



中國人民大學  
RENMIN UNIVERSITY OF CHINA



高瓴人工智能学院  
Gaoling School of Artificial Intelligence

# GraphGPT: Graph Instruction Tuning for Large Language Models

Data Intelligence Lab, University of Hong Kong. Baidu, Inc.

Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Lixin Su, Suqi Cheng, Dawei Yin and Chao Huang\*

# An intriguing example of prompt learning in drug-target affinity prediction

**GPT-4**

**Prompt:**  
Please estimate the binding affinity between the drug Afatinib and target EGFR.

**GPT-4:**  
As an AI language model, I cannot directly perform computational tasks like estimating the binding affinity between a drug and its target. Estimating the binding affinity between afatinib and the Epidermal Growth Factor Receptor (EGFR) requires specialized software and computational methods, such as molecular docking or molecular dynamics simulations. However, I can provide some background information on afatinib and EGFR. Afatinib is a tyrosine kinase inhibitor that specifically targets EGFR. It is used to treat non-small cell lung cancer (NSCLC) patients with EGFR mutations. Afatinib works by irreversibly binding to the ATP-binding site of EGFR, preventing the activation of downstream signaling pathways and ultimately inhibiting cancer cell growth and proliferation. To estimate the binding affinity between afatinib and EGFR, researchers typically use computational methods like molecular docking or molecular dynamics simulations in combination with experimental techniques like surface plasmon resonance, isothermal titration calorimetry, or enzyme inhibition assays. If you are interested in learning more about molecular docking or other methods to study protein-drug interactions, I'd be happy to provide more information.

**Prompt:**  
Please estimate the binding affinity between the drug Afatinib and target EGFR.

**GPT-4:**  
As an AI language model, I am unable to directly perform computational tasks like estimating the binding affinity between afatinib and EGFR. However, I can guide you through the process and recommend some steps and tools to help you perform the estimation yourself. To estimate the binding affinity between afatinib and EGFR, you can follow these general steps:

1. Obtain the 3D structures of both afatinib and EGFR: You can find the 3D structure of EGFR in the Protein Data Bank (PDB) database.
2. (<https://www.rcsb.org/>). For afatinib, you can obtain its 3D structure from databases like PubChem (<https://pubchem.ncbi.nlm.nih.gov/>) or DrugBank (<https://www.drugbank.ca/>).
3. Prepare the protein and ligand structures: Before performing molecular docking, you need to prepare the structures. This includes removing water molecules, adding hydrogen atoms, and assigning proper charges to both the protein and ligand. You can use molecular modeling software like UCSF Chimera (<https://www.cgl.ucsf.edu/chimera/>) or PyMOL (<https://pymol.org/>) for these tasks.
4. Perform molecular docking: Molecular docking is a computational method that predicts the preferred orientation of a ligand (in this case, afatinib) when it binds to a protein (EGFR) to form a stable complex. There are various molecular docking software packages available, such as AutoDock Vina (<http://vina.scripps.edu/>), GOLD (<https://www.ccdc.cam.ac.uk/solutions/csd-discovery/components/gold/>), or Glide (<https://www.schrodinger.com/glide>). These tools will provide you with a predicted binding affinity value.
5. Validate your results: To increase the reliability of your binding affinity estimation, it's essential to validate your results using different approaches, such as molecular dynamics simulations, experimental techniques like surface plasmon resonance, or comparing your results to known binding affinities reported in the literature.

Please note that these steps require a good understanding of molecular modeling, docking algorithms, and protein-ligand interactions. If you are unfamiliar with these topics, you may need to consult a specialist or learn more about these subjects before proceeding with the analysis.

**GPT-4**

**System message:**  
You are a drug assistant and should be able to help with drug discovery tasks. Given the SMILES sequence of a drug and the FASTA sequence of a protein target, you need to calculate the binding affinity score. You can think step-by-step to get the answer and call any function you want. The output should be a float number, which is the estimated affinity score without other words.

**Prompt:**  
Example 1:  
CC[C@H](C)[C@H](NC(=O)OC)C(=O)N1CCC[C@H]1c1ncc(-c2ccc3cc(-c4ccc5[nH]c([C@@H]6CCCCN6C(=O)C@@H](NC(=O)OC)[C@@H](C)OC)nc5c4)ccc3c2)[nH]1,  
SGSWSLRDVWDWICTVLTDFTKWLQSKLLPRIPGVFPLSCQRGYKGVWRGDGL..TMSEEASEDVVCC  
11.52

Example 2:  
CCCc1cccc(C(=O)CCC(=O)O)cc1,  
MELPNIMHPVAKLSTALAAMLSGCMPGE...PDSRAITHTARMADKLR  
2.68

Example 3:  
OC1ccc2cc(CO[C@H]3[C@H](O)[C@H](CO)[C@@H](S[C@H]4O[C@H](CO)[C@H](O)[C@H](O)C5=CC=C(C=C5)O)cc6oc5=O)[C@H]4O)[C@H](O)c1=O;c2c1,  
MMMLSLNNLQNIIYNPVIPFGTIPDQLDPGTLIVRGHVP..EINGDIHLLEVRSW  
4.08

Test input:  
{SMILES}  
{FASTA}

**GPT-4:**  
{Affinity score}

Figure 2.15: Few-shot examples used in few-shot DTA evaluations.

Prompt learning, instruction tuning, in-context learning help the LLMs better adapt to various tasks.

Figure 2.12: An example of zero-shot drug-target affinity (DTA) prediction: The GPT-4 model does not directly predict DTA; rather, it offers valuable information, such as calculating affinity using relevant docking software.

The Impact of Large Language Models on Scientific Discovery: a Preliminary Study using GPT-4 (arXiv 2023)

1

# Limitation of LLMs in understanding graph structural contexts

**Input:** (a) ChatGPT with Node Content Only Token Length: 615

**Abstract:** The use of lower precision has emerged as a popular technique ...  
**Title:** TiM-DNN: Ternary in-Memory accelerator for Deep Neural Networks  
**Question:** Which arXiv CS sub-category does this paper belong to? ...

**Output:**  
cs.AR, cs.AI, cs.SY, cs.ET, cs.NE. The paper presents a hardware ...  
Therefore, the most likely category for this paper is cs.AR ...

**Input:** (b) ChatGPT with Node Content and Text-based Graph Structure Token Length: 4649

**Abstract:** The use of lower precision has emerged as a popular technique ...  
**Title:** TiM-DNN: Ternary in-Memory accelerator for Deep Neural Networks  
~~With it as central node (paper 0), a citation graph can be constructed.~~

**The list of neighbors:** Paper 1: ... , ... , Paper 102: ...  
**The citation relations:** Paper 0 cites Paper 1, ... , ... cites Paper 102.

