

Graph Prompting for Graph Learning Models: Recent Advances and Future Directions

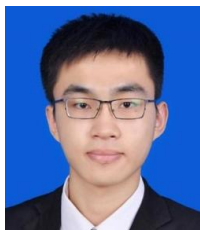
Presenters: Xingbo Fu, Zehong Wang, Jundong Li

University of Virginia, University of Notre Dame, University of Connecticut



KDD 2025 Tutorial
August 2025

Presenters



Xingbo Fu is a PhD candidate at the University of Virginia. His research interests focus mainly on graph learning, federated learning, and healthcare informatics. He has publications in major conferences and journals, such as KDD, ICLR, AAAI, ECAI, ICHI, SGIKDD Explorations, and TMLR. He brings industry experience through internships at Amazon and Netflix.



Zehong Wang is a PhD candidate at the University of Notre Dame. His research interests are broadly in machine learning and data mining, with a particular focus on foundation models. His work has appeared at top-tier conferences such as KDD, NeurIPS, ICML, IJCAI, NAACL, ACL, SDM, and WSDM. He has also applied his expertise in industry as an Applied Scientist Intern at Amazon.



Jundong Li is an Associate Professor at the University of Virginia. His research interests are generally in data mining and machine learning, with a focus on graph machine learning, trustworthy/safe machine learning, and LLMs. He has received several prestigious awards, including the SIGKDD Rising Star Award (2024), the SIGKDD Best Research Paper Award (2022), and the NSF CAREER Award (2022).



Tutorial Outline

| Time | Section |
|-----------------|---|
| 1:00pm ~ 1:30pm | Section 1: Introduction |
| 1:30pm ~ 2:00pm | Section 2: Graph Pre-training for Graph Prompting |
| 2:00pm ~ 3:00pm | Section 3: Techniques in Graph Prompting |
| 3:00pm ~ 3:30pm | Coffee Break |
| 3:30pm ~ 3:45pm | Section 4: Summary and Future Directions |
| 3:45pm ~ 4:00pm | Section 5: Q&A |

Tutorial Website & Survey Paper

Tutorial website

Slides & more information about this tutorial

<https://www.xingbofu.com/tutorials/kdd25-graph-promptng>



Graph Prompting for Graph Learning Models: Recent Advances and Future Directions

Xingbo Fu, Zehong Wang, Zihan Chen, Jiazheng Li, Yaochen Zhu, Zhenyu Lei, Cong Shen, Yanfang Ye,
Chuxu Zhang, Jundong Li

<https://arxiv.org/abs/2506.08326>



Section 1

Introduction

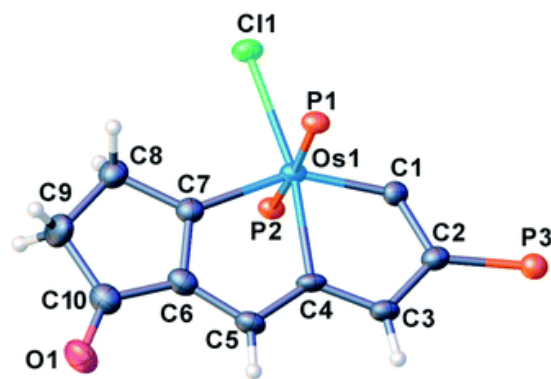
Jundong Li

Associate Professor

University of Virginia



Graphs



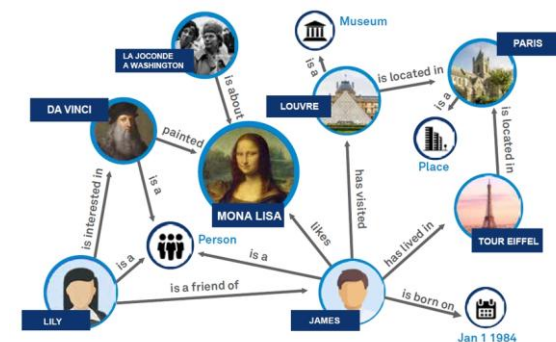
Molecular graphs



Social network



Transportation network

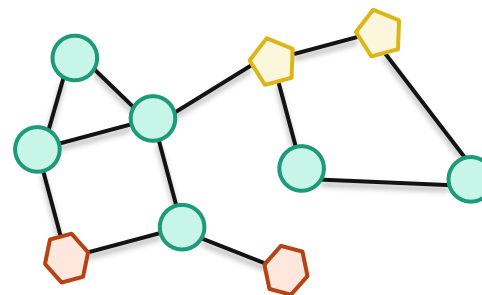


Knowledge graph

Modeling Real-World Data as Graphs

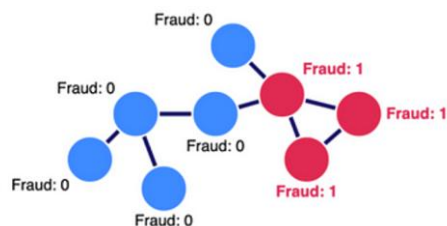
A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$

- $\mathcal{V} = \{v_1, v_2, \dots, v_N\}$: the node set
- $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$: the edge set
- $\mathbf{X} \in \mathbb{R}^{N \times F}$: node feature matrix
- $\mathbf{A} \in \{0,1\}^{N \times N}$: adjacency matrix

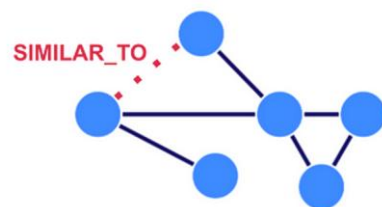


A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$

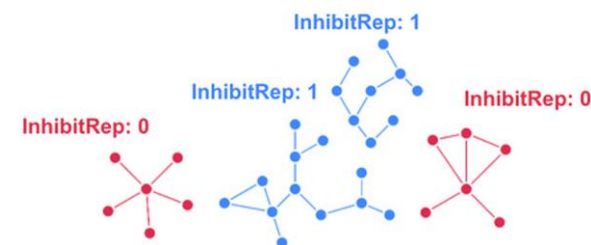
Tasks for graphs



Node-level tasks



Edge-level tasks

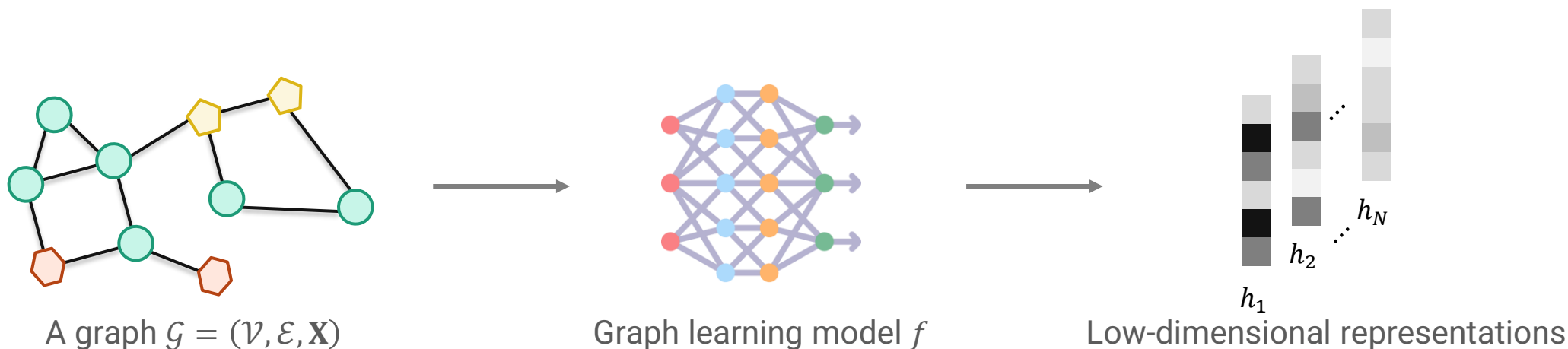


Graph-level tasks

Graph Representation Learning

Learn expressive graph representations by training graph learning models

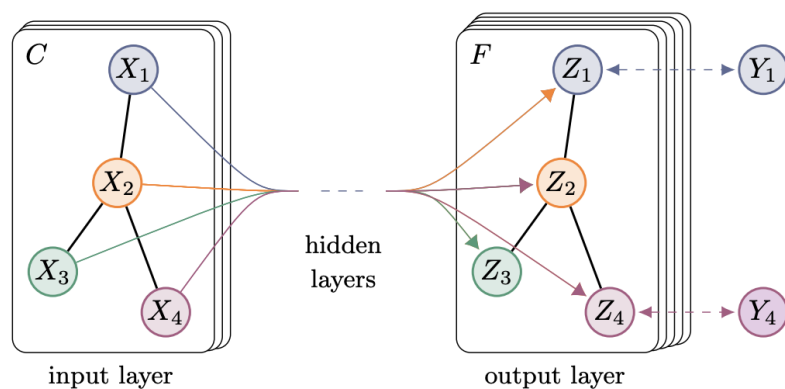
- Graph embeddings can be used for downstream tasks



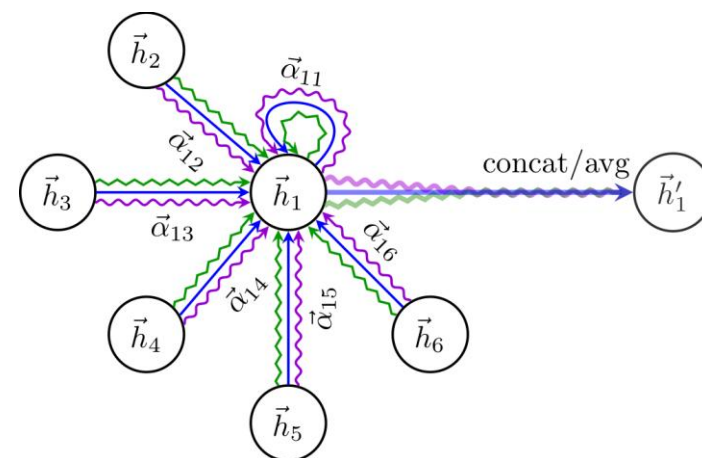
Graph Learning Models

Graph neural networks (GNN)

- Iteratively aggregate information in the neighborhood via the message-passing mechanism



Graph convolutional networks (GCN)¹

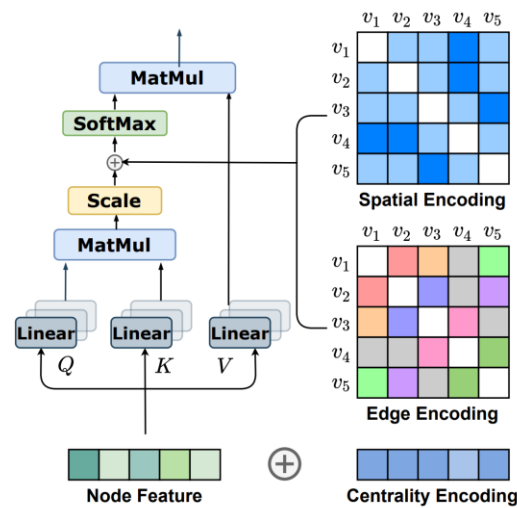


Graph attention networks (GAT)²

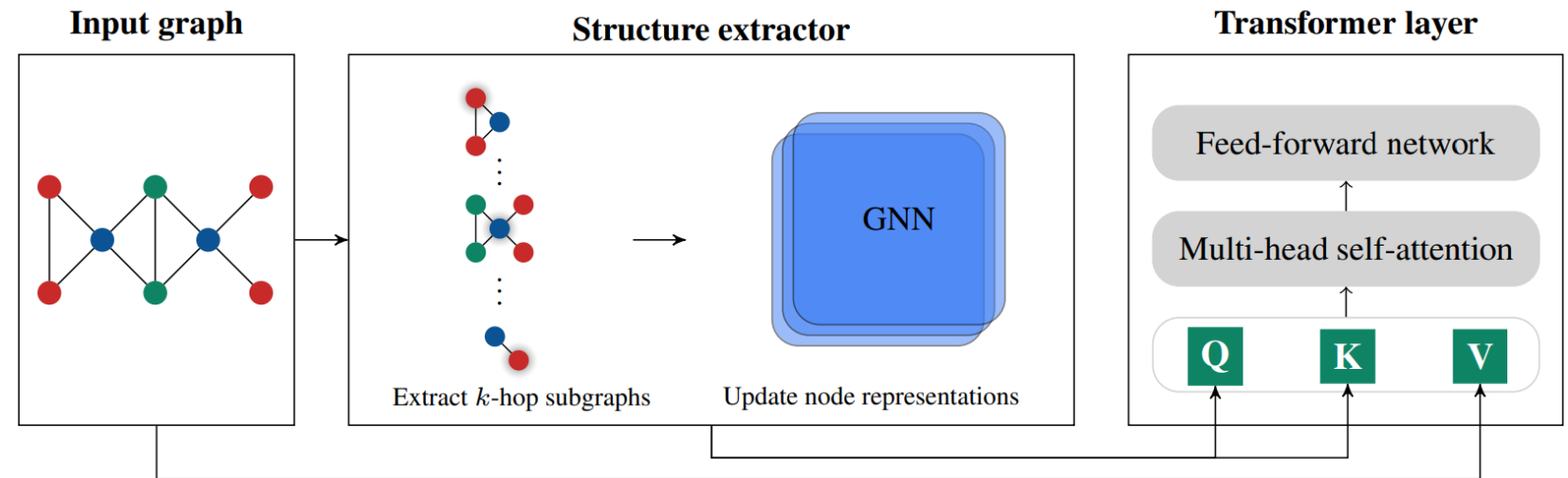
Graph Learning Models

Graph Transformers (GT)

- Apply Transformer architectures to graph data



Graphormer¹



Structure-aware Transformer (SAT)²

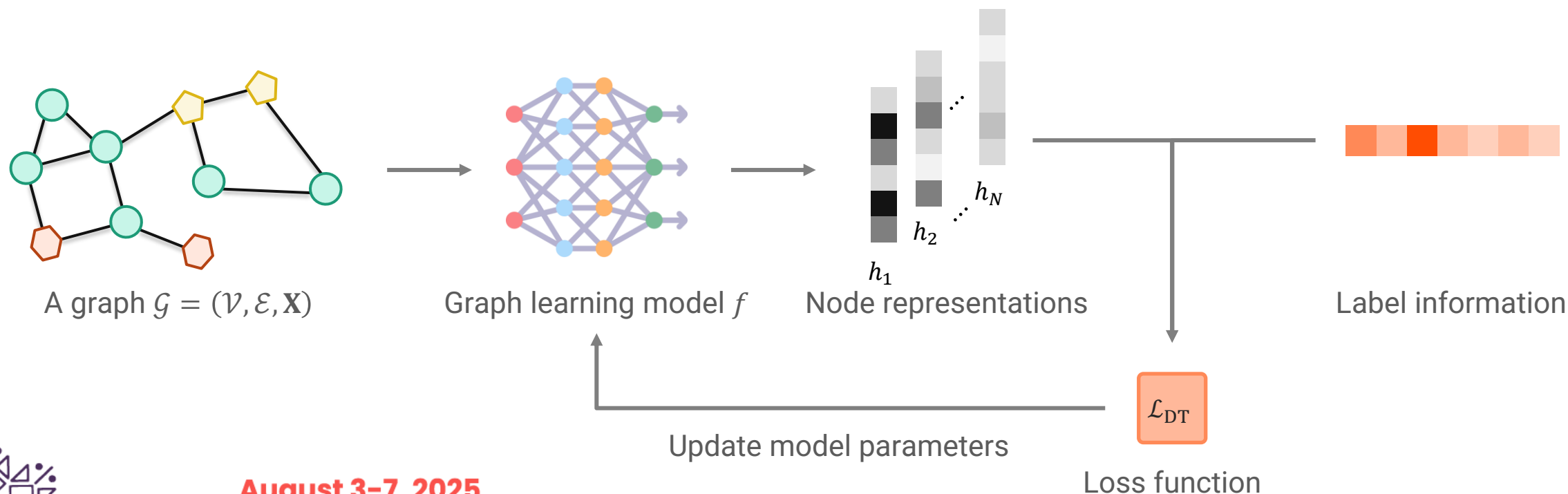
[1] Ying, Chengxuan, et al. "Do Transformers Really Perform Badly for Graph Representation?." NeurIPS 2021.

[2] Chen, Dexiong, et al. "Structure-Aware Transformer for Graph Representation Learning." ICML 2022.

Traditional Training Paradigm for Graph Learning Models

End-to-end training

- Optimize graph learning models based on abundant label information



Limitations of Traditional Training Paradigm

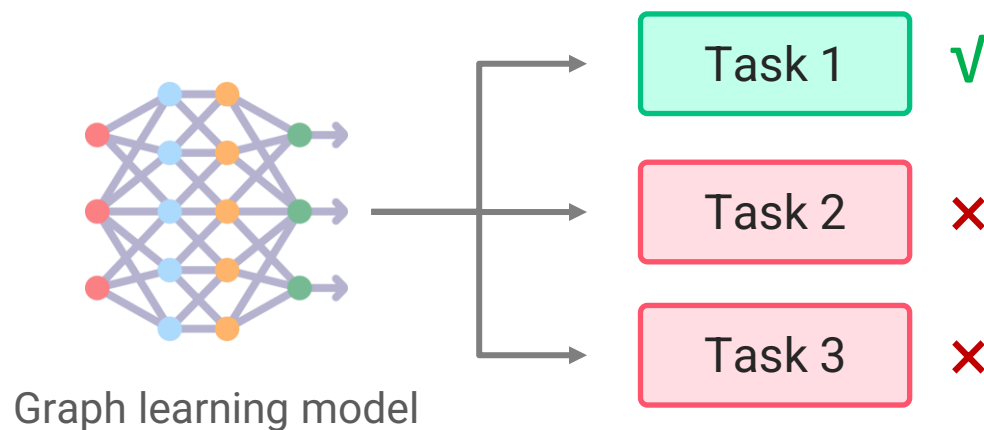
Rely heavily on label information

- Sufficient labeled graph data may be inaccessible in practice



Poor generalization

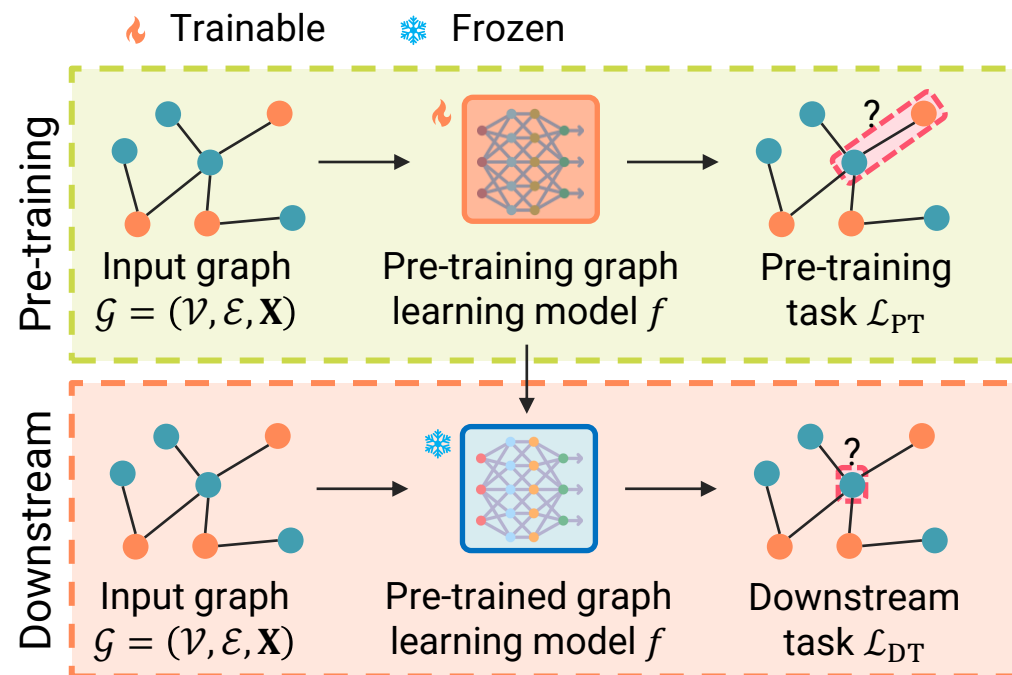
- Graph learning models cannot be well generalized to other downstream tasks



Graph Pre-training via Self-supervised Learning

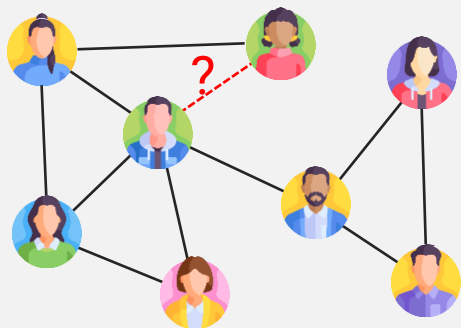
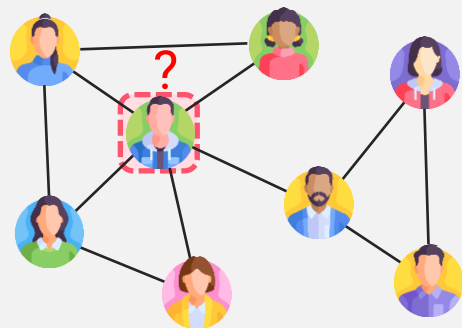
Learn generalizable graph embeddings without label information

- Pre-training stage
 - Graph learning models are learned by solving hand-crafted auxiliary tasks \mathcal{L}_{PT}
 - Supervision signals are acquired from graph data itself
- Downstream tasks
 - Pre-trained graph learning models directly generate graph embeddings used for downstream tasks \mathcal{L}_{DT}



