MORE DOMAINS, NO GAINS? FLO: FAST–SLOW RECIPROCAL LEARNING FOR MULTI-DOMAIN GRAPH PRE-TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Inspired by the great success of foundation models in the development of artificial general intelligence, Graph Foundation Models (GFMs) have recently attracted significant attention and have the potential to advance graph intelligence. Exhibiting such intelligence requires effectively learning knowledge from large-scale, multi-domain graph data during pre-training. Increasing domain diversity during pre-training is expected to yield richer and more generalizable knowledge. Most existing methods pursue this goal by designing domain alignment strategies that help a single graph model learn from multiple domains. However, we observe that existing methods struggle to benefit from increasing domain diversity. Through empirical and theoretical analysis, we reveal two key limitations. Firstly, pre-training in most GFMs is confined to a single message-passing paradigm. Since propagation flattens relations among neighboring nodes, the graph model can hardly perceive the knowledge beyond propagation, namely the attribute-predictable component of these relations. Secondly, domain alignment strategies often rely on randomly initialized parameters that are jointly updated with the graph model using immediate loss feedback, leaving early knowledge acquisition vulnerable to cross-domain interference. To address these issues, we propose FLO, a Fast–sLow reciprOcal approach for multi-domain graph pre-training. Specifically, a propagation-free attribute branch predicts the graph model's relations without message passing and returns these predictions to refine the model. We further stabilize knowledge acquisition by pairing a fast process that adapts to the current domain with a slow process that integrates knowledge across domains. On 12 datasets, FLO benefits from increasing domain diversity and achieves the best average performance in cross-domain and cross-task transfer.

1 INTRODUCTION

In recent years, foundation models (Bommasani et al., 2021) have achieved great success in advancing the development of artificial general intelligence, fundamentally reshaping various fields of machine learning. Inspired by this progress, Graph Foundation Models (GFMs) have emerged as a research frontier, attracting significant attention for their potential to advance graph intelligence (Mao et al., 2024; Liu et al., 2025; Li et al., 2024; He et al., 2025; Zhu et al., 2024; Zhao et al., 2025; Sun et al., 2025a; Xia & Huang, 2026; Sun et al., 2026). By modeling complex relational structures inherent in real-world systems, GFMs hold the promise of establishing a unified paradigm for graph machine learning (Liu et al., 2024; Wang et al., 2025b; Bevilacqua et al., 2025). However, exhibiting such highly generalizable intelligence imposes a critical challenge: these models must be capable of effectively extracting and learning universal knowledge from large-scale, multi-domain graph data during pre-training (Hu et al., 2020; Qiu et al., 2020; You et al., 2020).

Increasing domain diversity during pre-training is expected to yield richer and more generalizable knowledge. To achieve this, most existing methods[1] focus on carefully designing domain alignment strategies to help a single graph model reconcile the heterogeneity across diverse domains. Typically, these methods rely on unifying feature spaces or introducing learnable alignment tokens to

[1] More related work is discussed in Appendix C.

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.

bridge semantic and structural gaps across domains. For the former, these approaches project heterogeneous node attributes into a shared, domain-agnostic space (Shan et al., 2026b; Eliasof et al., 2026; Shen et al., 2025; Sun et al., 2025b; Shan et al., 2026a). For instance, FUG (Zhao et al., 2024b) designs a dimension encoder to capture transferable distributions among attributes, whereas TIG (Zhao et al., 2026) proposes transfer-invariant metadata, namely feature-dimension relations shared across graphs, and mines it to guide the pre-training process. For the latter, these methods explicitly reconcile complex cross-domain differences by augmenting the Graph Neural Network (GNN) encoder with auxiliary learnable parameters (Zhao et al., 2024a; Yuan et al., 2025; Zhuo & Luo, 2026). For example, MDGPT (Yu et al., 2024) employs learnable domain tokens as semantic anchors across various datasets, while SAMGPT (Yu et al., 2025) introduces structure tokens to align distinct topological characteristics. With these strategies, a graph model is expected to gain more knowledge as more source domains are added.

However, we observe a counter-intuitive phenomenon: as shown in Figure 1(a), existing methods struggle to benefit from increasing domain diversity. This suggests that domain alignment is a prerequisite rather than a guarantee of knowledge acquisition. Through detailed empirical and theoretical analysis, we identify two limitations that remain after domain alignment. **Firstly,** after domain alignment, the graph model still forms node relations mainly through message passing (Wang et al., 2026; Yang et al., 2026). Propagation smooths the representations of neighboring nodes (Li et al., 2018; Cai & Wang, 2020) and flattens their relations. The model can thus hardly perceive the knowledge beyond propagation, namely the attribute-predictable component of these relations. **Secondly,** the aforementioned alignment components (e.g., domain or structure tokens) often rely on randomly initialized parameters that are jointly updated with the graph model using immediate loss feedback. Given the severe data heterogeneity, this immediate joint optimization paradigm leaves early knowledge acquisition vulnerable to cross-domain interference, ultimately destabilizing the pre-training process (Figure 1(b)–(c)).

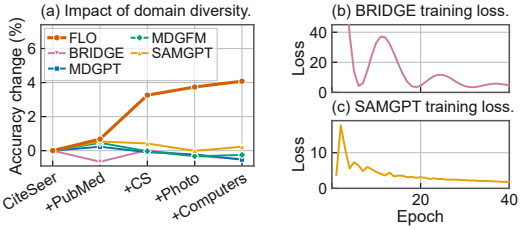


Figure 1: Motivation experiments with Cora as the target graph. (a) Accuracy change as source domains are added. (b)–(c) Pre-training loss of two baselines.

To address the above limitations, we propose FLO, a multi-domain graph pre-training approach with Fast–sLow reciprOcal learning. Specifically, we predict the relations of the graph model without message passing and return the predictions as feedback, so that the graph model acquires knowledge beyond propagation. We further pair a fast process that adapts to the current domain with a slow process that integrates knowledge across domains (McClelland et al., 1995; Polyak & Juditsky, 1992; Zhang et al., 2019; Cha et al., 2021; Arani et al., 2022; Busbridge et al., 2023; Morales-Brotons et al., 2024), and take the exchanged relations from the slow process. This protects early knowledge acquisition from cross-domain interference. After pre-training, we deploy only the graph model on unseen graphs, so FLO adds no parameters at inference. As shown in Figure 1(a), the accuracy of FLO increases steadily as source domains are added. Our contributions are summarized as follows:

- We find that existing methods struggle to benefit from increasing domain diversity. We reveal two limitations behind this: message passing alone can hardly perceive knowledge beyond propagation, and randomly initialized alignment parameters updated by immediate loss feedback leave early knowledge acquisition vulnerable to cross-domain interference.
- We propose FLO, which refines the node relations induced by message passing with feedback grounded in node attributes alone and takes this feedback from a slow process that integrates knowledge across domains. We prove that the returned relations lead the graph model toward the knowledge beyond propagation without domain-specific parameters, while the slow process filters cross-domain interference from the feedback.
- Experiments on 12 datasets show that FLO achieves the best average performance in cross-domain and cross-task transfer and benefits from increasing domain diversity.

2 PRELIMINARIES

Unless otherwise specified, we use calligraphic letters for sets, bold uppercase letters for matrices, bold lowercase letters for vectors, and blackboard bold letters for spaces.

Graph data. A graph $G = (\mathcal{V}, \mathcal{T})$ has node set $\mathcal{V}$ with n nodes, edge set $\mathcal{T}$, adjacency matrix $\mathbf{A}$, and feature matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ with rows $\mathbf{x}_i$. In this paper, we consider undirected graphs. Let $\widetilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$, $\widetilde{\mathbf{D}}_{ii} = \sum_j \widetilde{\mathbf{A}}_{ij}$, and $\widehat{\mathbf{A}} = \widetilde{\mathbf{D}}^{-1/2}\widetilde{\mathbf{A}}\widetilde{\mathbf{D}}^{-1/2}$.

Graph neural networks. A Graph Convolutional Network (GCN) layer (Kipf & Welling, 2017) computes:

$$\mathbf{H}^{(\ell+1)} = \sigma\big(\widehat{\mathbf{A}}\mathbf{H}^{(\ell)}\mathbf{W}^{(\ell)}\big), \tag{1}$$

where $\mathbf{W}^{(\ell)}$ is a learnable weight matrix and σ is a nonlinear activation. It aggregates the representations of neighboring nodes through message passing, and this propagation smooths them (Li et al., 2018). The Dirichlet energy $\mathcal{E}(\mathbf{Y}) = \mathrm{tr}(\mathbf{Y}^\top(\mathbf{I} - \widehat{\mathbf{A}})\mathbf{Y})$ measures how much a signal $\mathbf{Y}$ varies across edges, and propagation lowers it (Section 3.4).

Multi-domain pre-training. We are given S unlabeled source graphs $\{G_s\}_{s=1}^{S}$ with adjacency matrices $\mathbf{A}_s$ and raw feature matrices $\mathbf{X}_s^{\mathrm{raw}}$, whose feature dimensions and feature semantics differ across graphs. We treat each graph dataset as a domain. The goal is to pre-train a graph encoder on the source graphs without any labels, such that it produces node representations for a target graph from an unseen domain without being updated on it. To align the feature dimensions of different graphs, we map each raw feature matrix to a common dimension d:

$$\mathbf{X}_s = \mathrm{RowNorm}\Big(\mathrm{Pad}_d(\widetilde{\mathbf{X}}_s\mathbf{V}_s)\Big), \qquad \widetilde{\mathbf{X}}_s = \mathrm{RowNorm}(\mathbf{X}_s^{\mathrm{raw}}), \tag{2}$$

where $\mathbf{V}_s$ contains up to d leading right singular vectors of $\widetilde{\mathbf{X}}_s$, Pad_d appends zeros when needed, and $\mathrm{RowNorm}$ normalizes nonzero rows to unit length. The target graph undergoes the same preprocessing, which uses no labels.

3 METHOD

As illustrated in Figure 2, FLO addresses two problems that remain after domain alignment: what relational knowledge a shared graph encoder acquires (Section 3.1), and how this knowledge stays stable as source domains alternate (Section 3.2). We then present the training objective and transfer procedure in Section 3.3, followed by the theoretical analysis in Section 3.4.

3.1 PROPAGATION–ATTRIBUTE RELATION REFINEMENT

Section 3.4 shows how propagation can flatten relations among nearby nodes. To address this limitation, we design Propagation–Attribute Relation Refinement (PARR), which pairs the graph encoder with a propagation-free attribute branch. The key is the information asymmetry between the two branches rather than the second branch itself: the attribute branch is deliberately denied access to propagation. This constraint makes its optimal relation prediction the attribute-predictable component $\overline{P}$ (Definition 3.1), turning reciprocal relation learning into a conditional prediction-and-refinement process rather than ordinary agreement between peer encoders (Zhang et al., 2018; Feng et al., 2022).

Specifically, the graph branch uses a two-layer GCN as the graph encoder f_G. The attribute branch f_M uses the identity matrix in place of $\widehat{\mathbf{A}}$ and is implemented as a two-layer MLP (Zhang et al., 2022; Tian et al., 2023; Zhou et al., 2025) with the same layer widths. The two branches have separate parameters θ_G and θ_M, each shared across source graphs. For source graph s, we sample two independent elementwise masks $\mathbf{M}_G, \mathbf{M}_M \in \{0,1\}^{n_s \times d}$ with keep probability p and compute:

$$\mathbf{Z}_G = f_G(\mathbf{X}_s \odot \mathbf{M}_G, \mathbf{A}_s; \theta_G), \qquad \mathbf{Z}_M = f_M(\mathbf{X}_s \odot \mathbf{M}_M; \theta_M). \tag{3}$$

Inspired by masked graph autoencoders (Hou et al., 2022), each branch $a \in \{G, M\}$ recovers the direction of the unmasked features through a linear decoder D_a:

$$\mathcal{L}_{\mathrm{rec}}^{a} = \frac{\gamma}{n_s}\sum_{i=1}^{n_s} \left\|\mathrm{RowNorm}(D_a(\mathbf{z}_i^a)) - \mathbf{x}_i\right\|_2^2, \tag{4}$$

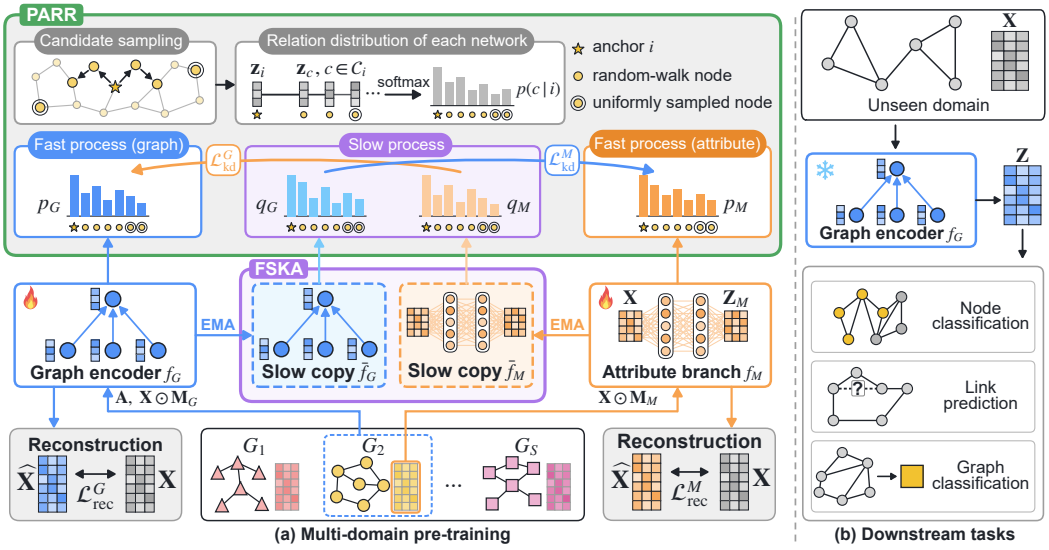


Figure 2: Overview of FLO. (a) Multi-domain pre-training: source graphs are visited in turn, and the graph branch f_G and the attribute branch f_M each reconstruct node features. In Fast–Slow Knowledge Acquisition (FSKA), each branch keeps a slow copy updated by an Exponential Moving Average (EMA) after each step. In Propagation–Attribute Relation Refinement (PARR), each branch learns the relation distribution of the other branch's slow copy over the same candidates. (b) Downstream tasks: only the frozen graph encoder f_G is applied to graphs from unseen domains.

where $\mathbf{z}_i^a$ and $\mathbf{x}_i$ are the i-th rows of $\mathbf{Z}_a$ and $\mathbf{X}_s$, and $\gamma > 0$ is a scaling coefficient. Since every nonzero row of $\mathbf{X}_s$ has unit norm by Eq. (2), the decoder predicts the direction of each node feature. Reconstruction keeps each representation tied to node content while the branch learns relations.

The two branches need not share embedding coordinates, so they exchange relation distributions instead of embeddings. At each step, we sample a set $\mathcal{I}$ of anchor nodes and, for every anchor i, a candidate set $\mathcal{C}_i$ shared by both branches and their slow copies (Section 3.2). The candidates are the anchor itself, the nodes visited by short random walks from i, and nodes sampled uniformly from the graph (Appendix B). Let $\mathbf{Z}$ be node representations with rows $\mathbf{z}_i$. Following knowledge distillation (Hinton et al., 2015; Park et al., 2019), in particular methods that match distributions of pairwise similarities (Yang et al., 2020; Fang et al., 2021; Zheng et al., 2021), we define the relation distribution of anchor i over its candidates as:

$$p_{\mathbf{Z}}(c \mid i) = \frac{\exp(\cos(\mathbf{z}_i, \mathbf{z}_c)/\tau)}{\sum_{c' \in \mathcal{C}_i} \exp(\cos(\mathbf{z}_i, \mathbf{z}_{c'})/\tau)}, \qquad c \in \mathcal{C}_i, \tag{5}$$

where $\tau > 0$ is a temperature. The relation distribution describes how similar the anchor is to each of its candidates. Relation distributions are unchanged by orthogonal transformations of the representations and positive rescaling of their rows (Appendix A.9).