**Question:** Which arXiv CS sub-category does this paper belong to? ...

**Output:**  
Based on the title and Abstract, the paper is likely to belong:  
1. cs.AR (Hardware Architecture) ...

**Ground Truth:** cs.LG, Machine Learning

## Text-based graph structure prompts:

Using text-based graph structure prompts results in an increase in the size of input tokens, posing challenges in practical applications.

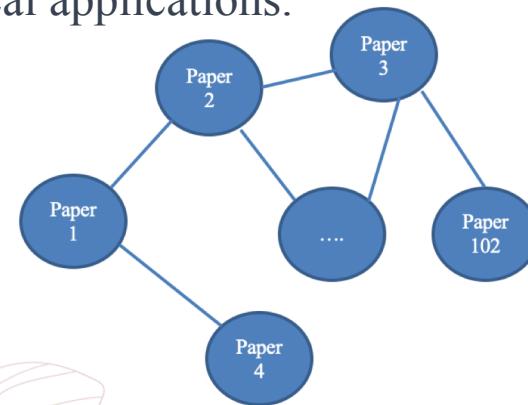
# Limitation of LLMs in understanding graph structural contexts

**Input:** (a) ChatGPT with Node Content Only Token Length: 615  
**Abstract:** The use of lower precision has emerged as a popular technique ...  
**Title:** TiM-DNN: Ternary in-Memory accelerator for Deep Neural Networks  
**Question:** Which arXiv CS sub-category does this paper belong to? ...  
**Output:**  
cs.AR, cs.AI, cs.SY, cs.ET, cs.NE. The paper presents a hardware ...  
Therefore, the most likely category for this paper is cs.AR ...

**Input:** (b) ChatGPT with Node Content and Text-based Graph Structure Token Length: 4649  
**Abstract:** The use of lower precision has emerged as a popular technique ...  
**Title:** TiM-DNN: Ternary in-Memory accelerator for Deep Neural Networks  
With it as central node (paper 0), a citation graph can be constructed.  
**The list of neighbors:** Paper 1: ... , ... , Paper 102: ...  
**The citation relations:** Paper 0 cites Paper 1, ... , ... cites Paper 102.  
**Question:** Which arXiv CS sub-category does this paper belong to? ...  
**Output:**  
Based on the title and Abstract, the paper is likely to belong:  
1. cs.AR (Hardware Architecture) ...  
**Ground Truth:** cs.LG, Machine Learning

## Text-based graph structure prompts:

Using text-based graph structure prompts results in an increase in the size of input tokens, posing challenges in practical applications.



GNNs



## 1. Graph Neural Networks(GNNs)

### Pros:

- Capture **structural information**

### Cons:

- The utilization of node and edge attitude is insufficient:  
Citation graph: title/abstract  
DDI graph: drug information/side effects description /medication scenarios
- Heavy reliance on supervised learning and poor robust performance.



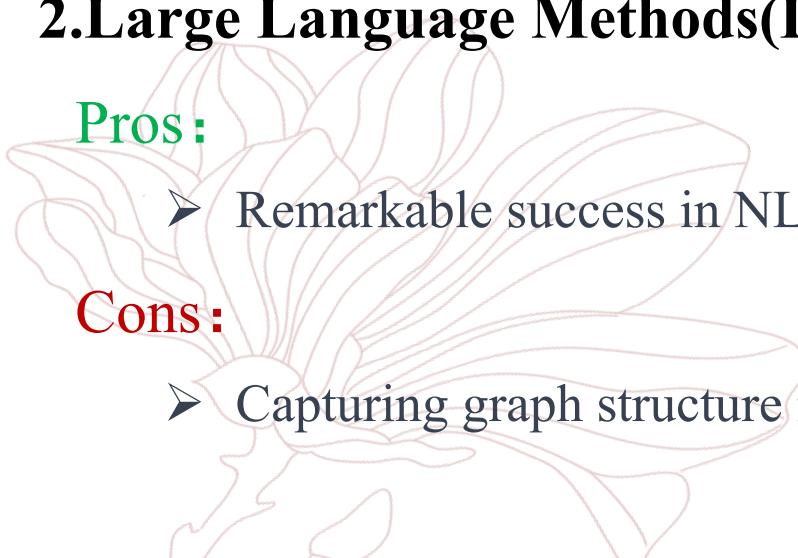
## 2. Large Language Methods(LLMs)

### Pros:

- Remarkable success in NLP tasks focusing on **text information**.

### Cons:

- Capturing graph structure is challenging.



# LLMs help graph-related tasks

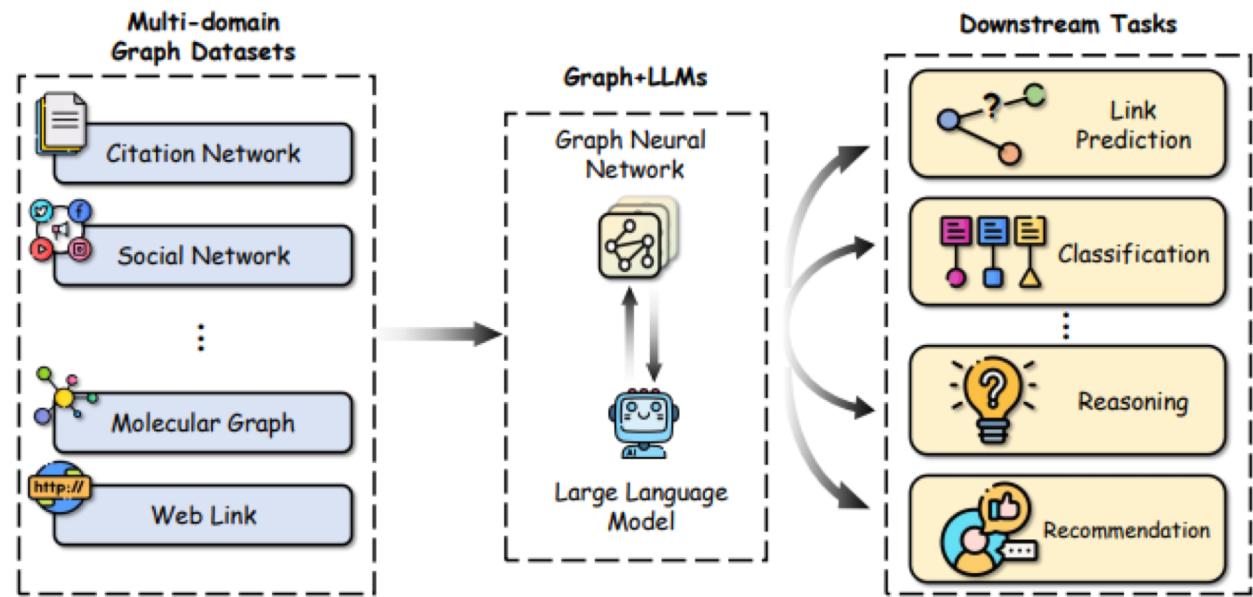


Figure 1: Across a myriad of graph domains, the integration of graphs and LLMs demonstrates success in various downstream tasks.