Objective Gap between Pre-training and Downstream Tasks

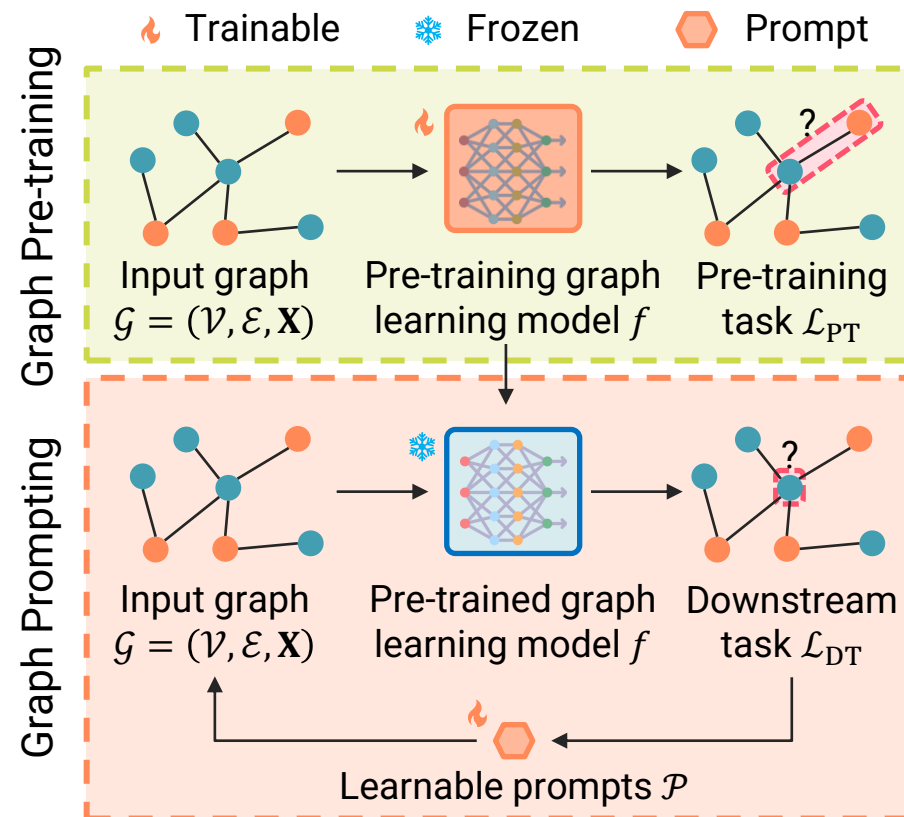
Example: edge prediction → node classification

| Phase | Pre-training stage | Downstream stage |
|---------------------|--|---|
| Illustration |  |  |
| Task | Edge prediction | Node classification |
| Involved embeddings | A pair of nodes | A single node |
| Prediction | Edge probabilities | Node class probabilities |

The “Pre-training, Prompting” Scheme

Two-stage adaptation of pre-trained graph learning models

- **Stage 1:** graph pre-training via self-supervised learning
- A graph learning model f is pre-trained on a pre-training task \mathcal{L}_{PT}
- **Stage 2:** graph prompting for adaptation
- Adapt the pre-trained graph learning model f for the downstream task \mathcal{L}_{DT} by learning extra prompts \mathcal{P}
- The pre-trained graph learning model f keeps frozen

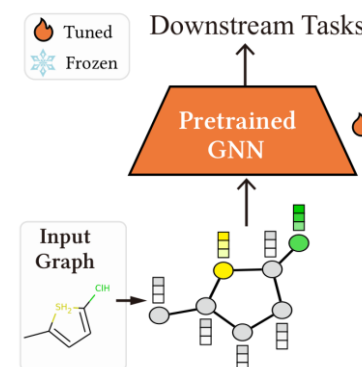


The “pre-training, prompting” scheme for graph learning models

Comparison between Graph Fine-tuning & Graph Prompting

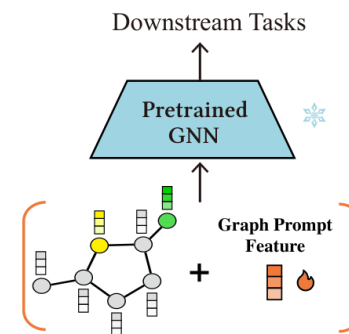
Graph fine-tuning

- Adapt pre-trained graph learning models by further tuning model parameters
- Tunable parameters: the whole graph learning model
- Unchanged part: graph data



Graph prompting

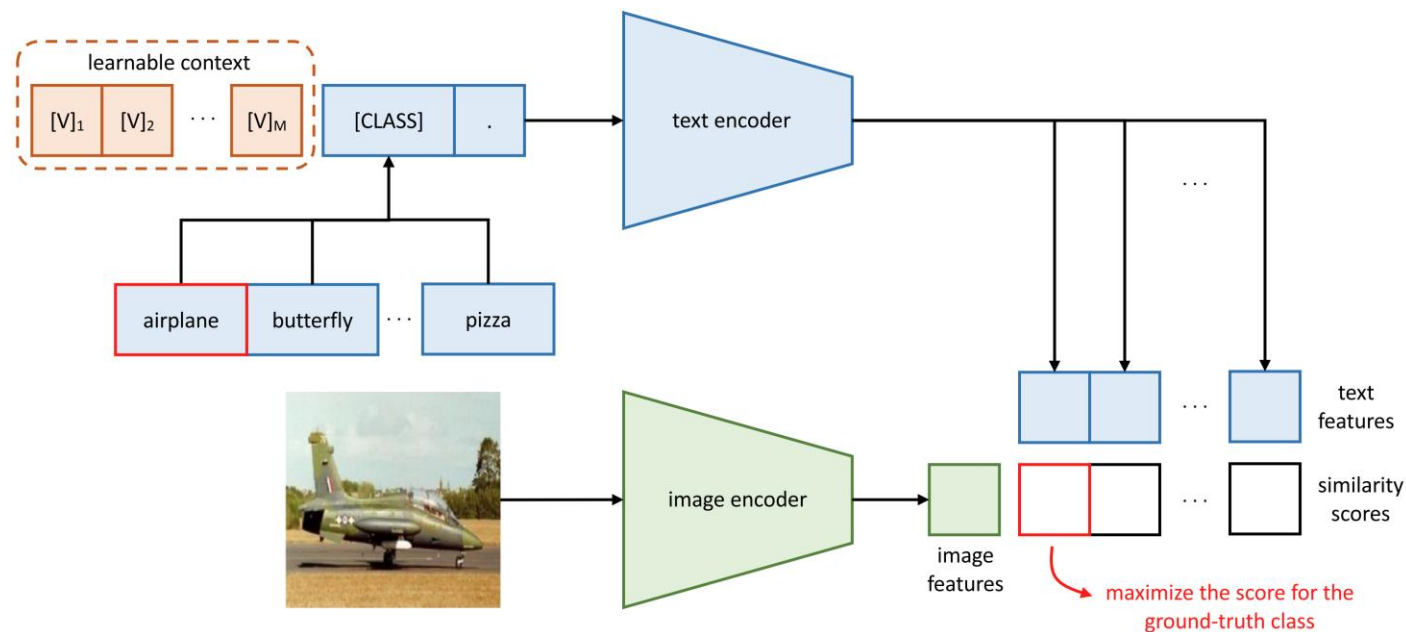
- Adapt pre-trained graph learning models by learning extra prompts
- Tunable parameters: additional prompts
- Unchanged part: the pre-trained graph learning model



Prompting in NLP and CV

Learning to modify the input data with extra trainable prompts

- Prompting in NLP: learn extra trainable tokens

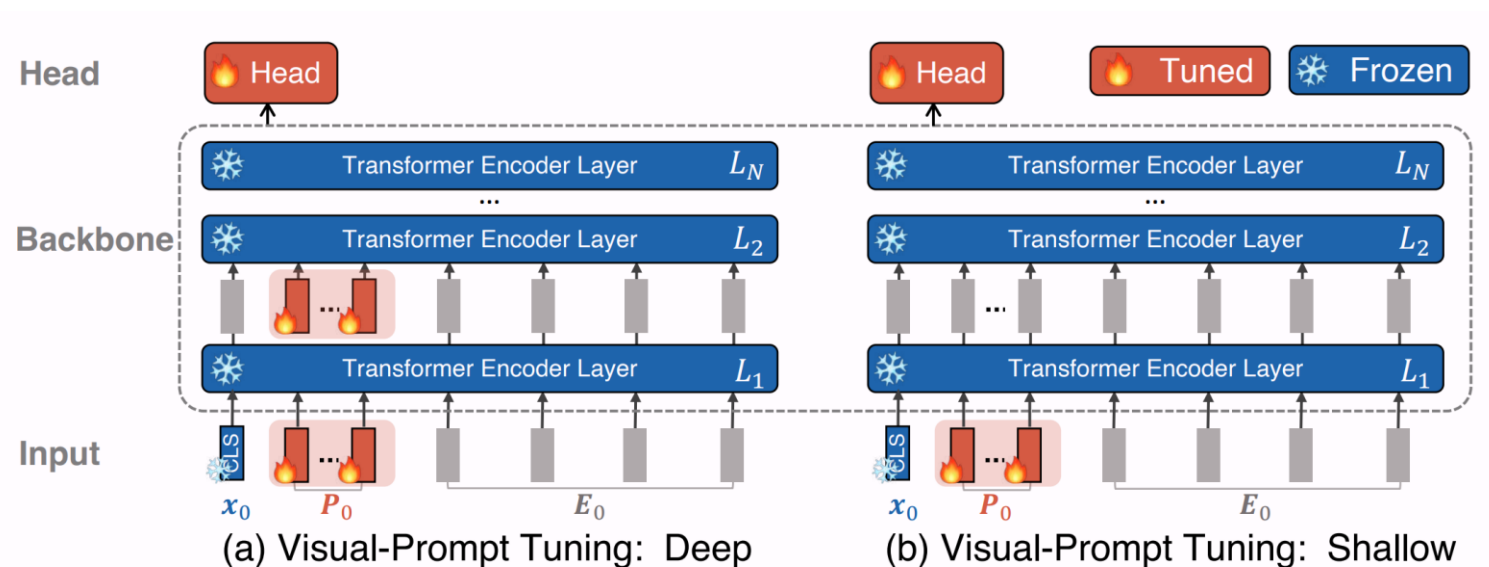


Context optimization (CoOP)¹

Prompting in NLP and CV

Learning to modify the input data with extra trainable prompts

- Prompting in CV: learn extra trainable patches



Visual prompt tuning (VPT)¹

Challenges in Prompting for Graph Learning Models

Non-Euclidean graph data

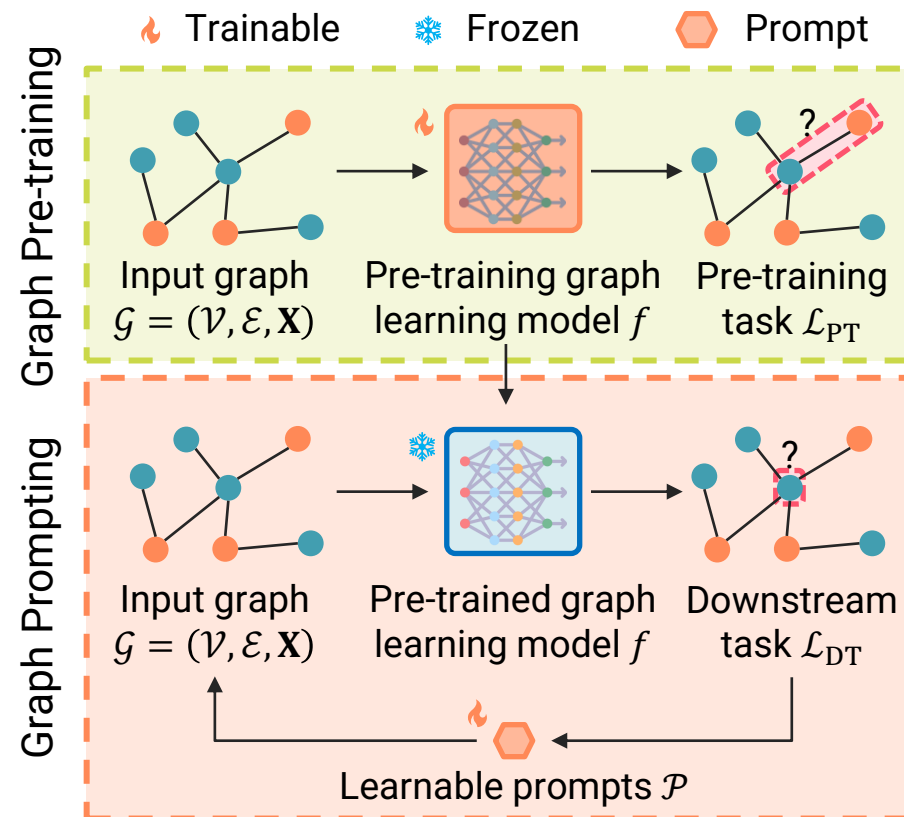
- Important structural information in graph data

Pre-training compatibility

- Pre-training strategies: generative/contrastive/...
- Should be compatible with various pre-training strategies

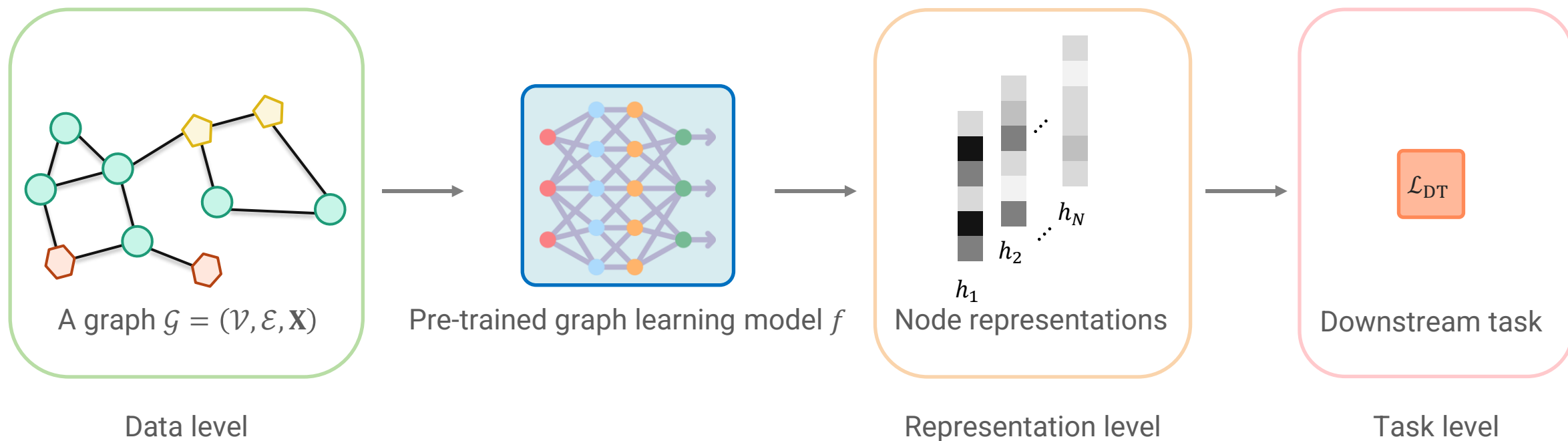
Downstream task universality

- Downstream tasks: node-level/graph-level/...
- Should be universal for different downstream tasks



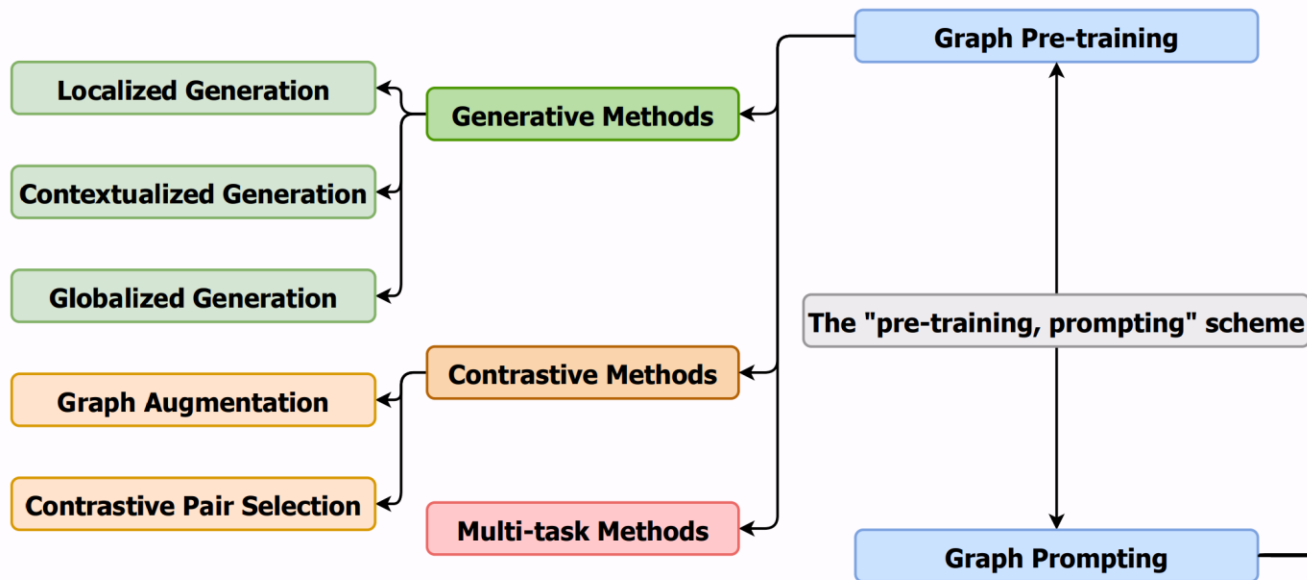
The "pre-training, prompting" scheme
for graph learning models

Where Can We Design Prompts for Graph Prompting?

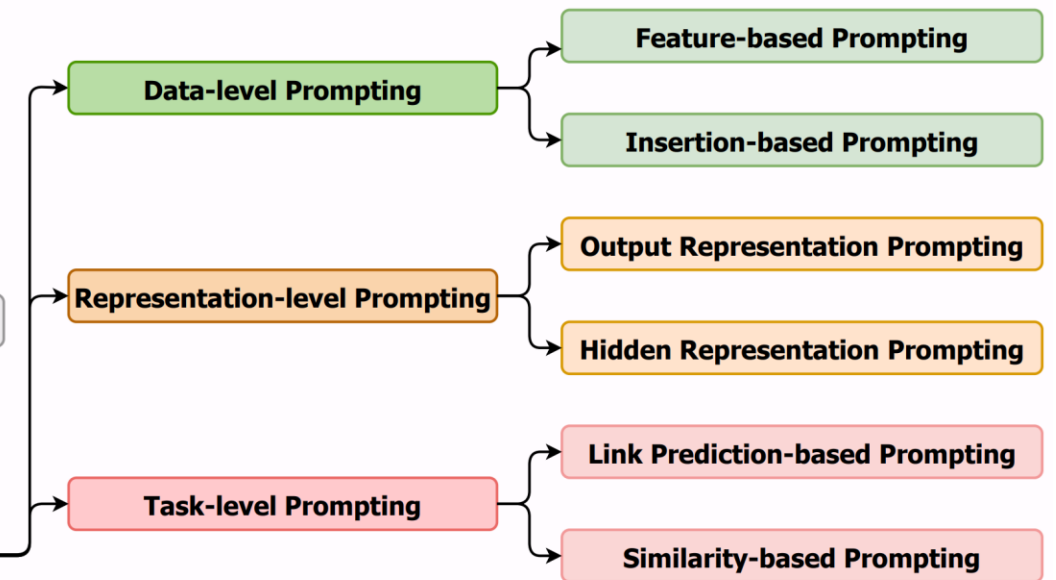


Key Techniques in the “Pre-training, Prompting” Scheme

Section 2



Section 3



Section 2

Graph Pre-training for Graph Prompting

Zehong Wang

PhD Candidate

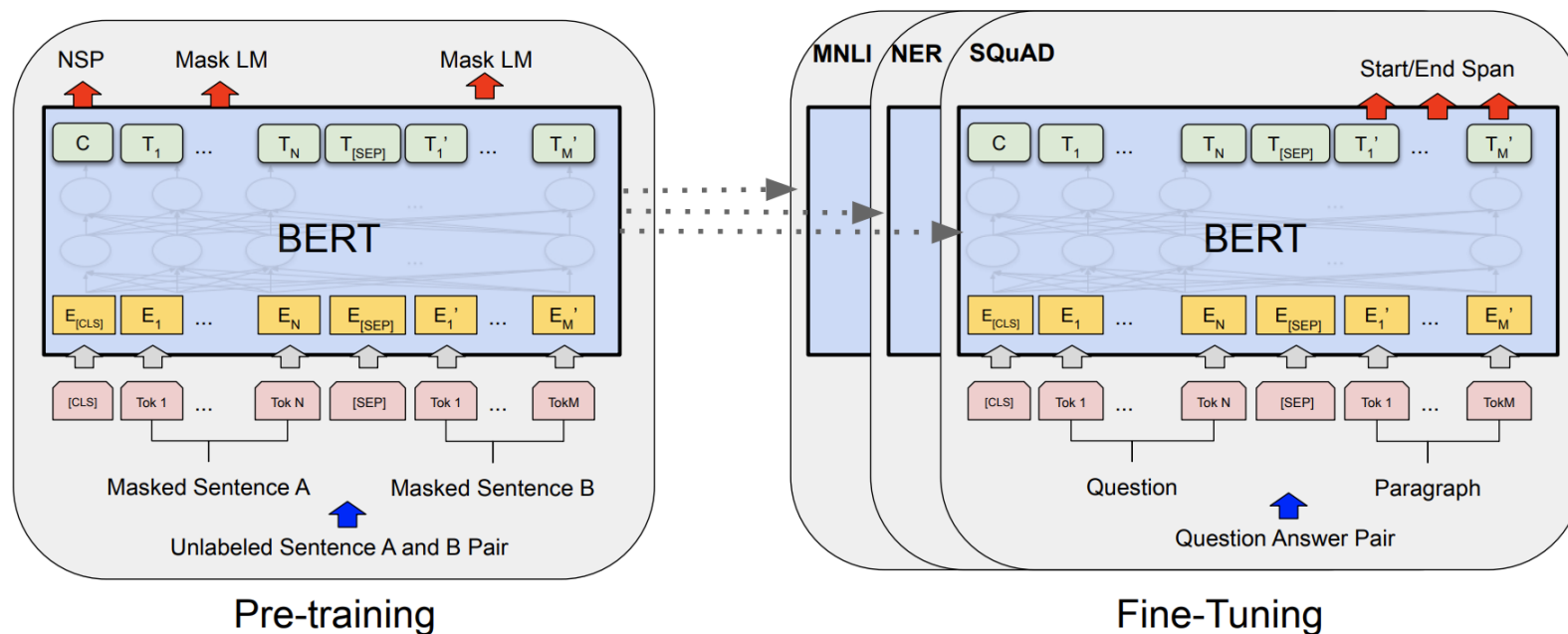
University of Notre Dame



From Supervised Learning to Self-supervised Learning

Generative Pre-training¹

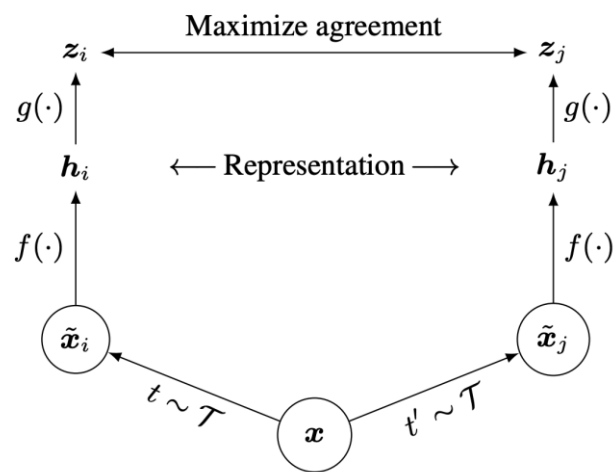
- Generate the masked parts
- Masked Token Prediction
- Next Sentence Prediction



