重庆大学
CHONGQING UNIVERSITY

All in One: Multi-Task Prompting for Graph Neural Networks

KDD 2023

汇报人：Nobody

时间：2023-10-17

All in One: Multi-Task Prompting for Graph Neural Networks

Part.01

作者信息

• 孙相国

---- 博士后

xiangguo sun

The Chinese University of Hong Kong
在 cuhk.edu.hk 的电子邮件经过验证

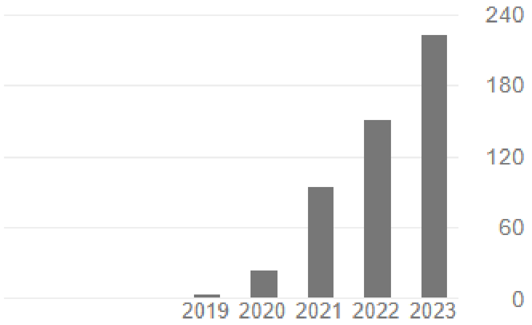
关注

创建我的个人资料

标题	引用次数	年份
Multi-level graph convolutional networks for cross-platform anchor link prediction CCF A H Chen, H Yin, X Sun, T Chen, B Gabrys, K Musial Proceedings of the 26th ACM SIGKDD international conference on knowledge …	105	2020
Heterogeneous hypergraph embedding for graph classification CCF B X Sun, H Yin, B Liu, H Chen, J Cao, Y Shao, NQ Viet Hung Proceedings of the 14th ACM international conference on web search and data …	70	2021
Who am I? Personality detection based on deep learning for texts CCF C X Sun, B Liu, J Cao, J Luo, X Shen 2018 IEEE international conference on communications (ICC), 1-6	70	2018

引用次数

	总计	2018 年至今
引用	498	497
h 指数	11	11
i10 指数	11	11



All in One: Multi-Task Prompting for Graph Neural Networks

Part.02

论文介绍

- **背景**

大语言模型的出色性能，仅用少量提示，就能适配下游任务

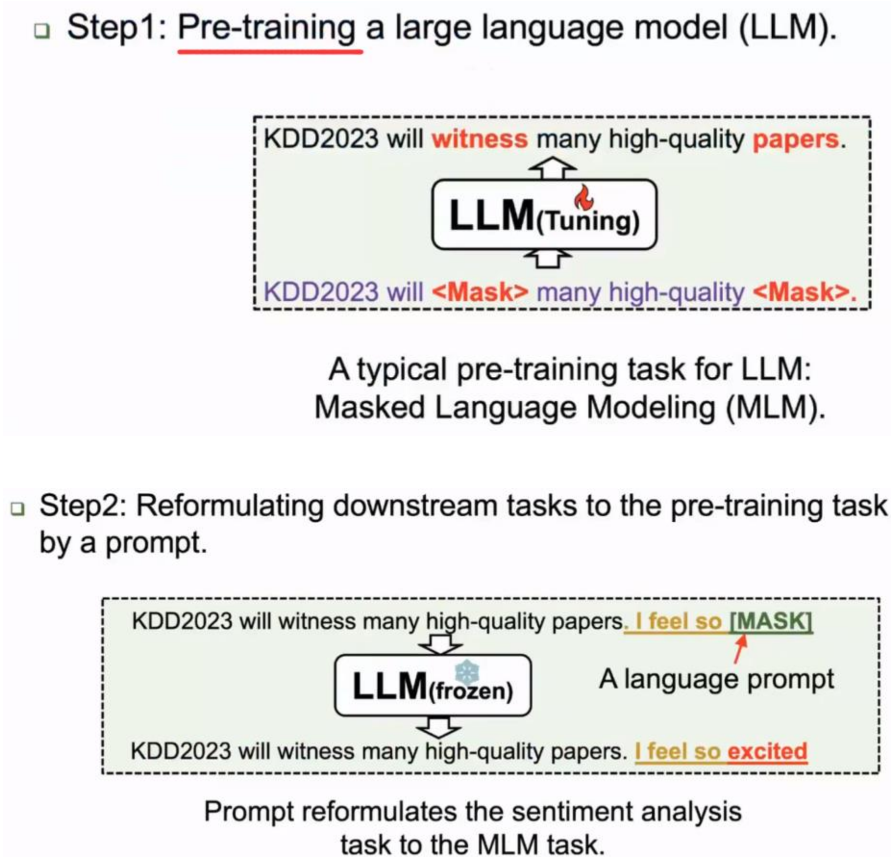
图结构数据的学习难以运用到大模型中，如何利用少量提示，使得预训练图模型适配图相关下游任务，激发出预训练模型的性能