A Survey of Graph Meets Large Language Model: Progress and Future Directions(arXiv 2023)

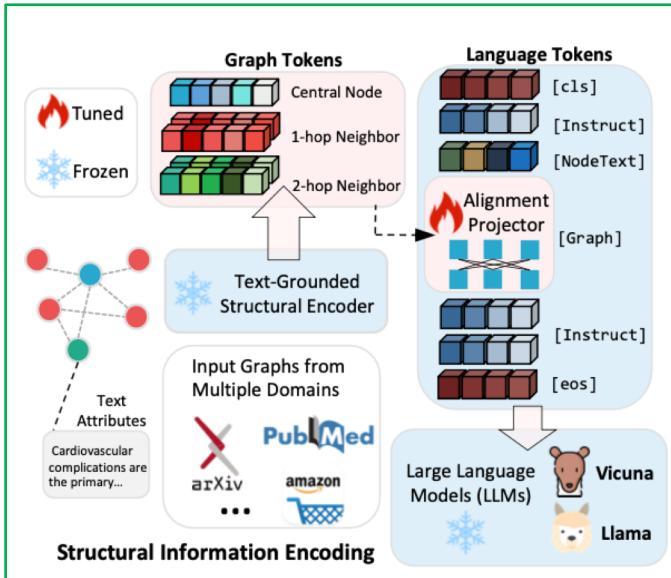
# Graph Instruction Tuning for Large Language Models

Input:	(a) ChatGPT with Node Content Only	Token Length: 615
<b>Abstract:</b>	The use of lower precision has emerged as a popular technique ...	
<b>Title:</b>	TiM-DNN: Ternary in-Memory accelerator for Deep Neural Networks	
<b>Question:</b>	Which arXiv CS sub-category does this paper belong to? ...	
Output:	cs.AR, cs.AI, cs.SY, cs.ET, cs.NE. The paper presents a hardware ...	
	Therefore, the most likely category for this paper is cs.AR ...	
Input:	(b) ChatGPT with Node Content and Text-based Graph Structure	Token Length: 4649
<b>Abstract:</b>	The use of lower precision has emerged as a popular technique ...	
<b>Title:</b>	TiM-DNN: Ternary in-Memory accelerator for Deep Neural Networks	
	With it as central node (paper 0), a citation graph can be constructed.	
<b>The list of neighbors:</b>	Paper 1: ..., ..., Paper 102: ...	
<b>The citation relations:</b>	Paper 0 cites Paper 1, ..., ... cites Paper 102.	
<b>Question:</b>	Which arXiv CS sub-category does this paper belong to? ...	
Output:	Based on the title and Abstract, the paper is likely to belong: 1. cs.AR (Hardware Architecture) ...	
Input:	(c) GraphGPT	Token Length: 750
<b>Abstract:</b>	Given a citation graph: <graph> where the 0th node is the target paper,	
	with the following information:	
<b>Title:</b>	TiM-DNN: Ternary in-Memory accelerator for Deep Neural Networks	
<b>Question:</b>	Which arXiv CS sub-category does this paper belong to? ...	
Output:	Based on the title and abstract, we can identify the following CS sub-categories that are most likely to be relevant:1. cs.LG ...	
Ground Truth:	cs.LG, Machine Learning	

**Figure 1: Limitation of LLMs in understanding graph structural contexts with heavy reliance on textual data.**

GraphGPT aligns LLMs with **graph structural knowledge** with a graph **instruction tuning framework**.

# Additional Task explanation: clarify difference from GNN's node classification task



## The overall architecture of GraphGPT

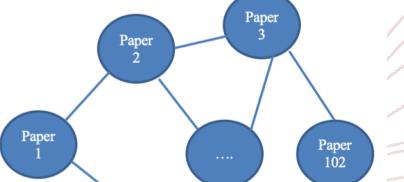


Table 5: Comparison of prediction results between our GraphGPT and ChatGPT.

**Title:** TiM-DNN: Ternary In-Memory Accelerator for Deep Neural Networks. (with 102 first-order and second-order neighbors)

**Ground-Truth Category:** cs.LG, Machine Learning

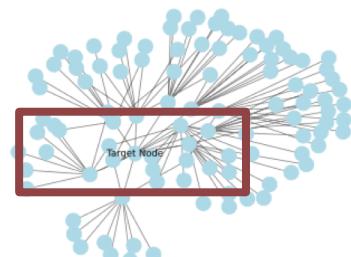
**ChatGPT (Node Content Only):** cs.AR, cs.AI, cs.SY, cs.ET, cs.NE. The paper presents a hardware accelerator for deep neural networks (DNNs) that is specifically designed to execute ternary DNNs, which represent weights and activations by signed ternary values. Therefore, the most likely category for this paper is cs.AR (Hardware Architecture), which focuses on the design and analysis of hardware and systems for AI and machine learning applications. The paper also involves the optimization of compute and storage requirements of DNNs, which falls under the category of cs.AI (Artificial Intelligence). Additionally, the paper proposes a programmable, in-memory accelerator, which is a type of embedded system, and therefore, it could be categorized under cs.SY (Systems and Control). Furthermore, the paper discusses the use of ternary values for DNNs, which is an emerging technique to optimize the compute and storage requirements of DNNs, and therefore, it could be categorized under cs.ET (Emerging Technologies). Lastly, the paper proposes a new bit-cell called Ternary Processing Cells (TPCs), which could be categorized under cs.NE (Neural and Evolutionary Computing).

**ChatGPT (Node Content and Text-based Graph Structure):** Based on the title "TiM-DNN: Ternary In-Memory Accelerator for Deep Neural Networks", the paper is likely to belong to the following arXiv CS sub-categories: 1. cs.AR (Hardware Architecture): The paper proposes a hardware accelerator, TiM-DNN, specifically designed for executing ternary deep neural networks. This falls under the domain of hardware architecture research. 2. cs.NE (Neural and Evolutionary Computing): The paper focuses on accelerating deep neural networks, which is a topic within the field of neural and evolutionary computing. 3. cs.CV (Computer Vision and Pattern Recognition): Deep neural networks are widely used in computer vision tasks, and the TiM-DNN accelerator could potentially be applied to accelerate computer vision algorithms. 4. cs.SY (Systems and Control): The TiM-DNN accelerator is a system-level solution for accelerating deep neural networks, so it could be categorized under systems and control research. 5. cs.DC (Distributed, Parallel, and Cluster Computing): While the paper does not explicitly mention distributed or parallel computing, the TiM-DNN accelerator is designed to optimize the compute and storage requirements of deep neural networks, which are often executed on distributed or parallel computing systems.

