Fused Gromov-Wasserstein Graph Mixup for Graph-level Classifications

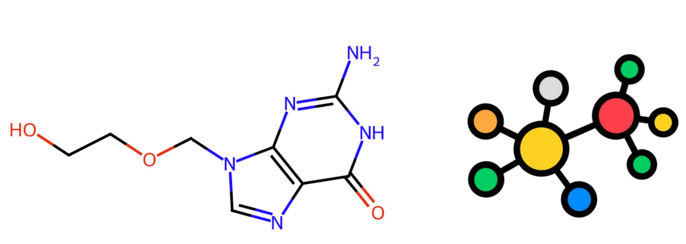
Xinyu Ma, Xu Chu*, Yasha Wang, Yang Lin, Junfeng Zhao, Liantao Ma, Wenwu Zhu

Presenter: Xinyu Ma, Peking University

2023.11

Background: Why Graph Data Augmentation?

- Graph Neural Networks (GNNs) have shown promising capabilities in various graph-level classification tasks:



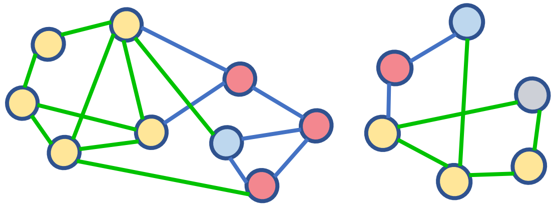
Molecular Property Prediction

Social Network Classification

- Yet GNNs still suffer from **data insufficiency** and **perturbations**
- Require the regularization of **Graph Data Augmentation** techniques:
  - Fortify GNNs against **potential noises and outliers** underlying **insufficient samples**
  - Enable **more robust and representative feature** learning

Challenges of Graph Data Augmentation

- Graphs are non-Euclidean data with distinctive properties:



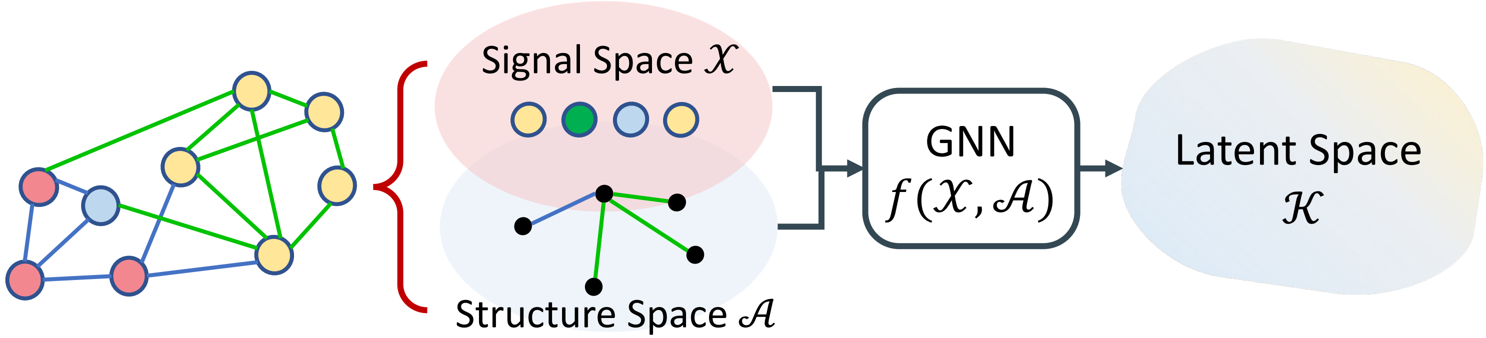
Irregular graph sizes

Misaligned Nodes

Diverse Topologies

  - **Require unique design to accommodate those properties**

- GNNs map two intertwined yet complementary data spaces (i.e., graph signal and graph structure spaces) to an aligned representation space



  - **A good graph data augmentation method should consider augmenting both graph signal and graph structure spaces**