大语言模型提示学习 ⟷ 关联？区别？ 预训练图模型提示学习

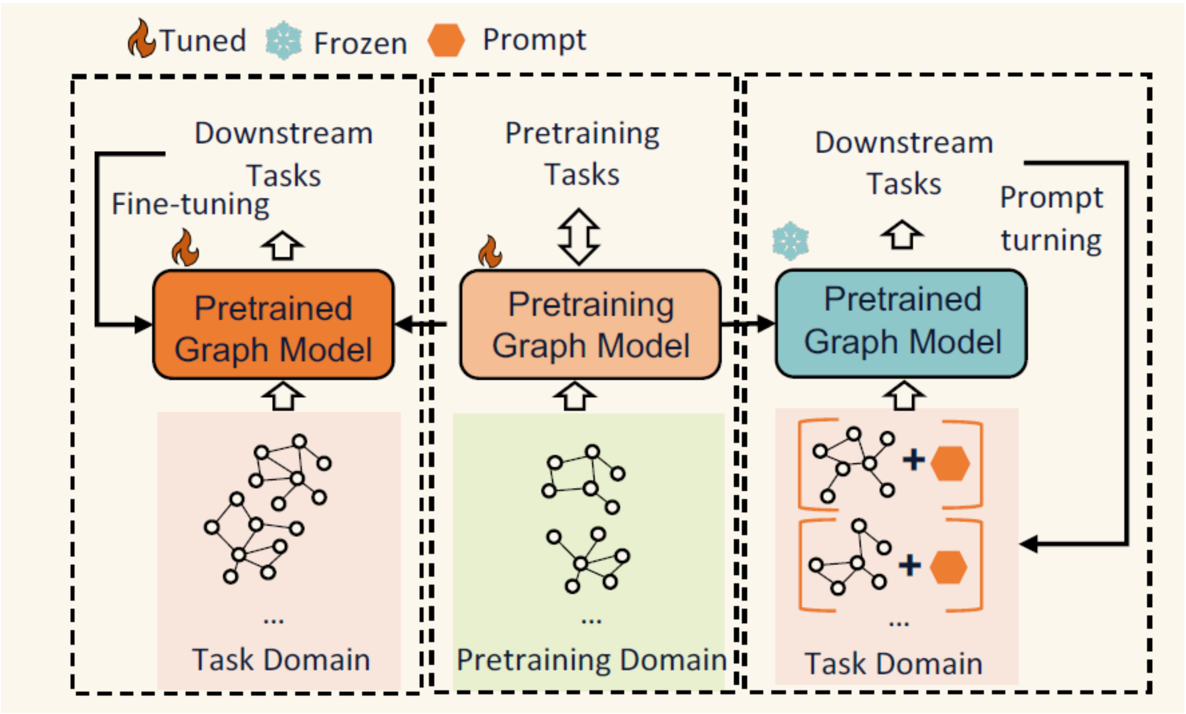
- Prompt Learning Prompt Learning 超强入门教程