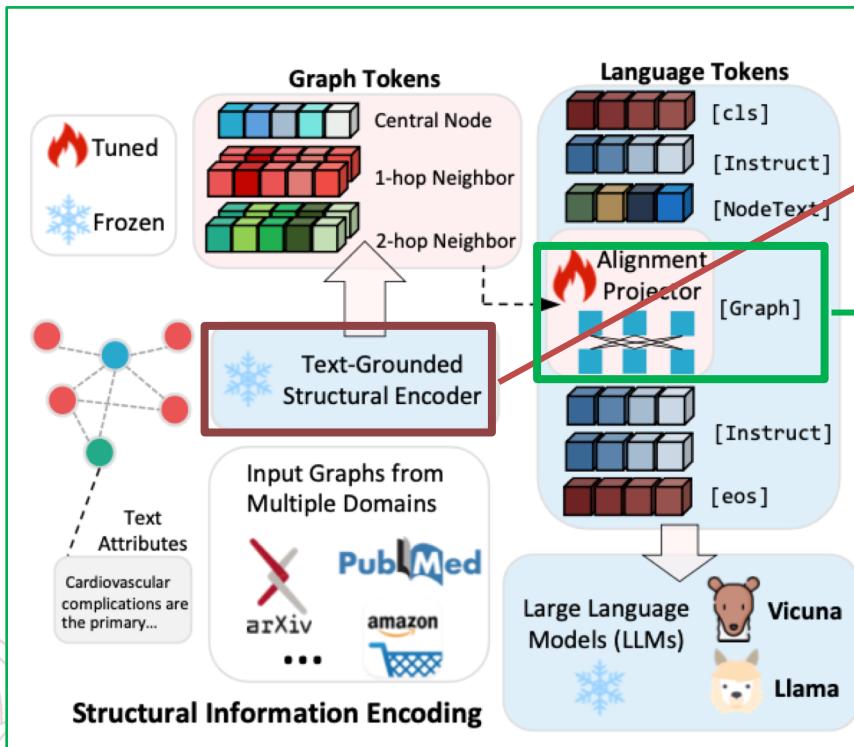
**GraphGPT:** Based on the title and abstract of the paper, we can identify the following arXiv CS sub-categories that are most likely to be relevant:

1. cs.LG (Machine Learning): This sub-category is relevant because the paper discusses the use of ternary DNNs for language modeling and image classification, which are both tasks that involve learning.
2. cs.AI (Artificial Intelligence): This sub-category is relevant because the paper proposes a new accelerator for DNNs, which is designed to optimize the compute and storage requirements of complex DNNs.
3. cs.NE (Numerical Analysis): This sub-category is relevant because the paper discusses the use of ternary DNNs, which involve signed ternary values, and the use of TiM tiles, which are specialized memory arrays that perform massively parallel signed vector-matrix multiplications on ternary values.
4. cs.SY (Systems and Control Theory): This sub-category is relevant because the paper discusses the implementation of TiM-DNN in 32nm technology using an architectural simulator calibrated with SPICE simulations and RTL synthesis.
5. cs.AR (Hardware Architecture): This sub-category is relevant because the paper proposes a new accelerator for DNNs, which is designed to be programmable and in-memory.

is designed to be programmable and in-memory.



# The overall architecture of GraphGPT



Step1: Graph ←→ Text → Graph token

- A text-graph grounding component

Step2: Graph token ←→ Language token

- A dual-stage instruction tuning
  - Self-supervised instruction tuning: **graph matching**
  - Task-specific instruction tuning: **node classification** and **link prediction**

The overall architecture of GraphGPT

# ➤ 1.Text-Structure alignment: a text-graph grounding component

Notions:

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

$\mathbf{C} = c_i \in \mathbb{R}^{l_i \times d}, 1 \leq i \leq N$  : the corresponding text contents

$l_i$  : the length of the text content for the  $i$ -th nodes.

$\hat{\mathbf{H}} \in \mathbb{R}^{N \times d}$  : encoded graph representations (pre-trained graph transformer)

$\hat{\mathbf{T}} \in \mathbb{R}^{N \times d}$  : encoded text representations (pre-trained vanilla transformer)

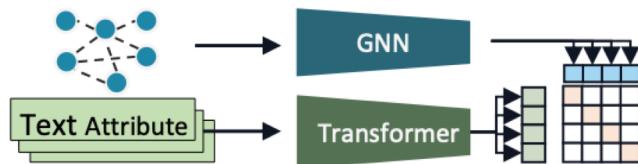


Figure 3: Workflow of text-structure alignment.

Alignment:

$$\mathbf{H} = f_{\mathbf{G}}(\mathbf{X}), \mathbf{T} = f_{\mathbf{T}}(\mathbf{C}), \hat{\mathbf{H}} = \text{norm}(\mathbf{H}), \hat{\mathbf{T}} = \text{norm}(\mathbf{T})$$

$$\Gamma_i = (g_i^{(1)}(\hat{\mathbf{H}}) g_i^{(2)}(\hat{\mathbf{T}})^T) \cdot \exp(\tau)$$

$$\mathcal{L} = \sum_i \frac{1}{2} \lambda_i (\text{CE}(\Gamma_i, \mathbf{y}) + \text{CE}(\Gamma_i^T, \mathbf{y}))$$

Where  $\mathbf{y} = (0, 1, \dots, n-1)^T$ ,  $g_i^{(1)}$  and  $g_i^{(2)}$  are transformation functions.