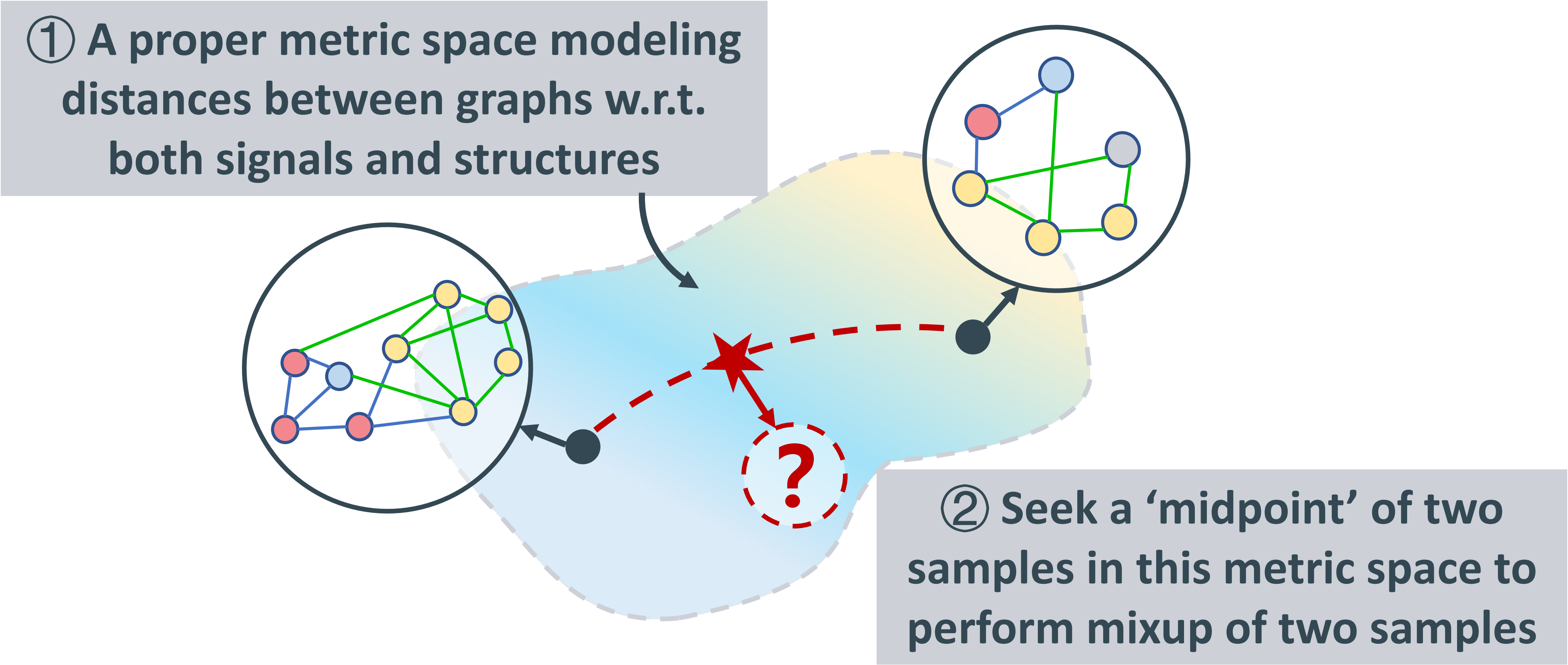
Previous Works

- Existing research considers the augmentation in the graph signal space and structure space **independently**
  - ifMixup[1] conducts Euclidean mixup in the graph signal space, yet **fails to preserve key topologies** of the original graphs.
  - $\mathcal{G}$-Mixup[2] realizes graph structure mixup based on the estimated graphons, yet **fails to assign semantically meaningful graph signals**.
- However, The graph signal and structure spaces are **NOT isolated from each other**, and there are strong entangling relations between them.