提示是给预先训练的模型的提示，让它更好地理解问题

1. 大语言模型提示学习

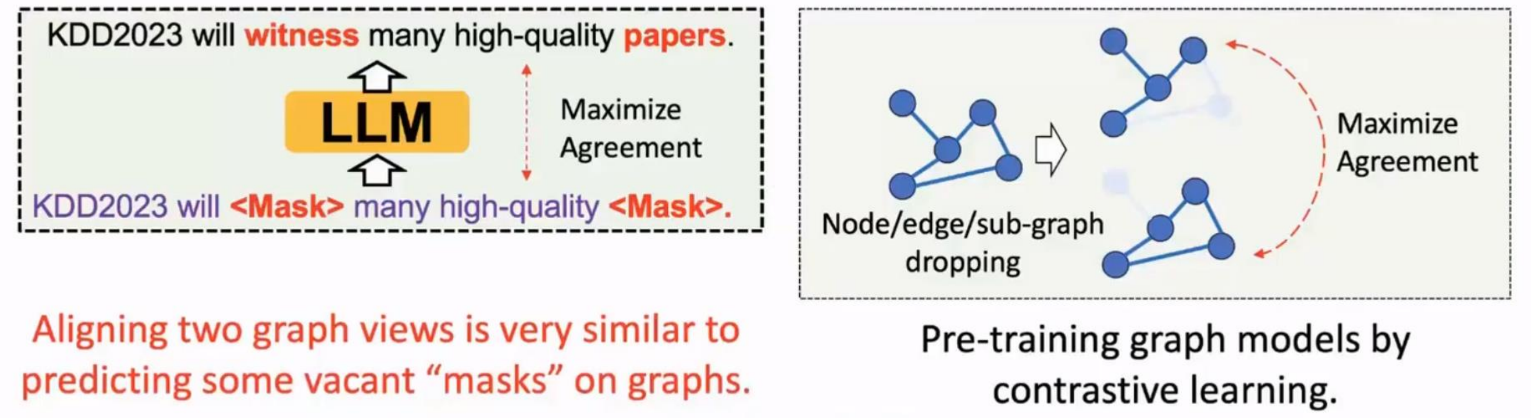


2. 图预训练模型提示学习

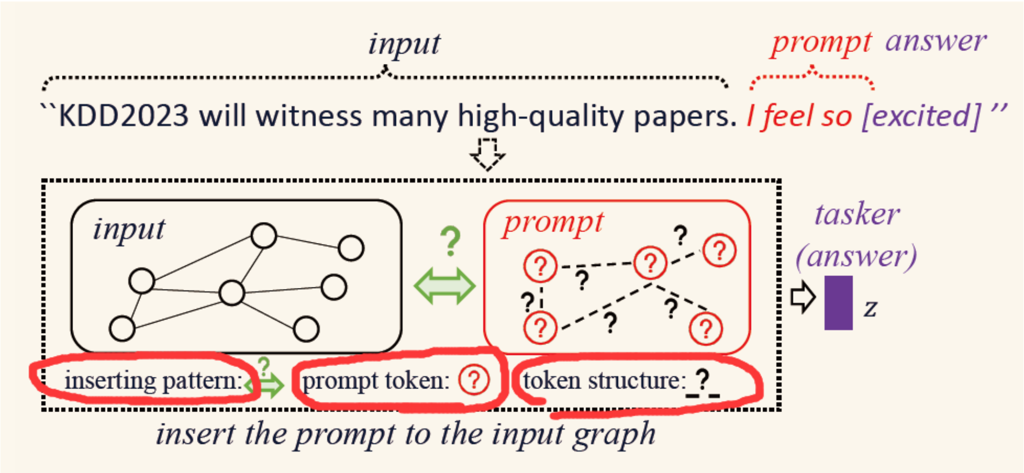


- **关联与区别**

1. 关联：预测MASK的部分



2. 区别：提示的结合

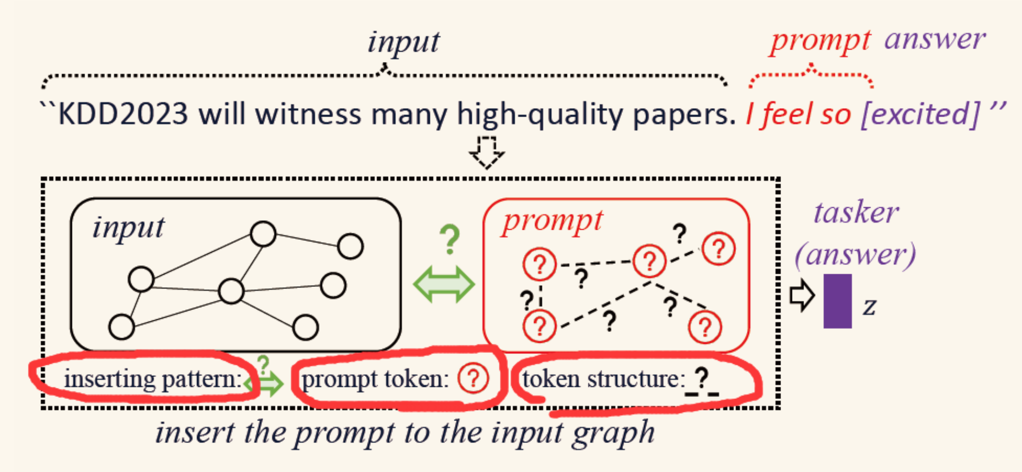


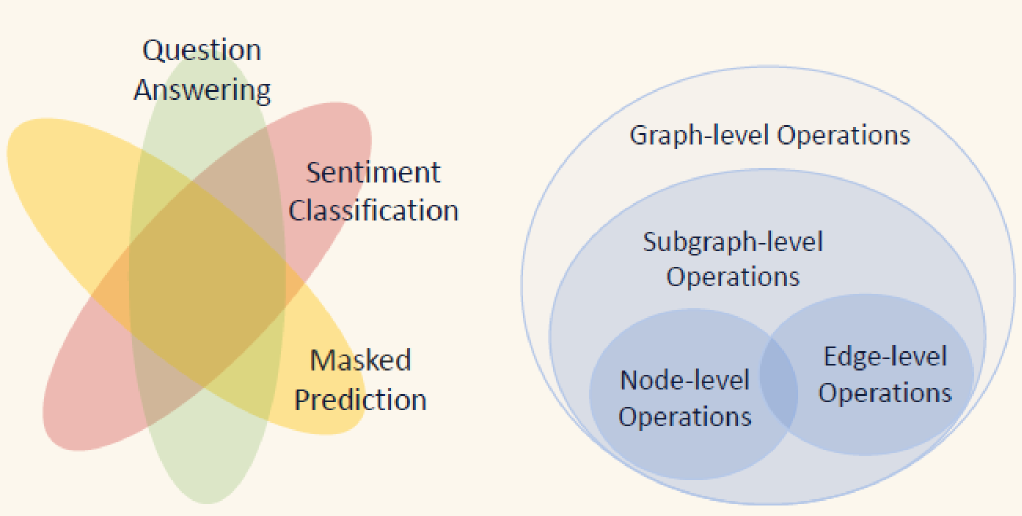
- **难点**

1. 提示词如何构建、提示词的结构如何定义、如何与原图结合

2. 如何统一各种下游图任务，已有工作只能适配于其中一种

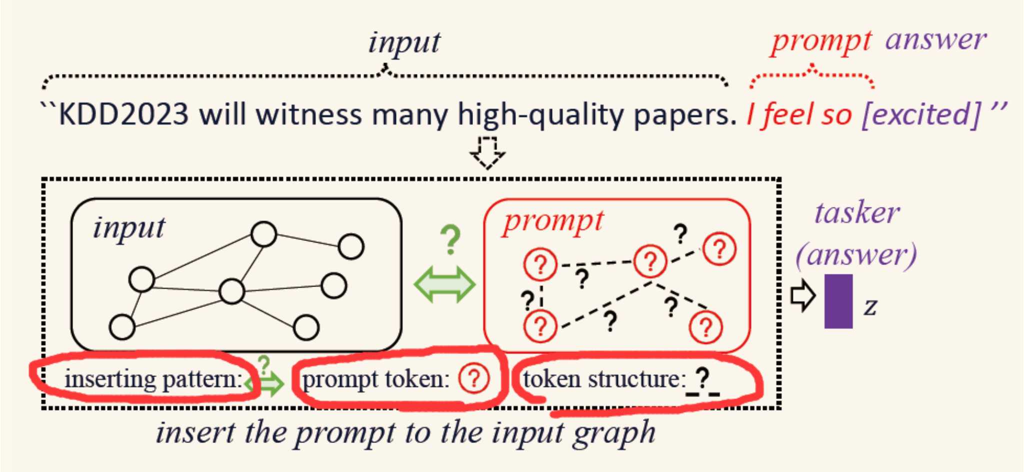
3. 少样本情况下，如何学习到能运用于快速适配到各种下游任务的初始嵌入





(a) NLP tasks (b) graph tasks

- **难点1** (a)提示词如何构建、(b)提示词的结构如何定义、(c)如何与原图结合



原图为$\mathcal{G}=(\mathcal{V},\mathcal{E})$，其中$\mathcal{V}=\{v_1,v_2,\cdots,v_N\}$为节点，$\mathcal{E}=\{(v_i,v_j|v_i,v_j\in\mathcal{V})\}$为边，每个节点的特征表示为$x_i\in\mathbb{R}^{1\times d}$。

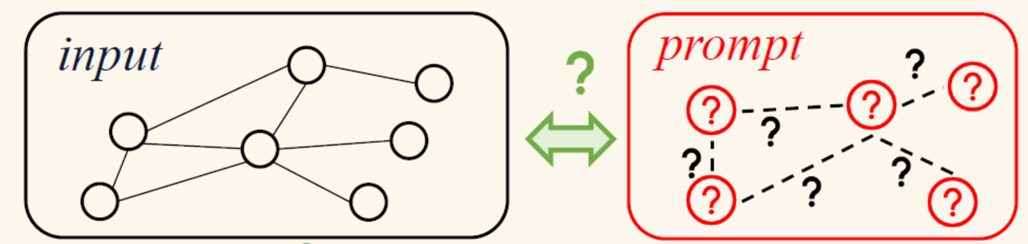
提示图为$\mathcal{G}_p=(\mathcal{P},\mathcal{S})$，其中$\mathcal{P}=\{p_1,p_2,\cdots,p_{|\mathcal{P}|}\}$为节点，$\mathcal{S}=\{(p_i,p_j|p_i,p_j\in\mathcal{P})\}$为边，每个节点的特征表示为$p_i\in\mathbb{R}^{1\times d}$。

