KDD2022

- GraphWorld: Fake Graphs Bring Real Insights for GNNs
- Few-shot Heterogeneous Graph Learning via Cross-domain Knowledge Transfer
- Instant Graph Neural Networks for Dynamic Graphs
- GraphMAE: Self-Supervised Masked Graph Autoencoders

GraphWorld: Fake Graphs Bring Real Insights for GNNs

John Palowitch, Anton Tsitsulin, Brandon Mayer, Bryan Perozzi

GraphWorld: Fake Graphs Bring Real Insights for GNNs

- How can we evaluate a GNN?
- Usually, we test the performance of a GNN on several datasets, e.g., OGB.

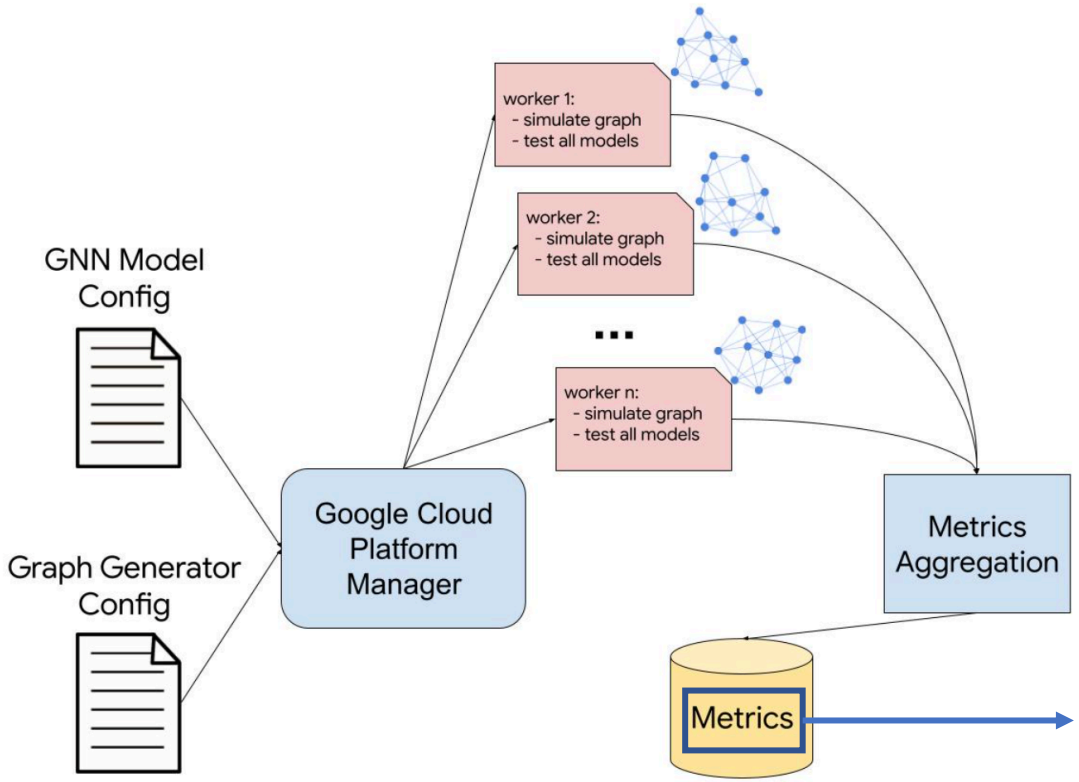
GraphWorld: Fake Graphs Bring Real Insights for GNNs

- But, using these benchmarked datasets will introduce a number of problems:
  - Inadequate generalization: hard to infer which models will generalize well to unseen datasets not in the benchmark
  - Architectural overfitting: new architectures are proposed only when they beat existing methods on these datasets
  - Lack of diversity: recent efforts on benchmark datasets mainly focus on the size of dataset

GraphWorld: Fake Graphs Bring Real Insights for GNNs

GraphWorld, a distributed framework for simulating diverse populations of GNN benchmark datasets

- Synthesize a fully-diverse “world” of potential graphs by probability models
- Running thousands of GNN experiments on synthetic data with less computational cost than one experiment on a large OGB dataset

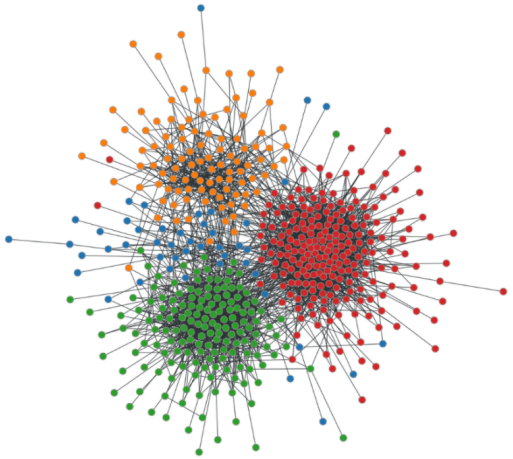


Task performance metrics. For example, accuracy for node classification

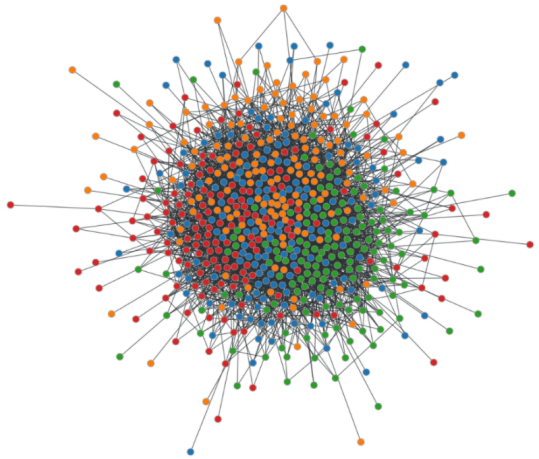
GraphWorld: Fake Graphs Bring Real Insights for GNNs

How to generate a graph? (for node classification task)