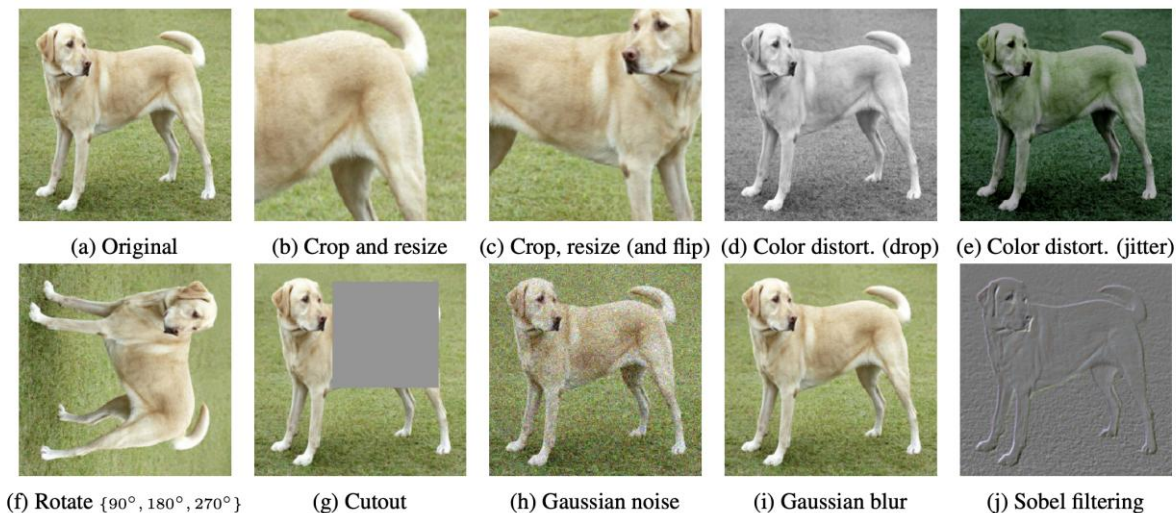
From Supervised Learning to Self-supervised Learning

Contrastive Pre-training¹

- Pull positives together and push negatives apart for discriminative representations



Contrastive learning¹

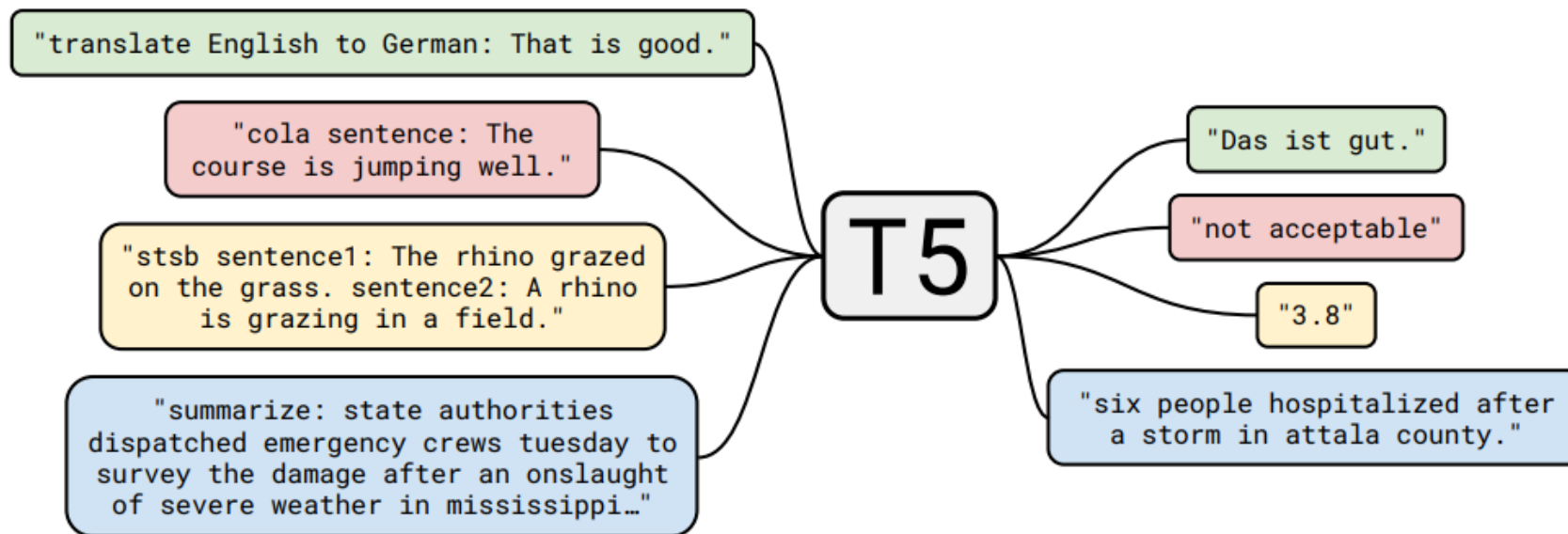


Data augmentation for harder tasks

From Supervised Learning to Self-supervised Learning

Multi-Task Pre-training¹

- A single model to optimize multiple tasks, improving model capability and generalization

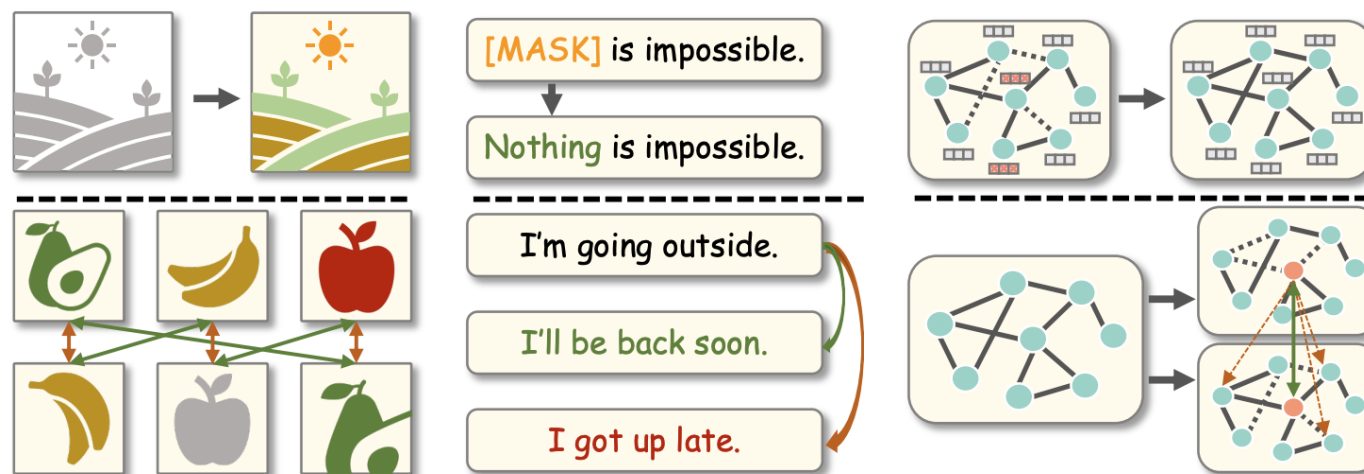


Pre-Training on Graphs

We need to carefully design pre-training tasks on graphs!

How to design pre-training tasks on graphs?

- Can we directly apply pre-training tasks on CV and NLP to graphs?
- Image/Text is in the Euclidean space
- Graph is in the non-Euclidean space

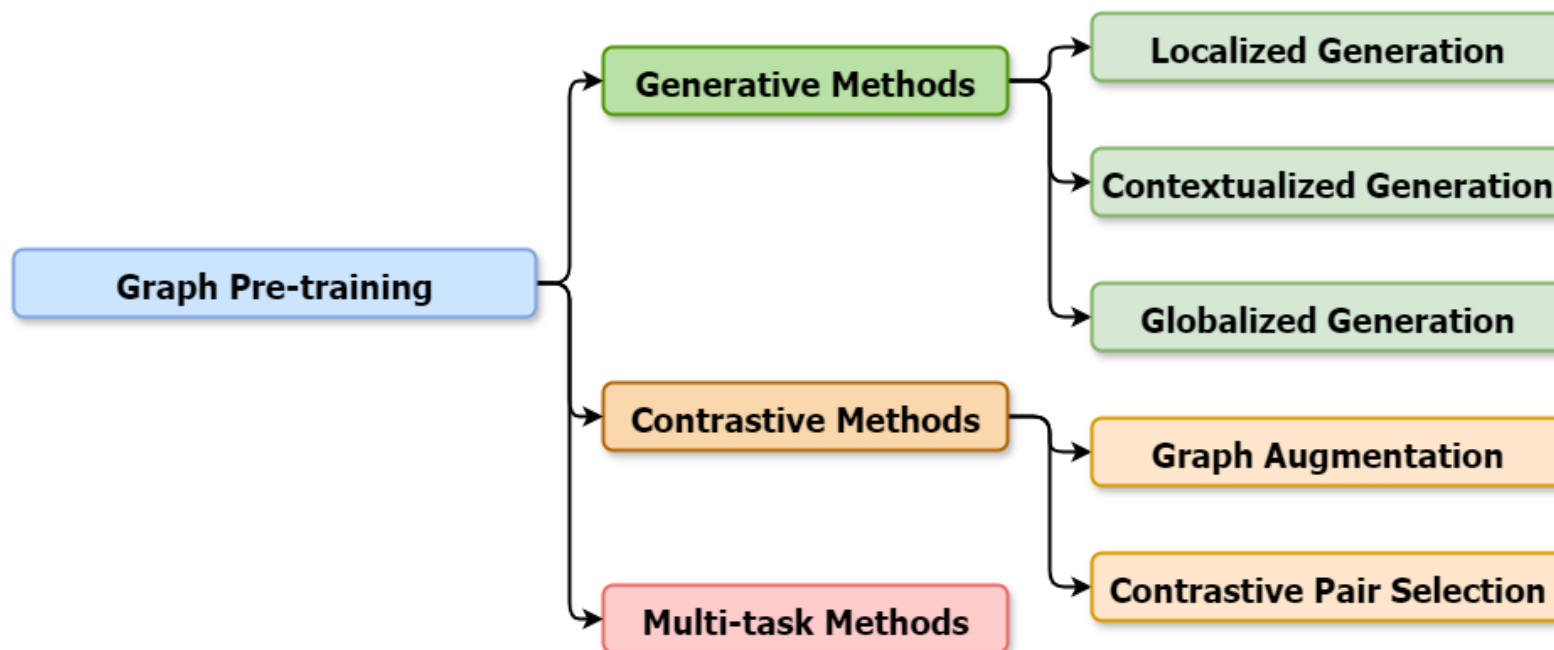


Toy examples of different pre-training tasks in CV, NLP, and graph learning¹

August 3-7, 2025

[1] Liu, Yixin, et al. "Graph Self-Supervised Learning: A Survey." TKDE 2022.

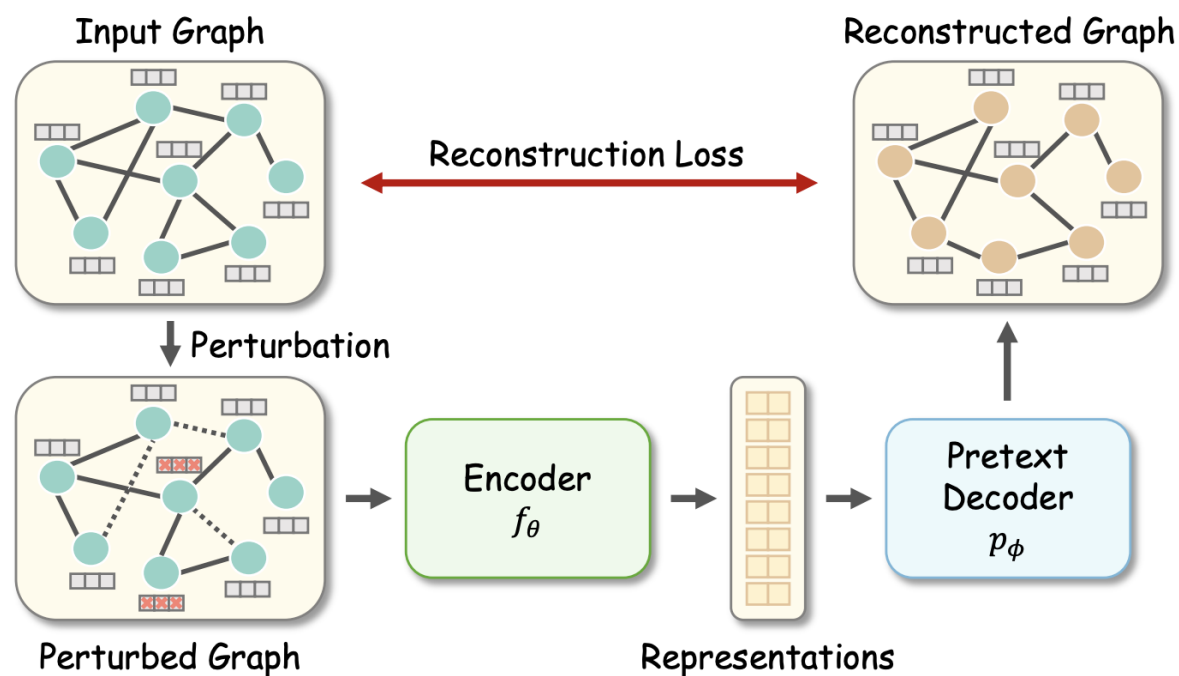
Taxonomy of Graph Pre-Training Techniques



Generative Methods

Intuition: Corrupt the graph and reconstruct the corrupted parts

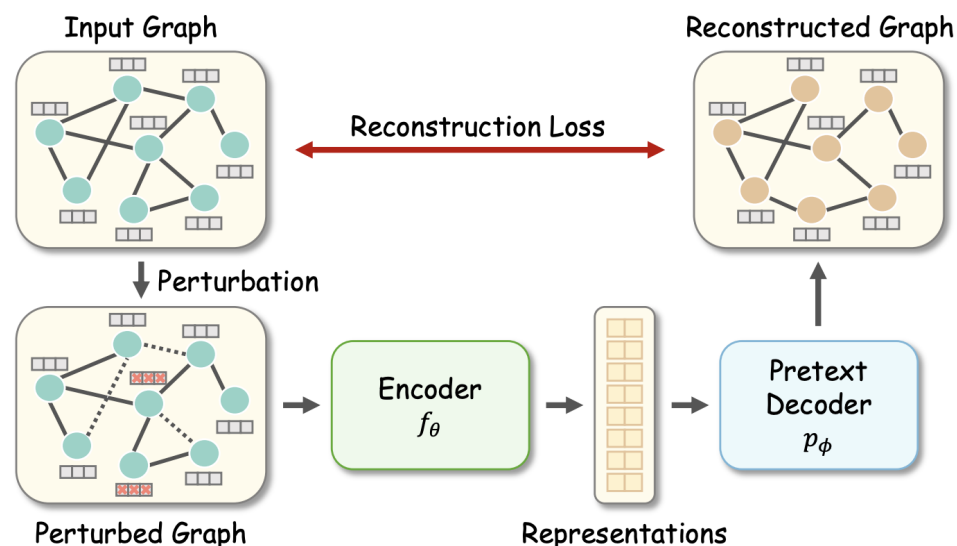
- Three types of generative methods
 - Localized generation
 - Contextualized generation
 - Globalized generation



An illustration of generative methods for graph pre-training¹

Generative Methods

Intuition: Corrupt the graph and reconstruct the corrupted parts



\mathcal{G} : the original graph

$\tilde{\mathcal{G}}$: the corrupted graph

θ : model parameters

Optimize the loss function: $\mathcal{L}_{PT} = \min_{\theta} \mathcal{L}(f(\tilde{\mathcal{G}}; \theta), \rho(\mathcal{G}))$

Reconstruction

Encode corrupted graph

Define reconstruction target

Generative Methods

Localized generation: reconstruct node features

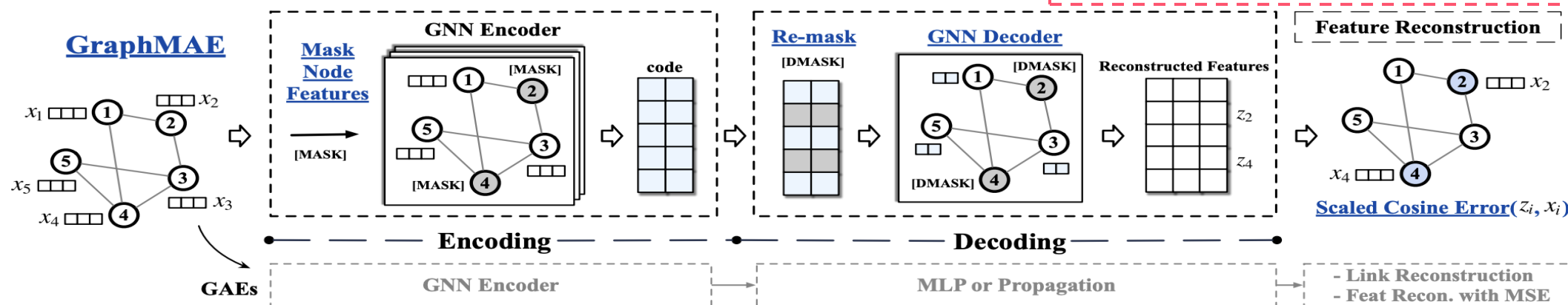
- GraphMAE¹: Self-Supervised Masked Graph Autoencoders

- Corruption is to mask the node features $\tilde{\mathcal{G}} = \text{mask}_{\text{node}}(\mathcal{G})$

- Encoding $f(\tilde{\mathcal{G}}; \theta)$ is to encode the corrupted graphs **twice**

- Reconstruction is to reconstruct the masked node features

Could be extended to centrality level or clustering coefficient



August 3-7, 2025

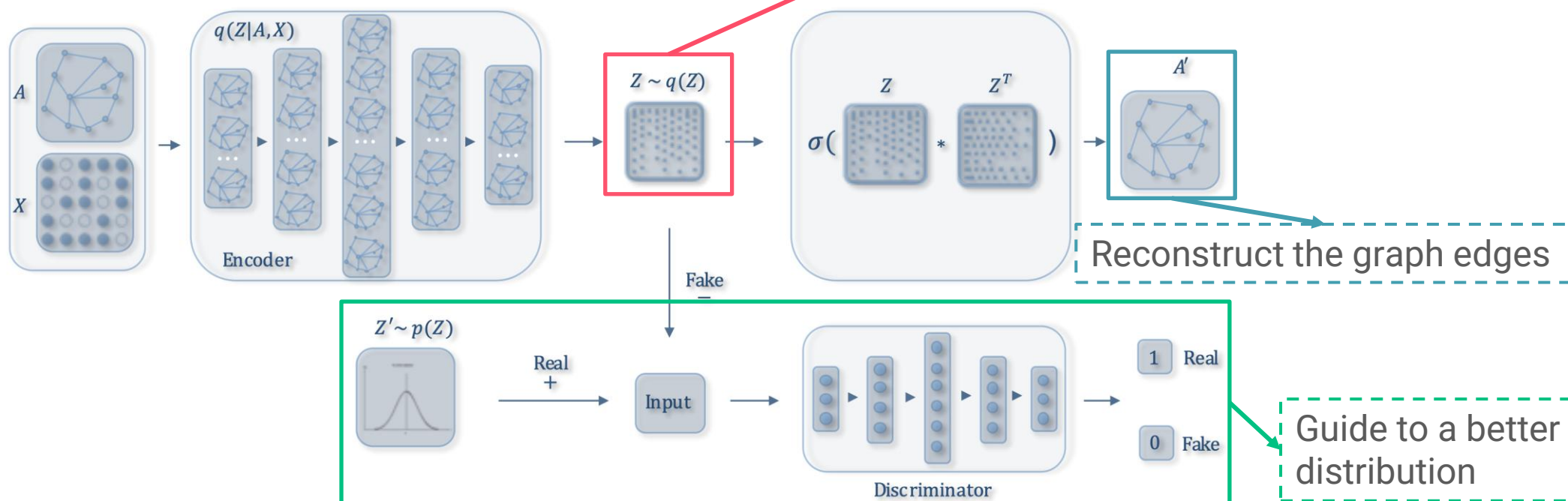
KDD2025

[1] Hou, Zhenyu, et al. "GraphMAE: Self-Supervised Masked Graph Autoencoders." KDD 2022.