Let p_G and p_M denote the relation distributions of $\mathbf{Z}_G$ and $\mathbf{Z}_M$, and let q_G and q_M denote the relation targets supplied by FSKA (Section 3.2). The two directions of reciprocal relation learning are then:

$$\mathcal{L}_{\mathrm{kd}}^{M} = \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} \mathrm{KL}\big(q_G(\cdot \mid i) \,\|\, p_M(\cdot \mid i)\big), \qquad \mathcal{L}_{\mathrm{kd}}^{G} = \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} \mathrm{KL}\big(q_M(\cdot \mid i) \,\|\, p_G(\cdot \mid i)\big), \tag{6}$$

where KL denotes the Kullback–Leibler divergence. Through $\mathcal{L}_{\mathrm{kd}}^{M}$, the attribute branch learns the relations formed by propagation from the attributes of the anchor and its candidates alone. Through $\mathcal{L}_{\mathrm{kd}}^{G}$, its predictions are returned to the graph encoder as relation targets. This exchange corrects the relations of the graph encoder toward those predictable from node attributes, with an error controlled by how well the attribute branch fits the relations of the graph branch (Section 3.4). Relation learning alone admits the trivial solution of uniform relations, which the reconstruction in Eq. (4) rules out by keeping each representation predictive of its own node features (Appendix A.6).

3.2 Fast–Slow Knowledge Acquisition

As source domains alternate, the pre-training loss of existing methods fluctuates sharply (Figure 1(b)–(c)). The parameters of each branch in PARR are also updated on one source graph at a time. If these parameters also supplied the relation targets, changes induced by each source graph would be passed to the other branch through these targets. To address this issue, we introduce Fast–Slow Knowledge Acquisition (FSKA), which separates the parameters that adapt to the current source graph from those that supply the relation targets.

Inspired by momentum teachers (Tarvainen & Valpola, 2017; He et al., 2020; Grill et al., 2020), each branch $a \in \{G, M\}$ maintains a slow copy $\overline{\theta}_a$ of its online parameters θ_a, initialized as $\overline{\theta}_a = \theta_a$. The online parameters are optimized on the current source graph and form the fast process, which acquires knowledge from the current domain. The slow copy receives no gradients and forms the slow process, which integrates this knowledge across domains. After each step, it is updated by an exponential moving average:

$$\overline{\theta}_a \leftarrow \mu\,\overline{\theta}_a + (1-\mu)\,\theta_a, \qquad a \in \{G, M\}, \tag{7}$$

where $\mu \in (0, 1)$ is the momentum coefficient, so each online update enters the relation targets only gradually. Each slow copy uses the same masked input and candidate sets as its online branch, and supplies the relation target q_a in Eq. (6). Combined with PARR, each fast branch learns from the slow copy of the other branch. PARR determines the relational target being exchanged, whereas FSKA controls the time scale at which this target enters reciprocal supervision.

3.3 Training Objective and Transfer

The overall objective combines reconstruction with the two directions of relation learning:

$$\mathcal{L} = \mathcal{L}^G_{\mathrm{rec}} + \mathcal{L}^M_{\mathrm{rec}} + \lambda\, r_e \left(\mathcal{L}^G_{\mathrm{kd}} + \mathcal{L}^M_{\mathrm{kd}}\right), \qquad r_e = \min(1, e/T_w), \tag{8}$$

where $\lambda > 0$ weights relation learning and r_e increases its weight linearly over the first T_w epochs. This warm-up lets reconstruction shape feature-dependent representations before relation learning reaches its full weight. Since the relation targets receive no gradients, each online branch a is optimized only through $\mathcal{L}^a_{\mathrm{rec}} + \lambda r_e \mathcal{L}^a_{\mathrm{kd}}$. In each epoch, we visit the source graphs in turn. At each visit, we take one optimization step on both online branches and their decoders, then update the slow copies. Each epoch thus takes S steps, one per source graph.

After pre-training, we discard the attribute branch, the decoders, and the slow copies. For an unseen target graph, we apply the frozen graph encoder f_G to its preprocessed features and adjacency matrix, without input masks or domain-specific encoder parameters. The complete algorithm, hyperparameters, and computational cost are given in Appendix B.

3.4 Theoretical Analysis

In this subsection, we analyze PARR and FSKA theoretically, measuring what a branch has learned by the information in its relation distributions. Proofs are given in Appendix A.

Propagation erodes relational information. We draw a source graph s with probability π_s, an anchor with its candidate slots, and the masked inputs of both branches. Let P be the relation distribution of the graph branch (Eq. (5)), and let C be a slot drawn from P. Let ξ contain the features of the anchor and its candidates in both masked inputs.

Definition 3.1 (Knowledge beyond propagation)**.** Let $\overline{P} = \mathbb{E}[P \mid \xi]$ denote the component of the graph-branch relation distribution predictable from propagation-free attribute observations, for a fixed candidate set. We call $\overline{P}$ the knowledge beyond propagation.

The information in the relations of the graph branch then splits as:

$$I(C; P) = I(C; \xi) + R, \qquad R = I(C; P \mid \xi) = \mathbb{E}\ \mathrm{KL}(P \| \overline{P}). \tag{9}$$

The first term is carried by the knowledge beyond propagation, and R is the part induced by propagation. Let $\mathcal{E}$ be the Dirichlet energy on graph s (Section 2), and let $\zeta_s < 1$ be the largest magnitude among the eigenvalues of $\widehat{\mathbf{A}}_s$ other than one.

Theorem 3.2 (Propagation flattens relations). *Let $\mathbf{Z} = \widehat{\mathbf{A}}_s^K \mathbf{Y}$, and let P_i^h be the relations of anchor i over its candidates within h hops. Then the following holds:*

$$\mathrm{KL}(P_i^h \| U) \le \beta_i \, \mathcal{E}(\mathbf{Z}) \le \beta_i \, \zeta_s^{2K} \, \mathcal{E}(\mathbf{Y}), \tag{10}$$

where U is uniform and β_i depends only on h, τ, the degree of i, and $\|\mathbf{z}_i\|_2$.

Since $I(C; P_i^h) \le \mathbb{E}\,\mathrm{KL}(P_i^h \| U)$, Eq. (10) gives a spectral decay bound if anchor norms are bounded away from zero uniformly in K. Input alignment leaves the spectral factor unchanged; the bound also depends on the initial energy and anchor norms.

Reciprocal relation learning removes the propagation-induced part. Since the attribute branch performs no propagation, its relation distribution Q is a function of ξ and contains no information induced by propagation.

Theorem 3.3 (Reciprocal relation learning). *(i) For every attribute branch Q, $\mathbb{E}\,\mathrm{KL}(P\|Q) = R + \mathbb{E}\,\mathrm{KL}(\overline{P}\|Q) \ge R$, with equality if and only if $Q = \overline{P}$. (ii) If the relations of the graph branch move toward a returned target Q, that is, $P' = (1-\eta)P + \eta Q$ with $\eta \in [0,1]$, then $R' \le (1-\eta)R$. If $Q = \overline{P}$, then in addition $I(C';\xi) = I(C;\xi)$. (iii) $\mathbb{E}\|P - \overline{P}\|_1^2 \le 2R$, and under the conditions of Appendix A.5, $R \le \kappa \sum_s \pi_s \sqrt{\mathcal{E}(\mathbf{X}_s)}$.*

Part (i) shows that learning from the graph branch drives the attribute branch toward the knowledge beyond propagation. Part (ii) shows that moving the relations a fraction η toward an attribute target removes at least a fraction η of R, and a target equal to $\overline{P}$ also keeps the information predicted by attributes. Part (iii) relates the size of this correction to the Dirichlet energy of each graph, and FLO makes it with parameters shared by all graphs. On our six graphs, the propagation-induced relation change correlates with the drop without relation learning ($r = 0.89$, Appendix A.8).

Slowly updated targets filter cross-domain interference. FLO visits the S source graphs in turn and takes one update on each. Over a window of training, we write each update as $\delta_t = \mathbf{g} + \mathbf{e}_{s(t)}$, where $\mathbf{g}$ is shared by all graphs and $\sum_s \mathbf{e}_s = \mathbf{0}$. The online parameters then follow $\mathbf{g}$ and oscillate with period S. We model the cross-domain interference of Section 1 as this oscillation. The large early loss fluctuations in Figure 1(b)–(c) motivate this model.

Proposition 3.4 (Slowly updated targets as a low-pass filter). *The slowly updated parameters $\overline{\theta}_t = \mu\overline{\theta}_{t-1} + (1-\mu)\theta_t$ follow $\mathbf{g}$ with a lag of $\mu/(1-\mu)$ steps. In the steady state, their period-S oscillation is at most ρ times that of the online parameters, where ρ is given by:*

$$\rho = \frac{1-\mu}{\sqrt{(1-\mu)^2 + 4\mu \sin^2(\pi/S)}}. \tag{11}$$

The relation targets inherit this attenuation up to a Lipschitz constant.

With $\mu = 0.99$ and $S = 5$, $\rho \approx 0.009$, so the oscillation shrinks by two orders of magnitude. In summary, PARR reduces the propagation-induced part R of the relations and keeps $\overline{P}$ (Theorem 3.3), and FSKA attenuates the oscillation between source graphs (Proposition 3.4).

4 Experiments

In this section, we evaluate FLO on six node classification, three link prediction, and three graph classification datasets. We aim to answer the following four research questions. **Q1**: Does FLO transfer effectively to unseen graph domains? **Q2**: Does FLO learn generalizable knowledge that transfers to unseen graph tasks? **Q3**: Does FLO benefit from increasing domain diversity while keeping pre-training stable? **Q4**: How do reciprocal relation learning and reconstruction contribute to the performance of FLO? Further results are given in Appendix D.

4.1 Experimental Setup

Datasets. For node classification, we use six graphs: the citation networks Cora, CiteSeer, and PubMed (Sen et al., 2008; Yang et al., 2016), the co-authorship network CS, and the co-purchase networks Photo and Computers (Shchur et al., 2018). For cross-task transfer, we use three link prediction datasets, Physics (Shchur et al., 2018), WikiCS (Mernyei & Cangea, 2020), and DBLP

Table 1: Accuracy (%) of cross-domain node classification with standard deviations. Each column is an unseen target graph. The best and second-best results are in bold and underlined.

Method	Cora	CiteSeer	PubMed	CS	Photo	Computers	Avg.
DGI	$75.47_{\pm 0.78}$	$65.76_{\pm 1.19}$	$82.83_{\pm 0.24}$	$92.82_{\pm 0.18}$	$91.48_{\pm 0.26}$	$87.24_{\pm 0.21}$	82.60
GRACE	$72.00_{\pm 0.36}$	$56.29_{\pm 0.71}$	$76.80_{\pm 0.37}$	$90.29_{\pm 0.09}$	$90.76_{\pm 0.15}$	$86.82_{\pm 0.15}$	78.83
BGRL	$79.43_{\pm 1.24}$	$64.46_{\pm 1.14}$	$83.91_{\pm 0.26}$	$92.63_{\pm 0.21}$	$92.76_{\pm 0.21}$	$88.98_{\pm 0.16}$	83.69
FUG	$82.79_{\pm 0.72}$	$70.52_{\pm 0.15}$	$84.42_{\pm 0.23}$	$92.41_{\pm 0.02}$	$92.31_{\pm 0.16}$	$87.30_{\pm 0.16}$	84.96
TIG	$\underline{84.39}_{\pm 1.08}$	$\mathbf{73.08}_{\pm 0.52}$	$\underline{86.45}_{\pm 0.13}$	$\underline{93.42}_{\pm 0.05}$	$93.27_{\pm 0.22}$	$89.51_{\pm 0.07}$	$\underline{86.68}$
GFT	$72.99_{\pm 0.94}$	$64.79_{\pm 0.99}$	$83.59_{\pm 0.12}$	$93.06_{\pm 0.27}$	$90.25_{\pm 0.18}$	$84.18_{\pm 0.19}$	81.48
MDGPT	$81.78_{\pm 0.47}$	$72.25_{\pm 0.49}$	$85.76_{\pm 0.24}$	$93.02_{\pm 0.08}$	$93.27_{\pm 0.07}$	$90.01_{\pm 0.09}$	86.02
SAMGPT	$81.64_{\pm 0.56}$	$71.59_{\pm 0.34}$	$85.44_{\pm 0.23}$	$93.04_{\pm 0.09}$	$92.98_{\pm 0.20}$	$89.91_{\pm 0.16}$	85.77
MDGFM	$81.60_{\pm 0.70}$	$71.66_{\pm 0.61}$	$85.63_{\pm 0.25}$	$92.84_{\pm 0.09}$	$\underline{93.33}_{\pm 0.15}$	$\mathbf{90.19}_{\pm 0.13}$	85.87
BRIDGE	$81.59_{\pm 0.65}$	$71.36_{\pm 0.51}$	$85.47_{\pm 0.23}$	$93.11_{\pm 0.05}$	$93.20_{\pm 0.13}$	$90.07_{\pm 0.20}$	85.80
FLO	$\mathbf{85.83}_{\pm 0.65}$	$\underline{72.50}_{\pm 0.91}$	$\mathbf{86.74}_{\pm 0.35}$	$\mathbf{93.62}_{\pm 0.05}$	$\mathbf{93.62}_{\pm 0.23}$	$\underline{90.13}_{\pm 0.07}$	**87.07**

(Bojchevski & Günnemann, 2018), and three graph classification datasets, COIL-RAG, ENZYMES, and PROTEINS (Morris et al., 2020). Dataset statistics are given in Appendix D.1.

Baselines. We compare FLO with ten methods in three categories: (1) graph self-supervised learning methods: DGI (Veličković et al., 2019), GRACE (Zhu et al., 2020), and BGRL (Thakoor et al., 2022); (2) feature unification methods: FUG (Zhao et al., 2024b) and TIG (Zhao et al., 2026); and (3) multi-domain graph pre-training methods: GFT (Wang et al., 2024), MDGPT (Yu et al., 2024), SAMGPT (Yu et al., 2025), MDGFM (Wang et al., 2025a), and BRIDGE (Yuan et al., 2025). All baselines are run with their official code and the settings recommended in their papers, as detailed in Appendix D.2, including the input preprocessing each method requires.

Evaluation protocol. All encoders are frozen during downstream evaluation, and for each task all methods use the same downstream classifier and the same data splits. For node classification, following FUG (Zhao et al., 2024b), we train an ℓ_2-regularized logistic regression classifier with a 10%/10%/80% training/validation/test split of the target nodes, and report the mean and standard deviation over three random splits. For few-shot node classification, following TIG (Zhao et al., 2026), we randomly label $k \in \{1, 3, 5\}$ nodes per class, train the same classifier, and report the mean and standard deviation over 500 sampled tasks shared by all methods. FLO shares one set of pre-training hyperparameters across all target graphs. We evaluate the final encoder without selecting checkpoints on the target graph. The protocol of each task and the hyperparameters of FLO are given in Appendices D.3 and B. The random seed is fixed to 0.

4.2 Cross-Domain Transfer (Q1)

To answer Q1, we adopt the leave-one-domain-out setting: each of the six graphs is held out in turn as the target graph, and the model is pre-trained without labels on the other five. **FLO gives the best average accuracy on unseen graph domains in both standard and few-shot node classification.**

Node classification. As shown in Table 1, FLO achieves the best average accuracy, 87.07%, and ranks first on four of the six target graphs. Feature unification methods align the input features of different graphs, and their encoders learn node relations only through message passing. FLO further refines these relations with those predicted from node attributes alone, so its encoder also acquires the knowledge beyond propagation (Theorem 3.3). Without this reciprocal relation learning, FLO falls below the best feature unification method on Cora and CiteSeer (Table 3). Multi-domain graph pre-training methods such as MDGPT and SAMGPT align domains with randomly initialized parameters updated by immediate loss feedback. FLO introduces no such parameters and takes its relation targets from slow copies, which filter cross-domain interference (Proposition 3.4).