- Node feature: Drawn from within-cluster multivariate normals
- Class label: Cluster assignments
- Graph structure: Stochastic Block Model [1]
  - A generative model for random graph, which tends to produce graphs containing communities
- Parameter: number of nodes, number of clusters, average degree, degree power-law, cluster size, in-cluster/out-cluster edge number, feature center distance



high homophily

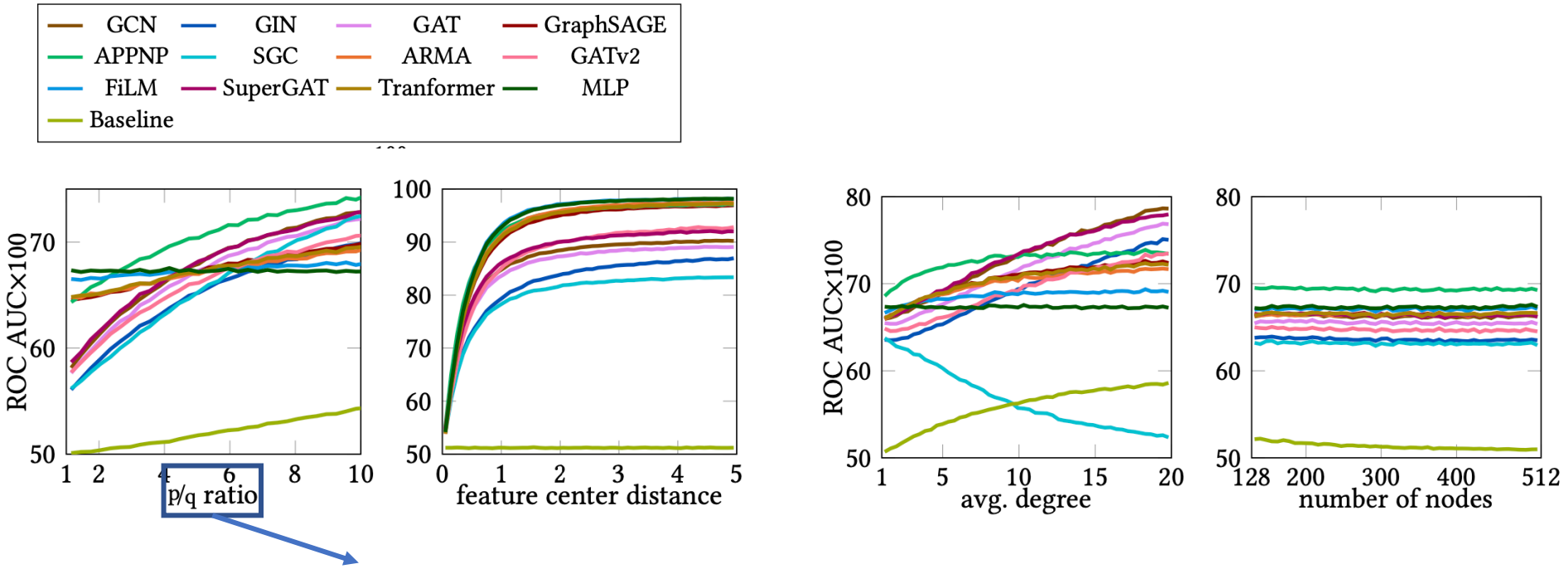


low homophily

[1] Emmanuel Abbe. 2017. Community detection and stochastic block models: recent developments. JMLR (2017).

GraphWorld: Fake Graphs Bring Real Insights for GNNs

Experimental results (x-axis shows different parameter)



p, q are the probabilities of in-cluster, out-cluster edges

GraphWorld: Fake Graphs Bring Real Insights for GNNs

Some interesting insights

- Number of vertices doesn't matter
  - The size of the graph (number of vertices) has negligible effect on test AUC
- Most models increased test AUC as the homophily (graph cluster signal) and feature-center-distance (feature cluster signal) increased
- Most GNNs can't count substructures (motif)
  - As shown in the following table, only GIN achieved better-than-mean-fitting MSE, along with simple linear regression with edge density as a feature, which out-performed all other GNNs

model	Mode 1	Mode 2	Mode 3
APPNP	1.04 ± 0.00	1.02 ± 0.00	1.03 ± 0.00
ARMA	1.03 ± 0.00	0.97 ± 0.00	0.92 ± 0.01
FiLM	1.06 ± 0.01	1.01 ± 0.00	1.01 ± 0.00
GAT	1.04 ± 0.00	1.02 ± 0.00	1.03 ± 0.00
GATv2	1.04 ± 0.00	1.02 ± 0.00	1.03 ± 0.00
GCN	1.04 ± 0.00	1.02 ± 0.00	1.03 ± 0.00
GIN	**0.86 ± 0.07**	**0.21 ± 0.01**	**0.33 ± 0.03**
GraphSAGE	1.04 ± 0.00	1.01 ± 0.00	1.01 ± 0.00
LR	**0.52 ± 0.00**	**0.52 ± 0.00**	**0.51 ± 0.03**
MLP	1.04 ± 0.00	1.02 ± 0.00	1.03 ± 0.00
SGC	1.08 ± 0.00	1.02 ± 0.00	1.03 ± 0.01
SuperGAT	1.04 ± 0.00	1.02 ± 0.00	1.03 ± 0.01
Transformer	1.04 ± 0.00	1.00 ± 0.00	1.00 ± 0.01

Table 1: Graph property prediction performance averages in terms of scaled MSE. Lower is better.

Mode1/2/3: different hyperparameter optimization strategy (see details in paper)

GraphWorld: Fake Graphs Bring Real Insights for GNNs

Some problems of this work

- Node features and label distribution are the most important factors to the node classification task. But it is hard to analyze impact of features based on the generative features.
- The graph generator is relatively simple. It will be better to see the performance of GNN on more diverse graphs.

Connection with our transfer learning project

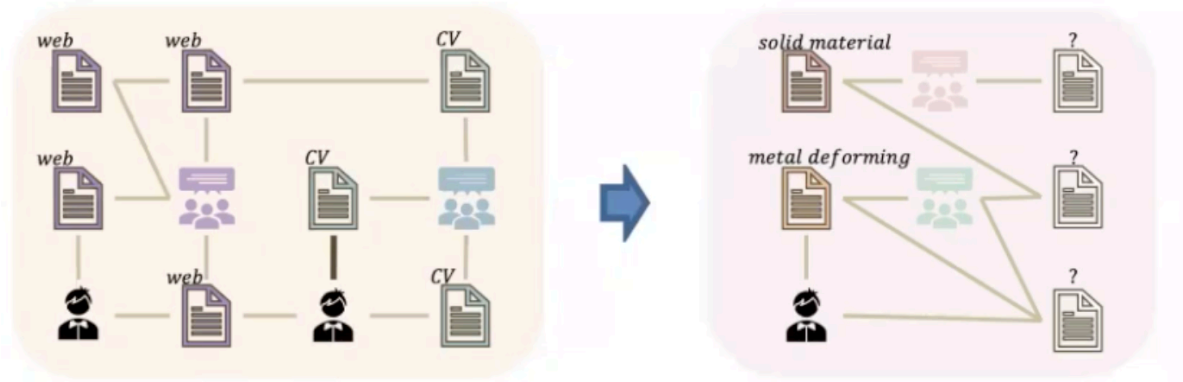
- As shown in the paper, graph-level task (motif counting) is a hard problem of GNN.
- Generative graph setting in GraphWorld can help us exploit many potential multi-graph combinations.
- For example, we would define two (ideally different) random graph generators, one with many labels, and one with few labels. We could then see how well the GNN is able to transfer knowledge from one graph to the other.