Generative Methods

Contextualized generation: reconstruct edges

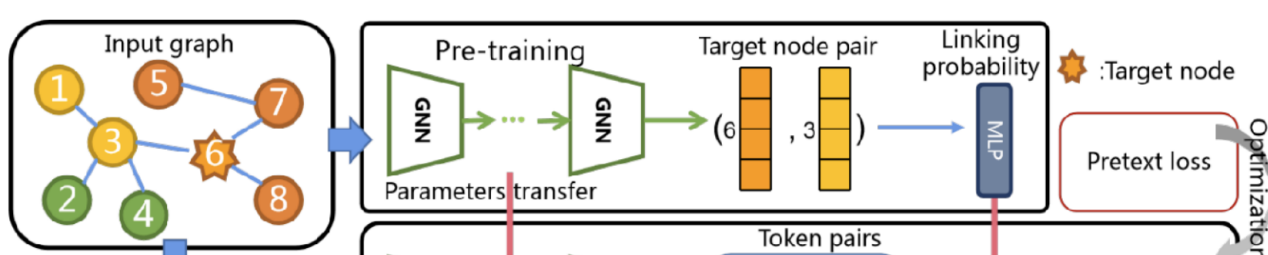
- ARVGA¹: Adversarially Regularized Graph Autoencoder for Graph Embedding



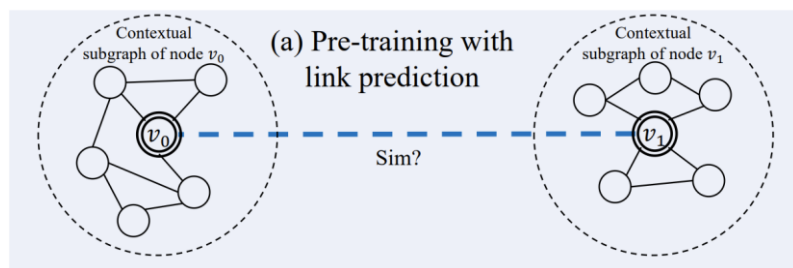
Generative Methods

Contextualized generation: reconstruct edges

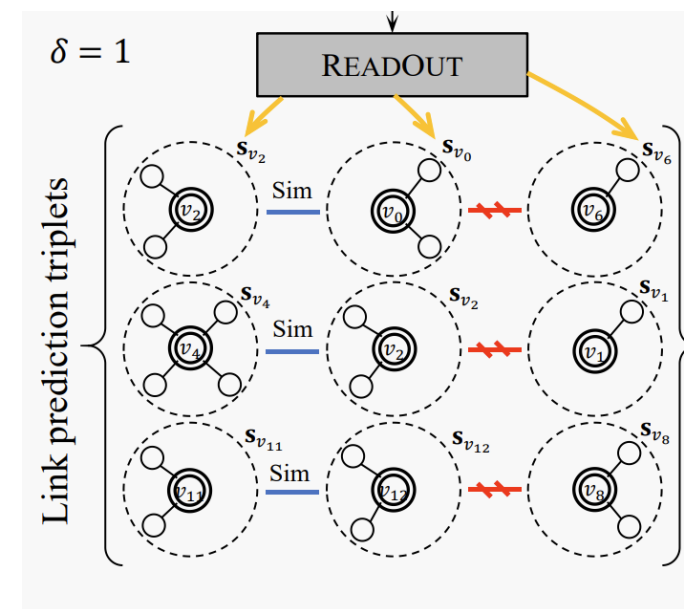
- Many graph prompting methods adapt contextualized generation for pre-training



GPPT¹



GraphPrompt²

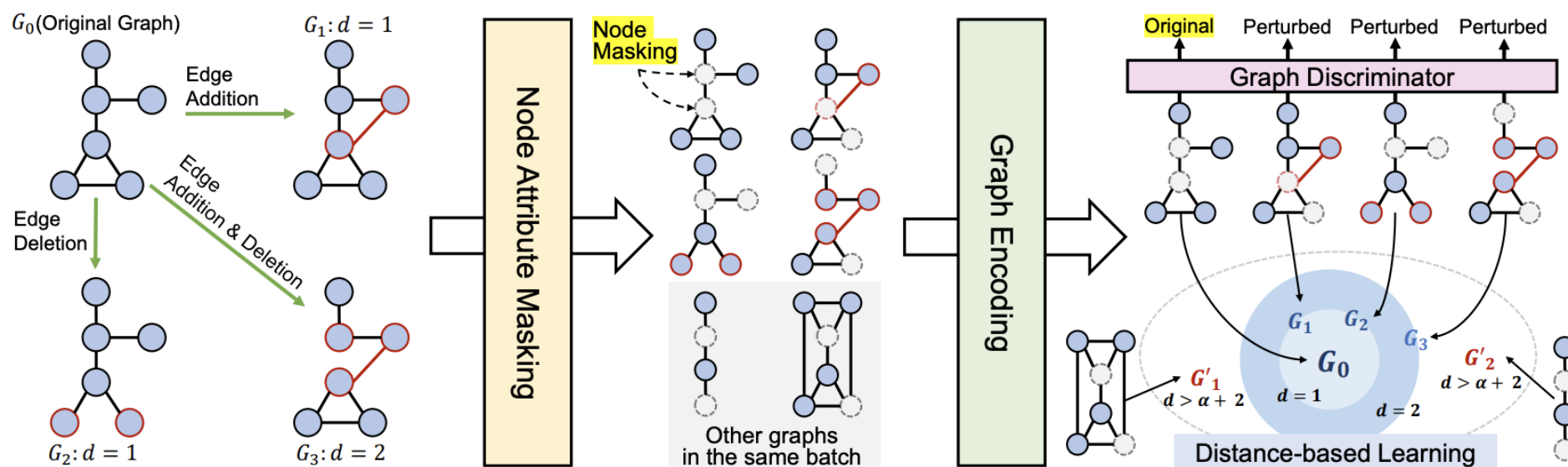


GraphPrompt+³

Generative Methods

Globalized generation: reconstruct graph-level property

- Corruption: distance-based edge masking and random node feature masking
- Reconstruction: reconstruct the edit distance between graphs



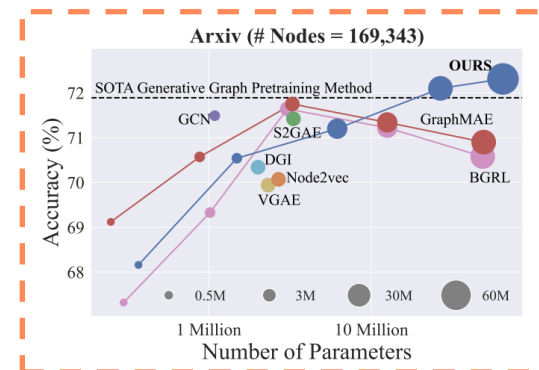
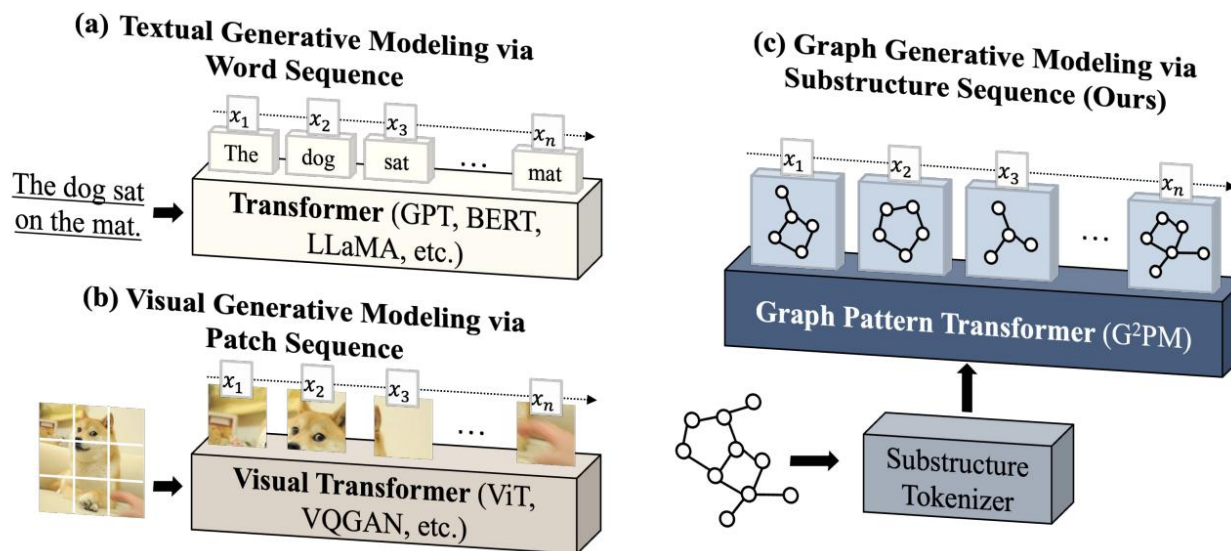
The workflow of D-SLA¹

[1] Kim, Dongki, Jinheon Baek, and Sung Ju Hwang. "Graph Self-supervised Learning with Accurate Discrepancy Learning." NeurIPS 2022.

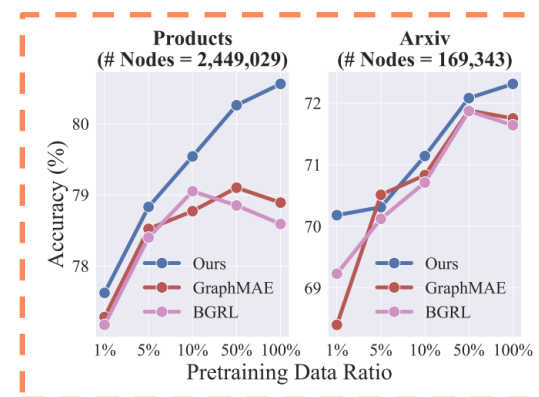
Generative Methods

Others

- Scalable Graph Generative Modeling via Substructure Sequences (G2PM)
- Follow the general transformer pretraining on other domains, like CV and NLP



Model Scaling

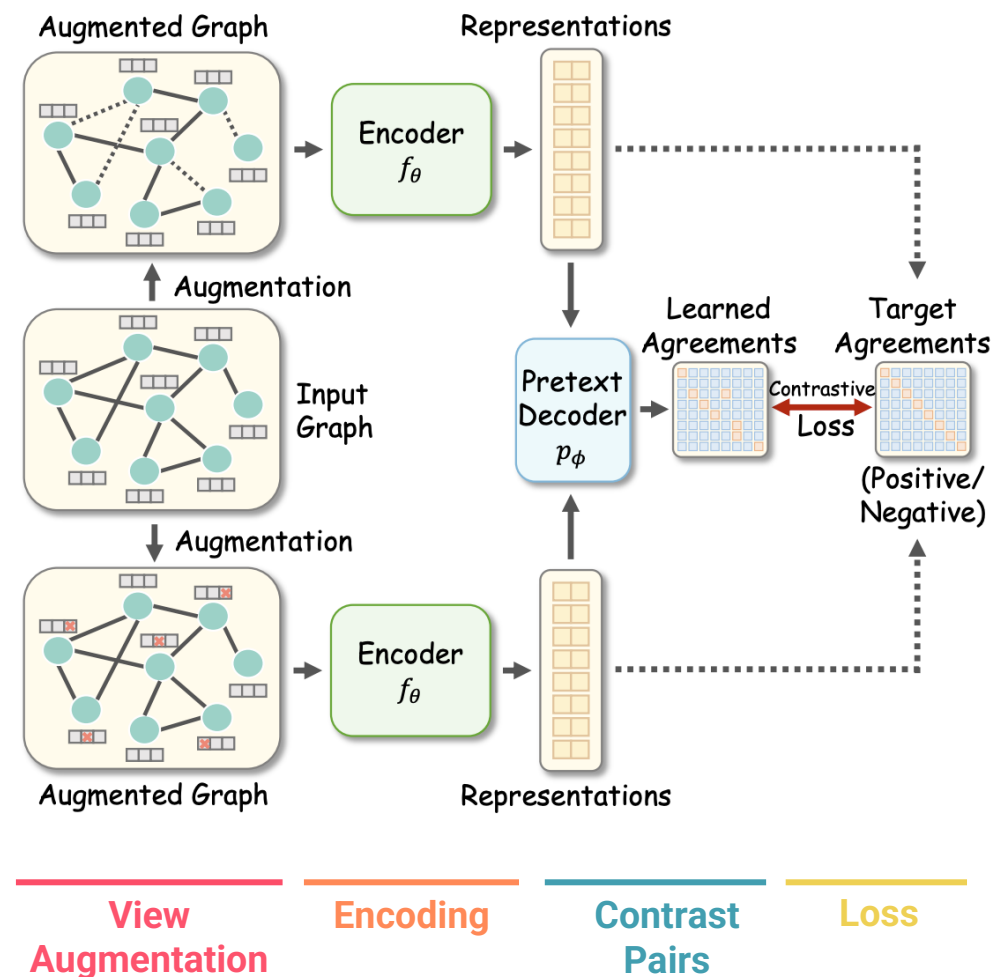


Data Scaling

Contrastive Methods

Main steps

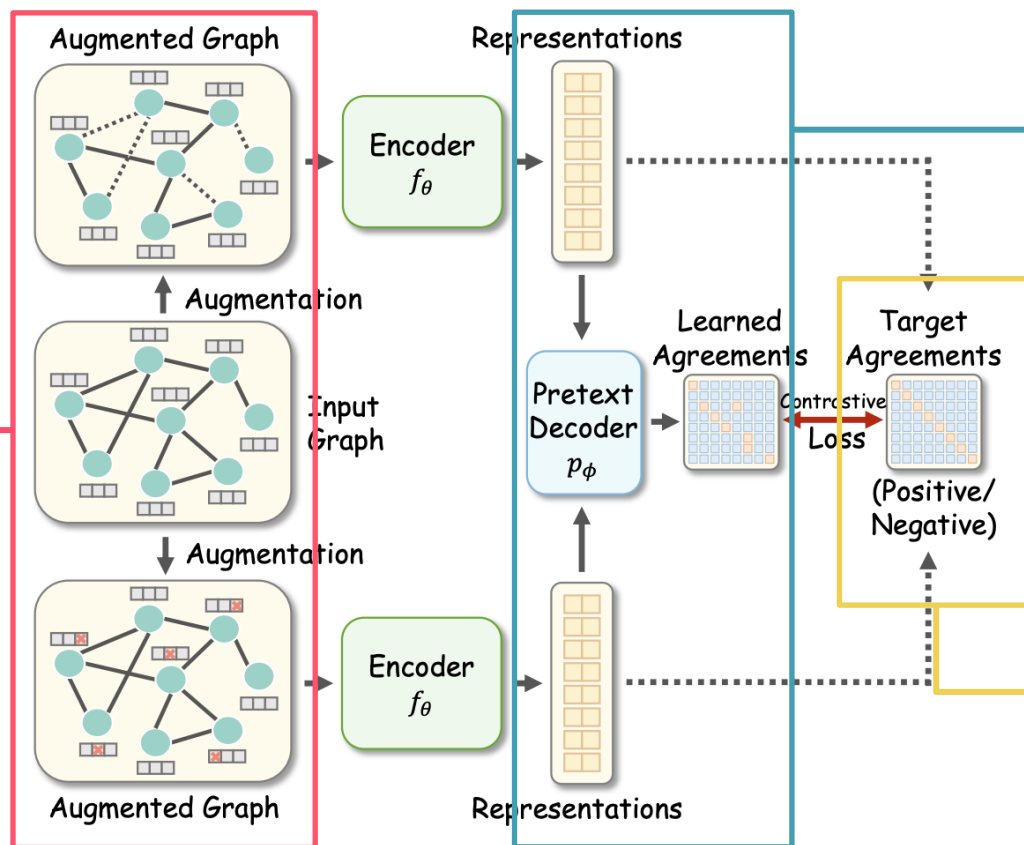
1. View augmentation
 - Corrupt multiple graphs $\tilde{\mathcal{G}}_k = \text{corrupt}_k(\mathcal{G})$.
2. Encoding
 - Encode each view $\mathbf{H}_k = f(\tilde{\mathcal{G}}_k)$
3. Contrast pairs
 - Pick the contrastive pairs and levels
4. Loss
 - Maximize the similarity between positive pairs while minimizing the similarity between negative pairs



Contrastive Methods

Augmentation:

1. **Feature**, e.g., node feature masking, node feature shuffling, etc.
2. **Structure**, e.g., edge perturbation, edge diffusion
3. **Substructure**, e.g., ego-graph sampling, random walk sampling, etc.



Contrast Levels:

1. **Local-Local**, e.g., node-to-node contrast, node-to-subgraph contrast.
2. **Local-Global**, e.g., node-to-graph contrast.
3. **Global-Global**, e.g., graph-to-graph contrast.

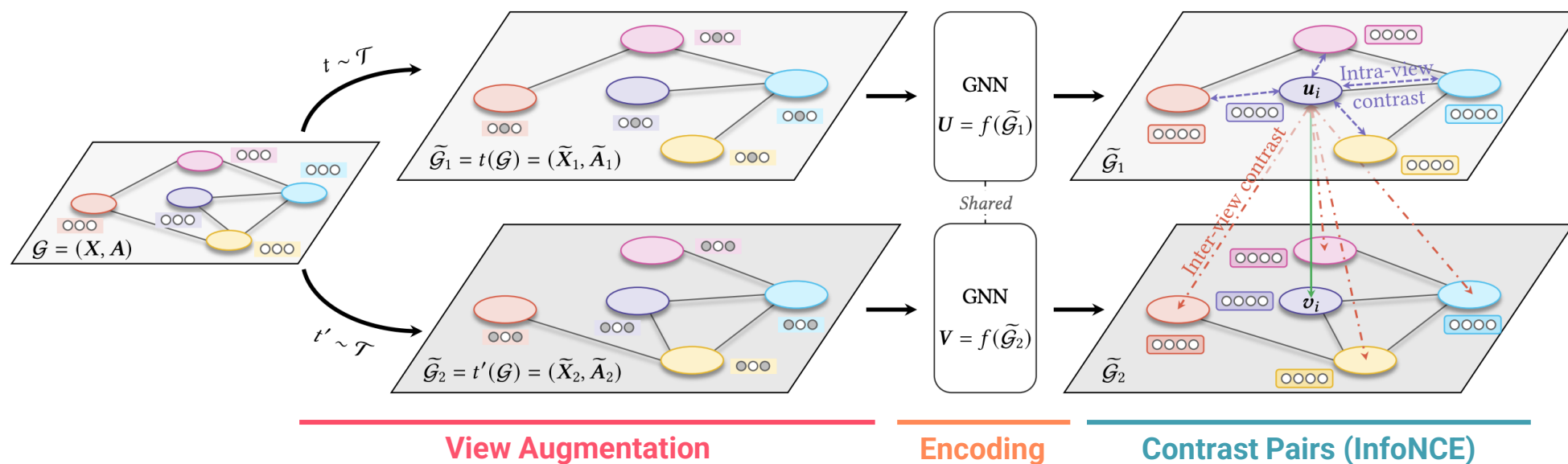
Loss:

1. InfoNCE
2. JS divergence
3. ...

Contrastive Methods

Local-to-local

- Augment two views of a single graph, treating the same node in these graphs as positives and the remaining as negatives



The workflow of local-to-local contrastive methods¹

Contrastive Methods

Local-to-local

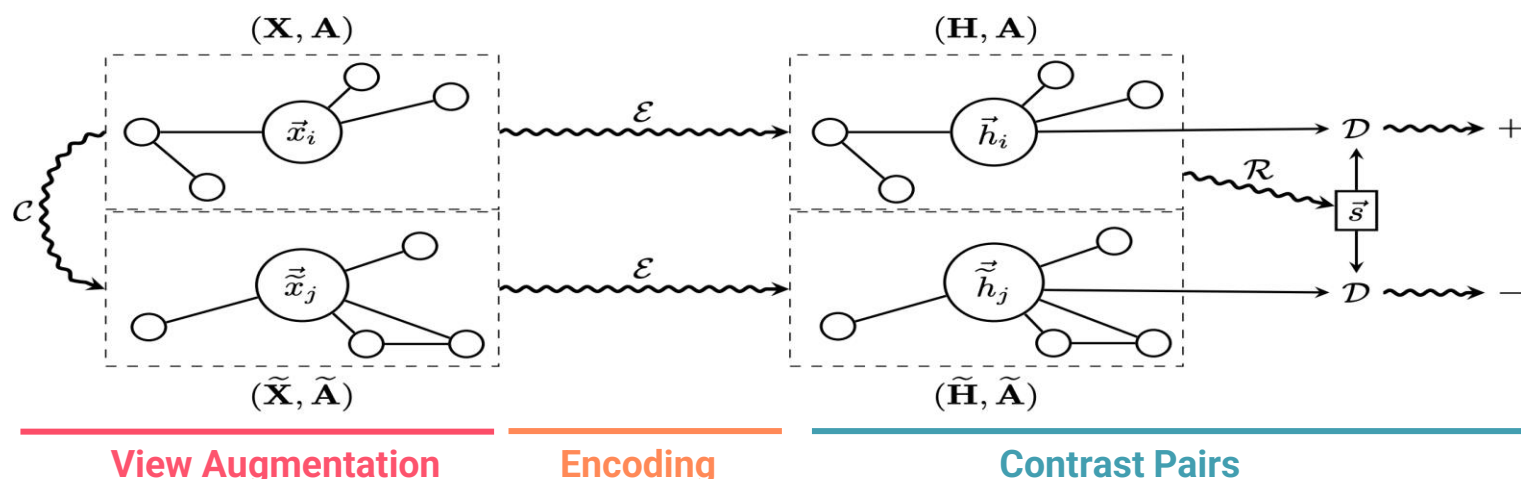
- Augment two views of a single graph, treating the same node in these graphs as positives and the remaining as negatives
- GRACE¹: **random** node feature masking + **random** edge masking
- GCA²: **centrality-based** node feature masking + **centrality-based** edge masking

$$\ell(\mathbf{u}_i, \mathbf{v}_i) = \log \frac{e^{\theta(\mathbf{u}_i, \mathbf{v}_i)/\tau}}{\underbrace{e^{\theta(\mathbf{u}_i, \mathbf{v}_i)/\tau}}_{\text{positive pair}} + \underbrace{\sum_{k \neq i} e^{\theta(\mathbf{u}_i, \mathbf{v}_k)/\tau}}_{\text{inter-view negative pairs}} + \underbrace{\sum_{k \neq i} e^{\theta(\mathbf{u}_i, \mathbf{u}_k)/\tau}}_{\text{intra-view negative pairs}}},$$