其中$|\mathcal{P}|\ll N$且$|\mathcal{P}|\ll d_h$

解决

- **难点1** (a)提示词如何构建、(b)提示词的结构如何定义、(c)如何与原图结合

(a)设计提示的数量，即节点数，学习这部分的节点特征即可



(b)这些提示节点该具有何种结构？形成提示图

$$\mathcal{A} = \bigcup_{\substack{i=1 \\ j=i+1}}^{|\mathcal{P}|-1} \{a_{ij}\}$$

(b.1)参数学习

$$(p_i, p_j) \in \mathcal{S} \text{ iff } \sigma(\mathrm{p}_i \cdot \mathrm{p}_j) < \delta$$

(b.2)点积阈值

$$\mathcal{S} = \emptyset.$$

(b.3)无结构

(c)提示图该如何与原始图结合？以发挥最大功效

为何不好?
本文好在?

前人工作：直接将学习到的下游任务提示，与所有节点特征相加 $\hat{x}_i = x_i + p, i \in \{1,2,\cdots,N\}$

ψ为结合函数，能将原始图与提示图结合：$\mathcal{G}_m = \psi(\mathcal{G}, \mathcal{G}_p)$

$$\hat{\mathrm{x}}_i = \mathrm{x}_i + \sum_{k=1}^{|\mathcal{P}|} w_{ik}\mathrm{p}_k \qquad w_{ik} = \begin{cases} \sigma(\mathrm{p}_k \cdot \mathrm{x}_i^T), & \text{if } \sigma(\mathrm{p}_k \cdot \mathrm{x}_i^T) > \delta \\ 0, & \text{otherwise} \end{cases}$$