## ➤ 2.Text-Structure alignment

### Alignment:

$$\mathbf{H} = f_{\mathbf{G}}(\mathbf{X}), \mathbf{T} = f_{\mathbf{T}}(\mathbf{C}), \hat{\mathbf{H}} = \text{norm}(\mathbf{H}), \hat{\mathbf{T}} = \text{norm}(\mathbf{T})$$

$$\Gamma_i = (g_i^{(1)}(\hat{\mathbf{H}})g_i^{(2)}(\hat{\mathbf{T}})^{\top}) \cdot \exp(\tau)$$

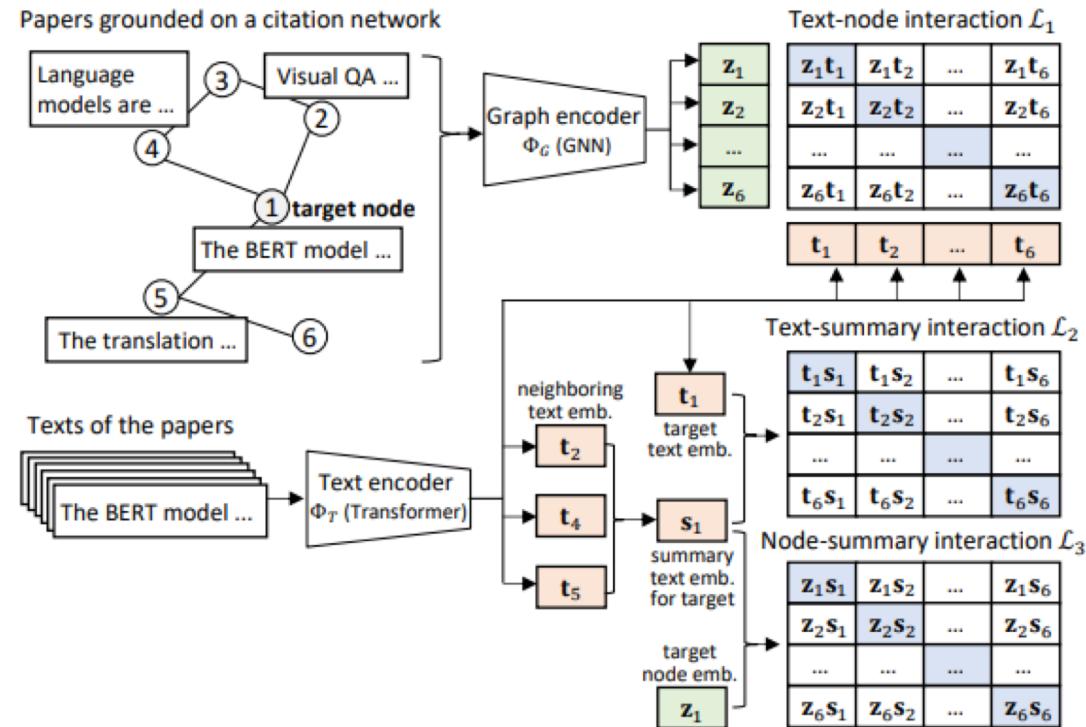
$$\mathcal{L} = \sum_i \frac{1}{2} \lambda_i (\text{CE}(\Gamma_i, \mathbf{y}) + \text{CE}(\Gamma_i^{\top}, \mathbf{y}))$$

Where  $\mathbf{y} = (0, 1, \dots, n-1)^{\top}$ ,  $g_i^{(1)}$  and  $g_i^{(2)}$  are transformation functions.

$$g_1^{(1)}(\hat{\mathbf{H}}) = \{\hat{\mathbf{H}}_i, 1 \leq i \leq N\}, g_1^{(2)}(\hat{\mathbf{T}}) = \{\hat{\mathbf{T}}_i, 1 \leq i \leq N\}$$

$$g_2^{(1)}(\hat{\mathbf{H}}) = \{\hat{\mathbf{H}}_i, 1 \leq i \leq N\}, g_2^{(2)}(\hat{\mathbf{T}}) = \left\{ \frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} \hat{\mathbf{T}}_j, 1 \leq i \leq N \right\}$$

$$g_3^{(1)}(\hat{\mathbf{H}}) = \{\hat{\mathbf{T}}_i, 1 \leq i \leq N\}, g_3^{(2)}(\hat{\mathbf{T}}) = \left\{ \frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} \hat{\mathbf{T}}_j, 1 \leq i \leq N \right\}$$



(a) Graph-grounded contrastive pre-training

## ➤ 2.Text-Structure alignment

### Alignment:

$$\mathbf{H} = f_{\mathbf{G}}(\mathbf{X}), \mathbf{T} = f_{\mathbf{T}}(\mathbf{C}), \hat{\mathbf{H}} = \text{norm}(\mathbf{H}), \hat{\mathbf{T}} = \text{norm}(\mathbf{T})$$

$$\Gamma_i = (g_i^{(1)}(\hat{\mathbf{H}})g_i^{(2)}(\hat{\mathbf{T}})^{\top}) \cdot \exp(\tau)$$

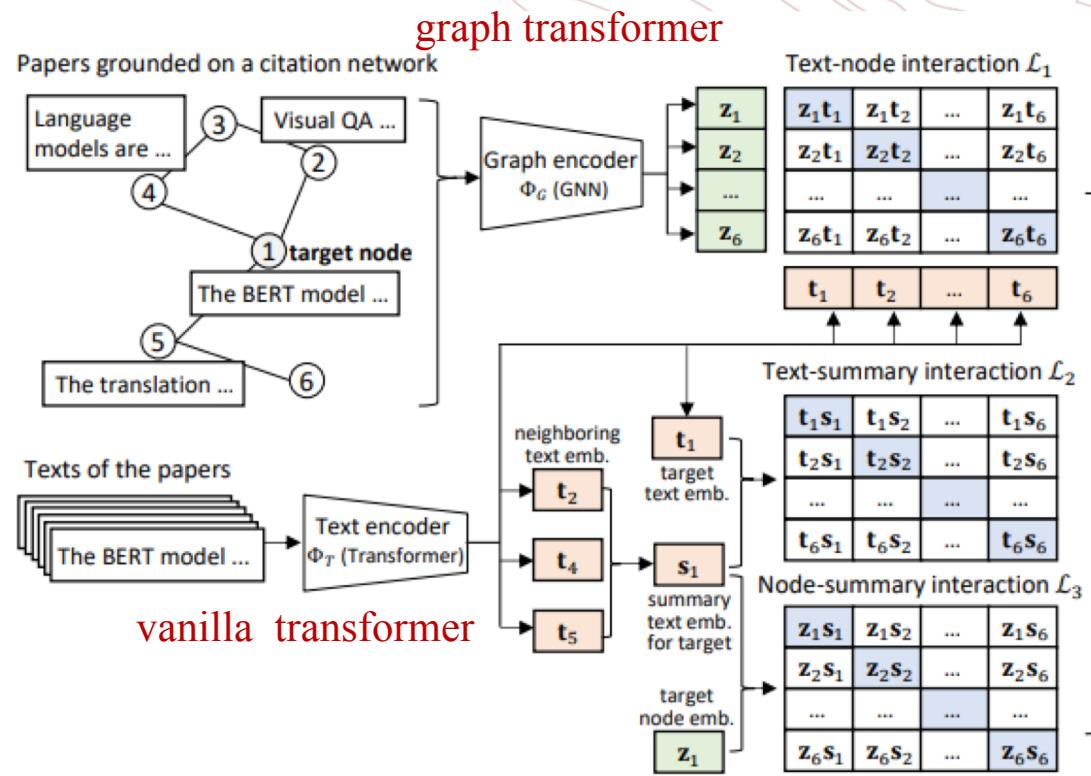
$$\mathcal{L} = \sum_i \frac{1}{2} \lambda_i (\text{CE}(\Gamma_i, \mathbf{y}) + \text{CE}(\Gamma_i^{\top}, \mathbf{y}))$$