A joint modeling of the interaction between graph signal and structure spaces is essential for graph data augmentation

Our Insights & Ideas

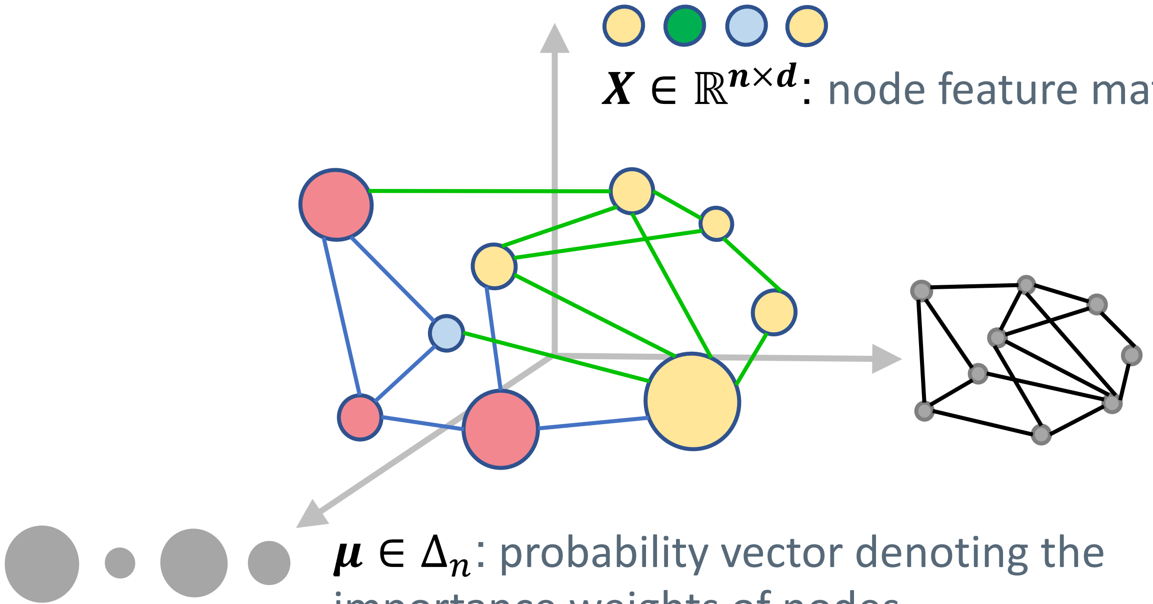
- Design a novel **graph mixup** method considering the **interaction of graph signal and graph structure spaces** to augment the input space



Methods

① A proper metric space modeling distances between graphs w.r.t. both signals and structures

- Fused Gromov-Wasserstein[3] Metric Space:
  - We define an undirected attributed graph with a tuple $(\boldsymbol{\mu}, \boldsymbol{X}, \boldsymbol{A})$:



$\boldsymbol{X} \in \mathbb{R}^{\boldsymbol{n} \times \boldsymbol{d}}$: node feature matrix

$\boldsymbol{A} \in \mathbb{S}_{\boldsymbol{n}}(\mathbb{R})$: symmetric structural relationship matrix (e.g., shortest path distance, adjacency, etc.)

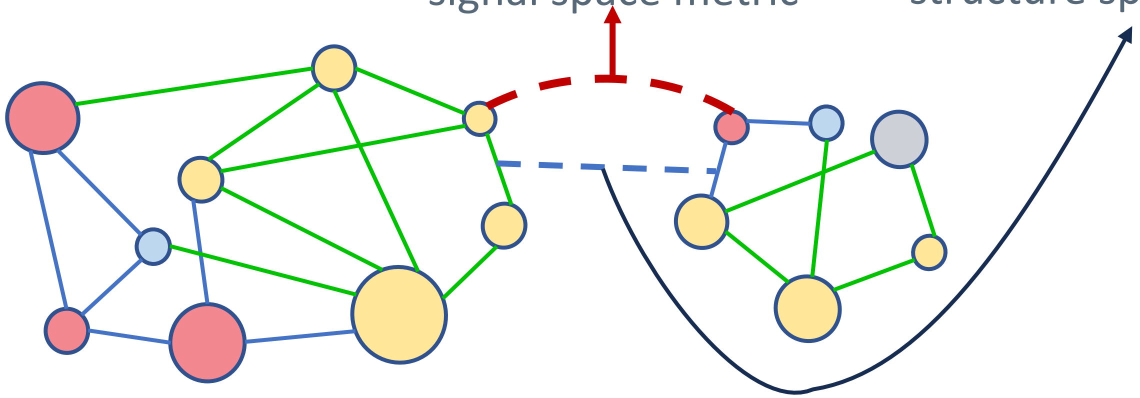
$\boldsymbol{\mu} \in \Delta_n$: probability vector denoting the importance weights of nodes