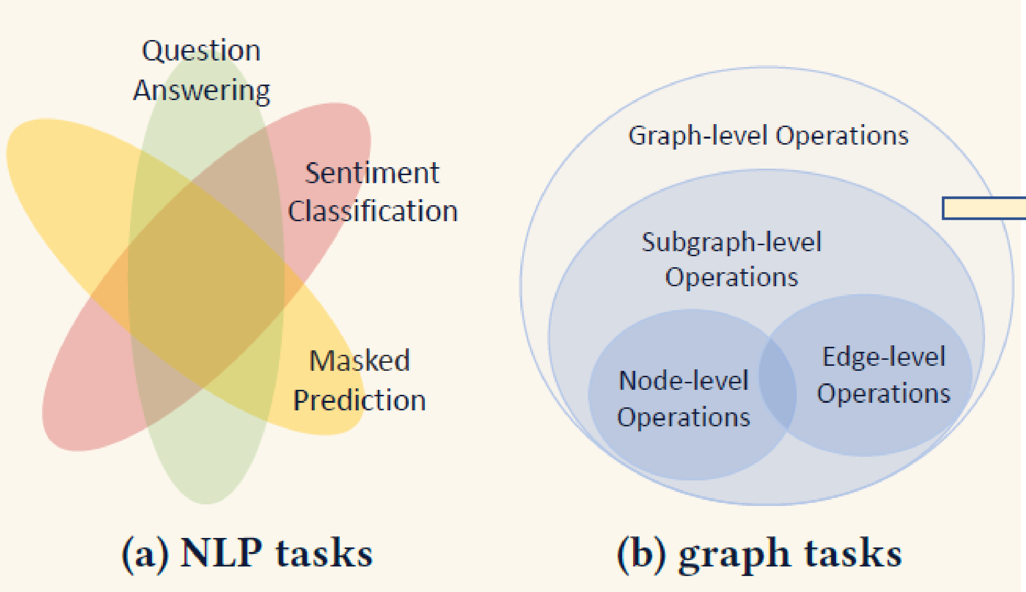
(c.1)点积加权

$$\hat{\mathrm{x}}_i = \mathrm{x}_i + \sum_{k=1}^{|\mathcal{P}|} \mathrm{p}_k.$$

(c.2)直接相加

- **难点2**　如何统一各种下游图任务，已有工作只能适配于其中一种

为什么之前的都是在整个图上进行操作的？节点分类、链路预测之类的呢？

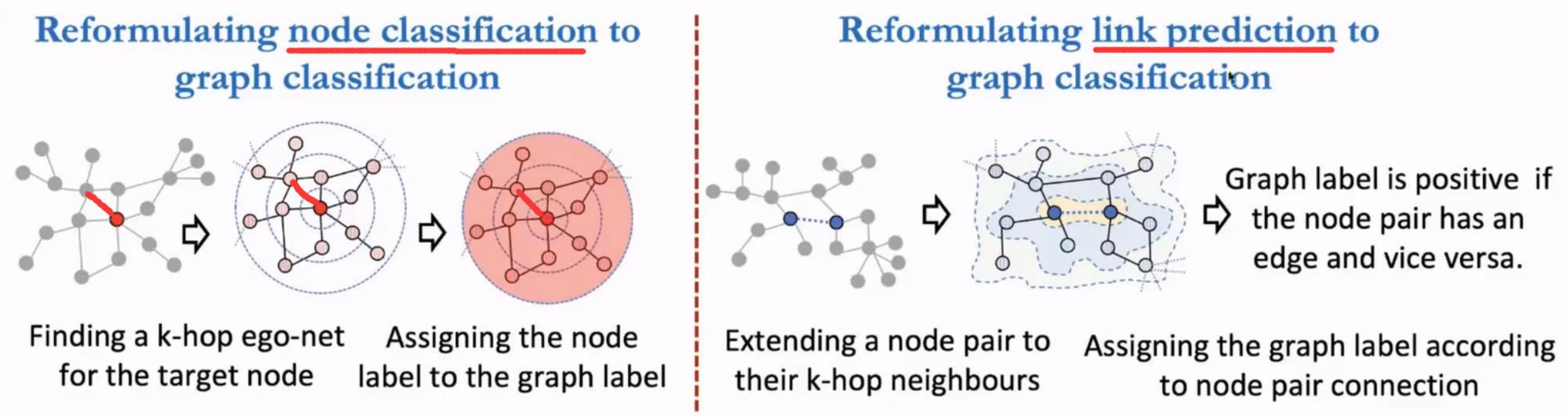


为何在图级别的空间上能更有效？
很多边、节点的操作都可以视作是图上的操作

如何适配（统一）多种图上的下游任务
(a)在什么空间上更有可能能适配图上的各种任务？

- **难点2** 如何统一各种下游图任务，已有工作只能适配于其中一种

(b)怎么将边、节点级别的任务转换为图级别的任务?



- **难点3**　少样本情况下，如何学习到能运用于快速适配到各种下游任务的初始嵌入

元学习

Algorithm 1: Overall Learning Process

Input: Overall pipeline $f_{\theta,\phi|\pi^*}$ with prompt parameter θ, pre-trained model with frozen parameter π^*, and task head parameterized by ϕ; Multi-task episodes $\mathcal{E} = \{\mathcal{E}_1, \cdots, \mathcal{E}_n\}$;

Output: Optimal pipeline $f_{\theta^*,\phi^*|\pi^*}$

Initialize θ and ϕ

while *not done* **do**

  `// inner adaptation`

  Sample $\mathcal{E}_i \in \mathcal{E}$ where $\mathcal{E}_i = (\mathcal{T}_{\mathcal{E}_i}, \mathcal{L}_{\mathcal{E}_i}, \mathcal{S}_{\mathcal{E}_i}, \mathcal{Q}_{\mathcal{E}_i})$

  for $\tau_{\triangleleft t} \in \mathcal{T}_{\mathcal{E}_i}, \triangleleft = g, n, \ell$ **do**　任务

    $\theta_{\tau_{\triangleleft t}}, \phi_{\tau_{\triangleleft t}} \leftarrow \theta, \phi$

    $\theta_{\tau_{\triangleleft t}} \leftarrow \theta_{\tau_{\triangleleft t}} - \alpha \nabla_{\theta_{\tau_{\triangleleft t}}} \mathcal{L}^{(\triangleleft)}_{\mathcal{D}^s_{\tau_{\triangleleft t}}} \left(f_{\theta_{\tau_{\triangleleft t}}, \phi_{\tau_{\triangleleft t}} | \pi^*} \right)$

    $\phi_{\tau_{\triangleleft t}} \leftarrow \phi_{\tau_{\triangleleft t}} - \alpha \nabla_{\phi_{\tau_{\triangleleft t}}} \mathcal{L}^{(\triangleleft)}_{\mathcal{D}^s_{\tau_{\triangleleft t}}} \left(f_{\theta_{\tau_{\triangleleft t}}, \phi_{\tau_{\triangleleft t}} | \pi^*} \right)$