Contrastive Methods

Local-to-global

- Contrast pairs: node embedding vs graph embedding
- Deep Graph Infomax¹



Augmentation:
node feature shuffling + edge perturbation

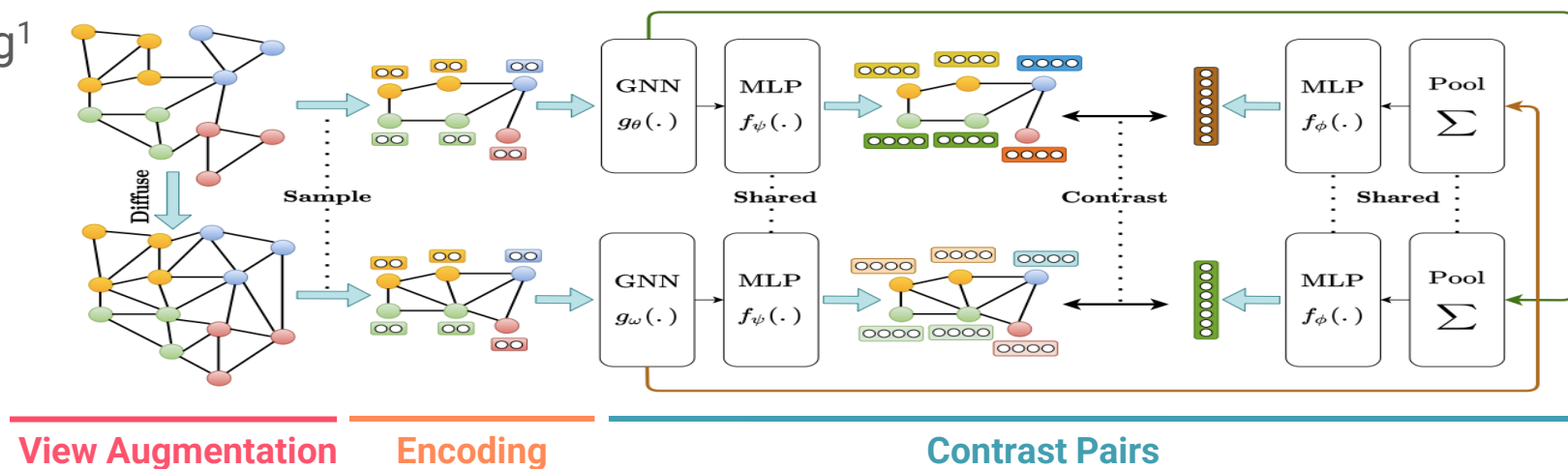
Loss:

$$\mathcal{L} = \frac{1}{N + M} \left(\sum_{i=1}^N \mathbb{E}_{(\mathbf{X}, \mathbf{A})} \left[\log \mathcal{D} \left(\vec{h}_i, \vec{s} \right) \right] + \sum_{j=1}^M \mathbb{E}_{(\tilde{\mathbf{X}}, \tilde{\mathbf{A}})} \left[\log \left(1 - \mathcal{D} \left(\vec{h}_j, \vec{s} \right) \right) \right] \right)$$

Contrastive Methods

Local-to-global

- Contrast pairs: node embedding vs graph embedding
- Multi-view contrastive learning¹



Augmentation:
Graph diffusion + subgraph sampling

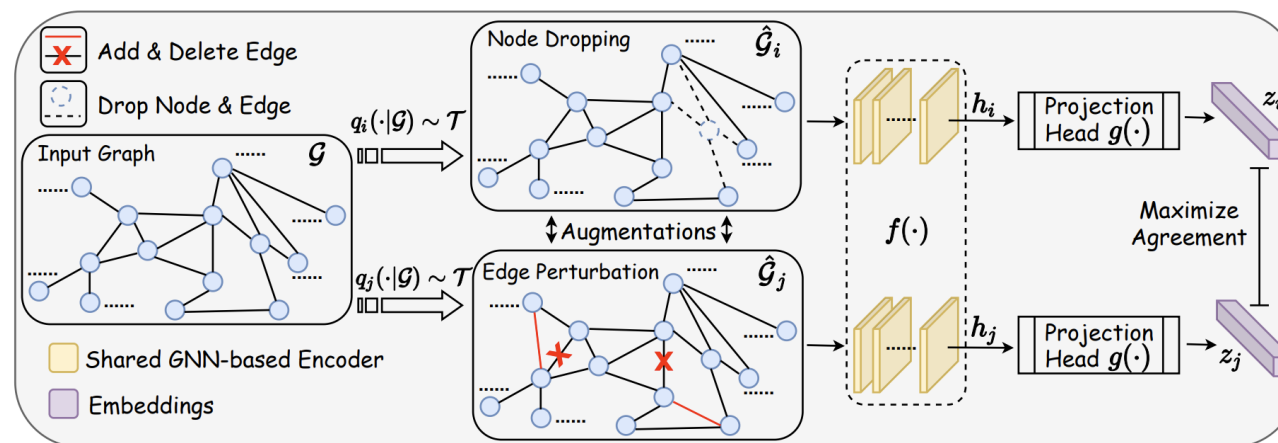
Loss:

$$\max_{\theta, \omega, \phi, \psi} \frac{1}{|\mathcal{G}|} \sum_{g \in \mathcal{G}} \left[\frac{1}{|g|} \sum_{i=1}^{|g|} \left[\text{MI}(\vec{h}_i^\alpha, \vec{h}_g^\beta) + \text{MI}(\vec{h}_i^\beta, \vec{h}_g^\alpha) \right] \right]$$

Contrastive Methods

Global-to-global

- Contrast pairs: graph embedding vs graph embedding
- GraphCL¹



View Augmentation

Encoding Contrast Pairs

Augmentation:
Random node dropping + edge perturbation

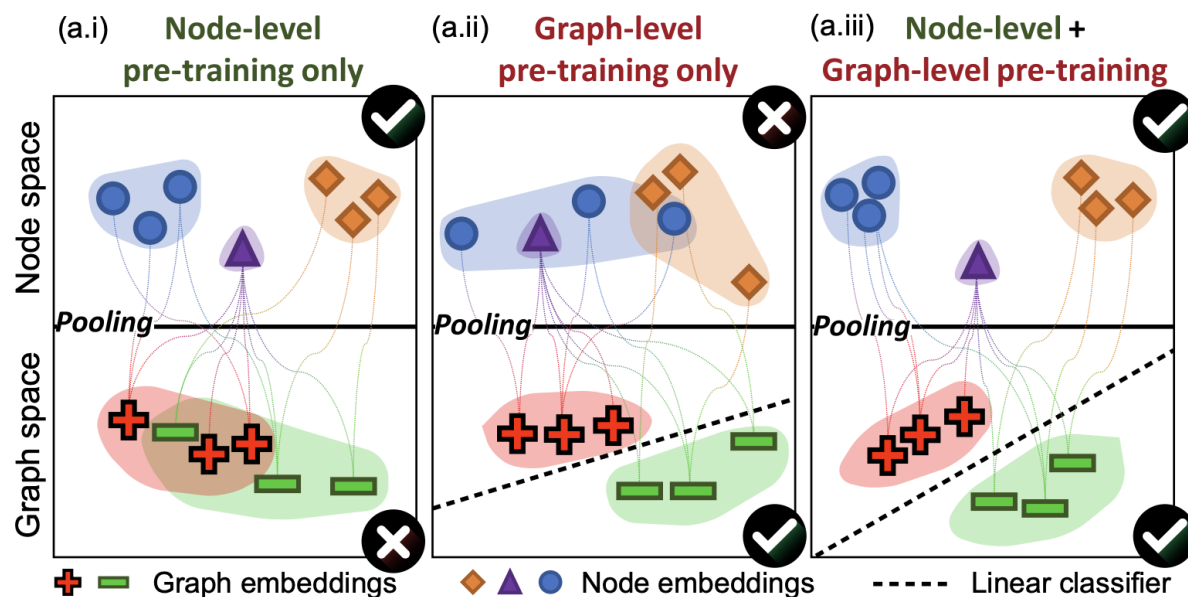
Loss:

$$\ell_n = -\log \frac{\exp(\text{sim}(\mathbf{z}_{n,i}, \mathbf{z}_{n,j})/\tau)}{\sum_{n'=1, n' \neq n}^N \exp(\text{sim}(\mathbf{z}_{n,i}, \mathbf{z}_{n',j})/\tau)}$$

Multi-task Methods

Jointly optimize multiple pre-training tasks

- Intuition: inject diverse capability into a single model



Multi-task methods¹

Multi-task Methods

Jointly optimize multiple pre-training tasks

- Intuition: inject diverse capability into a single model
- Node-level & graph-level
- Attribute prediction & structure prediction

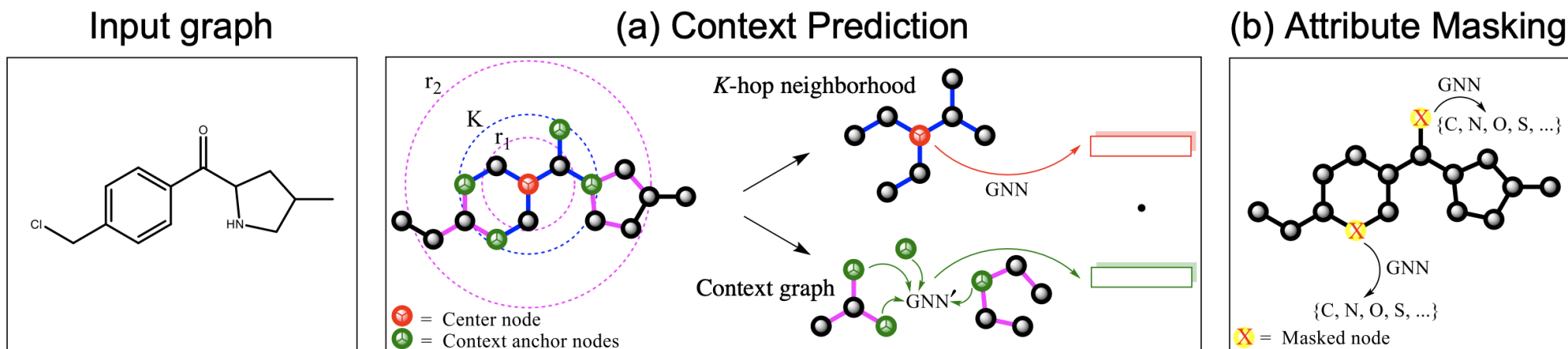
| | Node-level | Graph-level |
|----------------------|--------------------|----------------------------------|
| Attribute prediction | Attribute Masking | Supervised Attribute Prediction |
| Structure prediction | Context Prediction | Structural Similarity Prediction |

Different pre-training tasks¹

Multi-task Methods

Jointly optimize multiple pre-training tasks

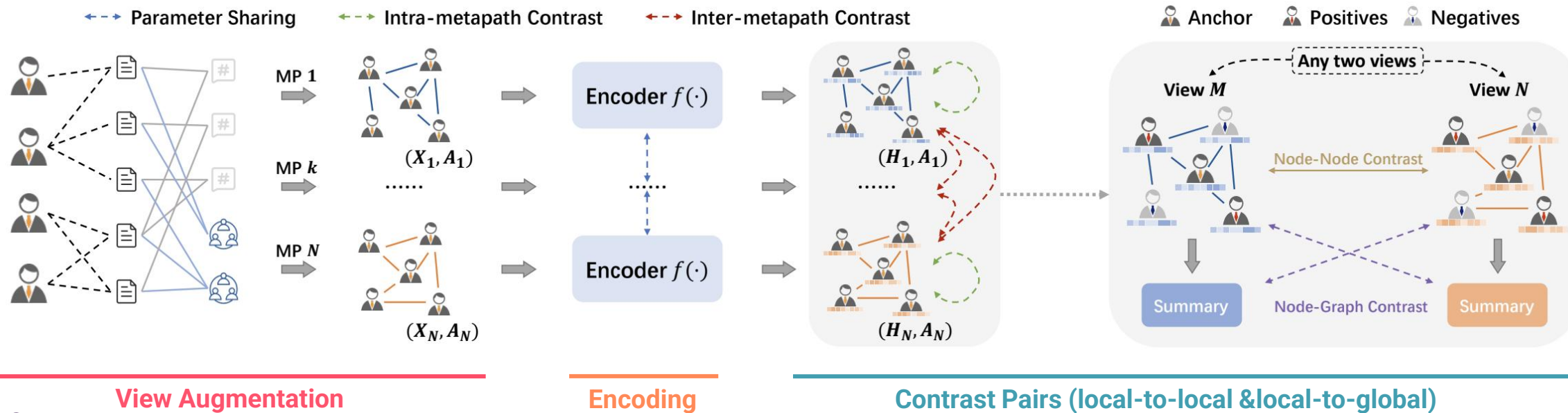
- Intuition: inject diverse capability into a single model



Multi-task Methods

Jointly optimize multiple pre-training tasks

- Heterogeneous graphs: Multi-view contrastive learning¹
- Use meta-paths on heterogeneous graphs to generate multiple views



View Augmentation

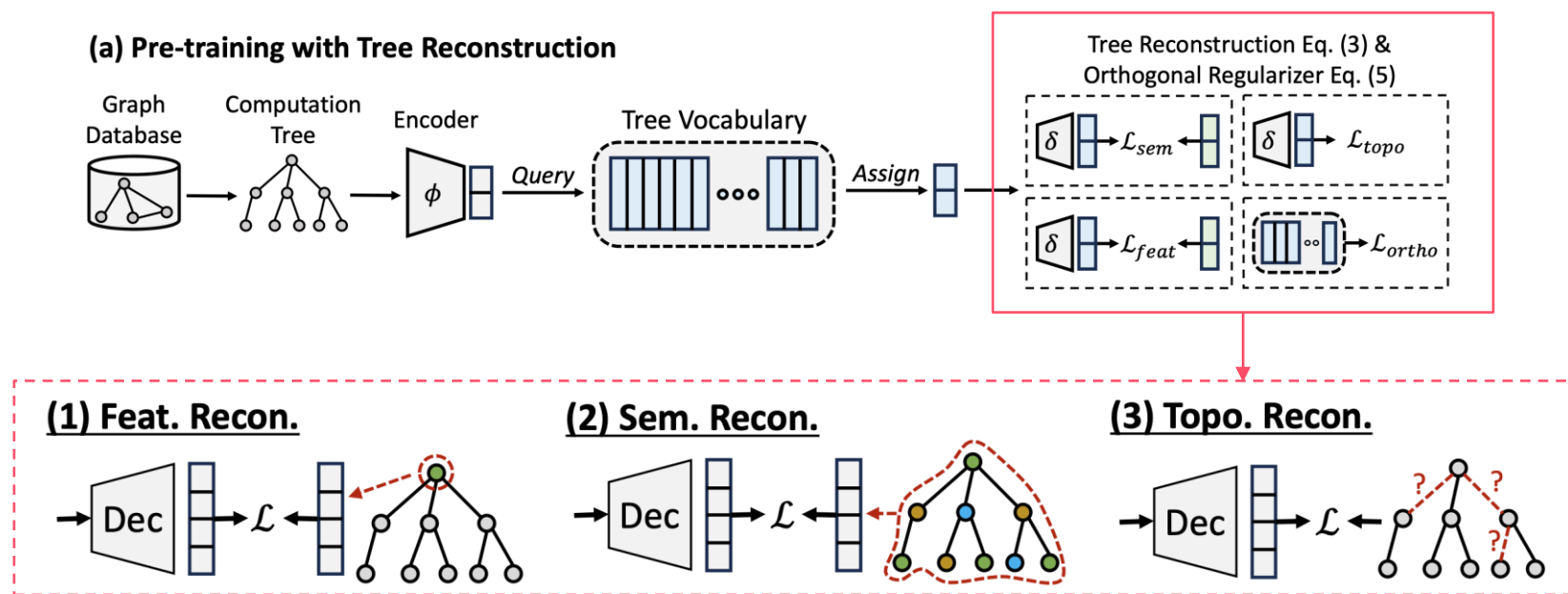
Encoding

Contrast Pairs (local-to-local & local-to-global)

Multi-task Methods

Jointly optimize multiple pre-training tasks

- GFT: Graph Foundation Model with Transferable Tree Vocabulary¹



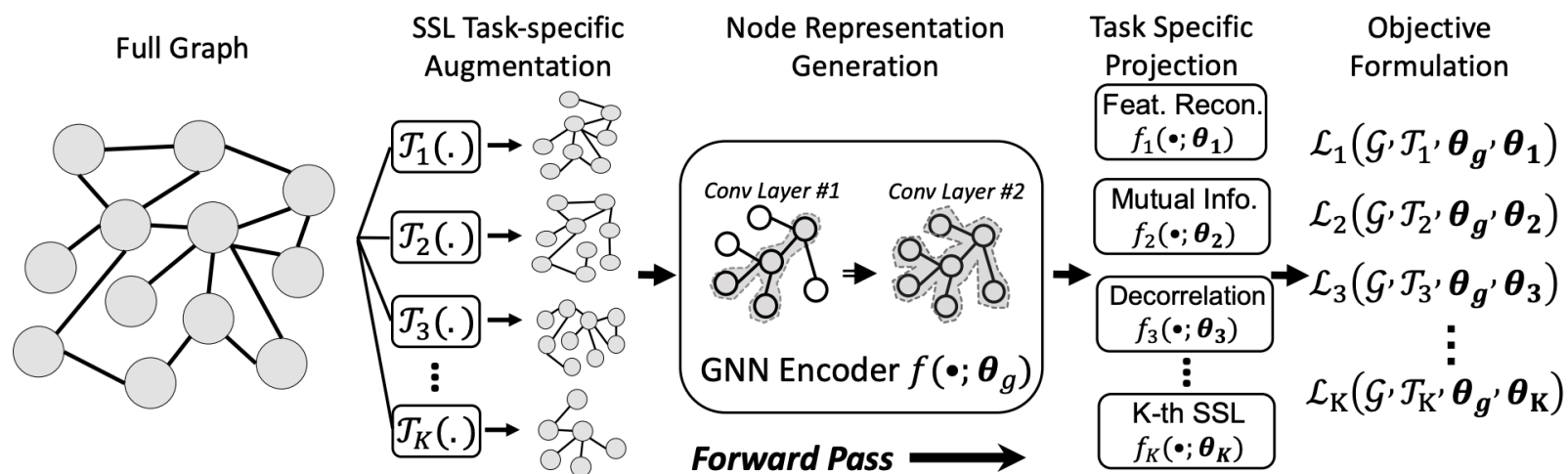
| | Node | Link | Graph | Avg. |
|-------------------------------------|--------------|--------------|--------------|--------------|
| <i>Different Pre-training Tasks</i> | | | | |
| n/a | 72.52 | 47.30 | 73.83 | 66.54 |
| w. \mathcal{L}_{sem} | <u>76.25</u> | 90.39 | <u>74.99</u> | <u>79.47</u> |
| w. \mathcal{L}_{feat} | <u>75.85</u> | <u>90.42</u> | 74.42 | 79.13 |
| w. \mathcal{L}_{topo} | 75.50 | 90.28 | 74.57 | 78.96 |
| GFT | 76.78 | 90.82 | 75.29 | 79.92 |

Jointly optimizing three-level tasks facilitates performance

Multi-task Methods

Jointly optimize multiple pre-training tasks

- Multi-task Self-supervised Graph Neural Networks Enable Stronger Task Generalization¹

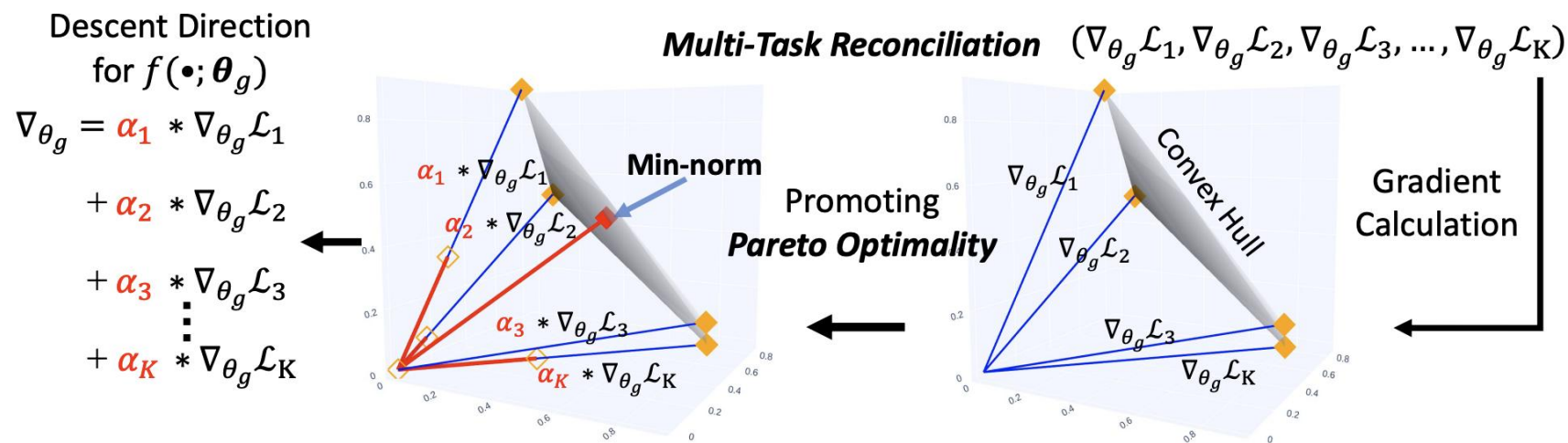


Multi-task learning on graphs

Multi-task Methods

Jointly optimize multiple pre-training tasks

- Multi-task Self-supervised Graph Neural Networks Enable Stronger Task Generalization¹