Few-shot node classification. FLO achieves the best average accuracy with one, three, and five labeled nodes per class (Figure 3), exceeding the strongest baseline by 3.71, 4.63, and 2.92 points, respectively. These margins are larger than in node classification, since with few labels the accuracy depends more on the relations among node representations, which FLO refines with the knowledge beyond propagation. The results of all methods on each target graph are given in Appendix D.4.

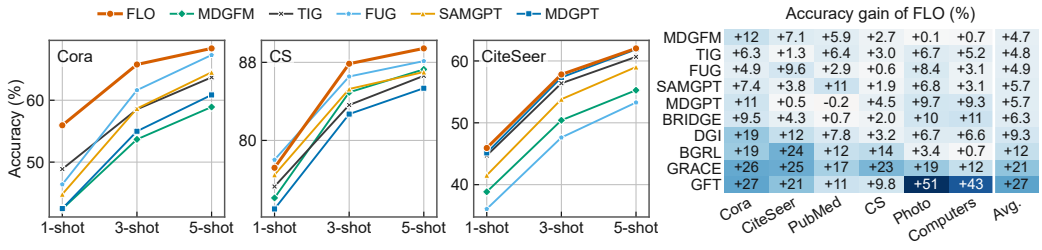


Figure 3: Few-shot node classification. Left: accuracy (%) on Cora, CS, and CiteSeer for FLO and the five baselines with the highest average accuracy. Right: accuracy gain (%) of FLO over each baseline, averaged over the 1-, 3-, and 5-shot settings; Avg. is the mean over the six target graphs.

4.3 Cross-Task Transfer (Q2)

To answer Q2, we pre-train the encoder only on the six node classification datasets and apply it, frozen, to three link prediction and three graph classification datasets. We compare FLO with the feature unification and multi-domain graph pre-training methods, which are designed for transfer across graphs. **FLO learns generalizable knowledge that transfers to unseen tasks on unseen graphs.** As shown in Table 2, FLO achieves the best average on both tasks, and its margin is largest on link prediction, 9.00 points in Hits@50 over the runner-up. Link prediction scores pairs of nodes, and FLO pre-trains the encoder to predict the relations between an anchor and its candidates. On ENZYMES and PROTEINS, whose nodes have only 21 and 32 features, FLO trails the best baseline by 0.40 and 0.58 points, within the standard deviations. One possible reason is that FLO learns the knowledge beyond propagation from node features, which are scarce on these graphs.

Table 2: Hits@50 (%) of link prediction and accuracy (%) of graph classification with standard deviations. The best and second-best results are in bold and underlined.

	Link prediction: Hits@50 (%)				Graph classification: accuracy (%)			
Method	Physics	WikiCS	DBLP	Avg.	COIL-RAG	ENZYMES	PROTEINS	Avg.
FUG	<u>55.99</u>$_{\pm 3.38}$	66.23$_{\pm 1.22}$	<u>64.18</u>$_{\pm 5.29}$	<u>62.13</u>	<u>84.97</u>$_{\pm 1.50}$	33.41$_{\pm 4.31}$	71.64$_{\pm 1.66}$	63.34
TIG	44.46$_{\pm 2.88}$	69.93$_{\pm 0.59}$	61.85$_{\pm 3.43}$	58.75	77.16$_{\pm 1.53}$	33.33$_{\pm 3.36}$	71.56$_{\pm 2.36}$	60.68
GFT	42.93$_{\pm 2.90}$	58.78$_{\pm 2.13}$	43.41$_{\pm 5.49}$	48.37	83.42$_{\pm 1.62}$	**35.88**$_{\pm 5.91}$	<u>72.37</u>$_{\pm 2.92}$	<u>63.89</u>
MDGPT	47.79$_{\pm 1.52}$	68.03$_{\pm 1.26}$	57.25$_{\pm 1.56}$	57.69	80.14$_{\pm 2.19}$	31.24$_{\pm 4.22}$	71.59$_{\pm 3.22}$	60.99
SAMGPT	49.82$_{\pm 1.86}$	70.12$_{\pm 0.73}$	57.60$_{\pm 1.75}$	59.18	81.91$_{\pm 1.99}$	29.57$_{\pm 4.13}$	70.65$_{\pm 2.71}$	60.71
MDGFM	51.22$_{\pm 2.39}$	<u>70.92</u>$_{\pm 1.23}$	55.09$_{\pm 6.77}$	59.08	81.86$_{\pm 1.91}$	33.73$_{\pm 4.32}$	**72.54**$_{\pm 2.07}$	62.71
BRIDGE	49.97$_{\pm 3.30}$	69.10$_{\pm 0.80}$	56.94$_{\pm 1.58}$	58.67	82.26$_{\pm 1.80}$	31.24$_{\pm 3.59}$	71.14$_{\pm 3.09}$	61.55
FLO	**67.37**$_{\pm 9.49}$	**72.75**$_{\pm 0.98}$	**73.28**$_{\pm 0.78}$	**71.13**	**86.59**$_{\pm 1.35}$	<u>35.48</u>$_{\pm 4.65}$	71.96$_{\pm 2.12}$	**64.68**

4.4 Domain Diversity and Training Stability (Q3)

To answer Q3, we vary the number of source domains and track the pre-training loss.

Domain diversity. We add source domains in a fixed order, skipping the target graph, and keep the pre-training budget fixed. As shown in Figure 4(a), from one to five source domains, FLO improves by 1.37–6.31 points on all six target graphs, whereas the four multi-domain baselines change by at most 0.63 points. **FLO benefits from increasing domain diversity.**

Training stability. We record the pre-training loss in the first 40 epochs, taking the loss of the graph encoder for FLO. As shown in Figure 4(b), the loss of FLO rises by at most 6.6% between consecutive epochs, whereas those of BRIDGE and SAMGPT rise by up to 249% and 371%. **FLO keeps pre-training stable as the source domains alternate.** These observations are complementary: reduced loss oscillation alone would not establish effective knowledge acquisition, while the consistent gains from additional source domains show that stability is accompanied by improved transfer. This pattern matches the intended roles of PARR and FSKA: PARR exposes complementary relational information, while FSKA consolidates it across domains.

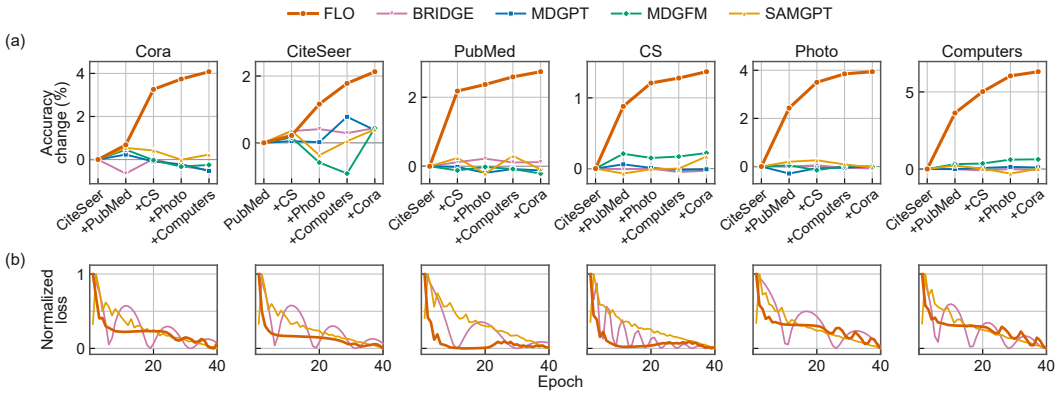


Figure 4: Domain diversity and training stability. (a) Accuracy change (%) as source domains are added in the order on the horizontal axis. (b) Min–max normalized log loss in the first 40 epochs.

4.5 Ablation Study (Q4)

To answer Q4, we remove or replace parts of FLO (Table 3). **Reciprocal relation learning improves the graph encoder through the returned relation targets, and reconstruction rules out the trivial solution of relation learning.** Removing both relation learning objectives lowers accuracy most on CiteSeer and Cora, the two graphs on which propagation changes the relations most (Section 3.4). Removing the slow attribute branch gives accuracy similar to removing both objectives, since the graph encoder then receives no returned targets (Theorem 3.3(ii)). Removing the slow graph branch lowers accuracy further, since the attribute branch is then not driven toward $\overline{P}$ (Theorem 3.3(i)) and its returned targets no longer correct the graph encoder. Removing reconstruction lowers accuracy to 30.33% on Cora and 20.66% on CiteSeer, close to the share of the largest class (Proposition A.2). The two directions thus serve distinct roles: graph-to-attribute transfer identifies the propagation-free conditional target, whereas attribute-to-graph transfer returns this target to the transferable graph encoder. A GCN attribute branch gives no gain over removing both relation learning objectives on any target graph, so the gain depends on a propagation-free attribute branch.

Table 3: Ablation study on node classification: accuracy (%) with standard deviations. w/o slow attribute branch removes $\mathcal{L}_{\mathrm{kd}}^{G}$, the relation targets returned to the graph branch; w/o slow graph branch removes $\mathcal{L}_{\mathrm{kd}}^{M}$, the relation targets that train the attribute branch. Removing one doubles the weight of the other, so the total relation weight is unchanged. The best results are in bold.

Variant	Cora	CiteSeer	PubMed	CS	Photo	Computers
w/o $\mathcal{L}_{\mathrm{kd}}$	$83.26_{\pm 0.84}$	$66.52_{\pm 1.31}$	$86.63_{\pm 0.23}$	$93.13_{\pm 0.09}$	$93.34_{\pm 0.07}$	$90.04_{\pm 0.12}$
w/o slow attribute branch	$83.37_{\pm 0.89}$	$66.98_{\pm 1.11}$	$86.67_{\pm 0.26}$	$93.03_{\pm 0.14}$	$93.47_{\pm 0.15}$	$90.04_{\pm 0.11}$
w/o slow graph branch	$64.65_{\pm 0.95}$	$42.01_{\pm 1.61}$	$80.57_{\pm 0.32}$	$82.58_{\pm 0.23}$	$89.62_{\pm 0.36}$	$84.83_{\pm 0.20}$
w/o $\mathcal{L}_{\mathrm{rec}}$	$30.33_{\pm 0.19}$	$20.66_{\pm 0.63}$	$82.48_{\pm 0.31}$	$76.14_{\pm 0.92}$	$59.72_{\pm 1.94}$	$52.40_{\pm 1.16}$
w/ GCN attribute branch	$79.74_{\pm 0.10}$	$55.56_{\pm 0.22}$	$86.60_{\pm 0.26}$	$91.52_{\pm 0.15}$	$92.02_{\pm 0.55}$	$89.48_{\pm 0.17}$
FLO	$\mathbf{85.83}_{\pm 0.65}$	$\mathbf{72.50}_{\pm 0.91}$	$\mathbf{86.74}_{\pm 0.35}$	$\mathbf{93.62}_{\pm 0.05}$	$\mathbf{93.62}_{\pm 0.23}$	$\mathbf{90.13}_{\pm 0.07}$

5 Conclusion

In this paper, we proposed FLO for multi-domain graph pre-training, where existing methods struggle to benefit from increasing domain diversity. PARR refines the node relations induced by message passing with feedback grounded in node attributes, and FSKA takes this feedback from a slow process that integrates knowledge across domains. We proved that PARR leads the graph encoder to the knowledge beyond propagation and FSKA filters cross-domain interference. On 12 datasets, FLO achieves the best average performance in cross-domain and cross-task transfer and benefits from increasing domain diversity, suggesting that domain alignment is a prerequisite rather than a guarantee of knowledge acquisition. Future work will extend FLO to graphs with scarce node features.

AI USE STATEMENT

Generative AI tools were used in this work to assist with language refinement, manuscript organization, and the retrieval of potentially relevant literature. All AI-assisted revisions and literature suggestions were independently examined by the authors against the manuscript, experimental records, and original cited sources before being incorporated. Generative AI was not used to produce experimental results or to independently determine scientific claims or conclusions. All scientific content was ultimately assessed and verified by the authors, who take full responsibility for the final manuscript.

REPRODUCIBILITY STATEMENT

Appendix B specifies the pre-training algorithm and all hyperparameters of FLO. Appendix D describes the datasets, baselines, and evaluation protocols. Complete proofs are given in Appendix A.

REFERENCES

Elahe Arani, Fahad Sarfraz, and Bahram Zonooz. Learning fast, learning slow: A general continual learning method based on complementary learning system. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=uxxFrDwrE7Y.

Beatrice Bevilacqua, Joshua Robinson, Jure Leskovec, and Bruno Ribeiro. Holographic node representations: Pre-training task-agnostic node embeddings. In *International Conference on Learning Representations*, volume 2025, pp. 64117–64152, 2025.

Aleksandar Bojchevski and Stephan Günnemann. Deep Gaussian embedding of graphs: Unsupervised inductive learning via ranking. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=r1ZdKJ-0W.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Dan Busbridge, Jason Ramapuram, Pierre Ablin, Tatiana Likhomanenko, Eeshan Gunesh Dhekane, Xavier Suau Cuadros, and Russell Webb. How to scale your EMA. *Advances in Neural Information Processing Systems*, 36:73122–73174, 2023.

Chen Cai and Yusu Wang. A note on over-smoothing for graph neural networks. *arXiv preprint arXiv:2006.13318*, 2020.

Junbum Cha, Sanghyuk Chun, Kyungjae Lee, Han-Cheol Cho, Seunghyun Park, Yunsung Lee, and Sungrae Park. SWAD: Domain generalization by seeking flat minima. *Advances in Neural Information Processing Systems*, 34:22405–22418, 2021.

Moshe Eliasof, Krishna Sri Ipsit Mantri, Beatrice Bevilacqua, Bruno Ribeiro, and Carola-Bibiane Schönlieb. Bridging input feature spaces towards graph foundation models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=Dt4XAIKYbf.

Zhiyuan Fang, Jianfeng Wang, Lijuan Wang, Lei Zhang, Yezhou Yang, and Zicheng Liu. SEED: Self-supervised distillation for visual representation. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=AHm3dbp7D1D.

Kaituo Feng, Changsheng Li, Ye Yuan, and Guoren Wang. FreeKD: Free-direction knowledge distillation for graph neural networks. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pp. 357–366, 2022.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent – a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *2020 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 9726–9735. IEEE, 2020.

Yufei He, Yuan Sui, Xiaoxin He, and Bryan Hooi. UniGraph: Learning a unified cross-domain foundation model for text-attributed graphs. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 1*, pp. 448–459, 2025.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Zhenyu Hou, Xiao Liu, Yukuo Cen, Yuxiao Dong, Hongxia Yang, Chunjie Wang, and Jie Tang. GraphMAE: Self-supervised masked graph autoencoders. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pp. 594–604, 2022.

Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HJlWWJSFDH.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=SJU4ayYgl.

Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Yuhan Li, Peisong Wang, Zhixun Li, Jeffrey Xu Yu, and Jia Li. ZeroG: Investigating cross-dataset zero-shot transferability in graphs. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1725–1735, 2024.

Hao Liu, Jiarui Feng, Lecheng Kong, Ningyue Liang, Dacheng Tao, Yixin Chen, and Muhan Zhang. One for all: Towards training one graph model for all classification tasks. In *International conference on learning representations*, volume 2024, pp. 20188–20210, 2024.

Jiawei Liu, Cheng Yang, Zhiyuan Lu, Junze Chen, Yibo Li, Mengmei Zhang, Ting Bai, Yuan Fang, Lichao Sun, Philip S Yu, et al. Graph foundation models: Concepts, opportunities and challenges. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(6):5023–5044, 2025.

Haitao Mao, Zhikai Chen, Wenzhuo Tang, Jianan Zhao, Yao Ma, Tong Zhao, Neil Shah, Mikhail Galkin, and Jiliang Tang. Position: Graph foundation models are already here. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=Edz0QXKKAo.