Few-shot Heterogeneous Graph Learning via Cross-domain Knowledge Transfer

Qiannan Zhang, Xiaodong Wu, Qiang Yang, Chuxu Zhang, Xiangliang Zhang

Few-shot Heterogeneous Graph Learning via Cross-domain Knowledge Transfer

- Cross-domain graph few-shot learning
  - Label is scarce in the novel graph
- How to extract the meta-knowledge from related-graphs in other domains for the novel graph with few-shot annotations?
  - Shared structure relevance



Few-shot Heterogeneous Graph Learning via Cross-domain Knowledge Transfer

- Input
  - Graph encoder parameter: θ_g, few-shot classifier: θ_c
  - A number of source domain datasets $\mathcal{S} = \{T_1, \dots, T_n\}$
  - A target domain dataset $\mathcal{T}$ with scarce labels
- Output
  - θ_g
- Objective: θ_g can conduct fast adaptation on $\mathcal{T}$ with few-shot labeled data after being trained with $\mathcal{S}$

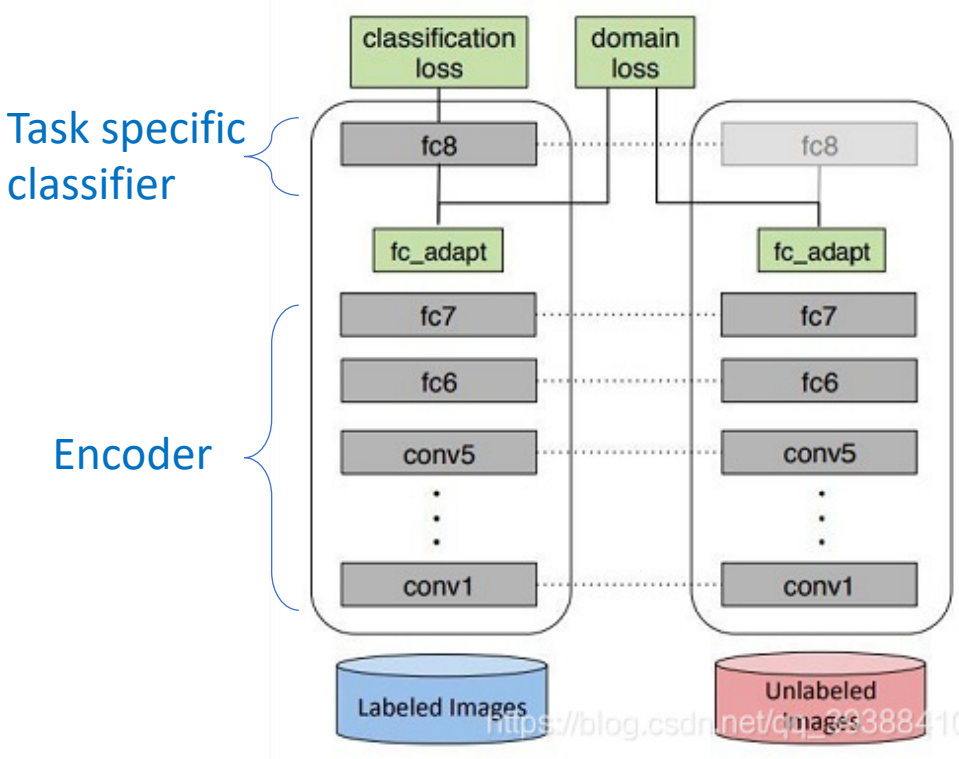
Few-shot Heterogeneous Graph Learning via Cross-domain Knowledge Transfer

- N-way K-shot setting
  - Dataset is splitted into support set and query set
    - Support set contains N classes, K samples per class
    - Query set contains Q examples per class
  - Model will be finetuned on support set and evaluated on query set
  - Usually, we will sample a set of few-shot tasks $T_i = \{\tau_1, \dots, \tau_m\}$ to conduct few shot learning

Few-shot Heterogeneous Graph Learning via Cross-domain Knowledge Transfer

How to handle domain shift?

- In other word, how can a encoder trained on one domain perform well on the other domain?
- Such encoder should project the data from source and target domain into a same feature space
- A common and successful way is to apply domain adaptation loss during training



During supervised training on source domain, enforce the encoder to generate same distribution on target domain

Few-shot Heterogeneous Graph Learning via Cross-domain Knowledge Transfer

Domain adaptation loss

- Measure the difference between two distributions
- For example, Max Mean Discrepancy (MMD) compares the higher order moments of two distributions

$$\mathrm{MMD}(X, Y) = \left\| \frac{1}{n} \sum_{i=1}^{n} \phi(x_i) - \frac{1}{m} \sum_{j=1}^{m} \phi(y_j) \right\|_H^2$$

Project function

Few-shot Heterogeneous Graph Learning via Cross-domain Knowledge Transfer

Meta learning

- Goal: solve new learning tasks using only a small number of training samples

Pipeline

- Sample a batch of virtual tasks $\{\tau_{vs,i}\}$
- Update graph encoder and classifier on support set of $\tau_{vs,i}$

$$(\theta_g', \theta_c') = (\theta_g, \theta_c) - \alpha \nabla_{\theta_g, \theta_c} \mathcal{L}_c^{sup}(\tau_{vs,i}; \theta_g, \theta_c),$$

Support set

- Optimize parameter based on query set

$$\mathcal{L}_{meta} = \sum_{\tau_{vs,i}} \mathcal{L}_c^{que}(\tau_{vs,i}; \theta_{g_i}', \theta_{c_i}')$$

Query set

Few-shot Heterogeneous Graph Learning via Cross-domain Knowledge Transfer

Cross domain graph meta learning

- Idea: Imitate domain shifts by splitting source domains as virtual source and virtual target domains during training

Pipeline

- Sample and pair a batch of virtual source tasks and virtual target tasks $\{\tau_{vs,i}, \tau_{vt,i}\}$
- Update graph encoder and classifier on support set of $\tau_{vs,i}$

$$(\theta_g', \theta_c') = (\theta_g, \theta_c) - \alpha \nabla_{\theta_g, \theta_c} \mathcal{L}_c^{sup}(\tau_{vs,i}; \theta_g, \theta_c),$$

- Adapt the graph encoder by domain adaptation loss and update the classifier based on $\tau_{vt,i}$

Domain adaptation

$$\theta_g^t = \theta_g' - \beta \nabla_{\theta_g'} \mathcal{L}_d(\tau_{vs,i}, \tau_{vt,i}; \theta_g', \theta_c'),$$

Domain adaptation loss

$$\theta_c^t = \theta_c' - \beta \nabla_{\theta_c'} \mathcal{L}_c^{sup}(\tau_{vt,i}; \theta_g^t, \theta_c'),$$

- Optimize parameter based on query set of source tasks and target tasks

$$\mathcal{L}_{meta} = \sum_{\tau_{vs,i}} \mathcal{L}_c^{que}(\tau_{vs,i}; \theta_{g_i}', \theta_{c_i}') + \sum_{\tau_{vt,i}} \mathcal{L}_c^{que}(\tau_{vt,i}; \theta_{g_i}^t, \theta_{c_i}^t),$$

Imitate that the model has been trained on source domain and applied in target domain

Few-shot Heterogeneous Graph Learning via Cross-domain Knowledge Transfer

Experiments

- Datasets
  - Aminer, U.S. Patents, Amazon
- Baseline
  - Graph embedding method, Graph neural network, Few-show learning method, Domain generalization technique
- Setting
  - leave-one-out setting: one domain is randomly chosen as the target domain with the remaining as source domains

Table 1: Paper classification accuracy (%) for AI, Mathematics & Interdisciplinary on AMiner data.