Where  $\mathbf{y} = (0, 1, \dots, n-1)^{\top}$ ,  $g_i^{(1)}$  and  $g_i^{(2)}$  are transformation functions.

$$g_1^{(1)}(\hat{\mathbf{H}}) = \{\hat{\mathbf{H}}_i, 1 \leq i \leq N\}, g_1^{(2)}(\hat{\mathbf{T}}) = \{\hat{\mathbf{T}}_i, 1 \leq i \leq N\}$$

$$g_2^{(1)}(\hat{\mathbf{H}}) = \{\hat{\mathbf{H}}_i, 1 \leq i \leq N\}, g_2^{(2)}(\hat{\mathbf{T}}) = \left\{ \frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} \hat{\mathbf{T}}_j, 1 \leq i \leq N \right\}$$

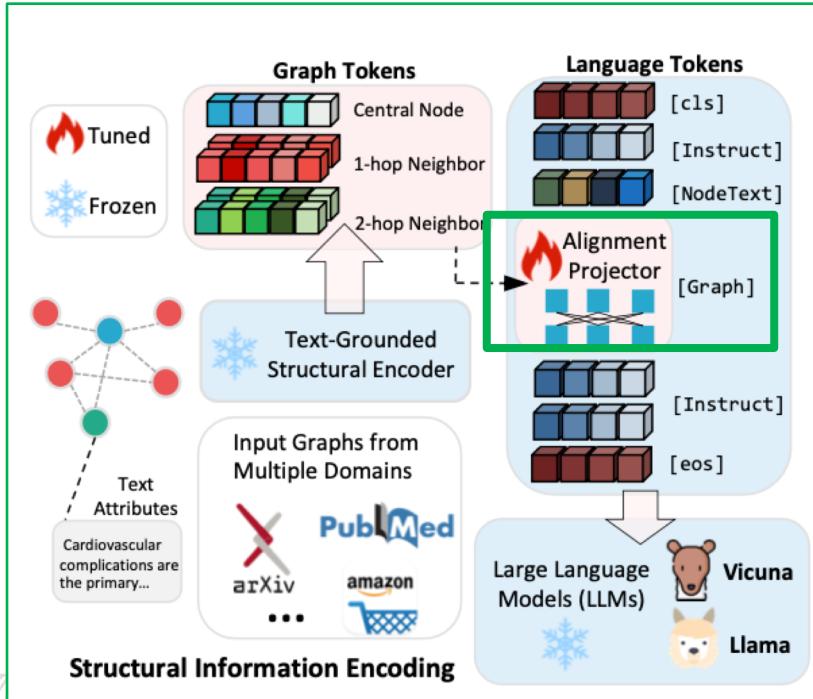
$$g_3^{(1)}(\hat{\mathbf{H}}) = \{\hat{\mathbf{T}}_i, 1 \leq i \leq N\}, g_3^{(2)}(\hat{\mathbf{T}}) = \left\{ \frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} \hat{\mathbf{T}}_j, 1 \leq i \leq N \right\}$$



(a) Graph-grounded contrastive pre-training

## ➤ 2.A dual-stage instruction tuning

Graph token  $\longleftrightarrow$  Language token



➤ A lightweight graph-text alignment projector.

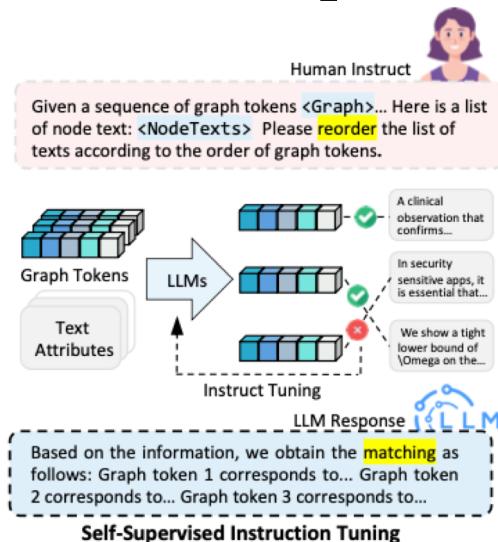
- This projector establishes the correspondence between graph node tokens and natural language tokens, thereby obtaining a token sequence for a large language model's input.

Original nature language token sequence: <graph>

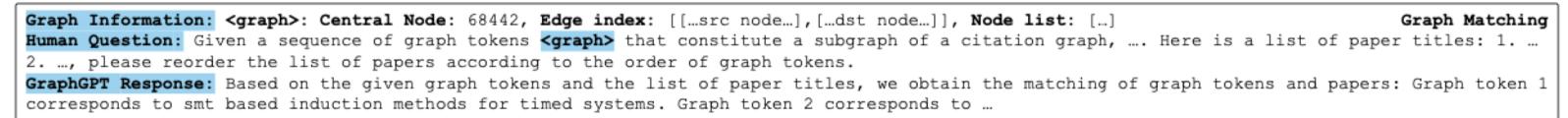
↑  
replace

Modified sequence: {<graph\_begin>, <graph\_token><sub>1</sub>, ..., <graph\_token><sub>n</sub>, <graph\_end>}

- 2.A dual-stage instruction tuning
- Graph token  $\longleftrightarrow$  Language token
- Self-supervised instruction tuning: graph matching