James L McClelland, Bruce L McNaughton, and Randall C O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, 102(3):419, 1995.

Péter Mernyei and Cătălina Cangea. Wiki-CS: A Wikipedia-based benchmark for graph neural networks. *arXiv preprint arXiv:2007.02901*, 2020.

Daniel Morales-Brotons, Thijs Vogels, and Hadrien Hendrikx. Exponential moving average of weights in deep learning: Dynamics and benefits. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=2M9CUnYnBA.

Christopher Morris, Nils M Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. TUDataset: A collection of benchmark datasets for learning with graphs. *arXiv preprint arXiv:2007.08663*, 2020.

Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3962–3971. IEEE, 2019.

Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. *SIAM journal on control and optimization*, 30(4):838–855, 1992.

Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. GCC: Graph contrastive coding for graph neural network pre-training. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 1150–1160, 2020.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Gallagher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93–106, 2008.

Lianze Shan, Jitao Zhao, Dongxiao He, Yongqi Huang, Zhiyong Feng, and Weixiong Zhang. MUG: Meta-path-aware universal heterogeneous graph pre-training. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 25260–25268, 2026a.

Lianze Shan, Jitao Zhao, Dongxiao He, Siqi Liu, Jiaxu Cui, and Weixiong Zhang. LEDA: Latent semantic distribution alignment for multi-domain graph pre-training. In *Proceedings of the ACM Web Conference 2026*, pp. 614–625, 2026b.

Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. *arXiv preprint arXiv:1811.05868*, 2018.

Yangyi Shen, Jincheng Zhou, Beatrice Bevilacqua, Joshua Robinson, Charilaos Kanatsoulis, Jure Leskovec, and Bruno Ribeiro. Zero-shot generalization of GNNs over distinct attribute domains. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 54701–54731. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/shen25p.html.

Li Sun, Zhenhao Huang, Suyang Zhou, Qiqi Wan, Hao Peng, and Philip Yu. RiemannGFM: Learning a graph foundation model from Riemannian geometry. In *Proceedings of the ACM on Web Conference 2025*, pp. 1154–1165, 2025a.

Li Sun, Zhenhao Huang, Silei Chen, Lanxu Yang, Junda Ye, Sen Su, and Philip Yu. Multi-domain Riemannian graph gluing for building graph foundation models. In *International Conference on Learning Representations*, volume 2026, pp. 37129–37169, 2026.

Yifei Sun, Yang Yang, Xiao Feng, Zijun Wang, Haoyang Zhong, Chunping Wang, and Lei Chen. Handling feature heterogeneity with learnable graph patches. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 1*, pp. 1313–1324, 2025b.

Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *Advances in neural information processing systems*, 30, 2017.

Shantanu Thakoor, Corentin Tallec, Mohammad Gheshlaghi Azar, Mehdi Azabou, Eva L Dyer, Remi Munos, Petar Veličković, and Michal Valko. Large-scale representation learning on graphs via bootstrapping. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=0UXT6PpRpW.

Yijun Tian, Chuxu Zhang, Zhichun Guo, Xiangliang Zhang, and Nitesh Chawla. Learning MLPs on graphs: A unified view of effectiveness, robustness, and efficiency. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Cs3r5KLdoj.

Petar Veličković, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rklz9iAcKQ.

Shuo Wang, Bokui Wang, Zhixiang Shen, Boyan Deng, and Zhao Kang. Multi-domain graph foundation models: Robust knowledge transfer via topology alignment. In *Forty-second International Conference on Machine Learning*, 2025a. URL https://openreview.net/forum?id=44Wq2xeRF0.

Yi Wang, Jitao Zhao, Di Jin, and Dongxiao He. Beyond feature and structure alignment: Learning transferable propagation knowledge for graph foundation models. *arXiv preprint arXiv:2607.28980*, 2026.

Zehong Wang, Zheyuan Zhang, Nitesh V Chawla, Chuxu Zhang, and Yanfang Ye. GFT: Graph foundation model with transferable tree vocabulary. *Advances in neural information processing systems*, 37:107403–107443, 2024.

Zehong Wang, Zheyuan Zhang, Tianyi Ma, Nitesh V Chawla, Chuxu Zhang, and Yanfang Ye. Towards graph foundation models: Learning generalities across graphs via task-trees. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 65518–65555. PMLR, 13–19 Jul 2025b. URL https://proceedings.mlr.press/v267/wang25eq.html.

Lianghao Xia and Chao Huang. AnyGraph: Graph foundation model in the wild. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 882–896, 2026.

Ankang Yang, Jitao Zhao, Dongxiao He, Liang Yang, Di Jin, and Weixiong Zhang. A graph foundation model with spectral parsing and prototype-guided spatial propagation. *arXiv preprint arXiv:2606.03315*, 2026.

Yiding Yang, Jiayan Qiu, Mingli Song, Dacheng Tao, and Xinchao Wang. Distilling knowledge from graph convolutional networks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7072–7081. IEEE, 2020.

Zhilin Yang, William Cohen, and Ruslan Salakhudinov. Revisiting semi-supervised learning with graph embeddings. In *International conference on machine learning*, pp. 40–48. PMLR, 2016.

Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. *Advances in neural information processing systems*, 33: 5812–5823, 2020.

Xingtong Yu, Chang Zhou, Yuan Fang, and Xinming Zhang. Text-free multi-domain graph pre-training: Toward graph foundation models. *arXiv preprint arXiv:2405.13934*, 2024.

Xingtong Yu, Zechuan Gong, Chang Zhou, Yuan Fang, and Hui Zhang. SAMGPT: Text-free graph foundation model for multi-domain pre-training and cross-domain adaptation. In *Proceedings of the ACM on Web Conference 2025*, pp. 1142–1153, 2025.

Haonan Yuan, Qingyun Sun, Junhua Shi, Xingcheng Fu, Bryan Hooi, Jianxin Li, and Philip S Yu. How much can transfer? BRIDGE: Bounded multi-domain graph foundation model with generalization guarantees. In *Forty-second International Conference on Machine Learning*, 2025.

Michael Zhang, James Lucas, Jimmy Ba, and Geoffrey E Hinton. Lookahead optimizer: k steps forward, 1 step back. *Advances in neural information processing systems*, 32, 2019.

Shichang Zhang, Yozen Liu, Yizhou Sun, and Neil Shah. Graph-less neural networks: Teaching old MLPs new tricks via distillation. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=4p6_5HBWPCw.

Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep mutual learning. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4320–4328. IEEE, 2018.

Haihong Zhao, Aochuan Chen, Xiangguo Sun, Hong Cheng, and Jia Li. All in one and one for all: A simple yet effective method towards cross-domain graph pretraining. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 4443–4454, 2024a.

Jianan Zhao, Zhaocheng Zhu, Mikhail Galkin, Hesham Mostafa, Michael Bronstein, and Jian Tang. Fully-inductive node classification on arbitrary graphs. In *International Conference on Learning Representations*, volume 2025, pp. 23968–23986, 2025.

Jitao Zhao, Di Jin, Meng Ge, Lianze Shan, Xin Wang, Dongxiao He, and Zhiyong Feng. FUG: Feature-universal graph contrastive pre-training for graphs with diverse node features. *Advances in Neural Information Processing Systems*, 37:4003–4034, 2024b.

Jitao Zhao, Yi Wang, Yawen Li, Dongxiao He, Di Jin, Zhiyong Feng, and Weixiong Zhang. Towards graph foundation model: Node feature transfer invariant modeling on general graphs. In *Proceedings of the ACM Web Conference 2026*, pp. 810–821, 2026.

Mingkai Zheng, Shan You, Fei Wang, Chen Qian, Changshui Zhang, Xiaogang Wang, and Chang Xu. ReSSL: Relational self-supervised learning with weak augmentation. *Advances in Neural Information Processing Systems*, 34:2543–2555, 2021.

Yizhen Zheng, Shirui Pan, Vincent Lee, Yu Zheng, and Philip S Yu. Rethinking and scaling up graph contrastive learning: An extremely efficient approach with group discrimination. *Advances in Neural Information Processing Systems*, 35:10809–10820, 2022.

Ziang Zhou, Zhihao Ding, Jieming Shi, Li Qing, and Shiqi Shen. TINED: GNNs-to-MLPs by teacher injection and Dirichlet energy distillation. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=nshtqLv4r4.

Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Deep graph contrastive representation learning. *arXiv preprint arXiv:2006.04131*, 2020.

Yun Zhu, Yaoke Wang, Haizhou Shi, Zhenshuo Zhang, Dian Jiao, and Siliang Tang. GraphControl: Adding conditional control to universal graph pre-trained models for graph domain transfer learning. In *Proceedings of the ACM Web Conference 2024*, pp. 539–550, 2024.

Wei Zhuo and Siqiang Luo. Modality-free graph in-context alignment. In *International Conference on Learning Representations*, volume 2026, pp. 21310–21336, 2026.

A PROOFS

We prove the results of Section 3.4 and one result on reconstruction. Table 4 lists what each result shows and the main tools of its proof. Appendix A.1 fixes the setting and two facts used throughout. Appendix A.8 relates the relation change measured on our graphs to the gain of PARR, and Appendix A.9 states the invariance of relation distributions.

Table 4: Results proved in this appendix.

Result	What it shows	Main tools	Proof
Eq. (9)	The relational information of the graph branch splits into an attribute-predicted part and the propagation-induced part R	Chain rule, tower property	A.2
Lemma A.1	Close embeddings give close relation distributions	Lipschitz bounds of cosine and softmax	A.3
Theorem 3.2	Propagation drives the relations among nearby nodes toward uniform	Edge form of the Dirichlet energy, spectral decomposition	A.4
Theorem 3.3	Targets returned by the attribute branch remove R	Conditional mean, convexity of KL, Pinsker's inequality	A.5
Proposition A.2	Reconstruction rules out uniform relations	Bounds of the decoder on directions	A.6
Proposition 3.4	Slow copies damp period-S oscillations by ρ	Frequency response, Parseval's identity	A.7

A.1 SETTING AND NOTATION

Expectations are taken over the sampled source graph s, anchor, candidate slots, and masks, with the slow copies held fixed in Theorems 3.2 and 3.3. The observation ξ includes the masked input of the attribute branch, so the targets of the attribute branch are functions of ξ. Relation distributions compared in a KL divergence share their support and are positive on the valid slots.

For graph s we write $\widetilde{\mathbf{A}} = \mathbf{A}_s + \mathbf{I}$, $\widetilde{d}_k = \sum_l \widetilde{\mathbf{A}}_{kl}$, $\widehat{\mathbf{A}}_s = \sum_r \omega_r \mathbf{u}_r \mathbf{u}_r^\top$, and $\mathbf{L} = \mathbf{I} - \widehat{\mathbf{A}}_s$, whose eigenvalues $1 - \omega_r$ lie in $[0, 2)$. For a nonzero vector $\mathbf{z}$, $\hat{\mathbf{z}} = \mathbf{z}/\|\mathbf{z}\|_2$. Theorem 3.2 treats linear propagation. ReLU does not increase the Dirichlet energy and a weight matrix scales it by at most $\|\mathbf{W}\|_{\mathrm{op}}^2$, so the contraction extends to bias-free ReLU layers (Cai & Wang, 2020).

Two facts. For nonzero $\mathbf{u}, \mathbf{b}$:

$$\|\hat{\mathbf{u}} - \hat{\mathbf{b}}\|_2 \le \frac{2\|\mathbf{u} - \mathbf{b}\|_2}{\|\mathbf{b}\|_2}, \tag{12}$$

which follows from $\|\hat{\mathbf{u}} - \hat{\mathbf{b}}\| \le \big|\|\mathbf{b}\| - \|\mathbf{u}\|\big|/\|\mathbf{b}\| + \|\mathbf{u} - \mathbf{b}\|/\|\mathbf{b}\|$. For nonzero vectors, $\cos(\mathbf{u}, \mathbf{b}) = 1 - \frac{1}{2}\|\hat{\mathbf{u}} - \hat{\mathbf{b}}\|_2^2$.

A.2 INFORMATION DECOMPOSITION

Eq. (9) separates what node attributes predict from what propagation adds. We split the information by the chain rule and evaluate the conditional part with the tower property.

Eq. (9) **(restated).** *$I(C; P) = I(C; \xi) + R$ with $R = I(C; P \mid \xi) = \mathbb{E}\ \mathrm{KL}(P \| \overline{P})$.*

Proof. **Step 1 (chain rule).** Given P, the slot C is drawn from P and does not depend on ξ, so $I(C; \xi \mid P) = 0$ and $I(C; P) = I(C; P, \xi) = I(C; \xi) + I(C; P \mid \xi)$.

Step 2 (conditional entropy). Given ξ alone, C follows $\mathbb{E}[P \mid \xi] = \overline{P}$. Hence $I(C; P \mid \xi) = H(C \mid \xi) - H(C \mid P, \xi) = \mathbb{E}H(\overline{P}) - \mathbb{E}H(P)$.

Step 3 (tower property). Since $\overline{P}$ is a function of ξ, $\mathbb{E}\sum_j P_j \log \overline{P}_j = \mathbb{E}\sum_j \overline{P}_j \log \overline{P}_j = -\mathbb{E}H(\overline{P})$. Therefore $\mathbb{E}\ \mathrm{KL}(P \| \overline{P}) = -\mathbb{E}H(P) + \mathbb{E}H(\overline{P}) = I(C; P \mid \xi)$. □

A.3 A Lemma on Stable Relations

Theorem 3.3(iii) and Proposition 3.4 compare relation distributions computed from two nearby sets of embeddings. Small changes of the embeddings give small changes of every cosine, and the softmax turns small changes of the logits into small changes of the log-probabilities.

Lemma A.1 (Stable relations). *Let $\mathbf{u}_k, \mathbf{b}_k$ be two embeddings of the anchor ($k = 0$) and its candidates, with $\|\mathbf{u}_k - \mathbf{b}_k\|_2 \le \epsilon$, $\|\mathbf{b}_k\|_2 \ge c > 0$, and $\mathbf{u}_k \ne \mathbf{0}$ for all k. Then their temperature-τ relation distributions satisfy:*

$$\max\{\mathrm{KL}(p_{\mathbf{U}} \| p_{\mathbf{B}}),\, \mathrm{KL}(p_{\mathbf{B}} \| p_{\mathbf{U}})\} \le \frac{8\epsilon}{\tau c}.$$

Proof. **Step 1 (directions).** By Eq. (12), $\|\hat{\mathbf{u}}_k - \hat{\mathbf{b}}_k\| \le 2\epsilon/c$ for every k.

Step 2 (cosines and logits). For unit vectors, $|\langle \mathbf{a}, \mathbf{b}\rangle - \langle \mathbf{a}', \mathbf{b}'\rangle| \le \|\mathbf{a} - \mathbf{a}'\| + \|\mathbf{b} - \mathbf{b}'\|$. Hence every cosine between the anchor and a candidate changes by at most $4\epsilon/c$, and every logit by at most $\Delta = 4\epsilon/(\tau c)$.

Step 3 (log-probabilities). A log-probability is a logit minus the log-sum-exp of all logits, and the log-sum-exp changes by at most the largest change of a logit. Hence every log-probability changes by at most 2Δ, and so does either KL divergence, which is an average of these differences. □

A.4 Proof of Theorem 3.2

Theorem 3.2 bounds relational variation among nearby candidates via energy. Small energy relative to the squared anchor norm makes embedding directions close. Candidate cosines are then close to one, giving nearly uniform relations. Steps 1–4 bound the relations by the energy, Step 5 bounds the energy by the spectrum, and Step 6 turns the bound into a bound on information.