Multi-task reconciliation via Pareto optimization

Multi-task Methods

Jointly optimize multiple pre-training tasks

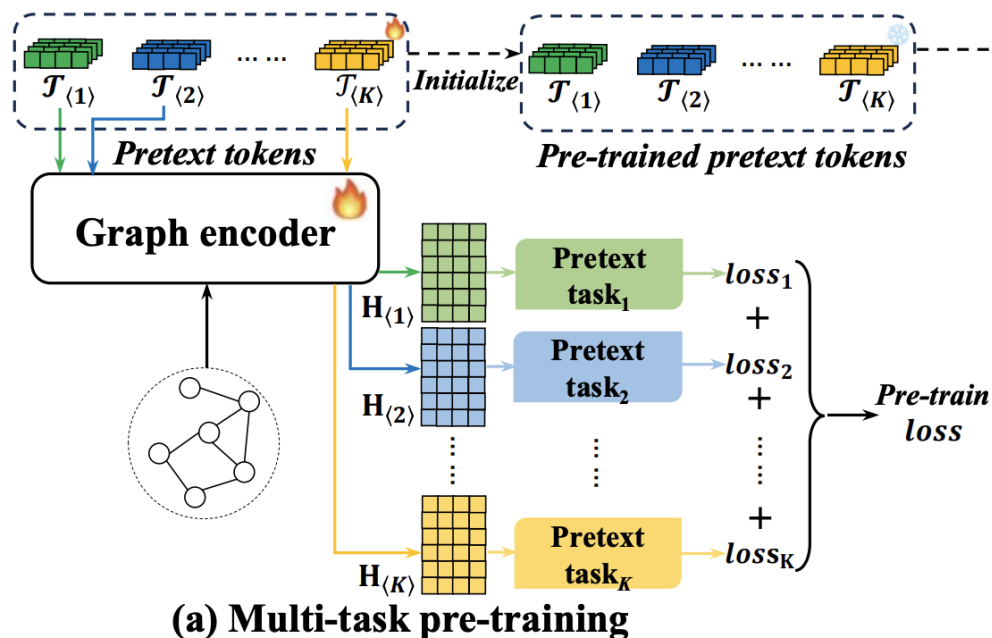
- Multi-task Self-supervised Graph Neural Networks Enable Stronger Task Generalization¹

| Method | WIKI.CS | PUBMED | AM.PHOTO | AM.COMP. | Co.CS | Co.PHY. | CHAM. | SQUIRREL | ACTOR | RANK |
|---------------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|------------|
| AVERAGE PERFORMANCE | | | | | | | | | | |
| FeatRec | 74.06 | 69.48 | 84.76 | 77.76 | <u>86.61</u> | 75.20 | 64.66 | 52.43 | 31.63 | 4.6 |
| TopoRec | 70.44 | 66.32 | 85.18 | 79.41 | 85.23 | 77.45 | 61.20 | 52.89 | 38.56 | 5.0 |
| RepDecor | 69.54 | 67.42 | 83.74 | 78.49 | 84.49 | 78.31 | 58.98 | 52.19 | 36.28 | 5.8 |
| MI-NG | 71.89 | 67.35 | 85.33 | 79.95 | 83.01 | 75.00 | 63.72 | 49.56 | 30.60 | 5.7 |
| MI-NSG | <u>75.59</u> | 69.25 | 82.99 | 80.85 | 86.02 | 80.30 | <u>64.69</u> | 53.89 | 38.22 | 3.6 |
| PARETOGNN | 76.03 | 72.48 | 86.58 | 82.57 | 87.80 | 83.35 | 65.21 | 55.31 | 40.76 | 1.0 |
| w/o Pareto | 74.64 | <u>69.82</u> | <u>85.82</u> | <u>82.09</u> | 86.54 | <u>82.32</u> | 64.37 | <u>54.90</u> | <u>40.12</u> | <u>2.4</u> |

Multi-task Methods

Jointly optimize multiple pre-training tasks

- MultiGPrompt¹: multi-task pre-training for better task adaptation in prompting



Section 3

Techniques in Graph Prompting

Xingbo Fu

PhD Candidate

University of Virginia



Problem Formulation of Graph Prompting

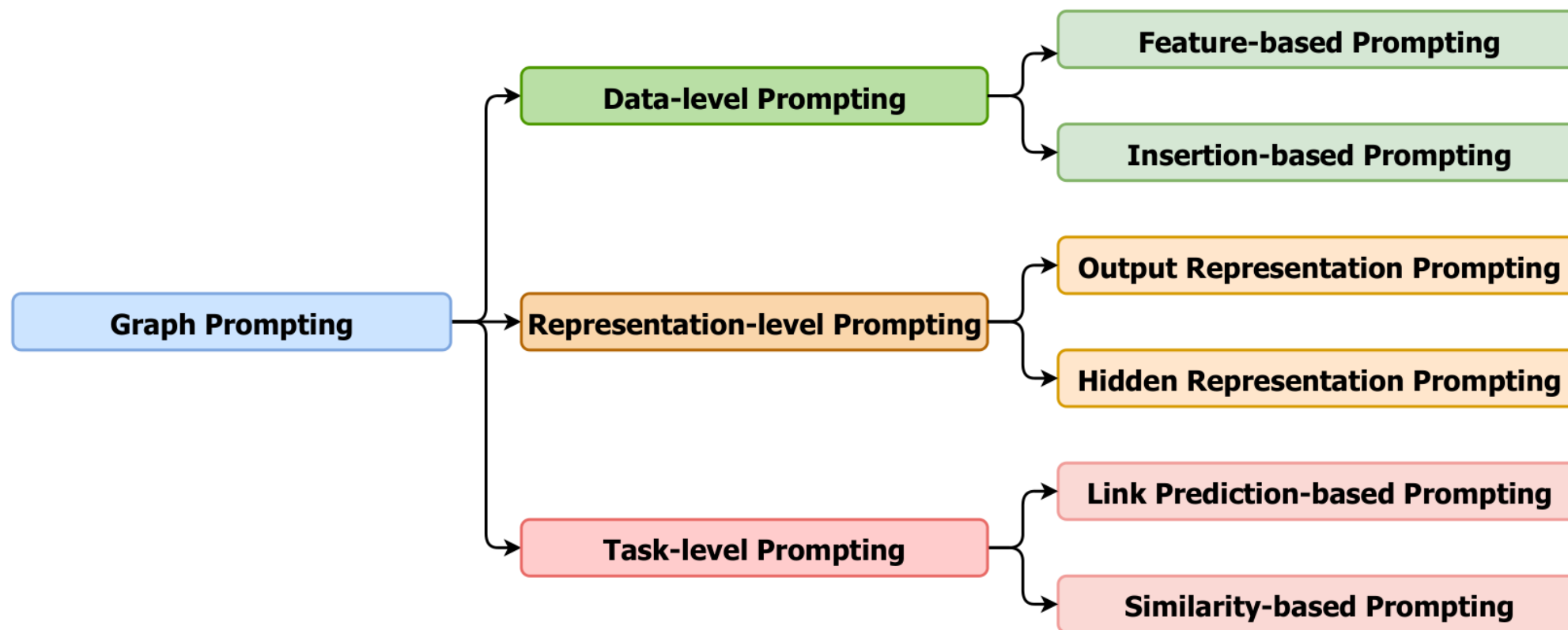
Pre-training compatibility

A graph learning model f is trained through *a pre-training task \mathcal{L}_{PT}* via self-supervised learning. During the prompting stage, graph prompting trains learnable prompts \mathcal{P} to *adapt the pre-trained graph learning model f to a specific downstream task \mathcal{L}_{DT}* .

Freeze pre-trained graph learning models

Downstream task universality

Taxonomy of Graph Prompting Techniques



Data-level Prompting

Learn trainable prompts at the data level

- Intuition: modify the input graph data to fit specific downstream tasks
- Formulation

Given the input graph $\mathcal{G} = (\mathbf{A}, \mathbf{X})$, data-level prompting \mathcal{T}^D transforms it into a prompted graph

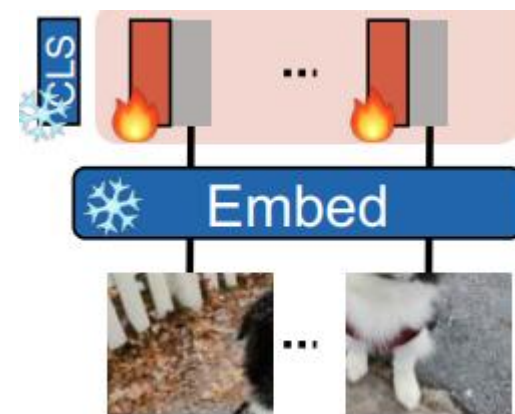
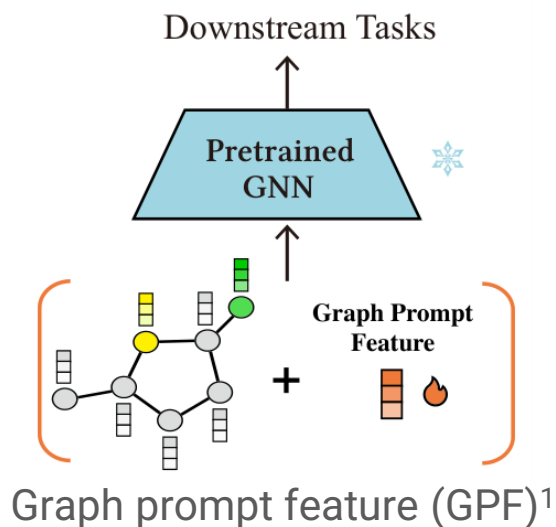
$\tilde{\mathcal{G}} = (\tilde{\mathbf{A}}, \tilde{\mathbf{X}}) = \mathcal{T}^D((\mathbf{A}, \mathbf{X}), \mathcal{P})$ with learnable prompts \mathcal{P} for adaptation.

- Two categories
 - Feature-based prompting
 - Insertion-based prompting

Data-level Prompting

Feature-based prompting

- Intuition: solely modify the feature matrix by learning prompt features
- Inspired by prompting techniques in NLP and CV
- The graph topology remains unchanged $\rightarrow \tilde{\mathcal{G}} = (\mathbf{A}, \tilde{\mathbf{X}}) = \mathcal{T}^D((\mathbf{A}, \mathbf{X}), \mathcal{P})$



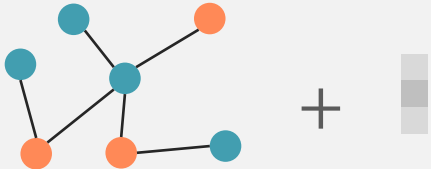
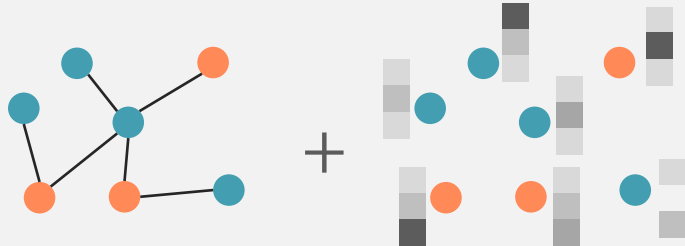
[1] Fang, Taoran, et al. "Universal Prompt Tuning for Graph Neural Networks." NeurIPS. 2023.

[2] Jia, Menglin, et al. "Visual Prompt Tuning." ECCV 2022.

Data-level Prompting

Feature-based prompting

- Goal: learn \mathbf{p}_i for $\tilde{\mathbf{x}}_i = \mathbf{x}_i + \mathbf{p}_i$
- Shared prompt features v.s. customized prompt features

| Shared prompt features | Customized prompt features |
|--|---|
|  |  |
| Fewer parameters | Increasing parameters in large-scale graphs |
| Easy to train | Prone to overfitting |
| Limited capability | More powerful |

Data-level Prompting

Feature-based prompting

- Customized prompt features with basis vectors¹

$$\tilde{\mathbf{x}}_i = \mathbf{x}_i + \mathbf{p}_i = \mathbf{x}_i + \sum_{m=1}^M \alpha_{i,m} \cdot \mathbf{b}_m$$

- GPF-plus¹: $\alpha_{i,m}$ as the softmax values of projected node feature \mathbf{x}_i

$$\alpha_{i,m} = \frac{\exp(a_m^T \mathbf{x}_i)}{\sum_{k=1}^M \exp(a_k^T \mathbf{x}_i)}$$

- SUPT²: α as the resulting scores derived from simple GNNs

$$\alpha = \hat{\mathbf{A}}^m \left(\mathbf{x} \oplus \sum_{m=1}^M \mathbf{b}_m \right) \mathbf{w}$$
$$\hat{\mathbf{A}} = (\mathbf{D} + \mathbf{I})^{-\frac{1}{2}} (\mathbf{A} + \mathbf{I}) (\mathbf{D} + \mathbf{I})^{-\frac{1}{2}}$$

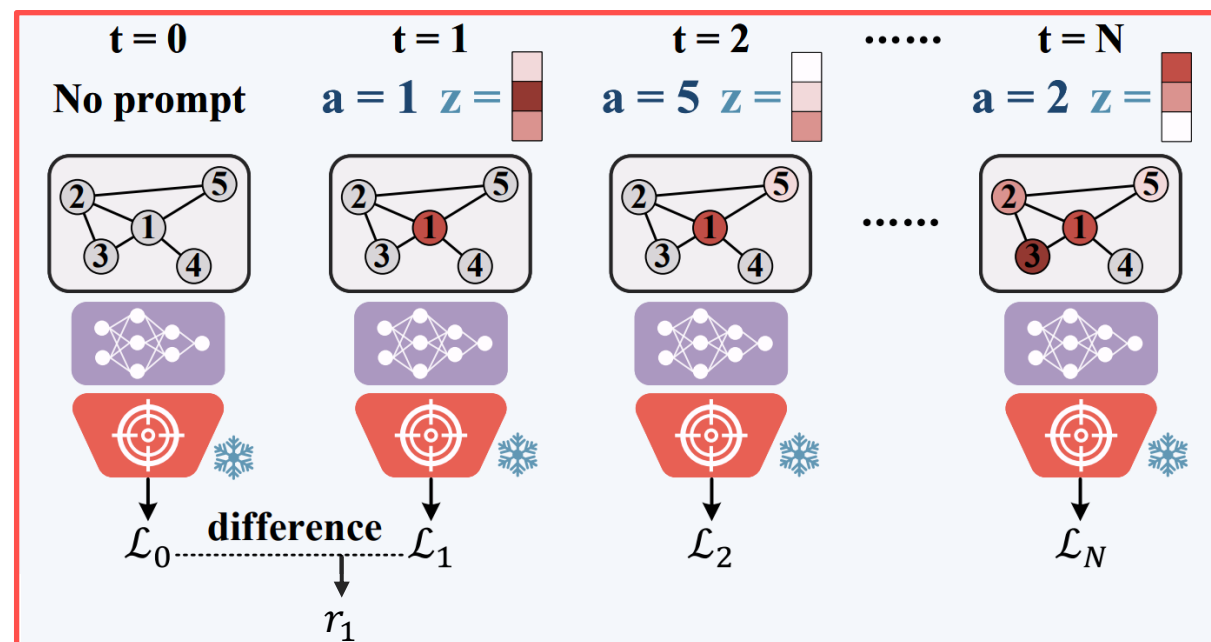
[1] Fang, Taoran, et al. "Universal Prompt Tuning for Graph Neural Networks." NeurIPS. 2023.

[2] Lee, Junhyun, et al. "Subgraph-level Universal Prompt Tuning." arXiv preprint arXiv:2402.10380 (2024).

Data-level Prompting

Feature-based prompting

- Customized prompt features by reinforcement learning with hybrid action space¹
- **Hybrid actions**: discrete node index to prompt and its corresponding continuous prompt features
- **States**: node representations
- **Reward function**: instant loss decrease
- **Policy network architecture**: H-PPO with two parallel actor networks and a single critic network



Data-level Prompting

Feature-based prompting

- Performance evaluation¹
- GPF+: prompts based on features may not obtain sufficient information
- SUPT: prompts based on both features and structures can improve performance
- RELIEF: selective methods are better than learning on all the nodes

| | Tuning Strategy | BBBP | Tox21 | ToxCast | SIDER |
|-------------|----------------------|------------------|------------------|------------------|------------------|
| Infomax | FT | 65.26 \pm 1.05 | 71.54 \pm 0.73 | 57.98 \pm 0.42 | 53.45 \pm 0.49 |
| | GPF | 66.30 \pm 0.94 | 71.26 \pm 0.30 | 58.33 \pm 0.27 | 53.65 \pm 0.34 |
| | GPF-plus | 66.12 \pm 1.27 | 71.54 \pm 0.63 | 58.43 \pm 0.27 | 53.76 \pm 1.15 |
| | SUPT _{soft} | 66.45 \pm 0.85 | 71.82 \pm 0.16 | 58.71 \pm 0.44 | 53.72 \pm 0.45 |
| | SUPT _{hard} | 66.14 \pm 0.85 | 71.55 \pm 0.37 | 58.65 \pm 0.33 | 53.82 \pm 0.49 |
| | RELIEF | 67.93 \pm 0.73 | 71.58 \pm 0.13 | 58.78 \pm 0.17 | 53.95 \pm 0.40 |
| AttrMasking | FT | 66.48 \pm 0.44 | 72.32 \pm 0.19 | 57.35 \pm 0.42 | 54.62 \pm 0.58 |
| | GPF | 66.67 \pm 0.60 | 72.31 \pm 0.30 | 58.01 \pm 0.27 | 55.65 \pm 1.92 |
| | GPF-plus | 66.29 \pm 0.36 | 72.75 \pm 0.38 | 57.91 \pm 0.38 | 55.05 \pm 1.20 |
| | SUPT _{soft} | 66.28 \pm 0.81 | 72.76 \pm 0.41 | 58.28 \pm 0.34 | 54.70 \pm 1.03 |
| | SUPT _{hard} | 66.93 \pm 0.91 | 72.75 \pm 0.41 | 58.18 \pm 0.44 | 55.20 \pm 1.26 |
| | RELIEF | 67.18 \pm 0.55 | 72.40 \pm 0.32 | 58.41 \pm 0.14 | 56.54 \pm 0.75 |
| ContextPred | FT | 62.82 \pm 0.83 | 70.11 \pm 0.38 | 57.68 \pm 0.69 | 56.68 \pm 0.78 |
| | GPF | 61.65 \pm 0.76 | 70.42 \pm 0.29 | 58.51 \pm 0.38 | 56.55 \pm 0.46 |
| | GPF-plus | 61.25 \pm 0.73 | 70.12 \pm 0.42 | 57.64 \pm 0.70 | 56.86 \pm 0.52 |
| | SUPT _{soft} | 61.85 \pm 1.56 | 70.37 \pm 0.14 | 57.82 \pm 0.37 | 56.57 \pm 0.50 |
| | SUPT _{hard} | 62.29 \pm 1.41 | 70.40 \pm 0.15 | 58.06 \pm 0.39 | 56.34 \pm 0.72 |
| | RELIEF | 62.99 \pm 0.67 | 70.51 \pm 0.14 | 58.58 \pm 0.04 | 56.89 \pm 0.25 |
| GCL | FT | 62.13 \pm 1.66 | 61.35 \pm 0.88 | 53.96 \pm 0.80 | 52.63 \pm 0.71 |
| | GPF | 61.58 \pm 1.81 | 59.92 \pm 1.29 | 54.44 \pm 0.31 | 51.21 \pm 0.56 |
| | GPF-plus | 62.19 \pm 1.45 | 60.13 \pm 0.52 | 54.43 \pm 0.43 | 50.90 \pm 0.78 |
| | SUPT _{soft} | 63.96 \pm 0.85 | 60.13 \pm 0.52 | 54.57 \pm 0.76 | 51.44 \pm 1.24 |
| | SUPT _{hard} | 64.15 \pm 0.96 | 60.56 \pm 0.20 | 54.72 \pm 0.71 | 51.60 \pm 0.85 |
| | All in One | 62.90 \pm 3.77 | 61.49 \pm 0.96 | 54.72 \pm 1.03 | 52.73 \pm 0.84 |
| | RELIEF | 64.80 \pm 0.69 | 61.45 \pm 0.35 | 55.03 \pm 0.39 | 52.82 \pm 0.59 |

Data-level Prompting

Feature-based prompting

- Quantifying Prompts Impact¹
- Prompt Coverage Ratio (PCR): the proportion of nodes prompted at least once during prompting
- Average Prompt Magnitude (APM): the absolute values of all entries in prompts

| Tuning Strategy | PCR | APM (10^{-2}) | OV. (10^{-2}) | ROC-AUC |
|----------------------|------|-------------------|-------------------|---------|
| GPF | 1.00 | 7.15 | 7.15 | 62.08 |
| GPF-plus | 1.00 | 6.69 | 6.69 | 62.38 |
| SUPT _{soft} | 1.00 | 6.46 | 6.46 | 62.39 |
| SUPT _{hard} | 0.65 | 6.10 | 3.97 | 62.68 |
| RELIEF | 0.61 | 6.03 | 3.68 | 63.72 |

No need for graph prompting on every node!