Methods

① A proper metric space modeling distances between graphs w.r.t. both signals and structures

- Fused Gromov-Wasserstein[3] Metric Space:
  - We define an undirected attributed graph with a tuple $(\boldsymbol{\mu}, \boldsymbol{X}, \boldsymbol{A})$
  - Fused Gromov-Wasserstein metric is formulated as an optimal transport problem optimizing the coupling $\boldsymbol{\pi} \in \Pi(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2) \coloneqq \{\boldsymbol{\pi} \in \mathbb{R}_+^{n_1 \times n_2} \mid \boldsymbol{\pi}\mathbf{1}_{n_2} = \boldsymbol{\mu}_1, \boldsymbol{\pi}^\top \mathbf{1}_{n_1} = \boldsymbol{\mu}_2\}$ between nodes in a fused metric space considering the interaction of structure and signals at minimum alignment costs.

$$\mathrm{FGW}_q(G_1, G_2) = \min_{\boldsymbol{\pi} \in \Pi(\boldsymbol{\mu}_i, \boldsymbol{\mu}_j)} \sum_{i,j,k,l} \left((1-\alpha) \underbrace{d\left(\boldsymbol{x}_1^{(i)}, \boldsymbol{x}_2^{(j)}\right)^q}_{\text{signal space metric}} + \alpha \underbrace{\left|\boldsymbol{A}_1(i,k) - \boldsymbol{A}_2(j,l)\right|^q}_{\text{structure space metric}} \right) \pi_{i,j}\pi_{k,l},$$



Methods

② Seek a 'midpoint' of two samples in this metric space to perform mixup of two samples

- Seek a synthetic graph $\tilde{G}$ at the 'midpoint' of source graphs G_1 and G_2

$$\arg \min_{\tilde{G} \in (\Delta_{\tilde{n}}, \mathbb{R}^{\tilde{n} \times d}, \mathbb{S}_{\tilde{n}}(\mathbb{R}))} \lambda \mathrm{FGW}(\tilde{G}, G_1) + (1-\lambda)\mathrm{FGW}(\tilde{G}, G_2),$$

- Traditional numeric solutions suffer high computation complexity. We enhance the computational efficiency by conducting Mirror Descent with a relaxed projection to polytope constraints:

Alternately projecting to row and column marginal constraints instead of directly to the strict polytope constraint