Theorem 3.2 (restated). *Let $\mathbf{Z} = \widehat{\mathbf{A}}_s^K \mathbf{Y}$, and let P_i^h be the relations of anchor i over its candidates within h hops. Then the following holds:*

$$\mathrm{KL}(P_i^h \| U) \le \beta_i\, \mathcal{E}(\mathbf{Z}) \le \beta_i\, \zeta_s^{2K}\, \mathcal{E}(\mathbf{Y}),$$

where U is uniform and $\beta_i = 2h\widetilde{d}_i/(\tau\|\mathbf{z}_i\|_2^2)$. Moreover, $I(C; P_i^h) \le \mathbb{E}\ \mathrm{KL}(P_i^h \| U)$.

Proof. **Step 1 (energy as edge differences).** With $\mathbf{y}_k = \mathbf{z}_k/\sqrt{\widetilde{d}_k}$:

$$\mathcal{E}(\mathbf{Z}) = \mathrm{tr}(\mathbf{Z}^\top \mathbf{L} \mathbf{Z}) = \tfrac{1}{2} \sum_{k,l} \widetilde{\mathbf{A}}_{kl} \|\mathbf{y}_k - \mathbf{y}_l\|_2^2 = \sum_{\{k,l\} \in E_s} \|\mathbf{y}_k - \mathbf{y}_l\|_2^2.$$

Step 2 (nearby nodes). Let v be a candidate within h hops and take a shortest path from i to v. Its edges are distinct, so by the Cauchy–Schwarz inequality $\|\mathbf{y}_i - \mathbf{y}_v\|_2 \le \sqrt{h\, \mathcal{E}(\mathbf{Z})}$.

Step 3 (cosines). Cosines ignore positive scaling, so $\hat{\mathbf{z}}_k = \hat{\mathbf{y}}_k$. By Eq. (12) and $\|\mathbf{y}_i\| = \|\mathbf{z}_i\|/\sqrt{\widetilde{d}_i}$:

$$\|\hat{\mathbf{z}}_i - \hat{\mathbf{z}}_v\|_2 \le \frac{2\sqrt{\widetilde{d}_i\, h\, \mathcal{E}(\mathbf{Z})}}{\|\mathbf{z}_i\|_2}, \qquad \cos(\mathbf{z}_i, \mathbf{z}_v) \ge 1 - \frac{2h\widetilde{d}_i\, \mathcal{E}(\mathbf{Z})}{\|\mathbf{z}_i\|_2^2}.$$

Step 4 (near-uniform relations). The anchor slot has cosine one, so all m logits lie in $[1/\tau - w,\ 1/\tau]$ with $w = \beta_i \mathcal{E}(\mathbf{Z})$. Every probability is then at most e^w/m, and $\mathrm{KL}(P_i^h \| U) = \sum_j p_j \log(m p_j) \le w$.

Step 5 (energy contraction). $\widehat{\mathbf{A}}_s$ and $\mathbf{L}$ share eigenvectors, so:

$$\mathcal{E}(\widehat{\mathbf{A}}_s^K \mathbf{Y}) = \sum_r (1 - \omega_r)\, \omega_r^{2K} \|\mathbf{u}_r^\top \mathbf{Y}\|_2^2 \le \zeta_s^{2K} \sum_r (1 - \omega_r) \|\mathbf{u}_r^\top \mathbf{Y}\|_2^2 = \zeta_s^{2K} \mathcal{E}(\mathbf{Y}),$$

because the terms with $\omega_r = 1$ vanish and all other eigenvalues satisfy $|\omega_r| \le \zeta_s$.

Step 6 (information). For a random distribution P and C drawn from it, $\mathbb{E}\ \mathrm{KL}(P \| U) = \log m - \mathbb{E} H(P) = \mathrm{KL}(\mathbb{E} P \| U) + I(C; P) \ge I(C; P)$. □

A.5 Proof of Theorem 3.3

Theorem 3.3 is the basis of PARR. The attribute branch sees only ξ, so the best it can do is the conditional mean $\overline{P}$. Mixing the graph relations with any function of ξ shrinks their distance to that mean. Part (i) identifies the best attribute target, Part (ii) shows what a returned target removes, and Part (iii) relates R to the Dirichlet energy.

Theorem 3.3 (restated). *(i) For every attribute branch Q, $\mathbb{E}\ \mathrm{KL}(P\|Q) = R + \mathbb{E}\ \mathrm{KL}(\overline{P}\|Q) \geq R$, with equality if and only if $Q = \overline{P}$. (ii) If the relations of the graph branch move toward a returned target Q, that is, $P' = (1-\eta)P + \eta Q$ with $\eta \in [0,1]$, then $R' \leq (1-\eta)R$. If $Q = \overline{P}$, then in addition $I(C';\xi) = I(C;\xi)$. (iii) $\mathbb{E}\|P - \overline{P}\|_1^2 \leq 2R$, and under the conditions of Step 2 of Part (iii) below, $R \leq \kappa \sum_s \pi_s \sqrt{\mathcal{E}(\mathbf{X}_s)}$.*

Proof. **Part (i).** Q and $\overline{P}$ are functions of ξ. Replacing P by $\mathbb{E}[P \mid \xi] = \overline{P}$ inside the expectation gives:

$$\mathbb{E}\ \mathrm{KL}(P\|Q) - \mathbb{E}\ \mathrm{KL}(P\|\overline{P}) = \mathbb{E}\sum_j P_j \log \frac{\overline{P}_j}{Q_j} = \mathbb{E}\sum_j \overline{P}_j \log \frac{\overline{P}_j}{Q_j} = \mathbb{E}\ \mathrm{KL}(\overline{P}\|Q) \geq 0,$$

with equality if and only if $Q = \overline{P}$ almost surely.

Part (ii). *Step 1 (new attribute-predicted relations).* Since Q is a function of ξ, the new attribute-predicted relations are $\overline{P}' = \mathbb{E}[P' \mid \xi] = (1-\eta)\overline{P} + \eta Q$.

Step 2 (convexity). The KL divergence is jointly convex, so for every realization:
$\mathrm{KL}\big((1-\eta)P + \eta Q \,\big\|\, (1-\eta)\overline{P} + \eta Q\big) \leq (1-\eta)\,\mathrm{KL}(P\|\overline{P}) + \eta\ \mathrm{KL}(Q\|Q) = (1-\eta)\,\mathrm{KL}(P\|\overline{P})$.
Taking expectations gives $R' \leq (1-\eta)R$.

Step 3 (the attribute-predicted part is kept). If $Q = \overline{P}$, then $\overline{P}' = \overline{P}$. The attribute-predicted information $I(C;\xi) = \mathbb{E}\ \mathrm{KL}(\overline{P}\|\mathbb{E}\overline{P})$ depends only on $\overline{P}$ and is therefore unchanged.

Part (iii). *Step 1 (Pinsker's inequality).* Pinsker's inequality gives $\|P - \overline{P}\|_1^2 \leq 2\,\mathrm{KL}(P\|\overline{P})$; taking expectations gives $\mathbb{E}\|P - \overline{P}\|_1^2 \leq 2R$.

Step 2 (a target computed without propagation). Assume that the graph branch computes relations from $\mathbf{U} = \widehat{\mathbf{A}}_s\mathbf{X}_s\mathbf{W}$, that ξ determines the rows of $\mathbf{B} = \mathbf{X}_s\mathbf{W}$ for the anchor and its candidates, and that these rows have norm at least c. The relations Q_B computed from $\mathbf{B}$ are a function of ξ, so Part (i) gives $R \leq \mathbb{E}\ \mathrm{KL}(P\|Q_B)$.

Step 3 (row changes). Each row changes by $\|(\mathbf{U} - \mathbf{B})_k\| = \|(\mathbf{L}\mathbf{X}_s\mathbf{W})_k\| \leq \|\mathbf{W}\|_{\mathrm{op}}\|\mathbf{L}\mathbf{X}_s\|_F$, and $\|\mathbf{L}\mathbf{X}_s\|_F^2 = \mathrm{tr}(\mathbf{X}_s^\top\mathbf{L}^2\mathbf{X}_s) \leq 2\,\mathcal{E}(\mathbf{X}_s)$ because the eigenvalues of $\mathbf{L}$ lie in $[0,2)$.

Step 4 (averaging over graphs). Lemma A.1 with $\epsilon = \|\mathbf{W}\|_{\mathrm{op}}\sqrt{2\mathcal{E}(\mathbf{X}_s)}$ bounds $\mathrm{KL}(P\|Q_B)$ on graph s. Averaging over s gives $R \leq \kappa\sum_s \pi_s\sqrt{\mathcal{E}(\mathbf{X}_s)}$ with $\kappa = 8\sqrt{2}\,\|\mathbf{W}\|_{\mathrm{op}}/(\tau c)$. □

Remark (a non-ideal target). For logits $\mathbf{a}_G$ with $p_G = \mathrm{softmax}(\mathbf{a}_G)$, a returned target Q induces the gradient $p_G - Q$ of $\mathrm{KL}(Q\|p_G)$. It differs from the gradient $p_G - \overline{P}$ of the ideal target by $\overline{P} - Q$, and Pinsker's inequality with Part (i) gives $\mathbb{E}\|\overline{P} - Q\|_2^2 \leq 2\,\mathbb{E}\ \mathrm{KL}(\overline{P}\|Q) = 2\big[\mathbb{E}\ \mathrm{KL}(P\|Q) - R\big]$. The error of the returned target is therefore set by how well the attribute branch fits the graph branch.

A.6 Reconstruction Rules Out Uniform Relations

Relation learning alone admits the trivial solution of uniform relations (Section 3.1). The proposition below shows that reconstruction rules it out. If two embeddings pointed in the same direction, they would decode to the same feature.

Proposition A.2. *Let a branch decode the unit feature $\mathbf{x}_k$ as $\mathrm{RowNorm}(D\mathbf{z}_k)$ with a linear decoder D of smallest singular value $s_{\min} > 0$, and let $\epsilon_k = \|\,\mathrm{RowNorm}(D\mathbf{z}_k) - \mathbf{x}_k\|_2$. Then the following holds:*

$$\|\hat{\mathbf{z}}_i - \hat{\mathbf{z}}_j\|_2 \geq \frac{s_{\min}}{2\|D\|_{\mathrm{op}}}\big(\|\mathbf{x}_i - \mathbf{x}_j\|_2 - \epsilon_i - \epsilon_j\big).$$

Proof. **Step 1 (the decoder on directions).** Let $g(\mathbf{z}) = D\mathbf{z}/\|D\mathbf{z}\|_2$. It ignores positive scaling, so $g(\mathbf{z}) = g(\hat{\mathbf{z}})$. By Eq. (12) and $\|D\hat{\mathbf{z}}_j\| \geq s_{\min}$, $\|g(\hat{\mathbf{z}}_i) - g(\hat{\mathbf{z}}_j)\| \leq 2\|D\|_{\mathrm{op}}\|\hat{\mathbf{z}}_i - \hat{\mathbf{z}}_j\|/s_{\min}$.

Step 2 (triangle inequality). The triangle inequality gives $\|\mathbf{x}_i - \mathbf{x}_j\| \leq \epsilon_i + \epsilon_j + \|g(\hat{\mathbf{z}}_i) - g(\hat{\mathbf{z}}_j)\|$. Combining with Step 1 proves the claim. □

Uniform relations would give every candidate the same cosine as the anchor slot, namely one, and hence $\hat{\mathbf{z}}_j = \hat{\mathbf{z}}_i$. The proposition rules this out for every candidate whose features differ from those of the anchor by more than $\epsilon_i + \epsilon_j$.

A.7 Proof of Proposition 3.4

Proposition 3.4 is the basis of FSKA. The slow update is a linear filter. It passes slow trends and damps oscillations whose period is short compared with $1/(1-\mu)$ steps. Steps 1–2 separate the shared trend from the oscillation, Steps 3–5 treat each part, and Step 6 carries the bound to the relation targets.

Proposition 3.4 (restated). *Let the source graphs be visited in turn, $\delta_t = \mathbf{g} + \mathbf{e}_{s(t)}$ with $\sum_s \mathbf{e}_s = \mathbf{0}$, and $\overline{\theta}_t = \mu\overline{\theta}_{t-1} + (1-\mu)\theta_t$ with $\overline{\theta}_0 = \theta_0$. Then $\overline{\theta}_t$ follows $\mathbf{g}$ with a lag of $\mu/(1-\mu)$ steps, and in the steady state the root mean square of its period-S oscillation over one period is at most ρ times that of θ_t.*

Proof. **Step 1 (decomposition).** $\theta_t = \theta_0 + t\,\mathbf{g} + \mathbf{p}_t$ with $\mathbf{p}_t = \sum_{j \leq t} \mathbf{e}_{s(j)}$. A full pass adds $\sum_s \mathbf{e}_s = \mathbf{0}$, so $\mathbf{p}_{t+S} = \mathbf{p}_t$.

Step 2 (linearity). Write F for the map from a sequence a_t to $\bar{a}_t = \mu\bar{a}_{t-1} + (1-\mu)a_t$ with $\bar{a}_0 = a_0$. It is linear and leaves constants unchanged, so $\overline{\theta}_t = \theta_0 + F[t\mathbf{g}]_t + F[\mathbf{p}]_t$.

Step 3 (the shared part). By induction on t, $F[t]_t = t - \frac{\mu}{1-\mu}(1-\mu^t)$. The shared part therefore passes with a lag that tends to $\mu/(1-\mu)$ steps.

Step 4 (the oscillation). Expand $\mathbf{p}_t = \mathbf{m} + \sum_{k=1}^{S-1} \mathbf{a}_k e^{\mathrm{i}2\pi kt/S}$. Substituting $e^{\mathrm{i}\omega t}$ into the recursion shows that F multiplies it by $H(\omega) = (1-\mu)/(1-\mu e^{-\mathrm{i}\omega})$, up to a transient that decays as μ^t. Since $H(0) = 1$, the steady state is $F[\mathbf{p}]_t = \mathbf{m} + \sum_k H(2\pi k/S)\,\mathbf{a}_k e^{\mathrm{i}2\pi kt/S}$.

Step 5 (the bound). For $1 \leq k \leq S-1$:

$$|H(2\pi k/S)|^2 = \frac{(1-\mu)^2}{1 - 2\mu\cos(2\pi k/S) + \mu^2} = \frac{(1-\mu)^2}{(1-\mu)^2 + 4\mu\sin^2(\pi k/S)} \leq \rho^2,$$

because $\sin^2(\pi k/S) \geq \sin^2(\pi/S)$. Parseval's identity over one period gives:

$$\frac{1}{S}\sum_{t=1}^{S} \|F[\mathbf{p}]_t - \mathbf{m}\|_2^2 = \sum_k |H(2\pi k/S)|^2 \|\mathbf{a}_k\|_2^2 \leq \rho^2\,\frac{1}{S}\sum_{t=1}^{S} \|\mathbf{p}_t - \mathbf{m}\|_2^2.$$

Step 6 (relation targets). If the embeddings are L-Lipschitz in the parameters and their rows have norm at least c, Lemma A.1 bounds the KL divergence between the targets at two parameter values by $8L/(\tau c)$ times their distance. The oscillation of the targets is therefore at most $8L/(\tau c)$ times the oscillation of the parameters that produce them. □

Remarks. The bound is attained when the oscillation has a single frequency, as for $S = 2$. For an arbitrary update sequence, F still multiplies every frequency component by $H(\omega)$, so components at the domain frequencies $2\pi k/S$ are attenuated at least by ρ. A per-step form also holds: $\overline{\theta}_t - \overline{\theta}_{t-1} = (1-\mu)(\theta_t - \overline{\theta}_{t-1})$, so $\mathrm{KL}(q(\overline{\theta}_t)\|q(\overline{\theta}_{t-1})) \leq \frac{8L}{\tau c}(1-\mu)\|\theta_t - \overline{\theta}_{t-1}\|_2$. Keeping ρ fixed as S grows requires $1-\mu$ to shrink roughly in proportion to $1/S$.