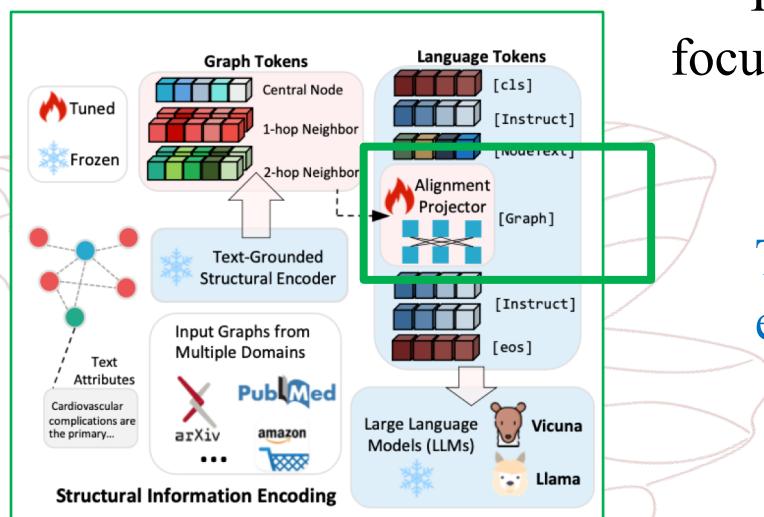


## ➤ Instruction Design



## ➤ Tuning Method

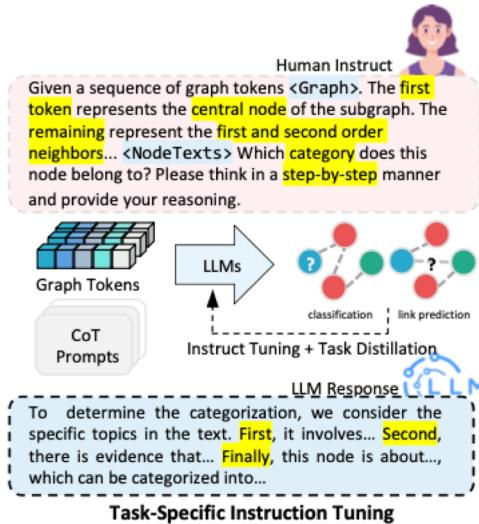
Keep the parameters of both the LLM and the graph encoder fixed, focusing solely on optimizing the parameters of the projector.



The objective of the LLM in the graph matching task is to align each graph token with its corresponding node text information.

## ➤ 2.A dual-stage instruction tuning Graph token $\longleftrightarrow$ Language token

### ➤ Task-specific instruction tuning : node classification and link prediction



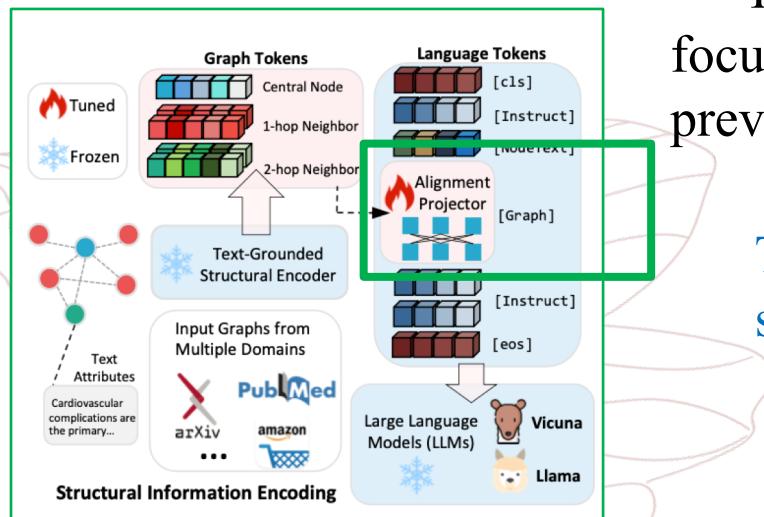
### ➤ Instruction Design

Graph Information: <graph>: Central Node: 2, Edge index: [...src node...], [...dst node...], Node list: [...]	Node Classification
<b>Human Question:</b> Given a citation graph: <graph> where the 0th node is the target paper, with the following information: Abstract: ... Title: ...	
Question: Which arXiv CS sub-category does this paper belong to? ...	
<b>GraphGPT Response:</b> cs.IT, cs.LG, cs.SP, cs.CV, cs.NA. The paper discusses the Restricted Isometry ... So, it is likely to belong to cs.IT...	
Graph Information: <graph>: Central Node 1: 8471, Edge index 1: [...src node...], [...dst node...], Node list 1: [...]	Link Prediction
<graph>: Central Node 2: 19368, Edge index 2: [...src node...], [...dst node...], Node list 2: [...]	
<b>Human Question:</b> Given a sequence of graph tokens: <graph> that constitute a subgraph of a citation graph, ... Abstract: ... Title: ... and the other sequence of graph tokens: <graph>, ... Abstract: ..., are these two central nodes connected? Give me an answer of "yes" or "no".	
<b>GraphGPT Response:</b> Yes, they are connected. Based on the first paper, .... And the second paper proposes ....	

### ➤ Tuning Method

Keep the parameters of both the LLM and the graph encoder fixed, focusing solely on optimizing the parameters of the projector from the previous stage.

The objective is to guide LLMs generate response that are better suited for the particular graph leaning task.



## ➤ 3. Chain-of-Thought(COT) Distillation

- Employ a distillation approach to extract valuable knowledge from a closed-source, powerful language model like ChatGPT.
- Employ the GPT3.5 to perform step-step reasoning for instruction.

---

### Node Classification (Arxiv in the COT manner)

**Input:** Given a citation graph: <graph> where the 0th node is the target paper, with the following information: Abstract: {abstract of the central node}. Title: {title of the central node}. Question: Which arXiv CS sub-category does this paper belong to? Give 5 likely arXiv CS sub-categories as a comma-separated list ordered from most to least likely, in the form "cs.XX". Please think about the categorization in a step by step manner and avoid making false associations. Then provide your reasoning.

**Output:** Based on the information, {reasoning process and answers}.

---

### Arxiv in the standard manner

**Input:** Given a citation graph: <graph> where the 0th node is the target paper, and other nodes are its one-hop or multi-hop neighbors, with the following information: Abstract: {abstract of the central node}. Title: {title of the central node}. Question: Which arXiv CS sub-category does this paper belong to? Give the most likely arXiv CS sub-categories of this paper directly, in the form "cs.XX" with full name of the category.