Methods	AI		Mathematics		Interdisciplinary	
	2-way	3-way	2-way	3-way	2-way	3-way
1-shot						
node2vec [7]	53.68	39.69	55.25	46.72	54.74	35.31
m2v [3]	65.22	59.27	74.49	67.02	71.33	60.99
GCN [12]	75.64	62.91	77.99	73.47	77.18	61.38
GraphSage [9]	76.89	61.60	75.02	75.64	74.69	60.14
HAN [33]	77.24	63.86	78.56	75.64	75.59	63.87
R-GCN [21]	76.91	63.59	79.45	75.94	77.54	65.42
ProtoNet [26]	74.63	64.55	85.70	79.13	78.71	65.71
MAML [5]	73.98	65.85	87.92	76.58	77.91	66.57
Proto-GCN	76.29	68.29	86.30	77.87	76.52	66.34
Proto-Sage	75.60	64.44	87.97	73.68	77.88	68.70
MAML-GCN	74.90	69.00	87.89	76.05	79.35	67.26
MAML-Sage	78.29	67.59	89.24	78.54	79.59	67.49
G-Meta [10]	76.50	63.27	86.75	75.60	78.43	65.27
GPN [16]	75.43	62.53	86.76	75.64	79.50	64.25
Meta-SGC [44]	77.41	61.85	87.57	77.86	75.69	67.31
AMM-GNN [32]	81.84	69.48	89.64	81.92	80.69	69.71
FC [14]	81.42	68.27	89.65	83.62	80.31	70.54
MLDG [13]	82.60	69.89	90.68	85.45	81.55	72.15
CrossHG-Meta	**85.36**	**72.43**	**91.45**	**89.54**	**82.93**	**74.63**
3-shot						
node2vec [7]	62.73	46.73	65.50	51.36	59.43	43.87
m2v [3]	79.96	71.72	86.35	74.57	82.80	68.80
GCN [12]	86.93	73.34	90.83	79.66	87.59	74.68
GraphSage [9]	85.26	73.01	88.25	80.39	86.09	72.96
HAN [33]	86.47	74.58	89.61	82.39	87.51	74.88
R-GCN [21]	86.78	75.64	90.16	83.57	88.26	75.91
ProtoNet [26]	83.81	70.37	91.07	87.41	87.89	75.01
MAML [5]	85.57	72.13	90.35	86.89	90.13	76.26
Proto-GCN	88.66	77.84	92.04	87.67	85.91	78.75
Proto-Sage	86.17	77.11	91.72	90.13	87.98	77.62
MAML-GCN	88.60	73.70	94.75	90.44	88.63	81.08
MAML-Sage	87.26	73.18	94.81	87.57	87.24	80.47
G-Meta [10]	82.37	72.53	92.65	87.56	87.04	78.83
GPN [16]	80.98	71.56	91.59	86.44	86.66	77.32
Meta-SGC [44]	83.50	71.57	93.45	87.68	85.86	79.43
AMM-GNN [32]	89.18	76.19	95.60	91.48	88.97	81.33
FC [14]	86.53	74.92	93.57	87.42	86.37	80.72
MLDG [13]	87.40	75.23	95.91	88.34	88.56	82.53
CrossHG-Meta	**90.16**	**79.20**	**96.63**	**92.18**	**90.07**	**85.34**

Table 1: Paper classification accuracy (%) for AI, Mathematics & Interdisciplinary on AMiner data.

Methods	AI		Mathematics		Interdisciplinary	
	2-way	3-way	2-way	3-way	2-way	3-way
1-shot						
node2vec [7]	53.68	39.69	55.25	46.72	54.74	35.31
m2v [3]	65.22	59.27	74.49	67.02	71.33	60.99
GCN [12]	75.64	62.91	77.99	73.47	77.18	61.38
GraphSage [9]	76.89	61.60	75.02	75.64	74.69	60.14
HAN [33]	77.24	63.86	78.56	75.64	75.59	63.87
R-GCN [21]	76.91	63.59	79.45	75.94	77.54	65.42
ProtoNet [26]	74.63	64.55	85.70	79.13	78.71	65.71
MAML [5]	73.98	65.85	87.92	76.58	77.91	66.57
Proto-GCN	76.29	68.29	86.30	77.87	76.52	66.34
Proto-Sage	75.60	64.44	87.97	73.68	77.88	68.70
MAML-GCN	74.90	69.00	87.89	76.05	79.35	67.26
MAML-Sage	78.29	67.59	89.24	78.54	79.59	67.49
G-Meta [10]	76.50	63.27	86.75	75.60	78.43	65.27
GPN [16]	75.43	62.53	86.76	75.64	79.50	64.25
Meta-SGC [44]	77.41	61.85	87.57	77.86	75.69	67.31
AMM-GNN [32]	81.84	69.48	89.64	81.92	80.69	69.71
FC [14]	81.42	68.27	89.65	83.62	80.31	70.54
MLDG [13]	<u>82.60</u>	<u>69.89</u>	<u>90.68</u>	<u>85.45</u>	<u>81.55</u>	<u>72.15</u>
CrossHG-Meta	**85.36**	**72.43**	**91.45**	**89.54**	**82.93**	**74.63**

Few-shot Heterogeneous Graph Learning via Cross-domain Knowledge Transfer

Problem to the paper

- How to handle the difference of different features (e.g., dimension) in one encoder? In my experience, two features that are too different cannot be calibrated using a simple domain adaptation loss.

