北京郵電大學
Beijing University of Posts and Telecommunications

华北电力大学
NORTH CHINA ELECTRIC POWER UNIVERSITY

UIC COLLEGE OF ENGINEERING
UNIVERSITY OF ILLINOIS AT CHICAGO

ICLR 2026 Oral

Multi-Domain Riemannian Graph Gluing for Building Graph Foundation Models

Li Sun, Zhenhao Huang, Silei Chen,

Lanxu Yang, Junda Ye, Sen Su, Philip S. Yu

Contact: *lsun@bupt.edu.cn* *huangzhenhao@ncepu.edu.cn*

GitHub: https://github.com/RiemannGraph/GraphGlue

A Foundational Problem in GFM

How is Knowledge Integrated or Transferred across Domains?

A Fresh Differential Geometry Perspective

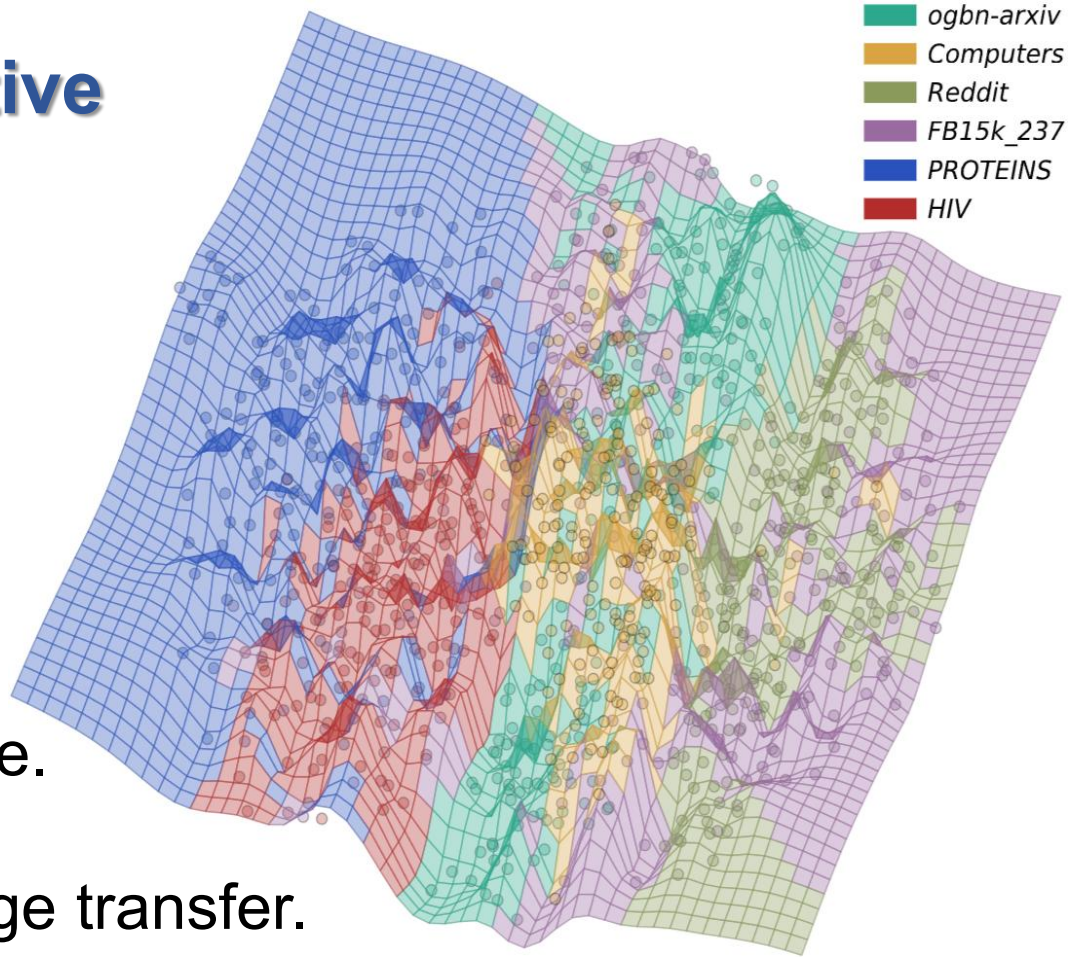
Neural Manifold Gluing

1. Each graph data is a local piece of manifold.

2. Each local geometric construction is different.

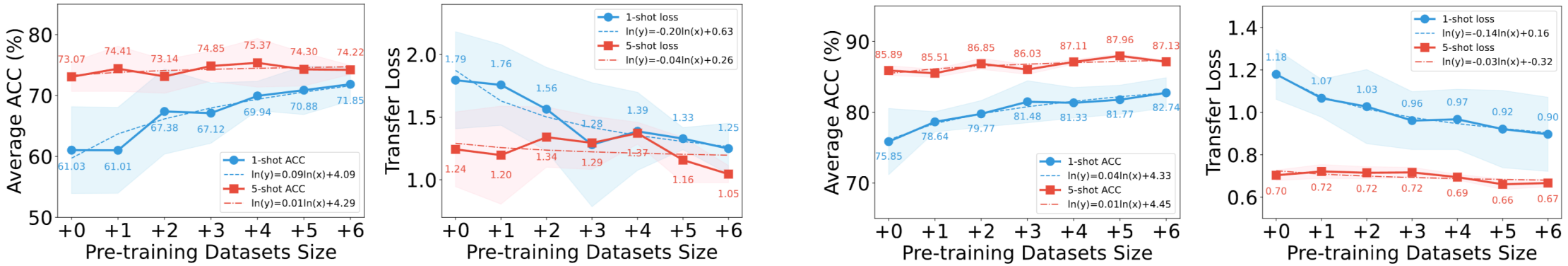
3. Glue local pieces together into a coherent whole.

