\big) + O\big(TL(Md + Nd^2)\big).$$

Although exact top-K candidate construction introduces a quadratic preprocessing cost, this step is performed only once before training. During training, MIRAGE operates on the candidate edge set Ω rather than all $O(N^2)$ node pairs. This is comparable to decoupled pipelines that first perform graph recovery and then apply GDA, where graph recovery often requires pairwise edge scoring or dense adjacency reconstruction. Therefore, the additional cost of MIRAGE mainly comes from making reconstruction transfer-aware, while the per-epoch computation remains candidate-based.