✅ **Single-loop Algorithm**

✅ **Faster Convergence**

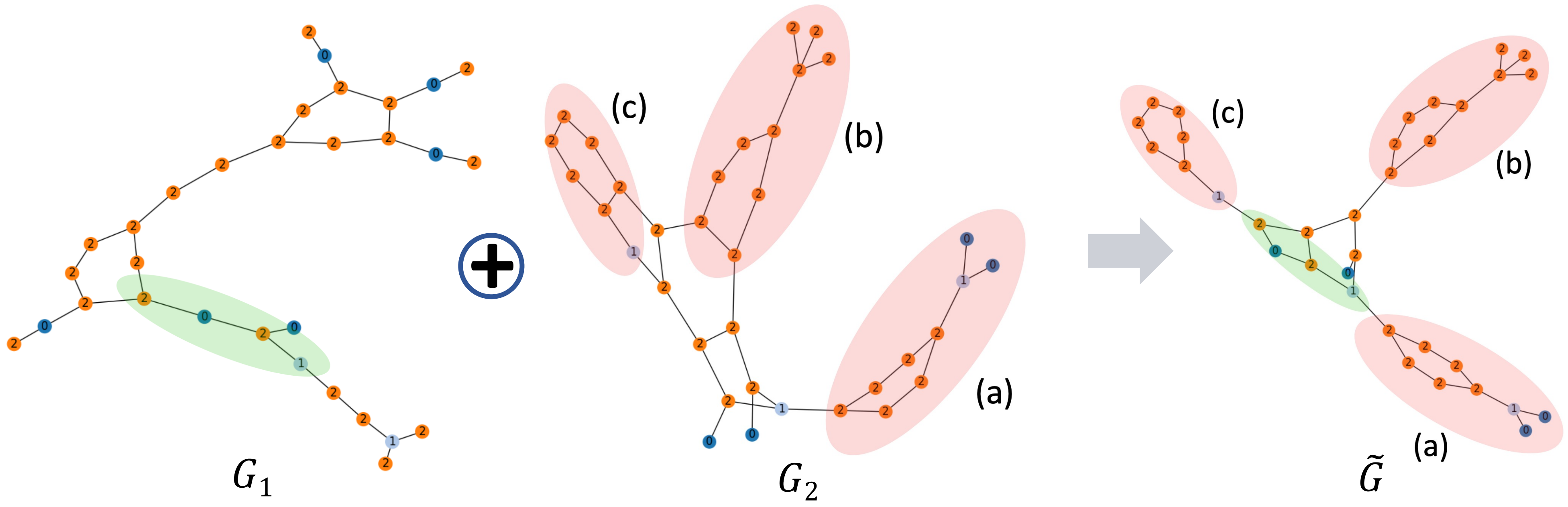
✅ **Ensuring Estimation Accuracy**

Algorithm 2 `FGWMixup`$_*$: Accelerated `FGWMixup`

Input: $\tilde{\boldsymbol{\mu}}, G_1 = (\boldsymbol{\mu_1}, \boldsymbol{X_1}, \boldsymbol{A_1}), G_2 = (\boldsymbol{\mu_2}, \boldsymbol{X_2}, \boldsymbol{A_2})$
Optimizing: $\tilde{\boldsymbol{X}} \in \mathbb{R}^{\tilde{n}\times d}, \tilde{\boldsymbol{A}} \in \mathbb{S}_{\tilde{n}}(\mathbb{R}), \boldsymbol{\pi}_1 \in \Pi(\tilde{\boldsymbol{\mu}}, \boldsymbol{\mu_1}), \boldsymbol{\pi}_2 \in \Pi(\tilde{\boldsymbol{\mu}}, \boldsymbol{\mu_2})$.
for k in *outer iterations* and not converged **do**:
  $\tilde{G}^{(k)} := (\tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{X}}^{(k)}, \tilde{\boldsymbol{A}}^{(k)})$
  $\boldsymbol{D}_1^{(k)} := \left(d(\tilde{\boldsymbol{X}}^{(k)}[i], \boldsymbol{X}_1[j])\right)_{\tilde{n}\times n_1}, \boldsymbol{D}_2^{(k)} := \left(d(\tilde{\boldsymbol{X}}^{(k)}[i], \boldsymbol{X}_2[j])\right)_{\tilde{n}\times n_2}$
  for i in $\{1,2\}$ **do**
    while not convergence **do**: ▷ Solve $\arg\min_{\boldsymbol{\pi}_i^{(k)}} \mathrm{FGW}(\tilde{G}^{(k)}, G_i)$
      $\boldsymbol{\pi}_i^{(k)} \leftarrow \boldsymbol{\pi}_i^{(k)} \odot \exp\left(\gamma(4\alpha\tilde{\boldsymbol{A}}^{(k)}\boldsymbol{\pi}_i^{(k)}\boldsymbol{A}_i - (1-\alpha)\boldsymbol{D}_i^{(k)})\right)$
      $\boxed{\boldsymbol{\pi}_i^{(k)} \leftarrow \mathrm{diag}(\tilde{\boldsymbol{\mu}}./\boldsymbol{\pi}_i^{(k)}\mathbf{1}_{\tilde{n}})\boldsymbol{\pi}_1^{(k)}}$ ▷ Bregman Projection on row constraint
      $\boldsymbol{\pi}_i^{(k)} \leftarrow \boldsymbol{\pi}_i^{(k)} \odot \exp\left(\gamma(4\alpha\tilde{\boldsymbol{A}}^{(k)}\boldsymbol{\pi}_i^{(k)}\boldsymbol{A}_i - (1-\alpha)\boldsymbol{D}_i^{(k)})\right)$
      $\boxed{\boldsymbol{\pi}_i^{(k)} \leftarrow \boldsymbol{\pi}_i^{(k)}\mathrm{diag}(\boldsymbol{\mu}_i./\boldsymbol{\pi}_i^{(k)\top}\mathbf{1}_{n_i})}$ ▷ Bregman Projection on column constraint
    end while
  end for
  Update $\tilde{\boldsymbol{A}}^{(k+1)} \leftarrow \frac{1}{\tilde{\boldsymbol{\mu}}\tilde{\boldsymbol{\mu}}^\top}(\lambda\boldsymbol{\pi}_1^{(k)}\boldsymbol{A}_1\boldsymbol{\pi}_1^{(k)\top} + (1-\lambda)\boldsymbol{\pi}_2^{(k)}\boldsymbol{A}_2\boldsymbol{\pi}_2^{(k)\top})$
  Update $\tilde{\boldsymbol{X}}^{(k+1)} \leftarrow \lambda\mathrm{diag}(1/\tilde{\boldsymbol{\mu}})\boldsymbol{\pi}_1^{(k)}\boldsymbol{X}_1 + (1-\lambda)\mathrm{diag}(1/\tilde{\boldsymbol{\mu}})\boldsymbol{\pi}_2^{(k)}\boldsymbol{X}_2$
end for
return $\tilde{G}^{(k)}, y_{\tilde{G}} = \lambda y_{G_1} + (1-\lambda)y_{G_2}$