Connection with our transfer learning project

- We both interest in cross-domain graph learning
- The main difference is this paper focuses on few-shot setting, that is how to learn a good initialization based on source graphs. But our attention is on the multi-graph joint learning setting, which is similar to the multi-task setting.
- I am curious of its domain adaptation part, and will take a look when the code is available

Instant Graph Neural Networks for Dynamic Graphs

Yanping Zheng, Hanzhi Wang, Zhewei Wei, Jiajun Liu, Sibo Wang

Instant Graph Neural Networks for Dynamic Graphs

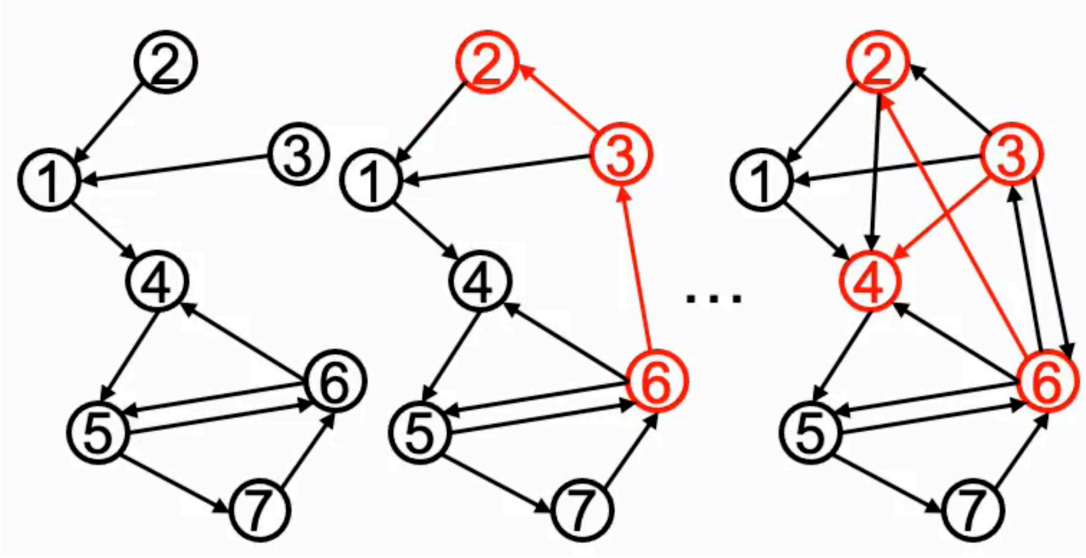
The problem setting is interesting and non-trivial: Continuous-Time Dynamic Graph (CTDG)

- How to instantly represent continuous changes of large-scale dynamic graphs with GNNs?
- Given an initial graph G_0
- CTDG focuses on a continuous graph evolving process: $\{event_1, \ldots, event_n\}$
- $event_i = (type_i, e_i)$ is an edge operation of $type_i$ (insert or delete) on edge e_i at time i

Previous dynamic graph problem setting: Discrete-Time Dynamic Graph (DTDG)

- A sequence of static graph snapshots taken at the time intervals
- A graph snapshot in a DTDG can be treated as the graph captured at time i of a CTDG, but the detailed evolving process is missing

Instant Graph Neural Networks for Dynamic Graphs



An example of dynamic graph

In real-world applications, the graph is constantly changing, making it time-consuming to retrain a GNN.
How can we adaptively refine the node representation learned from past data?

GraphMAE: Self-Supervised Masked Graph Autoencoders

Zhenyu Hou, Xiao Liu, Yukuo Cen, Yuxiao Dong, Hongxia Yang, Chunjie Wang, Jie Tang

GraphMAE: Self-Supervised Masked Graph Autoencoders

Contrastive SSL has been the dominant approach in recent years.

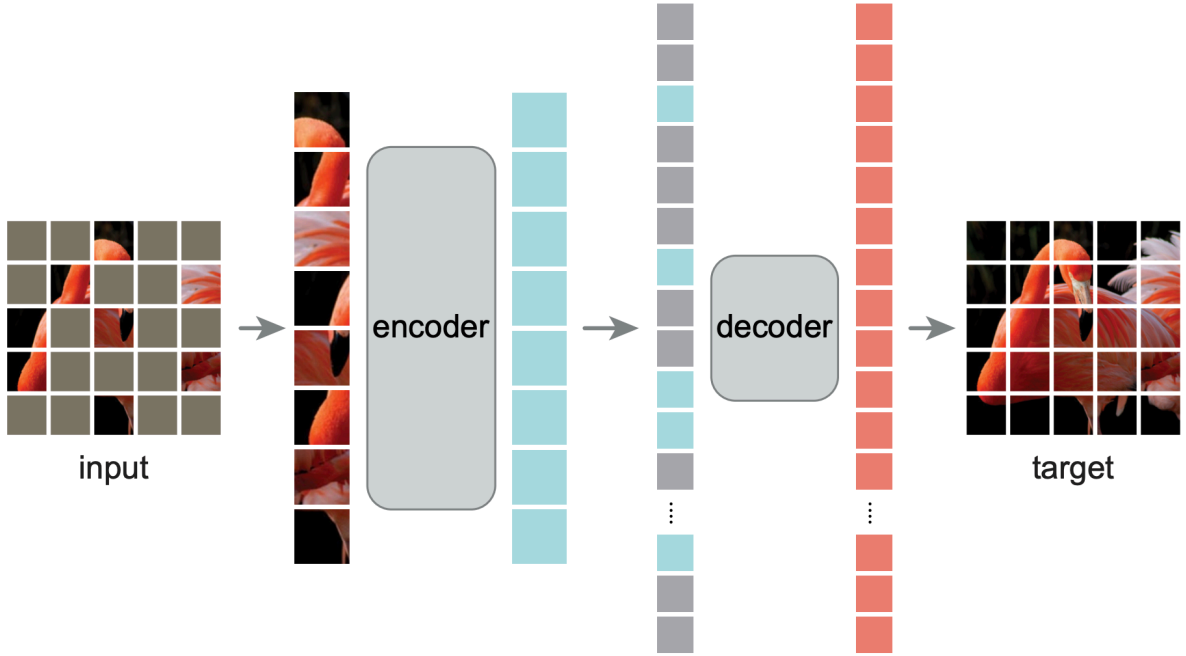
Contrastive learning heavily relies on complicated and elaborate designs,

- Negative sampling design
  - In-batch negatives (GRACE, GCA, GraphCL)
  - Dynamic queues as negatives (GCC,)
  - Shuffle node features as negatives (DGI, MVGRL)
- Architecture design
  - Asymmetric encoder, Projection head (BGRL, SimGRACE)
  - Feature de-correlation (CCA-SSG,)
- Data augmentation design
  - Node dropping, Edge perturbation, Subgraph Sampling (GraphCL, CCA-SSG, BGRL)
  - Graph Diffusion (MVGRL,), Random-walk (GCC,), Infomax Augmentation (Info-GCL)
  - …

- But, generative SSL can naturally avoid these issues

GraphMAE: Self-Supervised Masked Graph Autoencoders

- MAE[He, 2021] leads the revolution and opens the new era of generative SSL
  - Get rid of all constraints of contrastive SSL