4. The curvature determines the effort of knowledge transfer.



Contributions

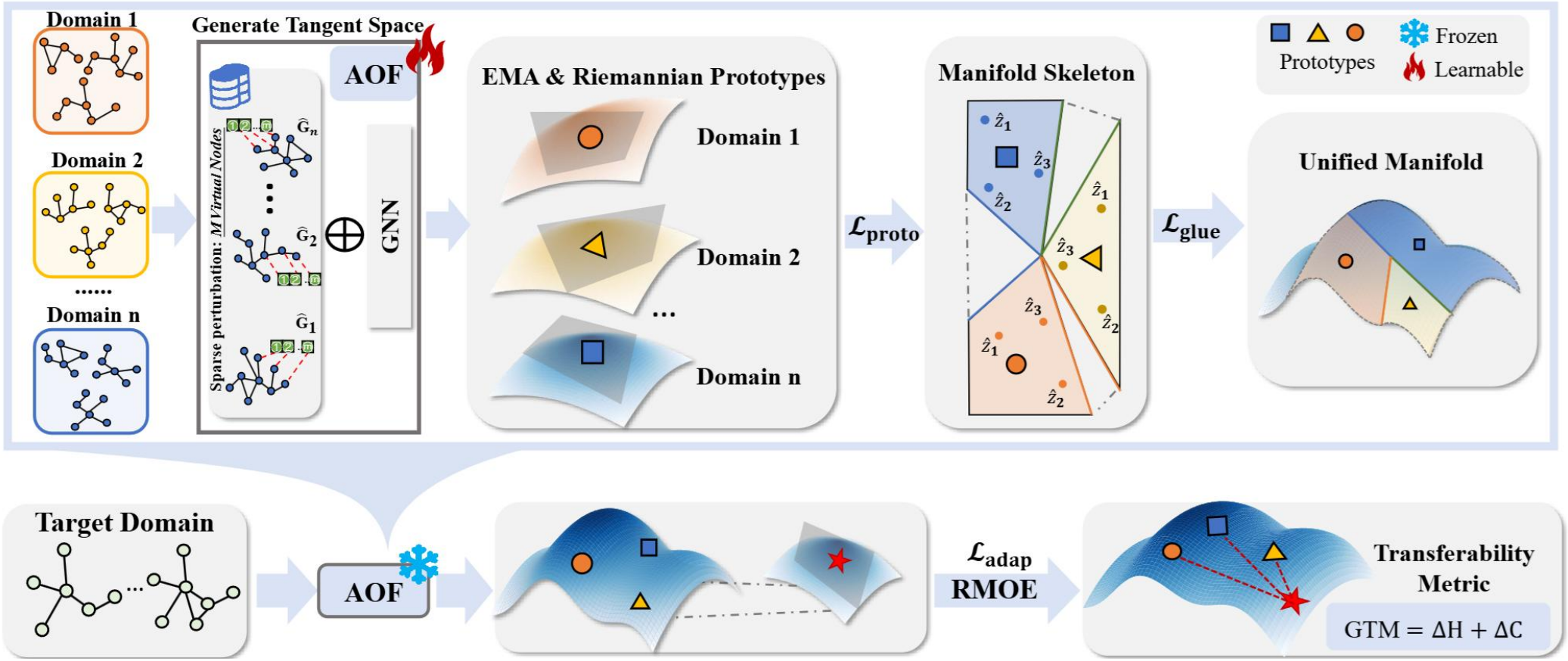
- **Foundational Problem.** How knowledge is integrated and transferred across different domains.
- **Neural Manifold Gluing.** A theory for understanding knowledge transfer.
- **GraphGlue.** A framework support mini-batch multi-domain pretraining.
- **Geometric Transfer Metric.** A metric to quantify GraphGlue's transferability.
- **Geometric Scaling Law .** Evaluation under cross-domain transfer learning empirically demonstrate it.



(a) Computers (b) Reddit

Figure: Geometric scaling law on (a) Computers and (b) Reddit datasets.

Overview of GraphGlue Framework



- **Pretraining.** Adaptive Orthogonal Frame & Holonomy Loss & Curvature Loss.
- **Adaptation.** Riemannian Prototypes & Riemannian Mixture-of-Experts.
- **Transferability.** Holonomy & Curvature Disagreement.