Experiments

- Qualitative Analysis:



✅ **Overall trident structure reserved in G_1** ✅ **Substructures from G_1 and G_2 are both adopted**

✅ **Both topologically alike and highly consistent in node features (denoted with ID and colors)**

Experiments

- Enhance GNN Performance
  - Experiments conducted on **five datasets** and **four GNN backbones** with various SOTA graph data augmentation methods

	Methods	PROTEINS vGIN	PROTEINS vGCN	NCI1 vGIN	NCI1 vGCN	NCI109 vGIN	NCI109 vGCN	IMDB-B vGIN	IMDB-B vGCN	IMDB-M vGIN	IMDB-M vGCN
MPNNs	vanilla	74.93(3.02)	74.75(2.60)	76.98(1.87)	76.91(1.80)	75.70(1.85)	75.89(1.35)	71.30(4.96)	72.30(4.34)	49.00(2.64)	49.47(3.76)
	DropEdge	73.59(2.50)	74.48(4.18)	76.47(2.85)	76.16(2.04)	75.38(2.05)	75.77(1.55)	73.30(3.85)	73.30(3.29)	49.47(2.66)	49.40(3.52)
	DropNode	74.48(2.91)	75.11(3.00)	76.89(1.25)	77.42(1.71)	73.98(2.16)	75.45(1.90)	71.50(3.23)	73.20(5.58)	49.80(3.29)	50.00(3.41)
	M-Mixup	74.40(3.00)	75.65(4.51)	76.45(3.39)	77.76(2.75)	75.41(2.78)	75.79(1.85)	72.20(4.83)	72.80(4.45)	49.13(3.25)	49.47(2.56)
	ifMixup	74.76(3.71)	74.04(2.27)	76.16(1.78)	77.37(2.56)	76.13(1.87)	76.74(1.56)	72.50(3.98)	72.40(5.14)	49.07(3.16)	49.73(4.67)
	$\mathcal{G}$-Mixup	74.84(2.99)	74.57(2.88)	76.42(1.79)	77.79(1.88)	75.55(2.32)	76.38(1.79)	72.40(4.82)	72.20(6.45)	49.47(4.73)	49.60(3.90)
	`FGWMixup`	75.02(3.86)	**76.01(3.19)**	**78.32(2.65)**	78.37(2.40)	76.40(1.65)	**76.79(1.81)**	73.00(3.69)	73.40(5.12)	**49.80(2.63)**	**50.80(4.06)**
	`FGWMixup`$_*$	**75.20(3.30)**	75.20(3.03)	77.27(2.71)	**78.47(1.74)**	**76.64(2.60)**	76.52(1.59)	**73.50(4.54)**	**74.00(2.90)**	49.20(3.38)	50.47(5.44)
	Methods	Graphormer	GraphormerGD	Graphormer	GraphormerGD	Graphormer	GraphormerGD	Graphormer	GraphormerGD	Graphormer	GraphormerGD
Graphormers	vanilla	75.47(3.16)	76.01(2.02)	61.56(3.70)	77.49(2.01)	65.54(3.04)	74.99(1.23)	70.40(5.00)	71.50(4.20)	48.87(4.10)	47.47(2.98)
	DropEdge	75.20(4.02)	75.12(3.22)	63.07(3.21)	74.94(2.44)	66.73(3.50)	74.73(3.22)	71.10(5.65)	72.30(3.93)	49.60(4.09)	46.67(3.85)
	DropNode	75.20(2.13)	76.28(3.49)	64.96(2.18)	76.20(1.95)	63.73(3.46)	74.78(2.07)	71.60(5.18)	71.30(5.18)	48.47(4.08)	47.67(2.83)
	M-Mixup	75.11(3.78)	74.39(3.83)	62.31(3.48)	75.47(1.45)	66.54(2.70)	74.61(1.86)	71.10(4.83)	70.50(4.70)	49.67(4.25)	48.00(3.85)
	$\mathcal{G}$-Mixup	75.74(3.12)	74.85(3.52)	63.07(4.40)	76.06(3.12)	65.03(2.98)	74.90(2.04)	72.10(6.38)	71.10(5.01)	46.93(5.18)	46.80(4.41)
	`FGWMixup`	**76.82(2.35)**	**77.18(3.48)**	**66.45(2.58)**	**78.20(1.88)**	67.36(3.21)	**76.01(3.04)**	**72.60(5.08)**	**72.40(4.48)**	49.73(3.80)	**48.87(4.03)**
	`FGWMixup`$_*$	76.19(3.20)	76.46(3.41)	64.26(3.25)	76.62(3.06)	**67.46(2.82)**	75.45(1.80)	71.70(4.17)	71.90(4.35)	**50.27(4.26)**	48.53(2.95)