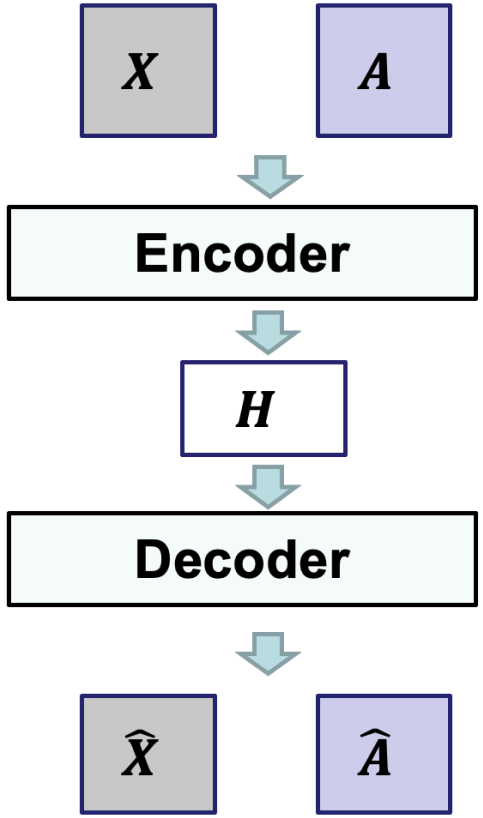


GraphMAE: Self-Supervised Masked Graph Autoencoders

Graph AutoEncoder

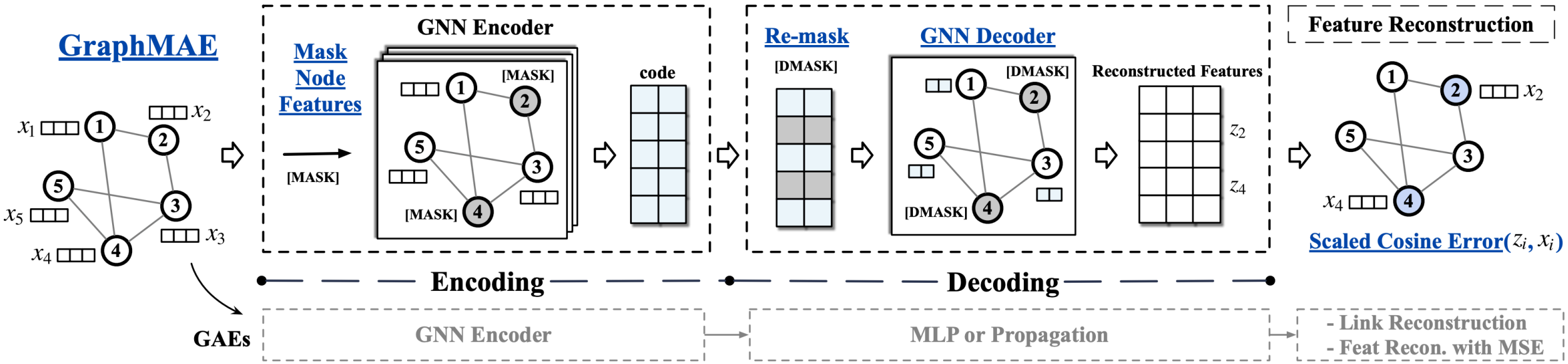
- $G = (V, A, X)$
  - $A \in \{0, 1\}^{N \times N}$: adjacency matrix,
  - $X \in \mathbb{R}^{N \times d}$: node features

- Encoding: $H = f_E(A, X)$,
- Decoding: $G' = f_D(A, H)$
- Reconstruction objectives:
  - graph structure (link)
  - Node features



GraphMAE: Self-Supervised Masked Graph Autoencoders

Overall pipeline

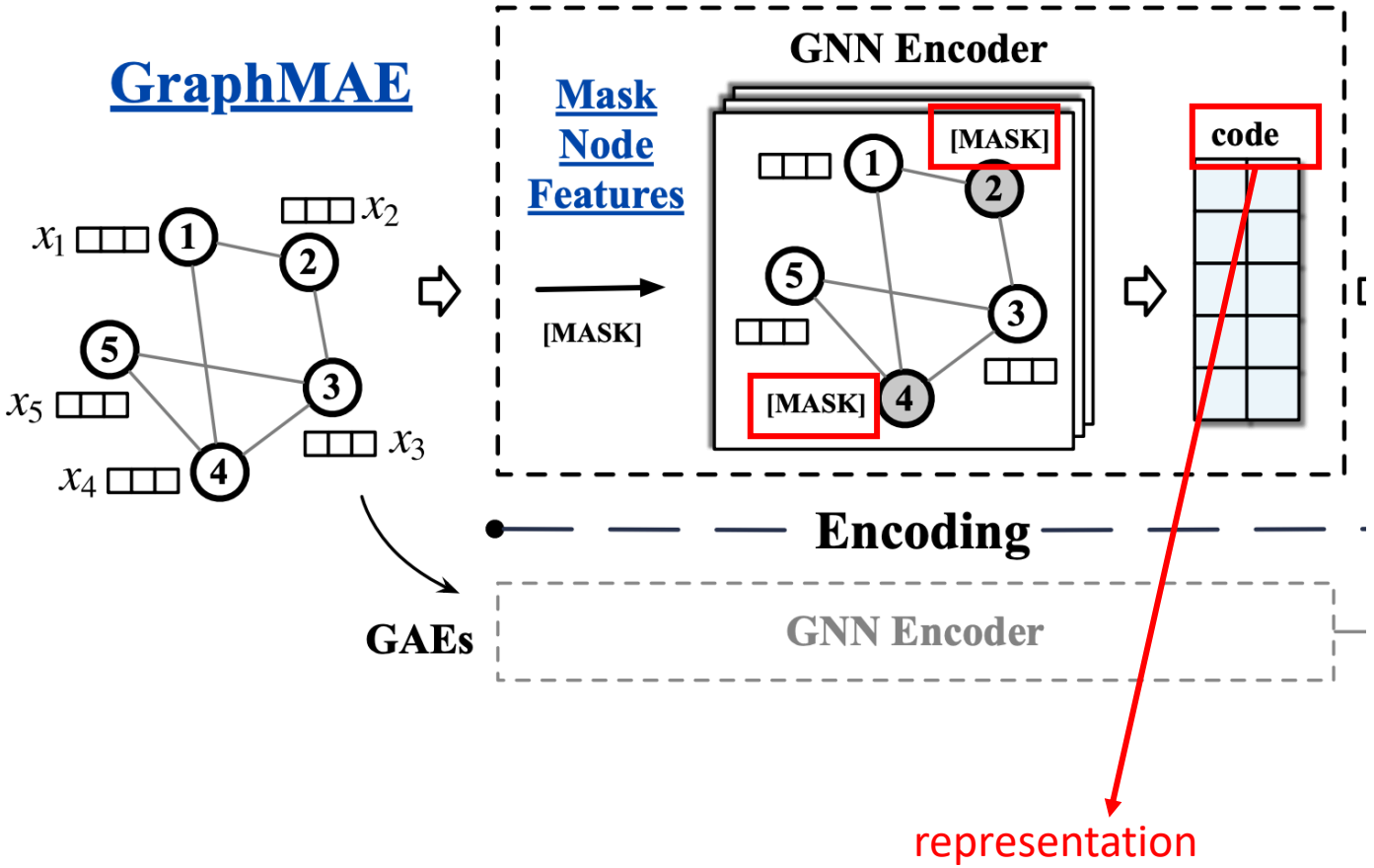


- Masked feature reconstruction
- GNN as decoder with re-mask decoding
- Scaled cosine error as the Criterion