A.8 Relation Change and the Effectiveness of PARR

Propagation flattening on real graphs. We first verify that the flattening characterized by Theorem 3.2 appears under the candidate construction of FLO, which also contains random-walk and uniformly sampled candidates. For each node, we compute its relation distribution over a candidate set of FLO (Eq. (5), $\tau = 0.2$, $\mathbf{W} = \mathbf{I}$) from $\mathbf{X}$, $\widehat{\mathbf{A}}\mathbf{X}$, and $\widehat{\mathbf{A}}^2\mathbf{X}$, and divide its KL divergence to the uniform distribution by the logarithm of the number of valid candidate slots, averaged over five draws of the candidate sets. Without any training, one propagation step reduces this divergence for at least 99.5% of the nodes on every graph, and a second step reduces it further for at least 99.9%. After two steps, the median divergence decreases by a factor of 2.0–2.3 on Cora, CiteSeer, PubMed, and CS, and by factors of 16 and 19 on Photo and Computers.

Relation change and the gain of PARR. We then test whether this relation change predicts when PARR helps. In the proof of Theorem 3.3(iii), R is bounded by $\mathbb{E}\ \mathrm{KL}(P\|Q_B)$, the divergence between relations computed with and without propagation. We measure this quantity on the six graphs of our experiments. For each graph, we take the preprocessed features $\mathbf{X}$ of Eq. (2) with $\mathbf{W} = \mathbf{I}$, sample 4,096 anchor nodes and their candidate sets as in FLO (the anchor, three random walks of length three, and 16 uniformly sampled nodes), and compute relation distributions (Eq. (5), $\tau = 0.2$) from $\widehat{\mathbf{A}}^K\mathbf{X}$ and from $\mathbf{X}$. Table 5 reports the average KL divergence for $K = 1, 2$ together with the accuracy drop on each target graph when both relation learning objectives are removed (Table 3). Propagation changes the relations most on CiteSeer and Cora, for both K, and these are the two target graphs on which relation learning contributes most; on the other four graphs, both the relation change and the drop are small (Figure 5). Across the six graphs, the Pearson correlation between the relation change and the drop is 0.89 for $K = 1$ and 0.81 for $K = 2$, and it stays between 0.86 and 0.92 for $K = 1$ when any one graph is removed. Since Q_B is a function of ξ, Theorem 3.3(i) gives $\mathbb{E}\ \mathrm{KL}(P\|Q_B) = R + \mathbb{E}\ \mathrm{KL}(\overline{P}\|Q_B) \geq R$, so the measured change bounds R from above and also contains the difference between the attribute-predictable relation and the relation computed from the input features; it is a graph-level measurement rather than an estimate of R_s. PARR therefore helps most on the graphs where propagation changes the relations most, which links its gain to the propagation-induced part R analyzed in Theorems 3.2 and 3.3.

Table 5: Relation change caused by propagation, $\mathbb{E}\ \mathrm{KL}(P\|Q_B)$, and the accuracy drop (%) when relation learning is removed.

	Cora	CiteSeer	PubMed	CS	Photo	Computers
Relation change, $K = 1$	0.70	**0.77**	0.41	0.44	0.53	0.53
Relation change, $K = 2$	**1.10**	1.09	0.83	0.67	0.61	0.61
Drop without $\mathcal{L}_{\mathrm{kd}}$	2.57	**5.98**	0.11	0.49	0.28	0.09

A.9 Relation Invariance

The two branches exchange relation distributions instead of embeddings (Section 3.1). This works because, for nonzero embedding rows, cosine relation distributions are invariant to a common orthogonal change of basis and to independent positive row scaling:

$$p_{\mathbf{D}_\nu \mathbf{Z}\mathbf{R}} = p_{\mathbf{Z}}, \qquad \mathbf{R}^\top\mathbf{R} = \mathbf{I}, \quad \mathbf{D}_\nu = \mathrm{diag}(\nu_1, \ldots, \nu_n), \quad \nu_i > 0.$$

Both operations leave every cosine in Eq. (5) unchanged, since $\cos(\nu_i\mathbf{R}^\top\mathbf{z}_i, \nu_c\mathbf{R}^\top\mathbf{z}_c) = \cos(\mathbf{z}_i, \mathbf{z}_c)$ for $\nu_i, \nu_c > 0$. Relation learning across the two branches therefore does not require their embeddings to share coordinates. Each slow copy tracks its own online branch, and no parameters are averaged between the graph branch and the attribute branch.

B Pre-training Algorithm and Implementation

This appendix gives the pre-training procedure of FLO (Algorithm 1), its hyperparameters (Table 6), the details of candidate sampling and input masking, and its computational cost (Table 7).

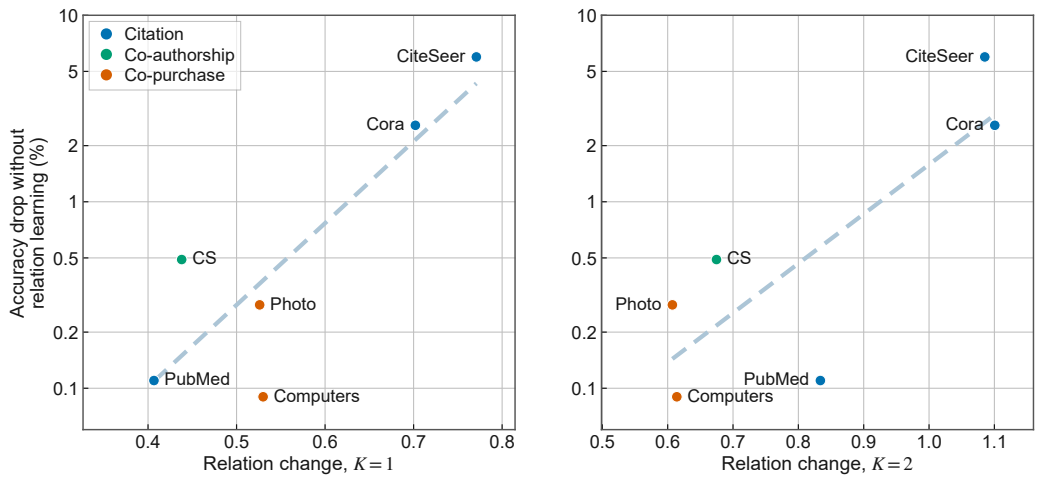


Figure 5: Relation change caused by propagation and the accuracy drop (%) when relation learning is removed, for each target graph. The values are those of Table 5. The vertical axis is logarithmic, and the dashed lines are least-squares fits of the logarithm of the drop.

Algorithm 1 Multi-domain pre-training of FLO

Input: source graphs $\{(\mathbf{A}_s, \mathbf{X}_s^{\text{raw}})\}_{s=1}^{S}$, epochs E, warm-up length T_w, relation weight λ, momentum μ, temperature τ.
Output: graph encoder $f_G(\cdot\,; \theta_G)$.

$\mathbf{X}_s \leftarrow$ Eq. (2) for $s = 1, \ldots, S$; initialize $\theta_G, \theta_M, D_G, D_M$; $\overline{\theta}_a \leftarrow \theta_a$ for $a \in \{G, M\}$
for $e = 1, \ldots, E$ **do**
  $r_e \leftarrow \min(1, e/T_w)$
  for $s = 1, \ldots, S$ **do**
    draw masks $\mathbf{M}_G, \mathbf{M}_M$; $\mathbf{Z}_G, \mathbf{Z}_M \leftarrow$ Eq. (3); $\mathcal{L}_{\text{rec}}^G, \mathcal{L}_{\text{rec}}^M \leftarrow$ Eq. (4)
    sample anchors $\mathcal{I}$ and candidate sets $\{\mathcal{C}_i\}_{i \in \mathcal{I}}$
    $p_G, p_M \leftarrow$ Eq. (5) on $\mathbf{Z}_G, \mathbf{Z}_M$; $q_G, q_M \leftarrow$ Eq. (5) on the slow copies, without gradients
    $\mathcal{L} \leftarrow$ Eq. (8) with $\mathcal{L}_{\text{kd}}^G, \mathcal{L}_{\text{kd}}^M$ from Eq. (6)
    update $\theta_G, \theta_M, D_G, D_M$ by one AdamW step on $\mathcal{L}$
    $\overline{\theta}_a \leftarrow \mu\,\overline{\theta}_a + (1-\mu)\,\theta_a$ for $a \in \{G, M\}$ Eq. (7)
return $f_G(\cdot\,; \theta_G)$

Algorithm. Each epoch visits the S source graphs in order and takes one update on each. The masks of the two branches are drawn independently. Both online branches and both slow copies use the same candidate sets, and each slow copy takes the same masked input as its online branch. After pre-training, the graph encoder is frozen and applied to the target graph without masks, and only the downstream readout is fitted.

Candidate sampling and masks. Three random walks each contribute every visited node over three steps. Sixteen additional nodes are drawn uniformly with replacement. Repeated nodes occupy separate slots; a global draw may equal the anchor or a neighbor. A walk without a valid next neighbor contributes an invalid slot, excluded from the softmax. Input masking samples a full $n_s \times d$ Bernoulli array independently for each branch; columns are not shared. The masks are not divided by their keep probability. The self slot remains valid. Row normalization uses an epsilon floor for zero-norm inputs. When only one direction of relation learning is used in the ablation, its weight is doubled, so that the total weight of relation learning is the same as in FLO.

GCN attribute branch. This variant in Table 3 uses a two-layer GCN with the widths and activation of the graph encoder as the attribute branch; everything else is as in FLO.

Table 6: Hyperparameters of FLO. The pre-training hyperparameters are shared across all target graphs; the last two rows apply only to few-shot node classification.

Component	Parameter	Value
Input	Common dimension d	1,024
Both branches	Hidden width; output width	512; 1,024
Graph branch	Architecture	GCN–PReLU–GCN
Attribute branch	Architecture	Linear–GELU–Linear
Decoders	Architecture	Separate linear maps
Mask	Elementwise keep probability	0.40
Reconstruction	Coefficient γ for mean squared row distance	$1/256$
Candidates	Self + walk visits + global draws	$1 + 3 \times 3 + 16 = 26$
Anchors	Maximum per source step N	4,096
Relations	Temperature τ; weight λ	0.2; 0.5
Warm-up	T_w	100 epochs
EMA	Momentum μ; update frequency	0.99; each source step
Optimizer	Algorithm; learning rate; weight decay	AdamW; 10^{-3}; 10^{-4}
Training	Epochs; pre-training seed	3,000; 0
Few-shot	Singular components kept for the target graph	Cora 16, CiteSeer 8, PubMed 8, CS 256, Photo 64, Computers 512
	Propagation steps with $\widehat{\mathbf{A}}$	Cora 5, others 1

Computational cost. In the main configuration, input and output widths both equal $d = 1024$. With hidden width h and $|\mathcal{C}_i| = 26$ candidate slots, a source step costs $O(n_s dh + n_s d^2 + |\mathcal{T}_s|(d+h))$ for the online branches, their slow copies, and reconstruction, plus $O(|\mathcal{I}||\mathcal{C}_i|d)$ for the relation distributions. The singular value decomposition is a one-time preprocessing cost. Transfer requires only the two-layer online GNN.

Table 7 reports the measured cost of all methods. The pre-training time is the median over 20 epochs, and the forward time is the median over 20 passes after 3 warm-up passes. FLO pre-trains about as fast as DGI and MDGPT, at 0.14–0.19 s per epoch, and faster than the other multi-domain graph pre-training methods. Its peak memory of 5.6 GB is the second lowest among all methods and less than a third of that of MDGPT, MDGFM, and BRIDGE. The extra cost of FLO lies in pre-training: the attribute branch and the two decoders raise its trainable parameters to 4.20M. At inference, FLO keeps only the graph encoder, a two-layer GNN with 1.05M parameters, a quarter of this number. Its forward pass is faster than those of MDGPT, SAMGPT, MDGFM, and BRIDGE on every target graph.

C Related Work

Graph pre-training and graph foundation models. Graph pre-training trains GNNs with self-supervised objectives, which are either contrastive (Veličković et al., 2019; You et al., 2020; Zhu et al., 2020; Thakoor et al., 2022) or generative (Hou et al., 2022). Early studies transfer structural patterns across graphs (Qiu et al., 2020) or pre-train a GNN at both the node and the graph level (Hu et al., 2020). Most of these methods pre-train within a single domain. Multi-domain graph pre-training has recently attracted growing interest as a route toward GFMs (Mao et al., 2024; Liu et al., 2025). One line of work represents node attributes as text and unifies them with language models. OFA casts node, link, and graph classification into a single prompt-graph format (Liu et al., 2024), ZeroG encodes node attributes and class semantics with a language model for zero-shot transfer across datasets (Li et al., 2024), and UniGraph pre-trains a cascaded language model and GNN with masked autoencoding (He et al., 2025). These methods rely on text to unify graphs from different domains.

Text-free multi-domain graph pre-training. For graphs without text, feature unification methods map the node attributes of different domains into a shared space. FUG captures transferable distributions among attributes with a dimensional encoder (Zhao et al., 2024b), and STAGE encodes statistical dependencies between attributes (Shen et al., 2025). PatchNet decomposes node attributes into learnable graph patches (Sun et al., 2025b), and TIG mines feature-dimension relations shared

Table 7: Computational cost. Top: pre-training time per epoch (s) with each graph held out as the target, peak GPU memory during pre-training, and trainable parameters. Bottom: forward time (ms) of the frozen encoder on each target graph and parameters at inference. All methods are measured on the same NVIDIA RTX 5090 GPU.

Method	Cora	CiteSeer	PubMed	CS	Photo	Computers	Memory (GB)	Params (M)
Pre-training time per epoch (s)								
DGI	0.189	0.189	0.145	0.148	0.177	0.159	19.6	0.39
GRACE	0.747	1.063	0.532	0.560	0.813	0.739	12.9	0.39
BGRL	0.102	0.102	0.099	0.097	0.076	0.064	3.5	0.53
FUG	0.167	0.165	0.152	0.133	0.138	0.110	10.6	10.49
TIG	0.231	0.230	0.209	0.201	0.183	0.152	14.7	3.81
GFT	15.30	15.15	11.98	10.90	11.88	8.59	14.2	12.80
MDGPT	0.179	0.165	0.126	0.128	0.153	0.142	25.2	0.15
SAMGPT	0.345	0.336	0.281	0.280	0.325	0.282	7.1	0.28
MDGFM	1.871	1.854	1.156	1.231	1.765	1.512	20.0	0.15
BRIDGE	0.784	0.809	0.586	0.600	0.722	0.661	27.2	0.15
FLO	0.188	0.178	0.150	0.147	0.169	0.136	5.6	4.20
Inference time per forward pass (ms)								
DGI	1.29	1.35	3.18	3.14	1.98	2.99	—	0.13
GRACE	1.12	1.18	3.33	3.87	3.30	6.03	—	0.26
BGRL	1.42	1.45	3.58	4.05	3.49	6.22	—	0.26
FUG	3.54	3.86	18.56	19.95	13.74	26.29	—	10.49
TIG	1.36	2.25	8.53	9.88	7.63	15.71	—	1.71
GFT	1.34	1.46	10.96	13.00	10.63	21.57	—	2.36
MDGPT	4.30	4.33	20.45	31.28	34.88	74.03	—	0.14
SAMGPT	4.34	4.36	20.61	31.55	35.40	74.67	—	0.14
MDGFM	4.40	4.39	20.66	32.00	46.15	95.42	—	0.14
BRIDGE	4.29	4.29	20.40	41.13	46.52	95.48	—	0.14
FLO	2.61	2.80	13.54	12.04	7.63	15.10	—	1.05

across graphs (Zhao et al., 2026). LEDA aligns the node attributes of different domains with a shared latent distribution (Shan et al., 2026b), and ALL-IN builds covariance-based representations in a shared random space (Eliasof et al., 2026). MUG applies universal pre-training to heterogeneous graphs through a dimension-aware encoder (Shan et al., 2026a). Some methods avoid a shared attribute space: GraphControl pre-trains on graph structure and injects the node attributes of the target graph as conditional inputs (Zhu et al., 2024), and GraphAny solves linear graph models in closed form on each new graph and combines them with attention (Zhao et al., 2025). Other methods add learnable alignment parameters to the model. GCOPE connects the source graphs through domain-specific virtual nodes (Zhao et al., 2024a), MDGPT learns domain tokens (Yu et al., 2024), and SAMGPT adds structure tokens (Yu et al., 2025). BRIDGE learns domain-invariant feature aligners (Yuan et al., 2025), and MF-GIA derives domain-specific transformations from gradient fingerprints (Zhuo & Luo, 2026). Another line seeks structural knowledge shared across domains. GFT treats computation trees as a transferable vocabulary (Wang et al., 2024), and RiemannGFM learns a vocabulary of trees and cycles in Riemannian space (Sun et al., 2025a). MDGFM aligns topologies across domains (Wang et al., 2025a), and AnyGraph routes graphs to a mixture of graph experts (Xia & Huang, 2026). GraphGlue merges graphs from different domains into one smooth Riemannian manifold (Sun et al., 2026), SPG learns a shared prototype geometry that guides propagation (Yang et al., 2026), and ProGFM learns propagation patterns shared across edges and feature dimensions (Wang et al., 2026). Beyond node classification, GFT and GIT pre-train a single model for node-, edge-, and graph-level tasks (Wang et al., 2024; 2025b), and holographic node representations serve tasks of any order after pre-training (Bevilacqua et al., 2025). Most of these methods align domains through the input features or through parameters trained jointly with a message-passing encoder.