✅ **Enhance GNN Test-time Generalizability:** Our methods consistently outperforms all SOTA augmentation methods under all settings

Experiments

- Enhance GNN Performance
  - Randomly corrupts 20/40/60% of training graph labels (i.e., switching to another random label)

Methods	IMDB-B 20%	IMDB-B 40%	IMDB-B 60%	NCI1 20%	NCI1 40%	NCI1 60%
vanilla	70.00(5.16)	59.70(5.06)	47.90(4.30)	70.58(1.29)	61.95(2.19)	48.25(4.87)
DropEdge	68.30(5.85)	59.40(5.00)	50.10(1.92)	69.51(2.27)	60.32(2.60)	49.61(1.28)
M-Mixup	70.70(5.90)	59.70(5.87)	50.90(1.81)	71.53(2.75)	63.24(2.59)	48.66(3.02)
$\mathcal{G}$-Mixup	67.50(4.52)	59.10(4.74)	49.40(2.87)	72.46(1.95)	63.26(4.39)	50.01(1.26)
FGWMixup	70.10(4.39)	**61.90(6.17)**	50.80(3.19)	**72.92(1.56)**	62.99(1.35)	**50.12(3.51)**
FGWMixup$_*$	**70.80(3.97)**	61.80(5.69)	**51.00(1.54)**	72.75(2.29)	**63.55(2.60)**	50.02(3.38)

✅ **Enhance GNN Robustness:** Resist label corruptions under various perturbation rates

Experiments

- Computational Efficiency Improvements:
  - Mixup clock time comparisons between FGWMixup using traditional solution and our accelerated version (FGWMixup*):

Datasets	Avg. Mixup Time (s) / Fold				
	PROTEINS	NCI1	NCI109	IMDB-B	IMDB-M
`FGWMixup`	802.24	1711.45	1747.24	296.62	212.53
`FGWMixup`$_*$	**394.57**	**637.41**	**608.61**	**85.69**	**74.53**
Speedup	2.03×	2.67×	2.74×	3.46×	2.85×

✅ **Higher mixup efficiency:** Up to 3.46× efficiency improvements

Thanks!
Q&A

Presenter: Xinyu Ma, Peking University

E-mail: maxinyu@pku.edu.cn