GraphMAE: Self-Supervised Masked Graph Autoencoders

Feature construction as the learning objective
Masked feature reconstruction

- Sample a subset of nodes
- Replace node feature with [MASK]
- Reconstruct the feature



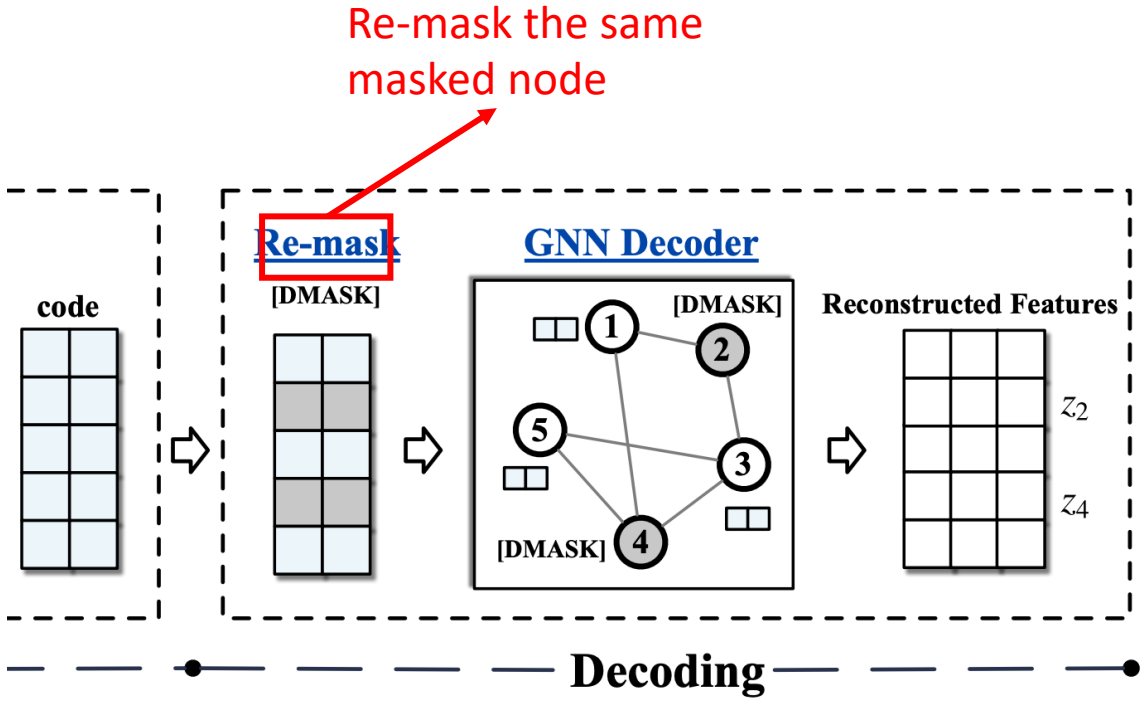
GraphMAE: Self-Supervised Masked Graph Autoencoders

Use GNN as the decoder

- A more expressive decoder helps reconstruct low informative features

Re-mask node features before decoder

- Re-mask the "masked" nodes



GraphMAE: Self-Supervised Masked Graph Autoencoders

MSE fails, especially for continuous features

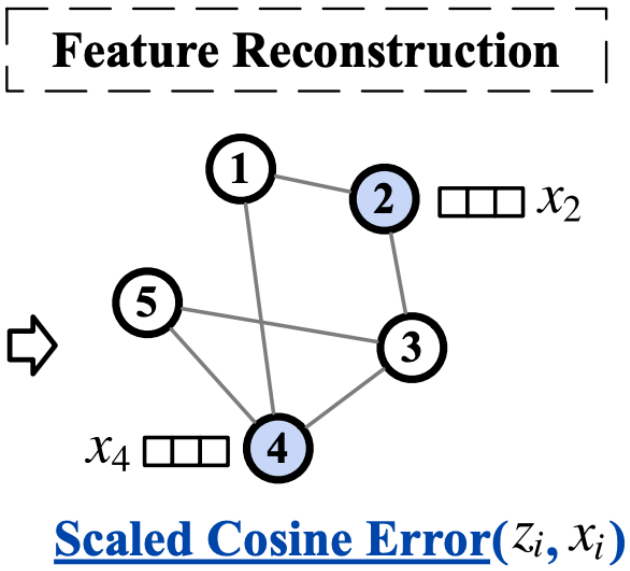
- *Sensitivity & low selectivity*

$$L_{MSE} = \frac{1}{|\tilde{V}|}\sum_{v_i \in \tilde{V}} (x_i - z_i)^2$$

Scaled cosine error as the criterion.

Cosine error & Scaled coefficient

$$\mathcal{L}_{\text{SCE}} = \frac{1}{|\widetilde{\mathcal{V}}|}\sum_{v_i \in \widetilde{\mathcal{V}}} \left(1 - \frac{\boldsymbol{x}_i^T \boldsymbol{z}_i}{\|\boldsymbol{x}_i\| \cdot \|\boldsymbol{z}_i\|}\right)^{\gamma},\ \gamma \geq 1,$$



GraphMAE: Self-Supervised Masked Graph Autoencoders

Node classification

Table 1: Experiment results in unsupervised representation learning for node classification. We report Micro-F1(%) score for PPI and accuracy(%) for the other datasets.

	Dataset	Cora	CiteSeer	PubMed	Ogbn-arxiv	PPI	Reddit
Supervised	GCN	81.5	70.3	79.0	71.74±0.29	75.7±0.1	95.3±0.1
	GAT	83.0±0.7	72.5±0.7	79.0±0.3	72.10±0.13	97.30±0.20	96.0±0.1
Self-supervised	GAE	71.5±0.4	65.8±0.4	72.1±0.5	-	-	-
	GPT-GNN	80.1±1.0	68.4±1.6	76.3±0.8	-	-	-
	GATE	83.2±0.6	71.8±0.8	80.9±0.3	-	-	-
	DGI	82.3±0.6	71.8±0.7	76.8±0.6	70.34±0.16	63.80±0.20	94.0±0.10
	MVGRL	83.5±0.4	73.3±0.5	80.1±0.7	-	-	-
	GRACE[1]	81.9±0.4	71.2±0.5	80.6±0.4	71.51±0.11	69.71±0.17	94.72±0.04
	BGRL[1]	82.7±0.6	71.1±0.8	79.6±0.5	71.64±0.12	73.63±0.16	94.22±0.03
	InfoGCL	83.5±0.3	**73.5±0.4**	79.1±0.2	-	-	-
	CCA-SSG[1]	84.0±0.4	73.1±0.3	81.0±0.4	71.24±0.20	73.34±0.17	95.07±0.02
	GraphMAE	**84.2±0.4**	73.4±0.4	**81.1±0.4**	**71.75±0.17**	**74.50±0.29**	**96.01±0.08**

GraphMAE: Self-Supervised Masked Graph Autoencoders

Graph classification

Table 2: Experiment results in unsupervised representation learning for graph classification. We report accuracy(%) for all datasets.