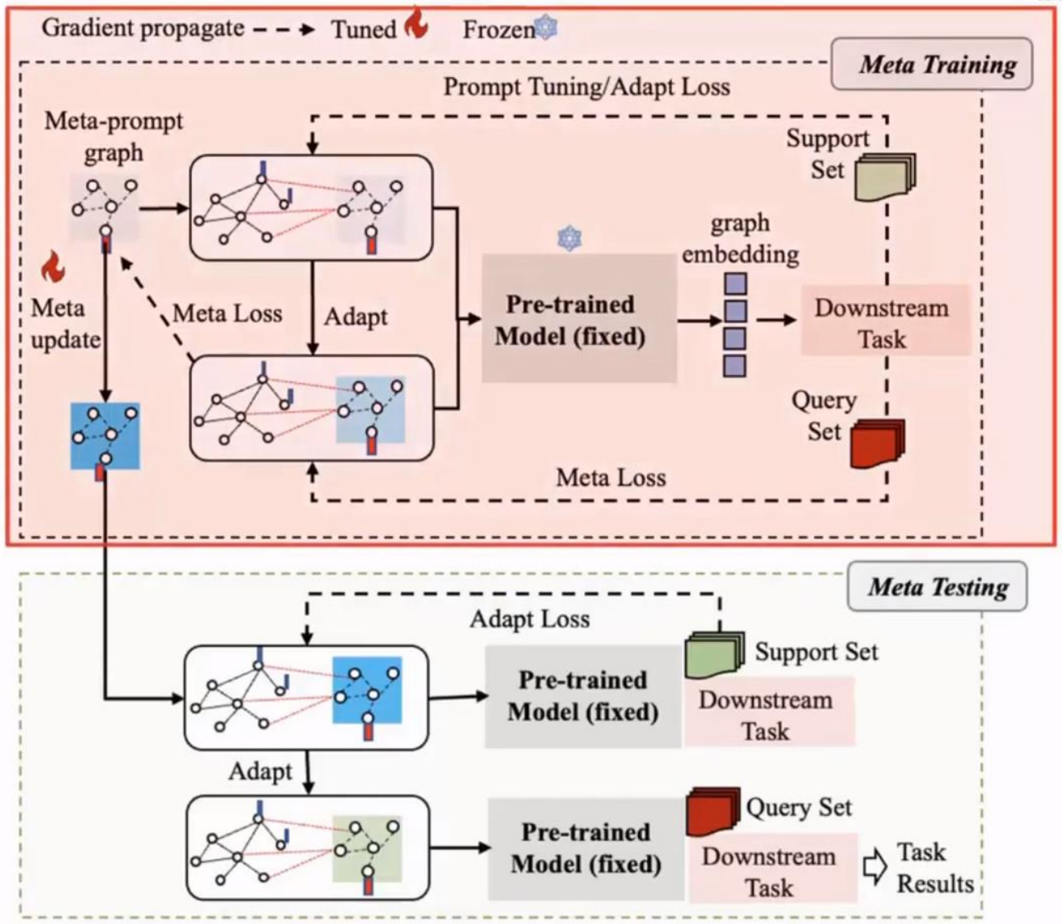
  end

  `// outer meta update`

  Update θ, ϕ by Equation (4) on $\mathcal{Q}_{\mathcal{E}_i} = \{\mathcal{D}^q_{\tau_{\triangleleft t}} | \tau_{\triangleleft t} \in \mathcal{T}_{\mathcal{E}_i}, \triangleleft = g, n, \ell\}$