Relational knowledge distillation. Knowledge distillation trains a student to match the softened outputs of a teacher (Hinton et al., 2015). Relational knowledge distillation transfers the relations among samples instead of the outputs of individual samples. RKD matches pairwise distances and

triplet angles (Park et al., 2019). LSP matches the local structure of each node between a teacher and a student graph convolutional network (Yang et al., 2020). SEED and ReSSL match the similarity distribution of each sample over a set of other samples (Fang et al., 2021; Zheng et al., 2021). Deep mutual learning removes the fixed teacher and lets a cohort of networks learn from the predictions of each other (Zhang et al., 2018). FreeKD lets two shallow GNNs exchange knowledge in directions chosen by reinforcement learning (Feng et al., 2022). On graphs, GLNN distills a trained GNN into a multilayer perceptron that needs no message passing at inference (Zhang et al., 2022). NOSMOG further distills the similarities between node representations (Tian et al., 2023), and TINED injects the parameters of the teacher and distills its Dirichlet energy (Zhou et al., 2025).

Slowly updated targets. Averaging the iterates of stochastic optimization accelerates its convergence (Polyak & Juditsky, 1992). Momentum teachers keep an exponential moving average of the parameters of a student network and use it to produce targets, as in Mean Teacher (Tarvainen & Valpola, 2017), MoCo (He et al., 2020), and BYOL (Grill et al., 2020). BGRL applies this design to graph self-supervised learning (Thakoor et al., 2022), and ReSSL uses it to produce relation targets (Zheng et al., 2021). Recent studies scale the momentum of the average with the batch size (Busbridge et al., 2023) and show that the averaged weights generalize better and are more robust to label noise (Morales-Brotons et al., 2024). Averaging the weights along the training trajectory improves generalization to unseen domains (Cha et al., 2021). Fast and slow weights also appear in optimization and continual learning. Lookahead moves a set of slow weights toward the fast weights every k steps (Zhang et al., 2019), and CLS-ER keeps short-term and long-term semantic memories of a working model, following complementary learning systems theory (McClelland et al., 1995; Arani et al., 2022).

D Experimental Details and Additional Results

This appendix describes the datasets, baselines, and evaluation protocols of Section 4. It then reports the few-shot results on every target graph, the computation of the loss curves in Figure 4(b), the effect of the pre-training budget, and the sensitivity of FLO to its hyperparameters.

D.1 Datasets

The statistics of all datasets are summarized in Table 8. Cora, CiteSeer, and PubMed are citation networks, where nodes are papers, edges are citations, and node features are bag-of-words vectors. CS is a co-authorship network, where nodes are authors and edges indicate co-authored papers. Photo and Computers are co-purchase networks from Amazon, where nodes are products and edges indicate that two products are frequently bought together. In the leave-one-domain-out setting, each graph is held out in turn as the target graph and the other five graphs are used for pre-training. For link prediction we use the co-authorship network Physics, the Wikipedia hyperlink network WikiCS, and the citation network DBLP, and for graph classification we use COIL-RAG, ENZYMES, and PROTEINS from the TUDataset collection; for both tasks, the encoder is pre-trained on all six node classification datasets.

All datasets are loaded with PyTorch Geometric: Cora, CiteSeer, and PubMed through `Planetoid`; Photo and Computers through `Amazon`; CS and Physics through `Coauthor`; WikiCS through `WikiCS`; DBLP through `CitationFull`; and COIL-RAG, ENZYMES, and PROTEINS through `TUDataset`.

D.2 Baselines

Graph self-supervised learning methods. DGI (Veličković et al., 2019) maximizes the mutual information between node representations and a summary vector of the graph, and takes the representations of a graph with shuffled node features as negative samples. GRACE (Zhu et al., 2020) builds two views of a graph by removing edges and masking features, and contrasts the representations of each node across the two views. BGRL (Thakoor et al., 2022) predicts the representations of one augmented view from another view and takes its targets from an encoder updated by an exponential moving average, so it needs no negative samples. Since these methods require a common

Table 8: Statistics of the datasets. Edges are counted once per undirected pair; for graph classification, the numbers of nodes and edges are averaged over the graphs of each dataset.

Dataset	Graphs	Nodes	Edges	Features	Classes	Domain
Node classification						
Cora	1	2,708	5,278	1,433	7	Citation
CiteSeer	1	3,327	4,552	3,703	6	Citation
PubMed	1	19,717	44,324	500	3	Citation
CS	1	18,333	81,894	6,805	15	Co-authorship
Photo	1	7,650	119,081	745	8	Co-purchase
Computers	1	13,752	245,861	767	10	Co-purchase
Link prediction						
Physics	1	34,493	247,962	8,415	–	Co-authorship
WikiCS	1	11,701	215,603	300	–	Web
DBLP	1	17,716	52,867	1,639	–	Citation
Graph classification						
COIL-RAG	3,900	3.01	3.02	64	100	Computer vision
ENZYMES	600	32.63	62.14	21	6	Bioinformatics
PROTEINS	1,113	39.06	72.82	32	2	Bioinformatics

input dimension, we reduce the features of every graph to a common dimension by singular value decomposition and pre-train each method on the source graphs for 1,000 epochs.

Feature unification methods. FUG (Zhao et al., 2024b) designs a dimension encoder with contrastive constraints that emulates the basis transformation of principal component analysis. The encoder maps graphs with different feature dimensions into a shared space without reshaping their features, and a global uniformity constraint replaces negative sampling in graph contrastive pre-training. TIG (Zhao et al., 2026) mines transfer-invariant metadata from node features, namely feature-dimension relations shared across graphs. It turns features of different dimensions into a unified structural representation and learns graph knowledge from it by self-supervised learning.

Multi-domain graph pre-training methods. GFT (Wang et al., 2024) treats computation trees as transferable patterns. It learns a tree vocabulary by reconstructing computation trees during pre-training and casts downstream tasks as tree classification. MDGPT (Yu et al., 2024) aligns the features of source domains with learnable domain tokens and adapts to the target domain with a unifying prompt and a mixing prompt. SAMGPT (Yu et al., 2025) further introduces structure tokens that modulate message passing in each domain to align structural patterns, and adapts with holistic and specific prompts. MDGFM (Wang et al., 2025a) aligns topologies across domains. It refines each graph with a structure learned from node features and propagated features, and pre-trains by contrasting the original and refined graphs. BRIDGE (Yuan et al., 2025) aligns multi-domain features with domain-invariant aligners during pre-training. It transfers source knowledge to the target through a lightweight mixture of experts and bounds the generalization error of this transfer.

All baselines are run with their official code (Table 9) and the settings recommended in their papers. Following their original settings, MDGPT, SAMGPT, MDGFM, and BRIDGE reduce the feature dimension of each graph to 50 with principal component analysis, and GFT obtains 768-dimensional features with singular value decomposition. Since TIG builds a node–feature graph whose edge weights are the feature values, the features of the cross-task datasets that take negative values or values above one (WikiCS, ENZYMES, and PROTEINS) are min–max scaled to $[0, 1]$ in each dimension before its preprocessing. For all methods, the representations produced by the pre-trained encoder are used for downstream evaluation.

D.3 Evaluation Protocols

Since each prior work covers only part of our evaluation scenarios, we follow the corresponding prior protocol in each scenario, and all encoders are frozen during downstream evaluation. For node

Table 9: Code of the baselines.

Method	Code	Method	Code
DGI	github.com/PetarV-/DGI	GFT	github.com/Zehong-Wang/GFT
GRACE	github.com/CRIPAC-DIG/GRACE	MDGPT	anonymous.4open.science/r/MDGPT
BGRL	github.com/nerdslab/bgrl	SAMGPT	github.com/blue-soda/SAMGPT
FUG	github.com/hedongxiao-tju/FUG	MDGFM	github.com/wbkzwqtzw/MDGFM
TIG	github.com/hedongxiao-tju/TIG	BRIDGE	github.com/RingBDStack/BRIDGE

classification, following FUG (Zhao et al., 2024b), we train a logistic regression classifier on the ℓ_2-normalized node representations. The nodes of the target graph are split into 20% and 80%; the regularization strength of the classifier is selected on five random 50%/50% training/validation splits of the 20%, which gives a 10%/10%/80% split, and the accuracy on the remaining 80% is averaged over three random splits. For few-shot node classification, following TIG (Zhao et al., 2026), we use the same 500 sampled k-shot tasks of each target graph for all methods and train a logistic regression classifier on the k labeled nodes of each class. For link prediction, the representations of each method are standardized column-wise, and an edge classifier with the same architecture is trained for all methods with early stopping on the validation edges; on Physics, WikiCS, and DBLP we use five random 85%/5%/10% splits of the edges; message passing uses only the training edges. For graph classification, the node representations of a graph are pooled by concatenating mean and max pooling, and the same two-layer classifier is trained on 20% of the graphs and tested on the remaining 80%, over 50 stratified splits. The pre-training hyperparameters of FLO are shared across all target graphs (Table 6), and FLO is evaluated at the final epoch of pre-training. The random seed is fixed to 0 for reproducibility.

Environment. All experiments run on one Linux server, and all methods are timed on the same type of GPU (Table 10).

Table 10: Experimental environment.

Component	Specification
Operating system	Ubuntu 22.04.5 LTS, Linux kernel 6.8.0
CPU	Intel Xeon Silver 4410Y, 48 threads
Memory	128 GB
GPU	NVIDIA GeForce RTX 5090, 32 GB
Software	Python 3.10.12, PyTorch 2.11.0 with CUDA 12.8, PyTorch Geometric 2.8.0 NumPy 2.2.6, scikit-learn 1.7.2

D.4 Few-Shot Node Classification

This subsection details the few-shot evaluation of Section 4.2 and reports the results on every target graph. The pre-trained encoder of FLO is the same as in node classification. In the few-shot setting, the number of singular components kept for the target graph in Eq. (2) is set for each target graph (Table 6), and the remaining dimensions are padded with zeros. Following GGD (Zheng et al., 2022), we further enhance the similarity relations between connected nodes by propagating the frozen representations over the target graph with $\widehat{\mathbf{A}}$ for a number of steps set for each target graph (Table 6). Tables 11, 12, and 13 report the results of all methods on each target graph; the results on Cora, CiteSeer, and CS are plotted in Figure 3. FLO achieves the best average accuracy in all three settings, and it ranks first on four, three, and three of the six target graphs with one, three, and five labeled nodes per class, respectively.

D.5 Pre-training Loss Curves

This subsection describes how the loss curves in Figure 4(b) are computed. Figure 4(b) uses the first 40 epochs of the same leave-one-domain-out runs as the main experiments. For FLO, each point is the per-epoch mean over the five source graphs of the graph encoder's objective, $\mathcal{L}^G_{\mathrm{rec}} + \lambda r_e \mathcal{L}^G_{\mathrm{kd}}$; BRIDGE and SAMGPT use their logged pre-training objectives. For each method and target graph

Table 11: Accuracy (%) of 1-shot node classification with standard deviations over the 500 tasks shared by all methods. The best and second-best results are in bold and underlined.

Method	Cora	CiteSeer	PubMed	CS	Photo	Computers	Avg.
DGI	33.09$_{\pm 5.51}$	33.79$_{\pm 5.60}$	47.10$_{\pm 6.45}$	71.82$_{\pm 6.36}$	56.42$_{\pm 8.15}$	42.48$_{\pm 9.90}$	47.45
GRACE	28.13$_{\pm 4.46}$	24.20$_{\pm 3.88}$	39.81$_{\pm 5.61}$	47.23$_{\pm 5.77}$	42.35$_{\pm 7.61}$	38.77$_{\pm 7.31}$	36.75
BGRL	34.19$_{\pm 4.42}$	25.04$_{\pm 3.75}$	43.37$_{\pm 5.66}$	57.58$_{\pm 5.84}$	60.32$_{\pm 10.14}$	50.95$_{\pm 11.17}$	45.24
FUG	46.40$_{\pm 8.37}$	36.07$_{\pm 6.81}$	52.23$_{\pm 9.47}$	**78.02**$_{\pm 5.88}$	63.62$_{\pm 9.77}$	51.27$_{\pm 12.82}$	54.60
TIG	48.88$_{\pm 8.80}$	44.71$_{\pm 9.02}$	50.94$_{\pm 10.00}$	75.28$_{\pm 6.22}$	63.52$_{\pm 10.47}$	49.47$_{\pm 13.08}$	55.47
GFT	27.25$_{\pm 4.78}$	26.78$_{\pm 4.31}$	45.36$_{\pm 6.00}$	63.49$_{\pm 6.42}$	20.01$_{\pm 4.37}$	14.62$_{\pm 5.12}$	32.92
MDGPT	42.49$_{\pm 7.94}$	45.13$_{\pm 8.81}$	**55.34**$_{\pm 8.32}$	72.98$_{\pm 6.61}$	55.92$_{\pm 8.31}$	41.05$_{\pm 8.44}$	52.15
SAMGPT	44.81$_{\pm 8.42}$	41.52$_{\pm 8.15}$	45.42$_{\pm 8.66}$	76.48$_{\pm 6.17}$	59.17$_{\pm 8.63}$	46.73$_{\pm 9.64}$	52.35
MDGFM	42.43$_{\pm 6.05}$	38.84$_{\pm 7.54}$	48.21$_{\pm 7.93}$	74.11$_{\pm 6.55}$	63.43$_{\pm 8.91}$	48.86$_{\pm 9.56}$	52.65
BRIDGE	42.77$_{\pm 6.37}$	40.70$_{\pm 7.61}$	53.57$_{\pm 7.78}$	75.14$_{\pm 6.36}$	53.23$_{\pm 7.21}$	38.89$_{\pm 7.11}$	50.72
FLO	**55.95**$_{\pm 8.41}$	**45.93**$_{\pm 8.99}$	55.26$_{\pm 9.59}$	77.20$_{\pm 6.24}$	**67.76**$_{\pm 9.55}$	**52.99**$_{\pm 10.76}$	**59.18**

Table 12: Accuracy (%) of 3-shot node classification with standard deviations over the 500 tasks shared by all methods. The best and second-best results are in bold and underlined.