GraphGlue: Learning Local Geometry

We generate Local basis of tangent space, **mimicking the directional derivate**

$$D_{\boldsymbol{v}} f \;=\; \lim_{t\to 0} \frac{f(\boldsymbol{p}+t\boldsymbol{v})-f(\boldsymbol{p})}{t},$$

Definition 4.1 (**(k, M)-sparse perturbation**). *Given a graph perturbation set that consists of M nodes $\mathbb{P} = \{p_i\}$ with parameters $\{\boldsymbol{p}_i\}$, for $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, the perturbed graph is denoted as $\hat{\mathcal{G}} = (\hat{\mathcal{V}}, \hat{\mathcal{E}}) := \mathcal{G} \oplus \mathbb{P} = \left(\mathcal{V} \cup \{p_m\}_{m=1}^{M}, \mathcal{E} \cup \{(v_{i_m}, p_m)\}_{i_m=1, m=1}^{k, M}\right)$, where (v_i, p_m) is a edge weighted by an attentive function $h(\boldsymbol{x}_i, \boldsymbol{p}_m)$, v_{i_m} are k nodes selected based on top-k $h(\boldsymbol{x}_i, \boldsymbol{p}_m)$.*

Definition 4.2 (**Adaptive Orthogonal Frame, AOF**). *With tangent vectors generated by the above perturbation and a graph encoder f_{GNN}, after QR-decomposition with length recovery, the adaptive orthogonal frame is $\{\boldsymbol{w}_m : \boldsymbol{z}^{(i)} \mapsto \boldsymbol{w}_m^{(i)} \in \mathbb{R}^d\}_{m=1}^{M}$ for every representation $\boldsymbol{z}^{(i)}$. There exists a dual frame $\{\boldsymbol{\theta}^m\}$ such that $\boldsymbol{\theta}^m(\boldsymbol{w}_l) = \delta_{ml}$, where δ_{ml} is the Kronecker delta.*

The length of the tangent vector, describing the space deformation, is **upper-bounded by the perturbation**.

Theorem 4.3 (**Upper bound of Tangent Vector Length, Appendix B.1**). *Given a connected $\mathcal{G}$ with N nodes, the adjacency matrix $\boldsymbol{A}$, the Laplacian $\boldsymbol{L}$, and the feature matrix of perturbation nodes $\boldsymbol{P}$, apply (k, M)-sparse perturbation to $\mathcal{G}$, suppose $\frac{kM}{N} = \varepsilon$, where $\varepsilon > 0$ is small, and added edge weights satisfy $\sum_l h(v_i, p_l) = 1$. Then, the upper bound $\|\boldsymbol{w}_m^{\boldsymbol{p}}\| \leq (1 + \varepsilon)\|\boldsymbol{P}\|$ holds, where $\boldsymbol{w}_m^{\boldsymbol{p}}$ is the component of $\boldsymbol{w}_m$ determined by perturbation.*

Based on above construction, **the point-wise matrix form** of Riemannian metric tensor is

$$\boldsymbol{G}_i = \boldsymbol{W}^{(i)\top}\boldsymbol{W}^{(i)} = \mathrm{diag}(\|\boldsymbol{w}_1\|^2, ..., \|\boldsymbol{w}_M\|^2),$$

GraphGlue: Local Pieces to Global Ones

The gluing boundary can be defined by the adjacency in graph topology.

Definition 4.4 (Edge Tangent Translation). *Given an edge $(i,j) \in \mathcal{E}$, the tangent spaces of its two endpoints $T_{\boldsymbol{z}^{(i)}}U_i$ and $T_{\boldsymbol{z}^{(j)}}U_j$, and the Riemannian metric of $T_{\boldsymbol{z}^{(i)}}U_i$ denoted as $\boldsymbol{G}_i$, the edge tangent translation is defined as a linear map $\boldsymbol{P}^{(i,j)} : T_{\boldsymbol{z}^{(i)}}U_i \to T_{\boldsymbol{z}^{(j)}}U_j$ on edge $(i,j) \in \mathcal{E}$ as*

$$\boldsymbol{P}^{(i,j)} = \boldsymbol{G}_j^{-1/2}\left(\boldsymbol{G}_j^{1/2}\boldsymbol{G}_i\boldsymbol{G}_j^{1/2}\right)^{1/2}\boldsymbol{G}_j^{-1/2}. \tag{2}$$

Theorem 4.5 (Tangent Edge Translation as Isometry, Appendix B.2). *The tangent edge translation in Definition 4.4 is the optimal solution of*

$$\min_{\boldsymbol{P}\in GL(M)} \left\|\boldsymbol{P}^\top \boldsymbol{G}_j \boldsymbol{P} - \boldsymbol{G}_i\right\|_F^2, \tag{3}$$

where GL *denotes the general linear group, such that $\boldsymbol{G}_j(\boldsymbol{P}^{(i,j)}\boldsymbol{u}, \boldsymbol{P}^{(i,j)}\boldsymbol{v}) = \boldsymbol{G}_i(\boldsymbol{u},\boldsymbol{v})$, which induces an isometry $\phi^{(i,j)}$ between manifold boundaries ∂U_i and ∂U_j.*

Theorem 4.6 (Existence of Global Metric, Appendix B.3). *Let $(\{\boldsymbol{G}_i\}_{i=1}^N, \{\boldsymbol{P}^{(i,j)}\}_{(i,j)\in\mathcal{E}})$ be local metrics and tangent edge translations. There exists a unique global continuous metric $\boldsymbol{G}$ on $(\bigcup_\phi)_{i=1}^N U_i$ such that the restriction of $\boldsymbol{G}|_{U_i} = \boldsymbol{G}_i$ for all i.*

We begin with the compatibility of metric **along edges**, which is necessary for the existence of a global metric.

GraphGlue: Local Pieces to Global Ones

Holonomy describes how the tangent vector changes *when traversing along a closed curve*, and define a **holonomy map** to measure the changes.