end

return $f_{\theta^*,\phi^*|\pi^*}$

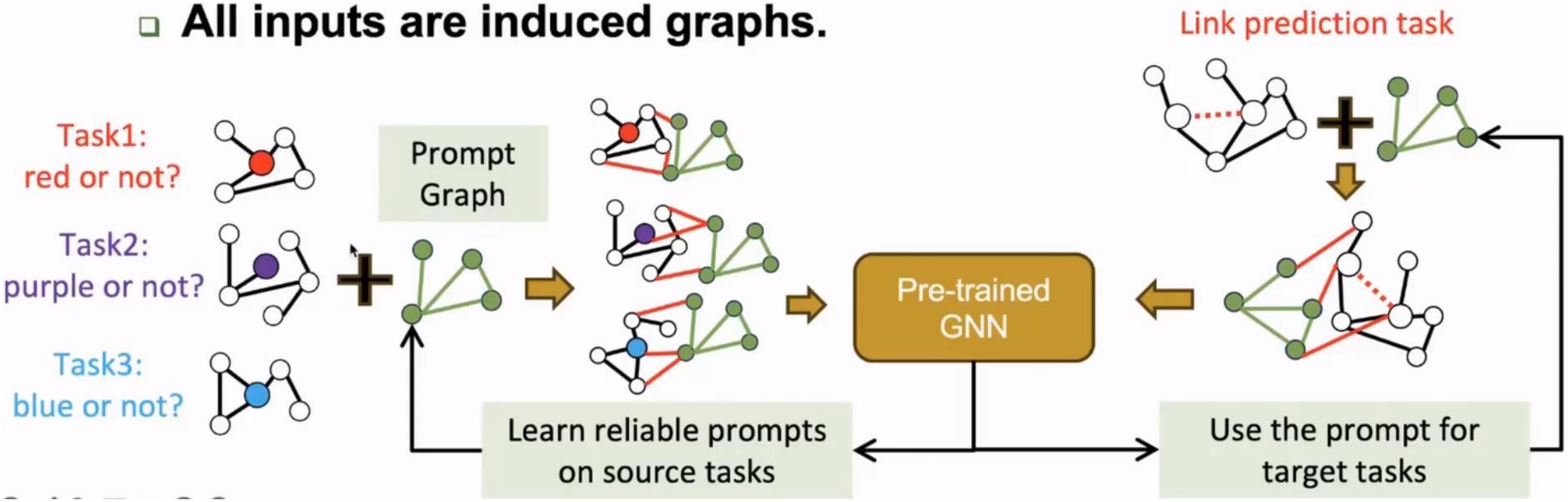


- **难点3** 少样本情况下，如何学习到能运用于快速适配到各种下游任务的初始嵌入

例子

An Example:

- **Target task:** link prediction.
- **Source tasks:** Node binary classification tasks.
  - Each task corresponds to one node class.
- **All inputs are induced graphs.**



为何有效

为何不好?
本文好在?

前人工作：直接将学习到的下游任务提示，与所有节点特征相加 $\hat{x}_i = x_i + p, i \in \{1,2,\cdots,N\}$

Fang et al. [1] proved that we can always learn an appropriate prompt token $p*$ making the following equation stand:

$$\varphi^*(\boldsymbol{A}, \boldsymbol{X} + p^*) = \varphi^*(g(\boldsymbol{A}, \boldsymbol{X})) + O_{p\varphi}$$

误差

- φ^*: pre-trained model
- p^*: a prompt token
- $\boldsymbol{A}, \boldsymbol{X}$: adjacent matrix and feature matrix
- $g(.)$: graph manipulation (e.g. "changing node features", "adding or removing edges/subgraphs" etc)

This means we can learn an appropriate token applied to the original graph to imitate any graph manipulation.

The error bound $O_{p\varphi}$ is related to: (1) some non-linear layers of the model (unchangeable), and (2) the quality of the learned prompt (changeable), which is promising to be further narrowed down by a more advanced prompt scheme.

$$\varphi^*(\underbrace{\boldsymbol{A}, \boldsymbol{X} + p^*}) = \varphi^*(g(\boldsymbol{A}, \boldsymbol{X})) + O_{p\varphi}$$

$$\Downarrow \qquad \vee \text{大于}$$

$$\varphi^*(\psi(\mathcal{G}, \mathcal{G}_p^*)) = \varphi^*(g(\boldsymbol{A}, \boldsymbol{X})) + O_{p\varphi}^*$$

We extend the standalone token p^* to a prompt graph $\mathcal{G}_p^*$ that has multiple prompt tokens with learnable inner structures and more advanced inserting pattern ψ to the original graph $\mathcal{G}$

All in One: Multi-Task Prompting for Graph Neural Networks

Part.03

实验

- **性能比较**

Table 2: Node-level performance (%) with 100-shot setting. IMP (%): the average improvement of prompt over the rest.

Training schemes	Methods	Cora			CiteSeer			Reddit			Amazon			Pubmed		
		Acc	F1	AUC	Acc	F1	AUC	Acc	F1	AUC	Acc	F1	AUC	Acc	F1	AUC
supervised	GAT	74.45	73.21	82.97	83.00	83.20	89.33	55.64	62.03	65.38	79.00	73.42	97.81	75.00	77.56	79.72
	GCN	77.55	77.45	83.71	88.00	81.79	94.79	54.38	52.47	56.82	95.36	93.99	96.23	53.64	66.67	69.89
	GT	74.25	75.21	82.04	86.33	85.62	90.13	61.50	61.38	65.56	85.50	86.01	93.01	51.50	67.34	71.91
pre-train + fine-tune	GraphCL+GAT	76.05	76.78	81.96	87.64	88.40	89.93	57.37	66.42	67.43	78.67	72.26	95.65	76.03	77.05	80.02
	GraphCL+GCN	78.75	79.13	84.90	87.49	89.36	90.25	55.00	65.52	74.65	96.00	95.92	98.33	69.37	70.00	74.74
	GraphCL+GT	73.80	74.12	82.77	88.50	88.92	91.25	63.50	66.06	68.04	94.39	93.62	96.97	75.00	78.45	75.05
	SimGRACE+GAT	76.85	77.48	83.37	90.50	91.00	91.56	56.59	65.47	67.77	84.50	84.73	89.69	72.50	68.21	81.97
	SimGRACE+GCN	77.20	76.39	83.13	83.50	84.21	93.22	58.00	55.81	56.93	95.00	94.50	98.03	77.50	75.71	87.53
	SimGRACE+GT	77.40	78.11	82.95	87.50	87.05	91.85	66.00	69.95	70.03	79.00	73.42	97.58	70.50	73.30	74.22
prompt	GraphCL+GAT	76.50	77.26	82.99	88.00	90.52	91.82	57.84	67.02	75.33	80.01	75.62	97.96	77.50	78.26	83.02
	GraphCL+GCN	79.20	79.62	85.29	88.50	91.59	91.43	56.00	68.57	78.82	96.50	96.37	98.70	72.50	72.64	79.57
	GraphCL+GT	75.00	76.00	83.36	91.00	91.00	93.29	65.50	66.08	68.86	95.50	95.43	97.56	76.50	79.11	76.00
	SimGRACE+GAT	76.95	78.51	83.55	93.00	93.14	92.44	57.63	66.64	69.43	95.50	95.43	97.56	73.00	74.04	81.89
	SimGRACE+GCN	77.85	76.57	83.79	90.00	89.47	94.87	59.50	55.97	59.46	95.00	95.24	98.42	78.00	78.22	87.66
	SimGRACE+GT	78.75	79.53	85.03	91.00	91.26	95.62	69.50	71.43	70.75	86.00	83.72	98.24	73.00	73.79	76.64
IMP (%)		1.47	1.94	1.10	3.81	5.25	2.05	3.97	5.04	6.98	4.49	5.84	2.24	8.81	4.55	4.62
Reported Acc of GPPT (Label Ratio 50%)		77.16	–	–	65.81	–	–	92.13	–	–	86.80	–	–	72.23	–	–
appr. Label Ratio of our 100-shot setting		~ 25%			~ 18%			~ 1.7%			~ 7.3%			~ 1.5%		