Method	Cora	CiteSeer	PubMed	CS	Photo	Computers	Avg.
DGI	45.56$_{\pm 5.08}$	44.96$_{\pm 4.42}$	55.58$_{\pm 5.12}$	85.26$_{\pm 2.08}$	70.17$_{\pm 5.23}$	56.98$_{\pm 7.27}$	59.75
GRACE	38.59$_{\pm 4.00}$	30.63$_{\pm 3.88}$	45.68$_{\pm 4.85}$	65.49$_{\pm 3.51}$	57.22$_{\pm 6.52}$	50.64$_{\pm 6.08}$	48.04
BGRL	46.74$_{\pm 4.09}$	31.66$_{\pm 3.58}$	51.06$_{\pm 5.51}$	75.12$_{\pm 3.29}$	72.63$_{\pm 7.23}$	62.05$_{\pm 8.43}$	56.54
FUG	61.63$_{\pm 5.42}$	47.62$_{\pm 4.94}$	60.82$_{\pm 6.32}$	86.55$_{\pm 1.70}$	65.91$_{\pm 8.90}$	58.85$_{\pm 10.17}$	63.56
TIG	58.53$_{\pm 6.17}$	56.39$_{\pm 5.03}$	56.48$_{\pm 7.81}$	83.63$_{\pm 3.04}$	68.24$_{\pm 10.30}$	56.31$_{\pm 11.03}$	63.26
GFT	37.84$_{\pm 4.25}$	35.49$_{\pm 3.97}$	51.90$_{\pm 5.65}$	78.95$_{\pm 2.74}$	22.25$_{\pm 4.74}$	17.49$_{\pm 6.30}$	40.65
MDGPT	54.99$_{\pm 6.30}$	57.34$_{\pm 4.97}$	**63.88**$_{\pm 6.02}$	82.70$_{\pm 2.45}$	65.47$_{\pm 6.54}$	53.34$_{\pm 7.18}$	62.95
SAMGPT	58.66$_{\pm 6.25}$	53.81$_{\pm 5.20}$	52.05$_{\pm 7.82}$	85.27$_{\pm 2.00}$	68.73$_{\pm 6.40}$	60.28$_{\pm 7.14}$	63.13
MDGFM	53.68$_{\pm 4.89}$	50.41$_{\pm 4.89}$	57.41$_{\pm 6.50}$	84.93$_{\pm 2.17}$	**76.69**$_{\pm 5.22}$	**62.95**$_{\pm 6.47}$	64.34
BRIDGE	56.36$_{\pm 5.15}$	53.38$_{\pm 4.94}$	62.98$_{\pm 6.00}$	85.53$_{\pm 1.96}$	65.52$_{\pm 5.55}$	51.56$_{\pm 6.21}$	62.55
FLO	**65.78**$_{\pm 5.37}$	**57.82**$_{\pm 5.37}$	63.73$_{\pm 6.42}$	**87.86**$_{\pm 1.82}$	75.74$_{\pm 6.26}$	62.90$_{\pm 7.00}$	**68.97**

separately, we plot $(\log L_e - \min_{t\le 40} \log L_t)/(\max_{t\le 40} \log L_t - \min_{t\le 40} \log L_t)$. Thus the figure compares changes over time rather than the magnitudes of different objectives. The largest one-epoch increases are computed from the unnormalized losses as $\max_{2\le e\le 40} L_e/L_{e-1} - 1$.

D.6 Effect of Pre-training Epochs

FLO is evaluated at the final epoch of a fixed budget of 3,000 epochs, without selecting checkpoints on the target graph (Section 4.1). This subsection examines whether the results depend on this

Table 13: Accuracy (%) of 5-shot node classification with standard deviations over the 500 tasks shared by all methods. The best and second-best results are in bold and underlined.

Method	Cora	CiteSeer	PubMed	CS	Photo	Computers	Avg.
DGI	53.07$_{\pm 3.79}$	50.55$_{\pm 3.54}$	60.34$_{\pm 4.47}$	87.77$_{\pm 1.20}$	74.10$_{\pm 4.33}$	63.16$_{\pm 4.49}$	64.83
GRACE	44.50$_{\pm 3.25}$	34.72$_{\pm 3.42}$	49.95$_{\pm 4.23}$	71.96$_{\pm 2.75}$	64.24$_{\pm 5.42}$	56.71$_{\pm 5.15}$	53.68
BGRL	53.28$_{\pm 3.37}$	35.74$_{\pm 3.11}$	56.07$_{\pm 4.68}$	81.12$_{\pm 2.22}$	77.83$_{\pm 5.18}$	67.13$_{\pm 6.08}$	61.86
FUG	67.31$_{\pm 3.23}$	53.28$_{\pm 3.86}$	64.51$_{\pm 4.64}$	88.13$_{\pm 1.13}$	66.29$_{\pm 7.50}$	62.83$_{\pm 8.18}$	67.06
TIG	63.70$_{\pm 4.59}$	60.66$_{\pm 3.45}$	59.64$_{\pm 6.58}$	86.62$_{\pm 1.98}$	69.20$_{\pm 8.48}$	60.80$_{\pm 8.49}$	66.77
GFT	44.13$_{\pm 3.73}$	40.85$_{\pm 3.38}$	56.09$_{\pm 4.75}$	82.69$_{\pm 1.62}$	24.92$_{\pm 4.83}$	19.76$_{\pm 6.37}$	44.74
MDGPT	60.86$_{\pm 3.97}$	61.87$_{\pm 3.28}$	67.59$_{\pm 4.71}$	85.35$_{\pm 1.51}$	70.45$_{\pm 4.98}$	60.12$_{\pm 5.19}$	67.71
SAMGPT	64.52$_{\pm 3.90}$	59.02$_{\pm 3.43}$	55.45$_{\pm 7.08}$	87.01$_{\pm 1.24}$	72.49$_{\pm 5.25}$	66.07$_{\pm 4.45}$	67.43
MDGFM	58.91$_{\pm 3.70}$	55.27$_{\pm 3.34}$	62.87$_{\pm 5.39}$	87.28$_{\pm 1.30}$	**80.65**$_{\pm 3.25}$	**68.51**$_{\pm 3.91}$	68.92
BRIDGE	62.45$_{\pm 3.43}$	58.70$_{\pm 3.29}$	**67.64**$_{\pm 4.69}$	87.68$_{\pm 1.28}$	71.09$_{\pm 4.29}$	58.77$_{\pm 4.91}$	67.72
FLO	**68.38**$_{\pm 4.03}$	**62.03**$_{\pm 3.52}$	67.34$_{\pm 4.87}$	**89.43**$_{\pm 1.21}$	77.43$_{\pm 5.42}$	66.44$_{\pm 4.61}$	**71.84**

budget. Figure 6 shows the accuracy on each target graph at pre-training budgets from 300 to 3,000 epochs, with the frozen graph encoder and the evaluation protocol of the main experiments. From 300 to 1,000 epochs, the accuracy rises by 1.00 points on CiteSeer and 0.35 points on Cora, the two graphs on which propagation changes the relations most (Appendix A.8), and changes by at most 0.16 points on the other four graphs. Between 1,000 and 3,000 epochs, the accuracy changes by at most 0.25 points on Cora, PubMed, CS, Photo, and Computers, and by 0.53 points on CiteSeer, where it still increases slowly. FLO therefore needs no target-specific early stopping, and all reported results use the fixed budget of 3,000 epochs.

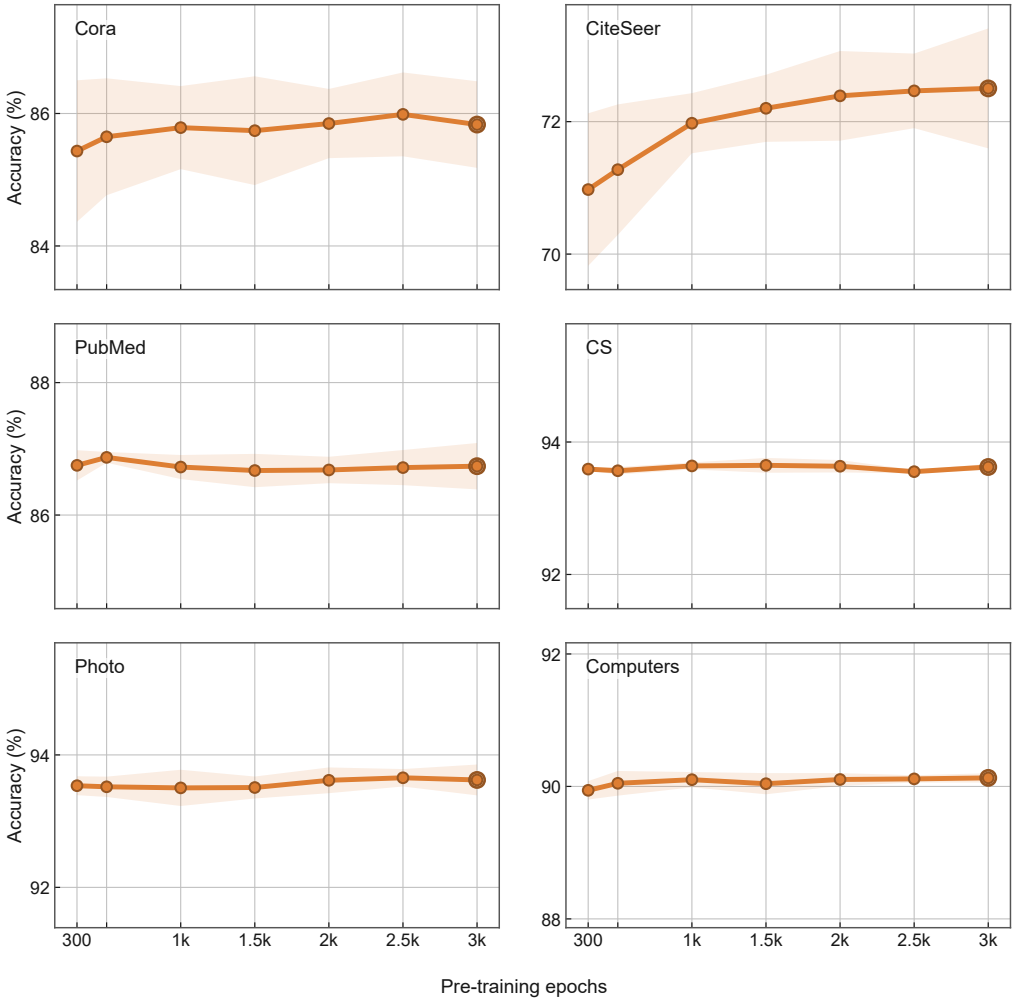


Figure 6: Accuracy (%) of node classification on each target graph at pre-training budgets of 300, 500, 1,000, 1,500, 2,000, 2,500, and 3,000 epochs. Shaded bands show the standard deviation over the three random splits of node classification, all panels share the same vertical span, and the circled point is the budget used in all other experiments.

D.7 Hyperparameter Sensitivity

We vary the temperature τ, the relation weight λ, and the momentum μ, each over five values around its default, and keep all other hyperparameters fixed. We evaluate every setting for 2,000 pre-training epochs under each of the six leave-one-domain-out configurations. As shown in Figure 7, the accuracy varies by at most 1.48 points on CiteSeer and by at most 0.86 points on the other graphs, and the default values are within 0.45 points of the best setting on every graph. Removing the warm-up of the relation weight lowers the accuracy by 0.36 points on Cora and 0.04 points on Photo, and extending it to 200 epochs changes the accuracy by at most 0.08 points.

We also vary the learning rate over $\{10^{-4}, 5 \cdot 10^{-4}, 10^{-3}, 5 \cdot 10^{-3}, 10^{-2}\}$ and the hidden width of both branches over $\{128, 256, 512, 1{,}024, 2{,}048\}$ with the same budget (Figure 7). From $5 \cdot 10^{-4}$ to 10^{-2}, the learning rate changes the accuracy by at most 0.56 points on every target graph. The smallest learning rate, 10^{-4}, lowers the accuracy by 0.83 points on Cora and 1.84 points on CiteSeer, and changes it by at most 0.49 points on the other graphs. Hidden widths from 128 to 1,024 change the accuracy by at most 0.87 points, and the default width of 512 gives the best accuracy on five of the six target graphs. A width of 2,048 lowers the accuracy on every target graph, by 3.06 points on CiteSeer and by at most 1.51 points on the others.

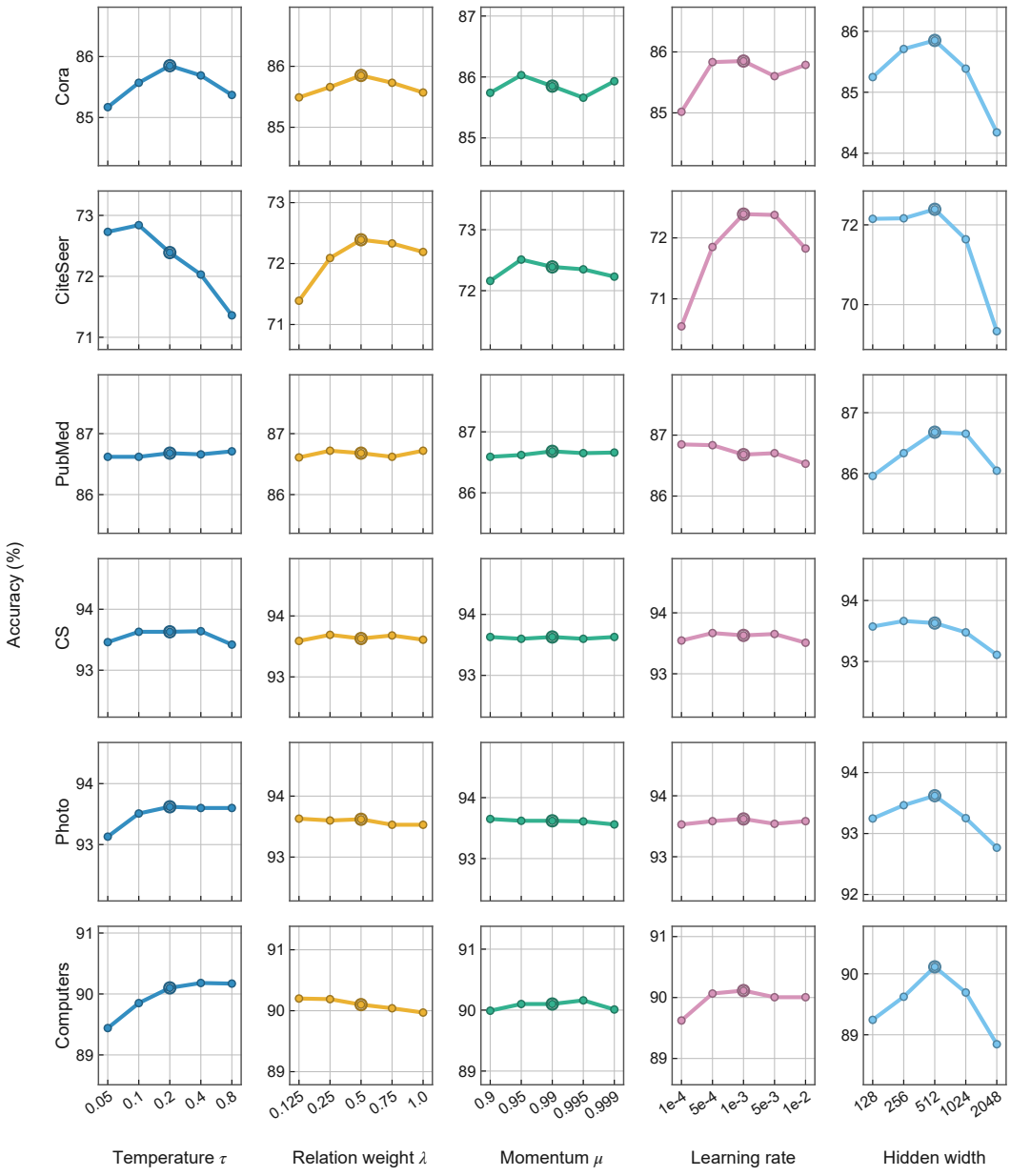


Figure 7: Accuracy (%) of node classification after 2,000 pre-training epochs when the temperature τ, the relation weight λ, the momentum μ, the learning rate, or the hidden width of both branches is varied. Each row is a target graph, and the circled point is the default value.