Definition 4.7 (Holonomy Map and Holonomy Loss). *Let $\mathcal{Z}_1(\mathcal{G})$ denote the real vector space of 1-cycles on graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ under symmetric difference. For any cycle $\mathcal{C} = (i_0, i_1, \ldots, i_L = i_0)$, its **holonomy map** is defined as the composition of transport maps along the path,*

$$\boldsymbol{H}(\mathcal{C}) := \prod_{\ell=0}^{L-1} \boldsymbol{P}^{(i_\ell, i_{\ell+1})} \in \mathrm{GL}(M). \tag{4}$$

*The collection $\mathbf{P} := \{\boldsymbol{P}^{(i,j)}\}$ is said to be **trivial** if $\boldsymbol{H}(\mathcal{C})$ is the identity map for $\forall \mathcal{C} \in \mathcal{Z}_1(\mathcal{G})$. Given the set of all triangles $\mathcal{A}_{ijk} = ((v_i, v_j), (v_j, v_k))$, the corresponding holonomy loss is formulated as*

$$\mathcal{L}_{holo}(\mathcal{G}) = \frac{1}{|\mathcal{A}|} \sum\nolimits_{\mathcal{A}_{ijk}} \|\boldsymbol{P}^{(k,i)} \boldsymbol{P}^{(j,k)} \boldsymbol{P}^{(i,j)} - \boldsymbol{I}\|_F^2. \tag{5}$$

Theorem 4.8 (Triangle Triviality, Appendix B.4). *If every edge belongs to at least one triangle, and $\boldsymbol{H}(\mathcal{T}) = \boldsymbol{I}$ for all triangular cycles $\mathcal{T}$ in $\mathcal{G}$, then $\boldsymbol{H}(\mathcal{C}) = \boldsymbol{I}$ for all cycles $\mathcal{C} \in \mathcal{Z}_1(\mathcal{G})$.*

When the holonomy map of triangles is **trivial**, the offset at the gluing boundaries is eliminated.

GraphGlue: Local Pieces to Global Ones

To eliminate "fold" that hinders knowledge transport along the manifold, we aim at making volume unit varies softly and smoothly across geodesic, avoiding huge volume change, which also relates to **the Ricci curvature**.

Theorem 4.9 (Ricci Curvature Estimation, Appendix B.5). *Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and an edge $(i, j) \in \mathcal{E}$, let $\boldsymbol{z}^{(i)}, \boldsymbol{z}^{(j)} \in \mathcal{M}$ be the corresponding embedded points, and $\gamma : [0, 1] \to \mathcal{M}$ be the unit-speed geodesic connecting them, i.e., $\gamma(0) = \boldsymbol{z}^{(i)}$, $\gamma(1) = \boldsymbol{z}^{(j)}$. The sign of the Ricci curvature along $\dot{\gamma}$ can be estimated by the ratio of metric determinants:*

$$r(\boldsymbol{z}^{(i)}, \boldsymbol{z}^{(j)}) := \frac{\det \boldsymbol{G}_i}{\det \boldsymbol{G}_j} \approx 1 - \frac{1}{3} Ric(\dot{\gamma}). \tag{6}$$

The glued manifold has achieved C^1 continuity, but making the metric tensor **smooth/C^∞** is almost impossible. Hence, we introduce **the Log-Determinant k-order Smoothness**.

Definition 4.10 (k-order Smoothness and Curvature Loss). *Define $g_i = \frac{1}{2} \log \det \boldsymbol{G}_i$ as a scalar field over $\mathcal{G}$, representing the logarithmic volume density at node v_i. We say the manifold structure exhibits **log-determinant smoothness** if $\boldsymbol{g} \in \mathbb{R}^{|\mathcal{V}|}$ minimizes the graph Dirichlet energy: $\mathcal{E}_{Dir}[\boldsymbol{g}] = \|\boldsymbol{L}^k \boldsymbol{g}\|^2$, where $\boldsymbol{L}$ is the (normalized) Laplacian of $\mathcal{G}$. In light of computational efficiency in practice, we define the curvature loss function of 2-order smoothness as follows,*

$$\mathcal{L}_{Curv}(\mathcal{G}) = \frac{1}{|\mathcal{A}|} \sum\nolimits_{\mathcal{A}_{ijk}} |\log(r_{ij}) - \log(r_{jk})|^2 \tag{7}$$

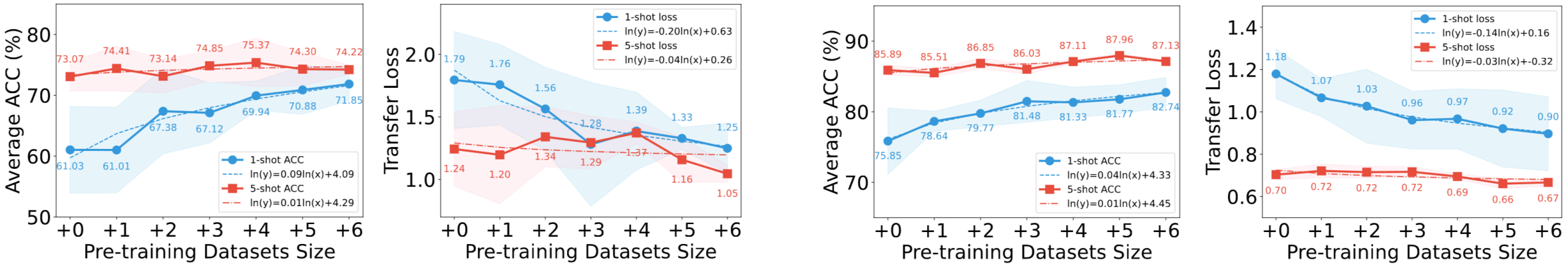
GraphGlue: Geometric Scaling Law

The Geometric Scaling Law

As the quantities of graphs increase, the glued manifold approximates an ideal manifold. **Larger quantities of datasets** improve model transferability with **a smoother manifold.**