• 训练参数和超参

Table 5: Tunable parameters comparison. RED (%): average reduction of the prompt method to others.

Methods	Cora	CiteSeer	Reddit	Amazon	Pubmed	RED (%)
GAT	~ 155K	~ 382K	~ 75K	~ 88K	~ 61K	95.4↓
GCN	~ 154K	~ 381K	~ 75K	~ 88K	~ 61K	95.4↓
GT	~ 615K	~ 1.52M	~ 286K	~ 349K	~ 241K	98.8↓
prompt	~ 7K	~ 19K	~ 3K	~ 4K	~ 3K	–

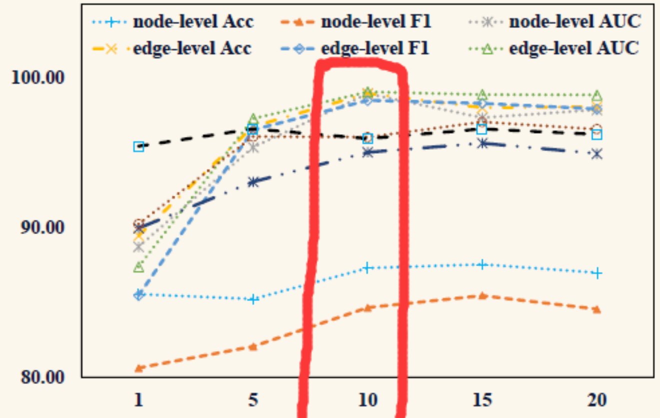


Figure 6: Impact of token numbers

Table 6: Error bound discussed by section 3.5.2 RED (%): average reduction of each method to the original error.

Prompt Solutions	Token Number	Drop Nodes	Drop Edges	Mask Features	RED (%)
Original Error (without prompt)	0	0.9917	2.6330	6.8209	-
Naive Prompt (Equation 5)	1	0.8710	0.5241	2.0835	66.70↓
Our Prompt Graph (with token, structure, and inserting patterns)	3	0.0875	0.2337	0.6542	90.66↓
	5	0.0685	0.1513	0.4372	93.71↓
	10	0.0859	0.1144	0.2600	95.59↓

$$\varphi^*(\underbrace{A, X + p^*}) = \varphi^*(g(A, X)) + O_{p\varphi}$$

大于

$$\varphi^*(\psi(\mathcal{G}, \mathcal{G}_p^*)) = \varphi^*(g(A, X)) + O^*_{p\varphi}$$

推测其提示图的形式更多样

- **消融**

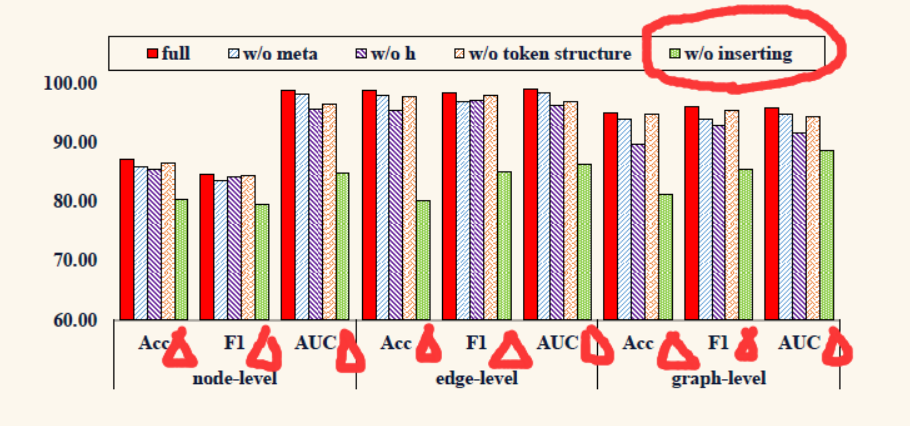


Figure 5: Effectiveness of main components

- **总结**
  - 其实本质上就是一个图增强方法，增强内容为可学习
  - 思考问题的方式，动机-> 存在的问题和挑战-> 如何解决
  - 提示图的插入（结合）及其重要
  - 似乎提示图的结构不是很重要

重慶大學

CHONGQING UNIVERSITY

谢谢大家