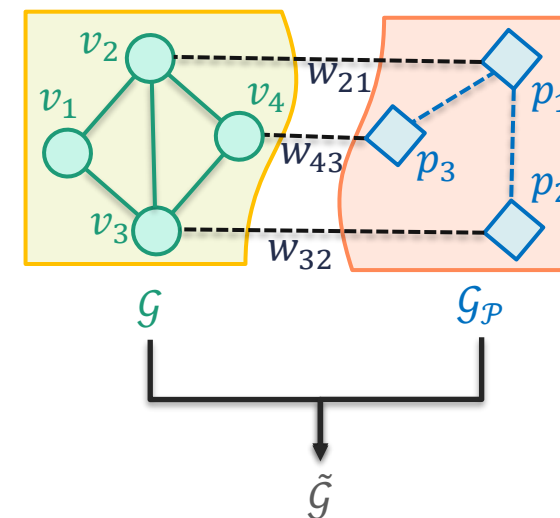
Data-level Prompting

Insertion-based prompting

- Intuition: insert additional prompt nodes as learnable prompts into the original graph

- $\tilde{\mathcal{G}}$ includes the prompt nodes and the original graph nodes
- $\mathcal{P} = \{p_1, p_2, \dots, p_M\}$: M learnable prompts as the feature vectors of the prompt nodes

- Key challenges
 - The connection among the prompt nodes
 - The connection among the prompt nodes and the original graph nodes



Data-level Prompting

The connection among the prompt nodes

- Solution 1: free learnable parameters a_{ij} indicating how possible p_i and p_j should be connected¹
- Solution 2: the dot product of each prompt node pair
 - Connect p_i and p_j if $\sigma(\mathbf{p}_i \cdot \mathbf{p}_j^T) > \delta^1$
- Solution 3: treat the prompt nodes as independent^{1,2}

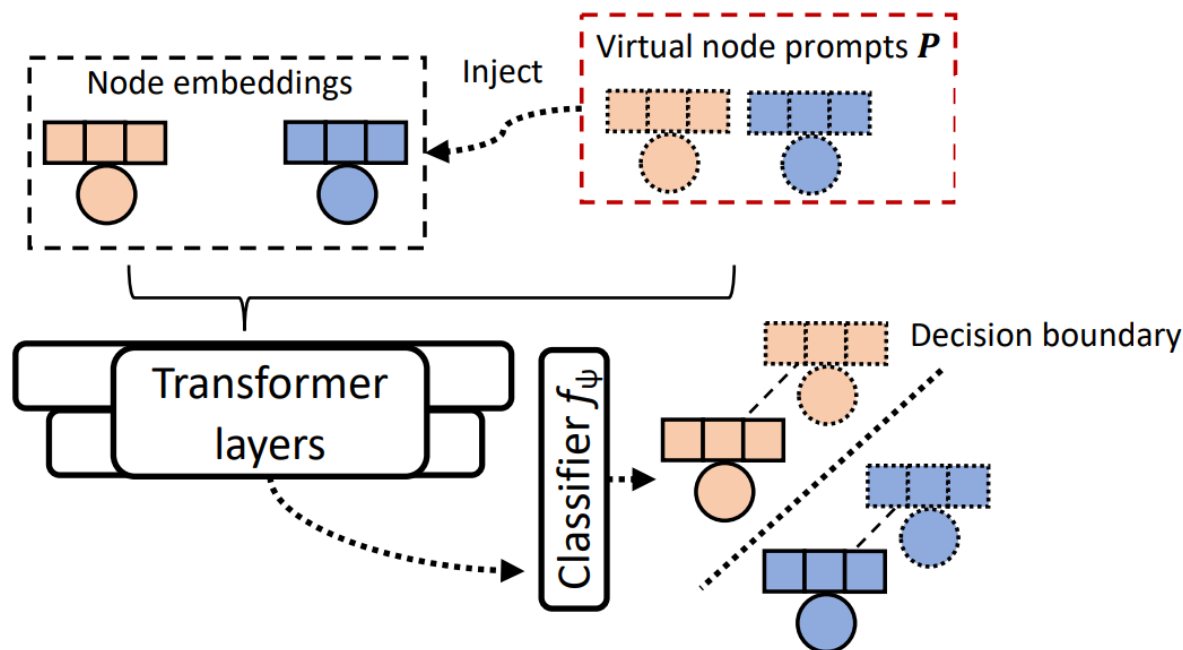
The connection among the prompt nodes and the original graph nodes

- Solution 1: the dot product between a prompt node and an original graph node
 - $w_{ij} = \sigma(\mathbf{x}_i \cdot \mathbf{p}_j^T) > \delta$ if $\sigma(\mathbf{x}_i \cdot \mathbf{p}_j^T) > \delta^1$
- Solution 2: $\tilde{\mathbf{x}}_i = \mathbf{x}_i + \sum_{m=1}^M \mathbf{p}_m^1$
- Solution 3: $N \times M$ free learnable parameters²

Data-level Prompting

Insertion-based prompting for **Graph Transformers** \longrightarrow *No worries about edge connections!*

- VNT¹: directly inject prompt nodes in the input of Graph Transformers



Representation-level Prompting

Learn trainable prompts at the representation level

- Intuition: apply learnable prompts to node representations
- Formulation

Given the (hidden) representation matrix $\mathbf{H}^{(l)}$ at the l -th layer, representation-level prompting \mathcal{T}^R transforms it into a prompted representation matrix $\tilde{\mathbf{H}}^{(l)} = \mathcal{T}^R(\mathbf{H}^{(l)}, \mathcal{P})$ with learnable prompts \mathcal{P} for adaptation.

- Two categories
 - Output representation prompting
 - Hidden representation prompting

Representation-level Prompting

Output representation prompting

- Intuition: directly modify the output representations after the final layer
- Only the output representations are modified: $\tilde{\mathbf{h}}^{(L)} = \mathcal{T}^R(\mathbf{h}^{(L)}, \mathcal{P})$
- Straightforward design: GraphPrompt¹
- Element-wise multiplication: $\tilde{\mathbf{h}}_v^{(L)} = \mathbf{p}_v \odot \mathbf{h}_v^{(L)}$
- \mathbf{p}_v is shared by all the nodes
- Limitation: shared \mathbf{p}_v is insufficient

Representation-level Prompting

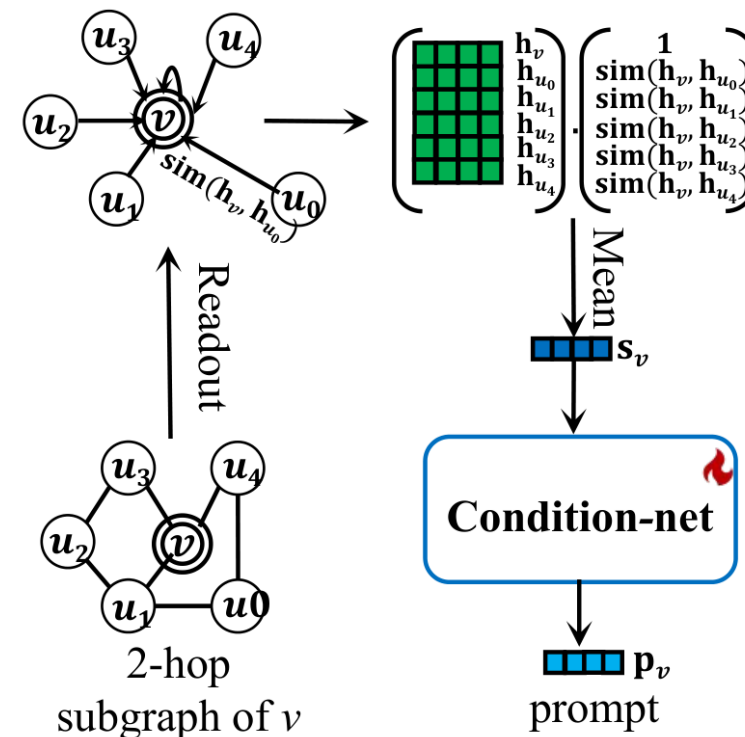
Output representation prompting

- Non-homophilic patterns of a node can be characterized by considering a multi-hop neighborhood around the node¹

$$\mathbf{s}_v = \frac{1}{\mathcal{N}_2(v)} \sum_{u_j \in \mathcal{N}_2(v)} \mathbf{h}_v^{(L)} \cdot \text{sim}(\mathbf{h}_v^{(L)}, \mathbf{h}_j^{(L)})$$

- Obtain customized \mathbf{p}_v through a condition-net²

$$\mathbf{p}_v = \varphi(\mathbf{s}_v) = \text{CondNet}(\mathbf{s}_v)$$



ProNoG¹

[1] Yu, Xingtong, et al. "Non-Homophilic Graph Pre-Training and Prompt Learning." KDD 2025.

[2] Zhou, Kaiyang, et al. "Conditional Prompt Learning for Vision-Language Models." CVPR 2022.

Representation-level Prompting

Output representation prompting

- ProNoG¹ performs well on non-homophilic graphs

| Methods | Wisconsin | Squirrel | Chameleon | Cornell |
|--------------|----------------------|---------------------|---------------------|----------------------|
| GCN | 21.39 ± 6.56 | 20.00 ± 0.29 | 25.11 ± 4.19 | 21.81 ± 4.71 |
| GAT | 28.01 ± 5.40 | 21.55 ± 2.30 | 24.82 ± 4.35 | 23.03 ± 13.19 |
| H2GCN | 23.60 ± 4.64 | 21.90 ± 2.15 | 25.89 ± 4.96 | 32.77 ± 14.88 |
| FAGCN | 35.03 ± 17.92 | 20.91 ± 1.79 | 22.71 ± 3.74 | 28.67 ± 17.64 |
| DGI | 28.04 ± 6.47 | 20.00 ± 1.86 | 19.33 ± 4.57 | 32.54 ± 15.66 |
| GRAPHCL | 29.85 ± 8.46 | 21.42 ± 2.22 | 27.16 ± 4.31 | 24.69 ± 14.06 |
| DSSL | 28.46 ± 10.31 | 20.94 ± 1.88 | <u>27.92</u> ± 3.93 | 20.36 ± 5.38 |
| GRAPHACL | <u>34.57</u> ± 10.46 | <u>24.44</u> ± 3.94 | 26.72 ± 4.67 | <u>33.17</u> ± 16.06 |
| GPPT | 27.39 ± 6.67 | 20.09 ± 0.91 | 24.53 ± 2.55 | 25.09 ± 2.92 |
| GRAPHPROMPT | 31.48 ± 5.18 | 21.22 ± 1.80 | 25.36 ± 3.99 | 31.00 ± 13.88 |
| GRAPHPROMPT+ | 31.54 ± 4.54 | 21.24 ± 1.82 | 25.73 ± 4.50 | 31.65 ± 14.48 |
| ProNoG | 44.72 ± 11.93 | 24.59 ± 3.41 | 30.67 ± 3.73 | 37.90 ± 9.31 |

Representation-level Prompting

Hidden representation prompting

- Intuition: modify node representations with layer-wise prompts
- More parameters provide more flexibility
- GraphPrompt+¹: an advanced version of GraphPrompt
- Learnable prompts at each layer $\mathcal{P} = \{\mathbf{p}^{(0)}, \mathbf{p}^{(1)}, \mathbf{p}^{(2)}, \dots, \mathbf{p}^{(L)}\}$
- Prompted representation fusion with learnable coefficients

$$\tilde{\mathbf{H}} = \sum_{l=0}^L w^{(l)} \cdot \tilde{\mathbf{H}}^{(l)}$$

Representation-level Prompting

Hidden representation prompting

- EdgePrompt+¹: learn edge prompts for each edge
- $\mathbf{e}_{ij}^{(l)}$: prompt vector on edge (v_i, v_j) at the l -th layer
- Without edge prompts

$$\mathbf{h}_i^{(l)} = \text{COMB}^{(l)} \left(\mathbf{h}_i^{(l-1)}, \text{AGG}^{(l)} \left(\left\{ \mathbf{h}_j^{(l-1)} : v_j \in \mathcal{N}(v_i) \right\} \right) \right)$$

- With edge prompts

$$\mathbf{h}_i^{(l)} = \text{COMB}^{(l)} \left(\mathbf{h}_i^{(l-1)}, \text{AGG}^{(l)} \left(\left\{ \mathbf{h}_j^{(l-1)} : v_j \in \mathcal{N}(v_i) \right\}, \left\{ \mathbf{e}_{ij}^{(l)} : v_j \in \mathcal{N}(v_i) \right\} \right) \right)$$

Representation-level Prompting

Hidden representation prompting

- EdgePrompt+¹: learn edge prompts for each edge
- Compute edge prompts as the weighted average of the anchor prompts

$$\mathbf{e}_{ij}^{(l)} = \sum_{m=1}^M b_{ijm}^{(l)} \mathbf{p}_m^{(l)}$$

M anchor prompts at the l -th layer

- $\mathbf{b}_{ij}^{(l)}$ is computed using a score function $\varphi^{(l)}$

$$\mathcal{P}^{(l)} = \{\mathbf{p}_1^{(l)}, \mathbf{p}_2^{(l)}, \dots, \mathbf{p}_M^{(l)}\}$$

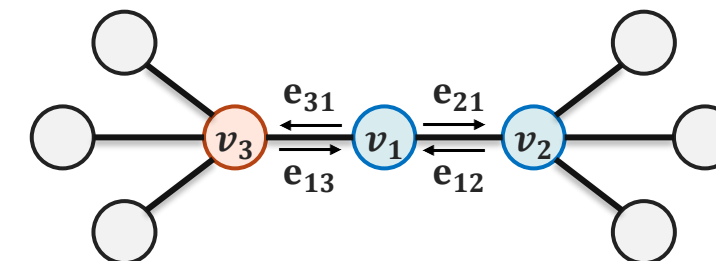
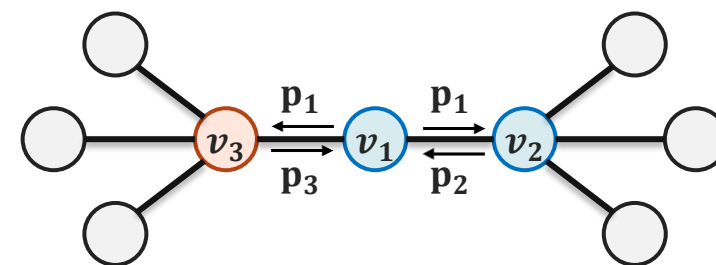
$$\mathbf{b}_{ij}^{(l)} = \text{Softmax} \left(\varphi^{(l)}(v_i, v_j) \right)$$

$$\varphi^{(l)}(v_i, v_j) = \text{LeakyReLU} \left(\left[\mathbf{h}_i^{(l-1)} \parallel \mathbf{h}_j^{(l-1)} \right] \cdot \mathbf{W}^{(l)} \right)$$

Representation-level Prompting

Hidden representation prompting

- Why EdgePrompt+ works?
- **Prompting on nodes**: add learnable prompts to node features or representations¹
- Limitation: \mathbf{p}_i will be uniformly aggregated by neighboring nodes
- **Prompting on edges**: add learnable prompts on edges²
- \mathbf{e}_{ij} provides customized prompts to different neighboring nodes
- Each node broadcasts distinct prompts to its neighbors



Representation-level Prompting

Hidden representation prompting

- EdgePrompt+ is more powerful than EdgePrompt
→ Customized edge prompts > shared edge prompts
- EdgePrompt+ is compatible with both generative and contrastive pre-training strategies
- GPF-plus is competitive among data-level prompting methods

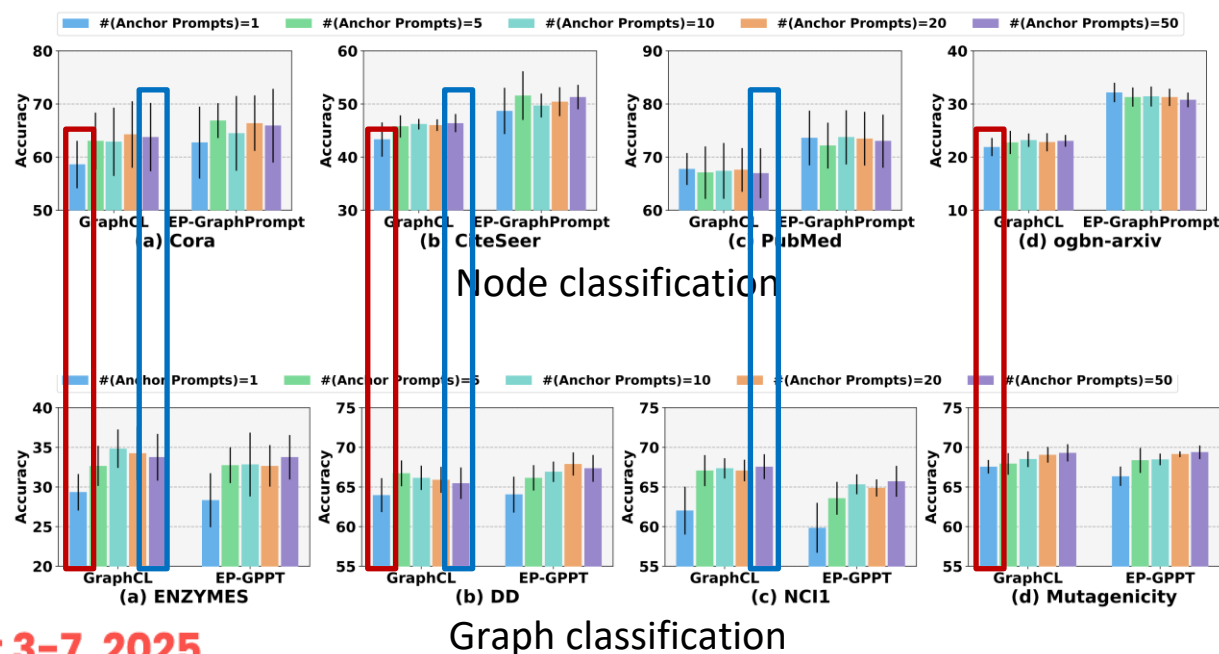
| Pre-training Strategies | Tuning Methods | ENZYMES | DD | NCI1 | NCI109 | Mutagenicity |
|-------------------------|---------------------------|--|--|--|--|--|
| GraphCL | Classifier Only | 30.50 \pm 1.16 | 62.89 \pm 2.19 | 62.49 \pm 1.95 | 61.68 \pm 0.93 | 66.62 \pm 1.87 |
| | GraphPrompt | 27.83 \pm 1.61 | 64.33 \pm 1.79 | 63.19 \pm 1.71 | 62.18 \pm 0.48 | 67.62\pm0.65 |
| | ALL-in-one | 25.92 \pm 0.55 | 66.54 \pm 1.82 | 57.52 \pm 2.61 | 62.74 \pm 0.78 | 63.43 \pm 2.53 |
| | GPF | 30.08 \pm 1.25 | 64.54 \pm 2.22 | 62.66 \pm 1.83 | 62.29 \pm 0.90 | 66.54 \pm 1.85 |
| | GPF-plus | 31.00 \pm 1.50 | 67.26 \pm 2.29 | 64.56 \pm 1.10 | 62.84 \pm 0.22 | 66.82 \pm 1.63 |
| | EdgePrompt EdgePrompt+ | 29.50 \pm 1.57 34.00\pm1.25 | 64.16 \pm 2.13 67.98\pm2.05 | 63.05 \pm 2.11 66.30\pm2.54 | 62.59 \pm 0.93 66.52\pm0.91 | 66.87 \pm 1.88 67.47\pm2.37 |
| SimGRACE | Classifier Only | 27.07 \pm 1.04 | 61.77 \pm 2.40 | 61.27 \pm 3.64 | 62.12 \pm 1.10 | 67.36 \pm 0.71 |
| | GraphPrompt | 26.87 \pm 1.47 | 62.58 \pm 1.84 | 62.45 \pm 1.52 | 62.41 \pm 0.69 | 68.03 \pm 0.78 |
| | ALL-in-one | 25.73 \pm 1.18 | 65.16 \pm 1.47 | 58.52 \pm 1.59 | 62.01 \pm 0.66 | 64.43 \pm 1.00 |
| | GPF | 28.53 \pm 1.76 | 65.64 \pm 0.70 | 61.45 \pm 3.13 | 61.90 \pm 1.26 | 67.19 \pm 0.74 |
| | GPF-plus | 27.33 \pm 2.01 | 67.20 \pm 1.56 | 61.61 \pm 2.89 | 62.84 \pm 0.23 | 67.69 \pm 0.64 |
| | EdgePrompt EdgePrompt+ | 29.33 \pm 2.30 32.67\pm2.53 | 63.97 \pm 2.14 67.72\pm1.62 | 62.02 \pm 3.02 67.07\pm1.96 | 62.02 \pm 1.03 66.53\pm1.30 | 67.55 \pm 0.85 68.31\pm1.36 |
| EP-GPPT | Classifier Only | 29.08 \pm 1.35 | 62.12 \pm 2.82 | 56.85 \pm 4.35 | 62.27 \pm 0.78 | 66.30 \pm 1.78 |
| | GraphPrompt | 26.67 \pm 1.60 | 61.61 \pm 1.91 | 58.77 \pm 0.97 | 62.16 \pm 0.89 | 66.37 \pm 1.17 |
| | ALL-in-one | 24.92 \pm 1.33 | 63.61 \pm 2.12 | 59.14 \pm 2.12 | 59.70 \pm 1.37 | 64.86 \pm 1.60 |
| | GPF | 28.33 \pm 1.73 | 63.48 \pm 2.08 | 58.14 \pm 4.16 | 62.52 \pm 1.39 | 66.10 \pm 0.96 |
| | GPF-plus | 29.25 \pm 1.30 | 66.92\pm2.34 | 62.93 \pm 3.23 | 64.13 \pm 1.42 | 67.57 \pm 1.45 |
| | EdgePrompt EdgePrompt+ | 28.33 \pm 3.41 32.75\pm2.26 | 64.03 \pm 2.26 66.16\pm1.60 | 59.85 \pm 3.15 63.58\pm2.07 | 62.98 \pm 1.44 65.15\pm1.60 | 66.36 \pm 1.22 68.35\pm1.57 |
| EP-GraphPrompt | Classifier Only | 31.33 \pm 3.22 | 62.58 \pm 2.40 | 62.09 \pm 2.31 | 60.19 \pm 1.71 | 65.13 \pm 0.81 |
| | GraphPrompt | 30.20 \pm 1.93 | 64.72 \pm 1.98 | 62.57 \pm 1.45 | 62.32 \pm 0.95 | 65.85 \pm 0.65 |
| | ALL-in-one | 29.07 \pm 1.16 | 65.60 \pm 2.38 | 58.67 \pm 2.42 | 57.69 \pm 1.08 | 64.66 \pm 0.76 |
| | GPF | 30.93 \pm 1.76 | 66.21 \pm 1.66 | 61.80 \pm 2.78 | 62.27 \pm 1.18 | 65.61 \pm 0.59 |
| | GPF-plus | 30.67 \pm 3.06 | 67.50\pm2.45 | 62.59 \pm 2.09 | 61.98 \pm 1.60 | 65.51 \pm 1.10 |
| | EdgePrompt EdgePrompt+ | 30.80 \pm 2.09 33.27\pm2.71 | 65.87 \pm 1.35 67.47\pm2.14 | 61.75 \pm 2.49 65.06\pm1.84 | 62.33 \pm 1.65 64.64\pm1.57 | 65.77 \pm 0.90 66.42\pm1.31 |

Graph classification

Representation-level Prompting

Hidden representation prompting

- One anchor prompt is insufficient in most cases
- Too many anchor prompts may not improve performance (hard to train)



Task-level Prompting

Particularly classification tasks

Learn trainable prompts at the task level

- Intuition: reformulate downstream tasks into alternative forms
- Formulation

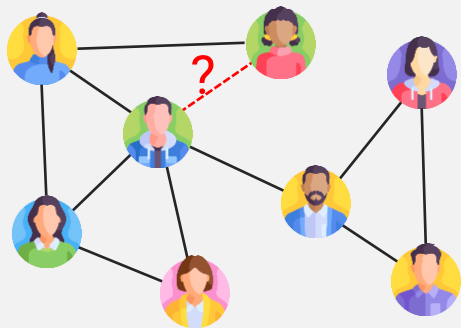
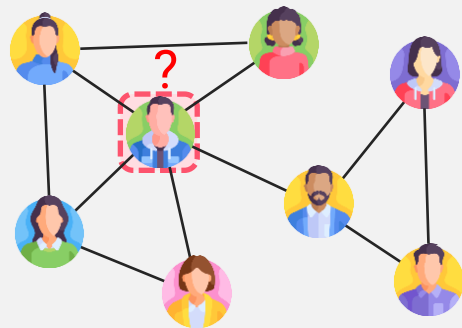
Given a downstream task \mathcal{L}_{DT} (e.g., node classification), task-level prompting \mathcal{T}^T transforms it into a different task $\tilde{\mathcal{L}}_{DT} = \mathcal{T}^T(\mathcal{L}_{DT}, \mathcal{P})$ with learnable prompts \mathcal{P} for adaptation.