Theorem 4.11 (Gluing into a Smooth Manifold, Appendix B.6). *For any graph dataset* $\mathbb{G}$*, if* $\mathbf{G}$ *is log-determinant* ∞*-order smooth, and* $\mathbf{P}$ *is trivial with induced metric-preserving diffeomorphism* ϕ*, then* $(\mathcal{F}, \mathbf{G}, \mathbf{P})$ *glues to a smooth Riemannian manifold* $(\mathcal{F}, \mathbf{G})$*, where* $\mathcal{F} = (\bigcup_{\phi})_{i=1}^{N} U_i$.

We incrementally incorporate *Pubmed, Photo, FacebookPagePage, WordNet18RR, MUATG and Lipophilicity* in order, referred to as +1, +2, +3, +4, +5 and +6, respectively. The observed logarithmic scaling supports our claim on the scaling law.



(a) Computers (b) Reddit

Figure: Geometric scaling law on (a) Computers and (b) Reddit datasets.

GraphGlue: Training with Prototypes

For multi-domain source graphs $\mathbb{S} = \{\mathcal{S}_1, ..., \mathcal{S}_K\}$, we associate each graph with a Riemannian prototype

$$(\boldsymbol{z}^{\mathcal{S}_k}, \log \boldsymbol{G}^{\mathcal{S}_k}) = \left(\frac{1}{|\mathcal{S}_k|} \sum_{\mathcal{G} \in \mathcal{S}_k} \boldsymbol{z}^{\mathcal{G}}, \frac{1}{|\mathcal{S}_k|} \sum_{\mathcal{G} \in \mathcal{S}_k} \log \boldsymbol{G}(\boldsymbol{z}^{\mathcal{G}}) \right).$$

Challenges

1. computation efficiency for large-scale graphs;
2. semantics distinction across different domains.

1. EMA for Riemannian prototyping

$$\boldsymbol{z}^{\mathcal{S}_k} \leftarrow \beta \boldsymbol{z}^{\mathcal{S}_k} + (1-\beta) \frac{1}{|\mathcal{B}_k|} \sum_{\mathcal{G} \in \mathcal{B}_k} \boldsymbol{z}^{\mathcal{G}}$$

$$\log \boldsymbol{G}^{\mathcal{S}_k} \leftarrow \beta \log \boldsymbol{G}^{\mathcal{S}_k} + (1-\beta) \frac{1}{|\mathcal{B}_k|} \sum_{\mathcal{G} \in \mathcal{B}_k} \log \boldsymbol{G}(\boldsymbol{z}^{\mathcal{G}}),$$

2. Sample-prototype contrastive loss

$$\mathcal{L}_{\text{proto}}(\mathcal{G}) = -\frac{1}{K} \sum_{k=1}^{K} \log \frac{\exp(\text{sim}(\boldsymbol{z}^{\mathcal{G}}, \boldsymbol{z}^{\mathcal{S}_k})/\tau)}{\sum_{j=1}^{K} \exp(\text{sim}(\boldsymbol{z}^{\mathcal{G}}, \boldsymbol{z}^{\mathcal{S}_j})/\tau)}.$$

Algorithm 1 *Training Procedure for* GRAPHGLUE

Require: Epoch index e, optimizer, datasets $\mathcal{D}_{\text{mix}}, \mathcal{D}_{\text{single}}, \mathcal{D}_{\text{multi}}$ with data name mappings.
Ensure: Updated model parameters Θ.

// Stage 1: Mix Training for Local Construction
Initial $\mathcal{L}_{\text{local}} = 0$
for each batch $\mathcal{B}$ in $\mathcal{D}_{\text{mix}}$ **do**
  $\boldsymbol{Z}, \boldsymbol{W} \leftarrow$ GRAPHGLUE($\mathcal{B}$)
  $\mathcal{L}_{\text{local}} \leftarrow$ ContrastiveLoss($\boldsymbol{Z}$)
  if $e \geq$ warmup_epochs **then**
    $\mathcal{L}_{\text{proto}} \leftarrow$ PrototypeLoss($\boldsymbol{Z}$, data_name)
    $\mathcal{L}_{\text{local}} \leftarrow \mathcal{L}_{\text{local}} + \mathcal{L}_{\text{proto}}$
  end if
  $\nabla_\theta \mathcal{L}_{\text{local}} \leftarrow$ Backward($\mathcal{L}_{\text{local}}$)
  OptimizerStep()
  Update Prototypes with $\boldsymbol{Z}, \boldsymbol{W}$ in Eq. (8)
end for

// Stage 2: Mix Training for Global Manifold Skeleton
for each batch $\mathcal{B}$ in $\mathcal{D}_{\text{mix}}$ **do**
  $\boldsymbol{Z}, \boldsymbol{W} \leftarrow$ GRAPHGLUE($\mathcal{B}$)
  $\mathcal{E}_{\text{knn}} \leftarrow$ Cross-Dataset_KNN_Graph($\boldsymbol{Z}$, data_name)
  $\mathcal{T} \leftarrow$ SampleTrianglePaths($\mathcal{E}_{\text{knn}}$, Number_Sampled, T_{sample})
  $\mathcal{L}_{\text{geo}} \leftarrow 0$
  for $t = 1$ to T_{sample} **do**
    $\mathcal{L}_{\text{geo}}$ += GeometricLoss($\boldsymbol{W}, \mathcal{T}[t]$) in Eq. (7) and Eq. (5)
  end for
  $\mathcal{L}_{\text{geo}} \leftarrow \mathcal{L}_{\text{geo}} / T_{\text{sample}}$
  $\nabla_\theta \mathcal{L}_{\text{geo}} \leftarrow$ Backward($\mathcal{L}_{\text{geo}}$)
  OptimizerStep()
end for