	Dataset	IMDB-B	IMDB-M	PROTEINS	COLLAB	MUTAG	REDDIT-B	NCI1
Supervised	GIN	75.1±5.1	52.3±2.8	76.2±2.8	80.2±1.9	89.4±5.6	92.4±2.5	82.7±1.7
	DiffPool	72.6±3.9	-	75.1±3.5	78.9±2.3	85.0±10.3	92.1±2.6	-
Graph Kernels	WL	72.30±3.44	46.95±0.46	72.92±0.56	-	80.72±3.00	68.82±0.41	80.31±0.46
	DGK	66.96±0.56	44.55±0.52	73.30±0.82	-	87.44±2.72	78.04±0.39	80.31±0.46
Self-supervised	graph2vec	71.10±0.54	50.44±0.87	73.30±2.05	-	83.15±9.25	75.78±1.03	73.22±1.81
	Infograph	73.03±0.87	49.69±0.53	74.44±0.31	70.65±1.13	89.01±1.13	82.50±1.42	76.20±1.06
	GraphCL	71.14±0.44	48.58±0.67	74.39±0.45	71.36±1.15	86.80±1.34	89.53±0.84	77.87±0.41
	JOAO	70.21±3.08	49.20±0.77	74.55±0.41	69.50±0.36	87.35±1.02	85.29±1.35	78.07±0.47
	GCC	72.0	49.4	-	78.9	-	**89.8**	-
	MVGRL	74.20±0.70	51.20±0.50	-	-	89.70±1.10	84.50±0.60	-
	InfoGCL	75.10±0.90	51.40±0.80	-	80.00±1.30	**91.20±1.30**	-	80.20±0.60
	GraphMAE	**75.52±0.66**	**51.63±0.52**	**75.30±0.39**	**80.32±0.46**	88.19±1.26	88.01±0.19	**80.40±0.30**

GraphMAE: Self-Supervised Masked Graph Autoencoders

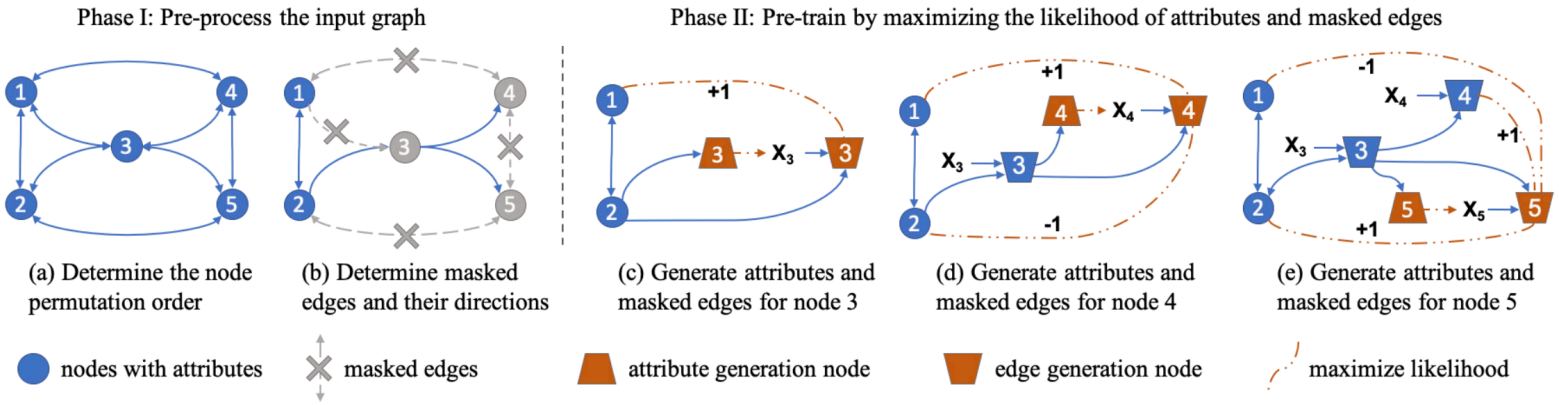
Transfer learning

Table 3: Experiment results in transfer learning on molecular property prediction benchmarks. The model is first pre-trained on ZINC15 and then finetuned on the following datasets. We report ROC-AUC(%) scores.

	BBBP	Tox21	ToxCast	SIDER	ClinTox	MUV	HIV	BACE	Avg.
No-pretrain	65.5±1.8	74.3±0.5	63.3±1.5	57.2±0.7	58.2±2.8	71.7±2.3	75.4±1.5	70.0±2.5	67.0
ContextPred	64.3±2.8	75.7±0.7	63.9±0.6	60.9±0.6	65.9±3.8	75.8±1.7	77.3±1.0	79.6±1.2	70.4
AttrMasking	64.3±2.8	**76.7±0.4**	**64.2±0.5**	61.0±0.7	71.8±4.1	74.7±1.4	77.2±1.1	79.3±1.6	71.1
Infomax	68.8 ±0.8	75.3 ±0.5	62.7 ±0.4	58.4 ±0.8	69.9±3.0	75.3 ±2.5	76.0 ±0.7	75.9 ±1.6	70.3
GraphCL	69.7±0.7	73.9±0.7	62.4±0.6	60.5±0.9	76.0±2.7	69.8±2.7	**78.5±1.2**	75.4±1.4	70.8
JOAO	70.2±1.0	75.0±0.3	62.9±0.5	60.0±0.8	81.3±2.5	71.7±1.4	76.7±1.2	77.3±0.5	71.9
GraphLoG	**72.5±0.8**	75.7±0.5	63.5±0.7	**61.2±1.1**	76.7±3.3	76.0±1.1	77.8±0.8	**83.5±1.2**	73.4
GraphMAE	72.0±0.6	75.5±0.6	64.1±0.3	60.3±1.1	**82.3±1.2**	**76.3±2.4**	77.2±1.0	83.1±0.9	**73.8**

GraphMAE: Self-Supervised Masked Graph Autoencoders

- Although this method contains some tricks, the total pipeline is easy to follow.
- It can perform well on PPI dataset which has very sparse node features (40% all-zero)
- It can be considered as a simplified version of GPT-GNN[1] which reconstructs node feature and edge as SSL task



- Generative task has great potential for us to explore
  - Generative task as SSL tasks
  - Autoregressive. Protein design, simulation
  - Diffusion model. DALLE-2, Molecule generation

[1] Hu, Ziniu, et al. "GPT-GNN: Generative pre-training of graph neural networks."