- Two categories
 - Link prediction-based prompting
 - Similarity-based prompting

Task-level Prompting

Link prediction-based prompting

- Recall: the gap between edge prediction for pre-training and node classification as the downstream task

| Phase | Pre-training stage | Downstream stage |
|---------------------|--|---|
| Illustration |  |  |
| Task | Edge prediction | Node classification |
| Involved embeddings | A pair of nodes | A single node |
| Prediction | Edge probabilities | Node class probabilities |

Task-level Prompting

Link prediction-based prompting

- Intuition: convert node classification to edge prediction
- Pre-training stage: edge prediction with positive and negative pairs

Positive pair



$$g(v_1, v_2) = 1$$

Negative pair



$$g(v_1, v_2) = 0$$

Pre-training objective

$$\min_{\pi_{\text{GNN}}, \phi_{\text{Proj}}} \sum_{(v_i, v_j)} \mathcal{L}_{\text{CE}} \left(\phi_{\text{Proj}}(\mathbf{h}_i, \mathbf{h}_j), g(v_i, v_j) \right)$$

Task-level Prompting

Link prediction-based prompting

- Intuition: convert node classification to edge prediction
- Downstream stage: construct positive and negative pairs using learnable task tokens

Positive pair



$$g(v_1, y_1) = 1$$

Negative pair



$$g(v_1, y_2) = 0$$

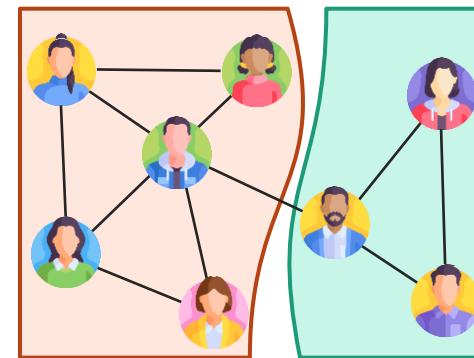
Downstream objective

$$\min_{\pi_{\text{GNN}}, \phi_{\text{Proj}}, \mathbf{E}} \sum_{(v_i, \mathbf{y}_c)} \mathcal{L}_{\text{CE}} \left(\phi_{\text{Proj}}(\mathbf{h}_i, \mathbf{e}_c), g(v_i, \mathbf{y}_c) \right)$$

Task-level Prompting

Link prediction-based prompting

- GPPT: one global task token \mathbf{E} is insufficient for all nodes¹
- Task tokens should vary with clusters
- For each cluster m , train an independent task token \mathbf{E}^m



$$\min_{\pi_{\text{GNN}}, \phi_{\text{Proj}}, \mathbf{E}^1, \dots, \mathbf{E}^M} \sum_{(v_i, y_c)} \mathcal{L}_{\text{CE}} \left(\phi_{\text{Proj}}(\mathbf{h}_i, \mathbf{e}_c^m), g(v_i, y_c) \right)$$

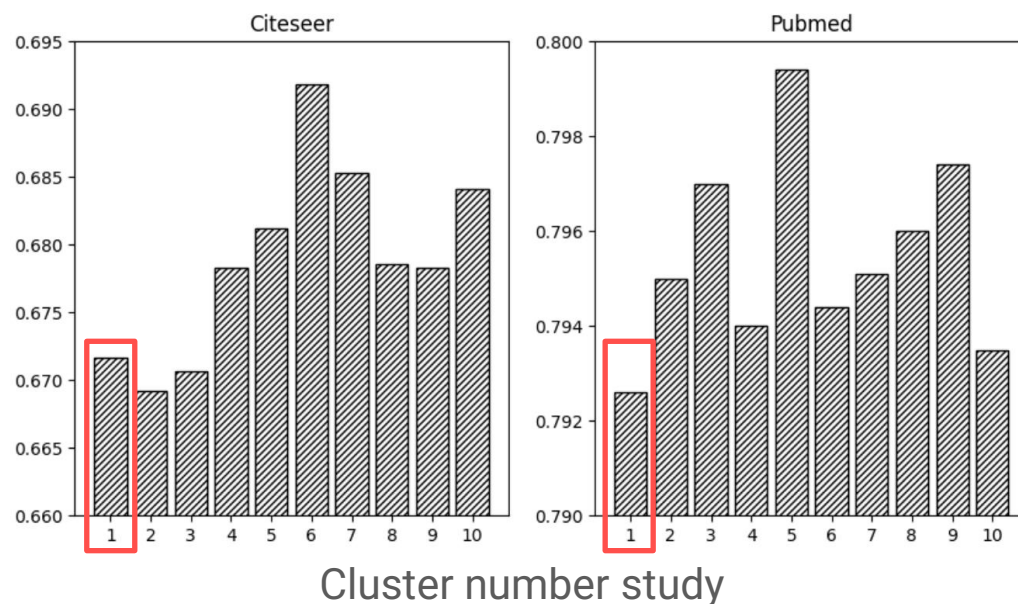
- Use structure tokens to replace node embeddings: $\mathbf{e}_i = a_i \cdot \mathbf{h}_i + \sum_{v_j \in \mathcal{N}(v_i)} a_j \cdot \mathbf{h}_j$
- Final objective

$$\min_{\pi_{\text{GNN}}, \phi_{\text{Proj}}, \mathbf{E}^1, \dots, \mathbf{E}^M} \sum_{(v_i, y_c)} \mathcal{L}_{\text{CE}} \left(\phi_{\text{Proj}}(\mathbf{e}_i, \mathbf{e}_c^m), g(v_i, y_c) \right) + \lambda \sum_m \|\mathbf{E}^M (\mathbf{E}^M)^T - \mathbf{I}\|_F^2$$

Task-level Prompting

Link prediction-based prompting

- The impact of cluster numbers in GPPT¹
- The performance of GPPT is significantly damaged with one cluster



Task-level Prompting

Similarity-based prompting

- Intuition: compare graph representations to class prototypes using contrastive loss

$$\min_{\mathcal{P}} \sum_{(x,y) \in \mathcal{D}} -\log \frac{\exp(\text{sim}(\mathbf{h}_i, \mathbf{s}_y)/\tau)}{\sum_{c \in \mathcal{Y}} \exp(\text{sim}(\mathbf{h}_i, \mathbf{s}_c)/\tau)}$$

Class prototypes

- How to get class prototypes?
 - GraphPrompt, GraphPrompt+, ProNoG: use the average of instance representations belonging to class c
 - HetGPT¹: initialized using average representation and tuned while prompting

Comparison of Graph Prompting Techniques

| Technique Categories | Prompting Stage | Prompting Strategies | Single Forward Pass | DT Universality |
|--------------------------------|----------------------|---------------------------------|---------------------|-----------------|
| Data-level prompting | Graph data | Feature-based prompting | ✗ | ✓ |
| | | Insertion-based prompting | ✗ | ✓ |
| Representation-level prompting | Node representations | Output representation prompting | ✓ | ✓ |
| | | Hidden representation prompting | ✗ | ✓ |
| Task-level prompting | Downstream tasks | Link prediction-based prompting | ✓ | ✗ |
| | | Similarity-based prompting | ✓ | ✗ |

Section 4

Summary and Future Directions

Xingbo Fu

PhD Candidate

University of Virginia



Summary

Graph prompting

- Objective gap between pre-training and downstream tasks
 - Example: link prediction → node classification
- Graph pre-training methods: the foundation step of graph prompting
 - Generative methods, contrastive method, multi-task methods
- Mainstream techniques in graph prompting
 - Data-level, representation-level, task-level techniques



Future Directions

Benchmarks and datasets

- Standard experimental settings for performance evaluation
- Cross-dataset & cross-domain evaluation
- One existing benchmark¹



☀ ProG: A Unified Python Library for Graph Prompting ☀

| [Quick Start](#) | [Paper](#) | [Media Coverage](#) | [Call For Contribution](#) |

Latest version **v0.2** PyTorch **v1.13.1** license **MIT** python **>=3.9**



[1] Zi, Chenyi, et al. "ProG: A Graph Prompt Learning Benchmark." NeurIPS 2024.

Future Directions

Theoretical foundation

- Empirical improvements → theoretical guarantees

Theorem 3. *Given a GPF-like prompt vector p_ω , if a GCN model F_θ has non-linear function layers but the model's weight matrix is row full-rank, then there exists an optimal ω for any input graph G such that $P_\omega(G) \in B_G$.*

Theorem 4. *Given the All-in-One-like prompt graph SG_ω (a subgraph containing prompt tokens and token structures), if a GCN model F_θ does not have any non-linear transformations, or has non-linear layers but the model's weight matrix is row full-rank, then there exists an optimal ω for any input graph G such that $P_\omega(G) \in B_G$.*

The upper bound of graph prompting¹

Future Directions

Theoretical foundation

- Empirical improvements → theoretical guarantees

Theorem 1. (*Universal Capability of GPF*) Given a pre-trained GNN model f , an input graph $\mathcal{G}: (\mathbf{A}, \mathbf{X})$, an arbitrary prompting function $\psi_t(\cdot)$, for any prompted graph $\hat{\mathcal{G}}: (\hat{\mathbf{A}} \in \mathbb{A}, \hat{\mathbf{X}} \in \mathbb{X})$ in the candidate space of the graph template $\mathcal{G}^* = \psi_t(\mathcal{G})$, there exists a GPF extra feature vector \hat{p} that satisfies:

$$f(\mathbf{A}, \mathbf{X} + \hat{p}) = f(\hat{\mathbf{A}}, \hat{\mathbf{X}}) \quad (11)$$

Universal capability of graph prompting¹

Future Directions

Theoretical foundation

- Empirical improvements → theoretical guarantees

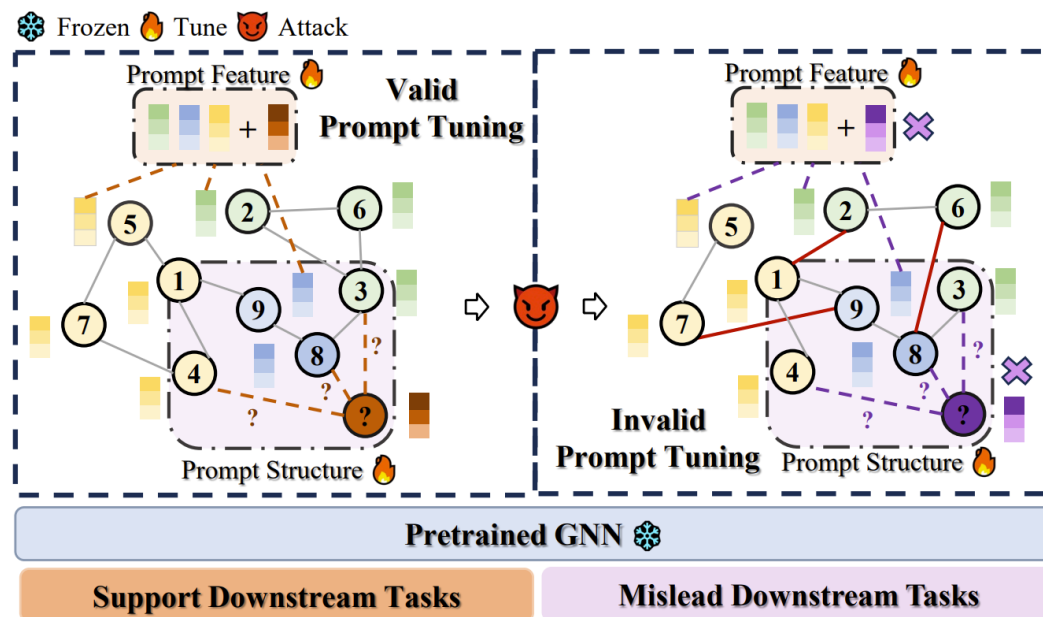
Theorem 1. *Given a random graph $\mathcal{G} \sim \text{CSBM}(\mu_1, \mu_2, p, q)$ and a pre-trained GCN model f , there always exist a set of $M \geq 2$ anchor prompts $\mathcal{P} = \{\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_M\}$ and the score vectors $\mathbf{b}_{i,j}$ for each edge (v_i, v_j) that improve the expected distance after GCN operation between classes c_1 and c_2 to T times without using edge prompts, where $T \in (1, 1 + \frac{p}{|p-q|}]$.*

Separability improvements by graph prompting¹

Future Directions

Attacks & defense in graph prompting

- A new attack and defense perspective by graph prompting

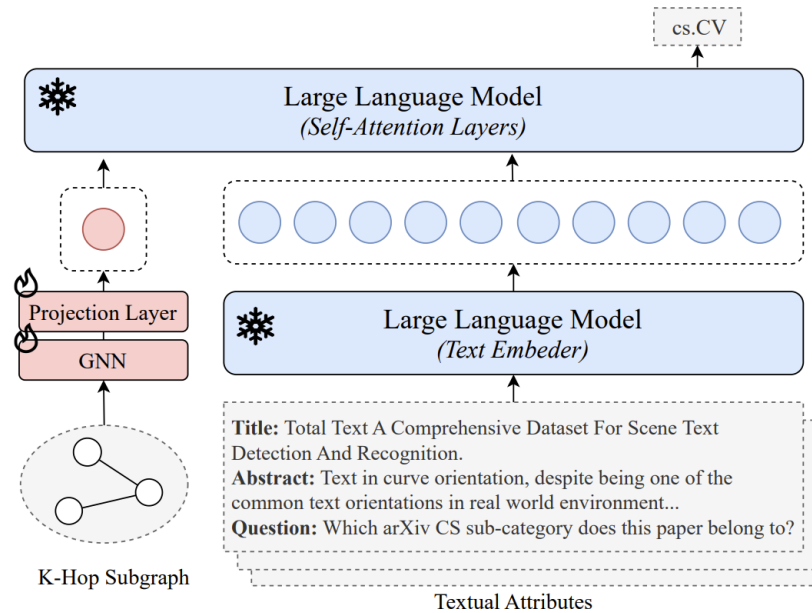


Adversarial attacks in graph prompting¹

Future Directions

LLM Incorporation for graph prompting

- Graph prompting with semantic information in graph data
- Text-attributed graphs (TAGs)

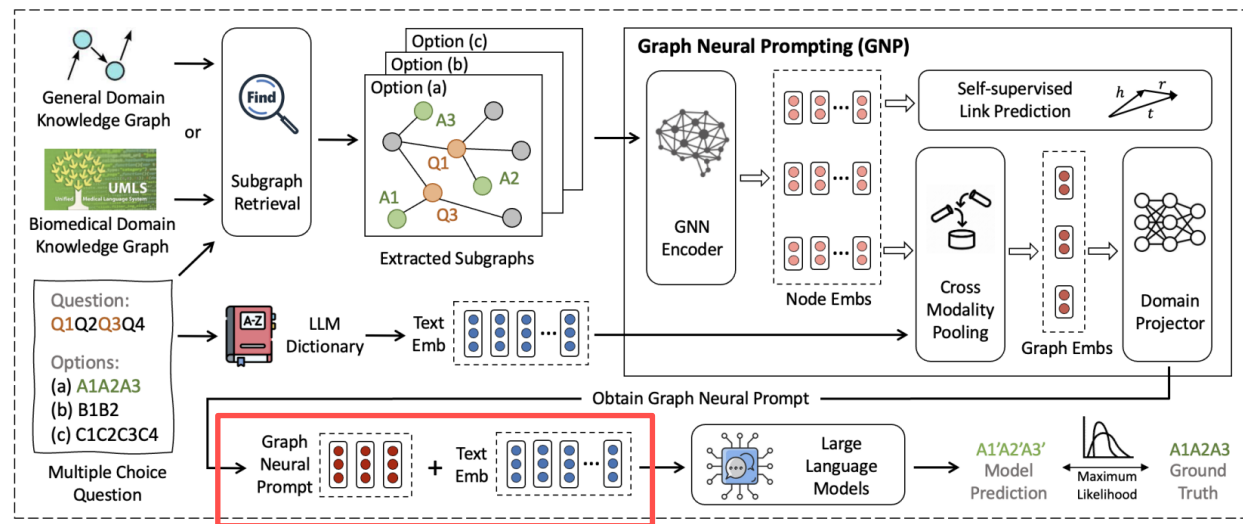


GraphPrompter¹

Future Directions

LLM Incorporation for graph prompting

- Graph prompting with semantic information in graph data
- Knowledge graphs (KGs)

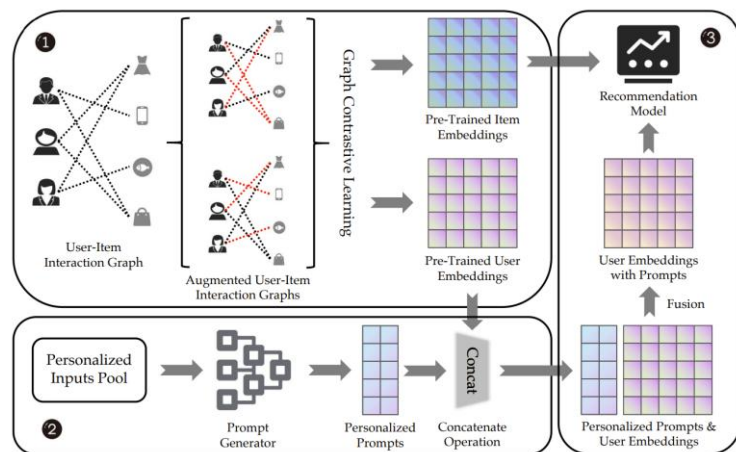


Graph Neural Prompting (GNP)¹

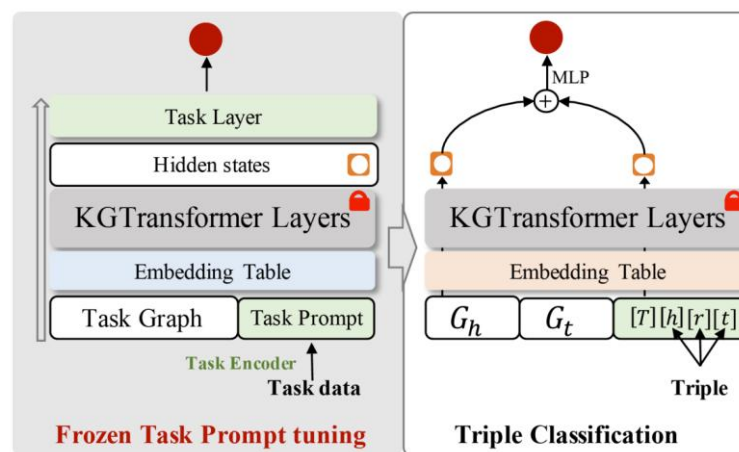
[1] Tian, Yijun, et al. "Graph Neural Prompting with Large Language Models." AAAI 2024.

Future Directions

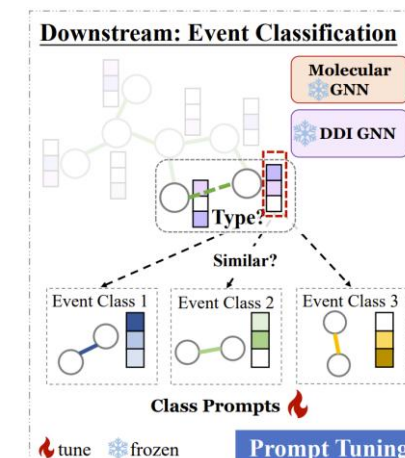
Applications in various domains



Recommendation systems¹



Knowledge engineering²



Biology and medicine³

[1] Yang, Haoran, et al. "An Empirical Study Towards Prompt-Tuning for Graph Contrastive Pre-Training in Recommendations." NeurIPS 2023.

[2] Zhang, Wen, et al. "Structure Pretraining and Prompt Tuning for Knowledge Graph Transfer." WWW 2023.

[3] Wang, Yingying, et al. "DDIPrompt: Drug-Drug Interaction Event Prediction based on Graph Prompt Learning." CIKM. 2024.

THANK YOU!