// Stage 3: Refine Local Manifold Structure For Each Dataset
for each dataset $\mathcal{D}_s$ in $\mathcal{D}_{\text{single}}$ **do**
  Load graph data G_s with edge set $\mathcal{E}_s$
  $\mathcal{T}_s \leftarrow$ SampleTrianglePaths($\mathcal{E}_s$, Number_Sampled, T_{local})
  for $t = 1$ to T_{local} **do**
    Construct mini-graph batch $\mathcal{B}_t$ from $\mathcal{T}_s[t]$
    $\mathbf{z}, \mathbf{z}_{\text{tan}} \leftarrow$ GRAPHGLUE($\mathcal{B}_t$)
    $\mathcal{L}_{\text{refine}} \leftarrow$ GeometricLoss($\boldsymbol{W}, \mathcal{T}_s[t]$) in Eq. (7) and Eq. (5)
    $\nabla_\theta \mathcal{L}_{\text{refine}} \leftarrow$ Backward($\mathcal{L}_{\text{refine}}$)
    OptimizerStep()
  end for
end for
for each dataset $\mathcal{D}_m$ in $\mathcal{D}_{\text{multi}}$ **do**
  for each batch ($\mathcal{B}$) in $\mathcal{D}_m$ **do**
    $\boldsymbol{Z}, \boldsymbol{W} \leftarrow$ GRAPHGLUE($\mathcal{B}$)
    $\mathcal{E}_{\text{knn}} \leftarrow$ Intra-Dataset_KNN_Graph($\boldsymbol{Z}$)
    $\mathcal{T} \leftarrow$ SampleTrianglePaths($\mathcal{E}_{\text{knn}}$, Number_Sampled, T_{sample})
    for $t = 1$ to T_{sample} **do**
      $\mathcal{L}_{\text{geo}}$ += GeometricLoss($\boldsymbol{W}, \mathcal{T}[t]$) in Eq. (7) and Eq. (5)
    end for
    $\mathcal{L}_{\text{geo}} \leftarrow \mathcal{L}_{\text{geo}} / T_{\text{sample}}$
    $\nabla_\theta \mathcal{L}_{\text{geo}} \leftarrow$ Backward($\mathcal{L}_{\text{geo}}$)
    OptimizerStep()
  end for
end for
return Optimized Model parameters Θ^*

GraphGlue: Consistent Adaptation

GraphGlue employs **prompt adaptation** and **Riemannian MoE** with **Geometric Consistent Regularization.**

Prompt Adaptation

For a target sample $\mathcal{G}^{\mathcal{T}}$, with the coordinates $z^{\mathcal{T}}$, local metric G_z and basis vectors of the tangent space $W^{\mathcal{T}} = [w_1^{\mathcal{T}}, \ldots, w_M^{\mathcal{T}}]$ given by the pre-trained model. We introduce a **learnable prompt matrix** $Q \in \mathbb{R}^{d \times d}$.

$$z^{\text{adapt}} = Qz^{\mathcal{T}}. \qquad G^{\text{adapt}} = \text{diag}\left(\|Qw_1^{\mathcal{T}}\|^2, \ldots, \|Qw_M^{\mathcal{T}}\|^2\right),$$

Riemannian MoE

- Each Riemannian prototy$(z^{\mathcal{S}_k}, \log G^{\mathcal{S}_k})$ serves as **an expert**.
- The weight for each expert is given by a gating funct$\beta_k = g_k(z^{\text{adapt}}, G^{\text{adapt}})$.
- This MoE generates $\log G^{\text{align}} = \sum_{k=1}^{K} \beta_k \log G^{\mathcal{S}_k}$.
- The final representation $z_{\text{task}} = [z^{\mathcal{T}}; \log G^{\text{adapt}}; \log G^{\text{align}}]$, where $z_{\text{task}} \in \mathbb{R}^{d+2M}$

Geometric Consistent Regularization

we construct a transfer graph $\mathcal{G}_0$ by connecting the target to its k-nearest prototypes

$$\mathcal{L}_{\text{adap}} = \mathcal{L}_{\text{task}}(z_{\text{task}}; y_{\text{task}}) + \lambda \mathcal{L}_{\text{glue}}, \qquad \mathcal{L}_{\text{glue}} = \mathcal{L}_{\text{holo}}(\mathcal{G}_0) + \mathcal{L}_{\text{curv}}(\mathcal{G}_0),$$

where λ balances task-specific learning with consistency, and y_{task} is the label of downstream task.

GraphGlue: Geometric Transfer Metric

We introduce **Geometric Transfer Metric** (GTM) which is defined as **the minimal geometric deformation** required to merge the target $\mathcal{G}^{\mathcal{T}}$ into the pre-trained manifold without disrupting its learned local geometry.

$$\mathrm{GTM}(\mathcal{G}^{\mathcal{T}};\mathcal{S}) = \Delta H + \Delta C, \qquad \Delta H = \mathcal{L}_{\mathrm{holo}}(\mathcal{G}_0), \qquad \Delta C = \mathcal{L}_{\mathrm{curv}}(\mathcal{G}_0).$$

1. **Holonomy disagreement** $\Delta \mathrm{H}$. It measures how the holonomy map deviates from identity along paths connecting the target to its nearest prototype.
2. **Curvature disagreement** $\Delta \mathrm{C}$. The natural interpretation is given as the “bending” or abrupt change in local volume.

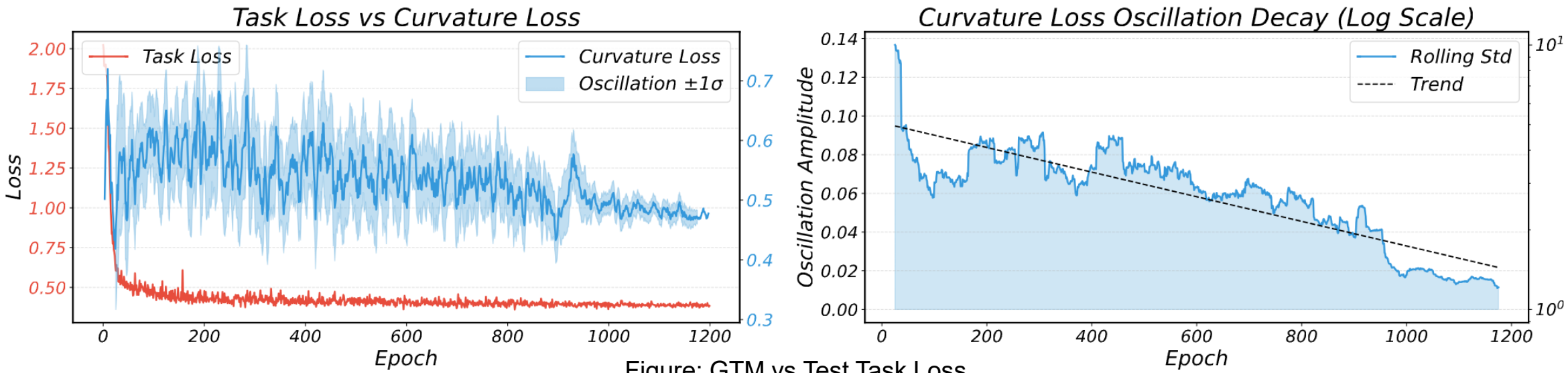


Figure: GTM vs Test Task Loss.

Experiments

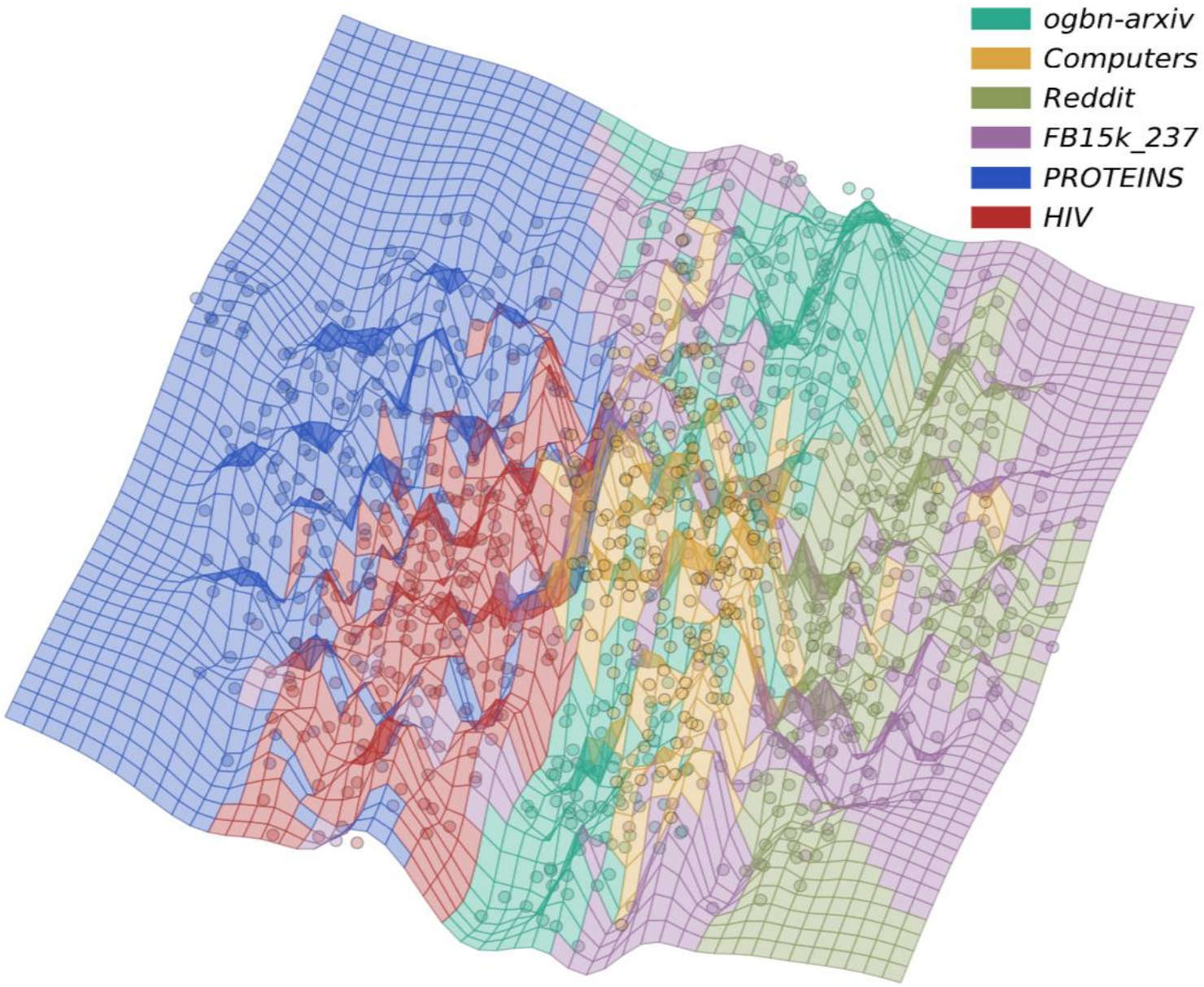
Table 1: Performance of cross-domain transfer on various downstream tasks, reported as mean ± std over 10 runs. The highest result is **bolded**, and the runner-up is underlined.

Model	Node Classification						Link Classification		Graph Classification	
	Arxiv		Computers		Reddit		FB15k_237		PROTEINS	
	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
GCN	12.6±1.7	27.6±2.1	33.8±3.8	65.7±4.2	11.1±2.1	28.3±1.0	32.1±2.3	52.4±1.8	50.1±13.0	55.0±9.9
GraphSAGE	14.6±3.7	26.1±2.2	35.4±8.2	66.7±4.4	14.6±2.3	22.2±1.1	35.7±2.1	58.9±1.5	58.9±2.7	60.4±1.3
GIN	11.2±2.0	26.0±2.4	44.7±6.0	69.5±3.5	18.5±1.8	29.0±1.6	38.2±2.5	63.7±1.7	54.2±13.5	58.8±5.0
GCC	12.6±2.0	26.8±2.1	34.8±6.1	62.6±3.1	54.7±5.6	65.2±1.5	47.8±1.9	73.6±1.2	59.2±7.9	64.2±3.0
DGI	13.3±3.3	27.1±2.3	35.2±7.5	61.0±3.2	60.0±4.8	62.7±2.2	42.5±2.0	68.3±1.4	53.1±8.4	53.3±6.2
GraphMAE	12.6±1.7	27.6±2.1	33.8±3.8	65.7±4.2	11.1±2.1	28.3±1.0	51.3±1.8	77.2±1.0	**60.1±13.0**	65.0±9.9
PRODIGY	28.4±2.2	33.6±2.8	45.3±4.1	52.7±3.6	35.6±3.2	42.3±2.9	53.5±1.0	72.1±6.9	48.9±5.4	55.2±4.7
GFT	26.5±2.4	36.7±1.9	54.6±4.0	69.1±3.5	58.8±2.5	66.2±1.4	58.0±1.3	79.1±1.6	55.4±5.8	62.1±3.5
RAGraph	18.7±2.5	32.3±1.7	46.2±4.3	62.3±3.7	52.5±3.4	63.0±1.3	52.1±3.0	64.5±2.5	51.4±5.1	58.6±2.8
SAMGPT	24.1±3.8	34.4±2.2	47.6±7.4	60.8±3.6	62.8±4.2	75.1±1.6	57.4±2.4	77.6±2.7	52.4±3.1	59.1±2.6
GCOPE	26.5±5.5	**39.1±1.9**	54.5±9.1	72.2±2.8	62.7±4.5	80.4±0.7	58.2±2.6	79.3±2.2	55.1±3.5	64.8±2.4
MDGFM	26.0±2.4	32.2±1.7	46.6±8.4	64.0±5.3	64.8±3.3	76.5±1.7	56.1±1.6	77.6±2.0	53.4±5.3	57.7±3.4
GRAPHGLUE	**28.8±5.2**	37.0±2.3	**59.5±7.0**	**73.2±0.7**	**67.1±3.3**	**85.0±1.1**	**59.7±5.2**	**81.5±2.3**	59.8±4.8	**65.3±2.4**

Table 13: Ablation study of GraphGlue's key components.

	Variants	Arxiv	Computers	Reddit	FB15k_237	PROTEINS	HIV
1-shot	w/o EMA	15.46±1.41	30.84±9.50	7.43±2.37	35.90±17.40	58.48±2.56	52.62±2.41
	w/o L_proto	9.57±4.56	31.24±10.36	37.90±9.34	43.59±8.13	58.49±2.60	54.10±2.76
	GRAPHGLUE	**29.73±2.56**	**61.03±7.13**	**68.42±4.68**	**60.89±2.11**	**69.12±4.19**	**58.53±8.20**
5-shot	w/o EMA	16.08±2.13	34.90±7.77	11.53±2.26	40.21±12.07	59.85±4.53	53.87±2.43
	w/o L_proto	15.26±9.91	36.63±11.84	46.13±14.14	64.56±16.42	61.64±6.28	55.50±2.70
	GRAPHGLUE	**39.98±1.67**	**74.15±2.38**	**84.89±0.68**	**79.52±1.75**	**73.94±2.38**	**62.18±2.50**

Visualization



Thanks for Listening

Contact

lsun@bupt.edu.cn

huangzhenhao@ncepu.edu.cn

GitHub: https://github.com/RiemannGraph/GraphGlue