**Output:** {the ground-truth answer}.



## ➤ Evaluation

- Task: node classification
- Dataset: OGB-arxiv, PubMed, and Cora

**Table 6: Statistical information of datasets.**

Dataset	# Nodes	# Edges	# Classes
ogb-arxiv	169,343	1,166,243	40
PubMed	19,717	44,338	3
Cora	25,120	182,280	70

- Baselines:
  - MLP
  - GNNs: GraphSAGE, GCN, GAT, RevGNN
  - Self-supervised approach: DGI
  - Knowledge distillation-enhanced GNNs: GKD,GLNN
  - Graph transformer: NodeFormer, DIFFormer
  - Open-sourced LLMs: Baichuan-7B,vicunna-7B-v1.1,-v1.5(text-attributed graph data)
- Implementation Details:
  - Base model: vicunna-7B-v1.1,-v1.5
  - Explore the performance of the model under different data mixtures by adopting various combinations of instruction data

# ➤ Evaluation

**Table 1: Performance comparison of various methods on node classification under both supervised and zero-shot settings.**

Dataset	Arxiv-Arxiv		Arxiv-PubMed		Arxiv-Cora		(Arxiv+PubMed)-Cora		(Arxiv+PubMed)-Arxiv	
Model	Accuracy	Macro-F1	acc	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1
MLP	0.5179	0.2536	0.3940	0.1885	0.0258	0.0037	0.0220	0.0006	0.2127	0.0145
GraphSAGE	0.5480	0.3290	0.3950	0.1939	0.0328	0.0132	0.0132	0.0029	0.1281	0.0129
GCN	0.5267	0.3202	0.3940	0.1884	0.0214	0.0088	0.0187	0.0032	0.0122	0.0008
GAT	0.5332	0.3118	0.3940	0.1884	0.0167	0.0110	0.0161	0.0057	0.1707	0.0285
RevGNN	0.5474	0.3240	0.4440	0.3046	0.0272	0.0101	0.0217	0.0016	0.1309	0.0126
DGI	0.5059	0.2787	0.3991	0.1905	0.0205	0.0011	0.0205	0.0011	0.5059	0.2787
GKD	0.5570	0.1595	0.3645	0.2561	0.0470	0.0093	0.0406	0.0037	0.2089	0.0179
GLNN	0.6088	0.3757	0.4298	0.3182	0.0267	0.0115	0.0182	0.0092	0.3373	0.1115
NodeFormer	0.5922	0.3328	0.2064	0.1678	0.0152	0.0065	0.0144	0.0053	0.2713	0.0855
DIFFormer	0.5986	0.3355	0.2959	0.2503	0.0161	0.0094	0.0100	0.0007	0.1637	0.0234
baichuan-7B	0.0946	0.0363	0.4642	0.3876	0.0405	0.0469	0.0405	0.0469	0.0946	0.0363
vicuna-7B-v1.1	0.2657	0.1375	0.5251	0.4831	0.1090	0.0970	0.1090	0.0970	0.2657	0.1375
vicuna-7B-v1.5	0.4962	0.1853	0.6351	0.5231	0.1489	0.1213	0.1489	0.1213	0.4962	0.1853
<b>GraphGPT-7B-v1.1-cot</b>	0.4913	0.1728	0.6103	0.5982	0.1145	0.1016	0.1250	0.0962	0.4853	0.2102
<b>GraphGPT-7B-v1.5-stage2</b>	<b>0.7511</b>	<b>0.5600</b>	0.6484	0.5634	0.0813	0.0713	0.0934	0.0978	0.6278	0.2538
<b>GraphGPT-7B-v1.5-std</b>	0.6258	0.2622	<b>0.7011</b>	<b>0.6491</b>	0.1256	0.0819	0.1501	0.0936	0.6390	0.2652
<b>GraphGPT-7B-v1.5-cot</b>	0.5759	0.2276	0.5213	0.4816	<b>0.1813</b>	<b>0.1272</b>	<b>0.1647</b>	<b>0.1326</b>	<b>0.6476</b>	<b>0.2854</b>
p-val	$2.26e^{-9}$	$1.56e^{-10}$	$2.22e^{-7}$	$1.55e^{-9}$	$1.04e^{-9}$	$9.96e^{-6}$	$7.62e^{-8}$	$1.97e^{-7}$	$1.5e^{-13}$	$4.63e^{-6}$

- Supervised task settings
- Zero-shot task settings
- “-stage2” indicates that only the second stage tuning is adopted.
- “-std” and “-cot” denotes the use of the standard and generated COT instruction datasets.

# ➤ Evaluation

**Table 1: Performance comparison of various methods on node classification under both supervised and zero-shot settings.**

Dataset	Arxiv-Arxiv		Arxiv-PubMed		Arxiv-Cora		(Arxiv+PubMed)-Cora		(Arxiv+PubMed)-Arxiv	
Model	Accuracy	Macro-F1	acc	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1
MLP	0.5179	0.2536	0.3940	0.1885	0.0258	0.0037	0.0220	0.0006	0.2127	0.0145
GraphSAGE	0.5480	0.3290	0.3950	0.1939	0.0328	0.0132	0.0132	0.0029	0.1281	0.0129
GCN	0.5267	0.3202	0.3940	0.1884	0.0214	0.0088	0.0187	0.0032	0.0122	0.0008
GAT	0.5332	0.3118	0.3940	0.1884	0.0167	0.0110	0.0161	0.0057	0.1707	0.0285
RevGNN	0.5474	0.3240	0.4440	0.3046	0.0272	0.0101	0.0217	0.0016	0.1309	0.0126
DGI	0.5059	0.2787	0.3991	0.1905	0.0205	0.0011	0.0205	0.0011	0.5059	0.2787
GKD	0.5570	0.1595	0.3645	0.2561	0.0470	0.0093	0.0406	0.0037	0.2089	0.0179
GLNN	0.6088	0.3757	0.4298	0.3182	0.0267	0.0115	0.0182	0.0092	0.3373	0.1115
NodeFormer	0.5922	0.3328	0.2064	0.1678	0.0152	0.0065	0.0144	0.0053	0.2713	0.0855
DiffFormer	0.5986	0.3355	0.2959	0.2503	0.0161	0.0094	0.0100	0.0007	0.1637	0.0234
baichuan-7B	0.0946	0.0363	0.4642	0.3876	0.0405	0.0469	0.0405	0.0469	0.0946	0.0363
vicuna-7B-v1.1	0.2657	0.1375	0.5251	0.4831	0.1090	0.0970	0.1090	0.0970	0.2657	0.1375
vicuna-7B-v1.5	0.4962	0.1853	0.6351	0.5231	0.1489	0.1213	0.1489	0.1213	0.4962	0.1853
<b>GraphGPT-7B-v1.1-cot</b>	0.4913	0.1728	0.6103	0.5982	0.1145	0.1016	0.1250	0.0962	0.4853	0.2102
<b>GraphGPT-7B-v1.5-stage2</b>	<b>0.7511</b>	<b>0.5600</b>	0.6484	0.5634	0.0813	0.0713	0.0934	0.0978	0.6278	0.2538
<b>GraphGPT-7B-v1.5-std</b>	0.6258	0.2622	<b>0.7011</b>	<b>0.6491</b>	0.1256	0.0819	0.1501	0.0936	0.6390	0.2652
<b>GraphGPT-7B-v1.5-cot</b>	0.5759	0.2276	0.5213	0.4816	<b>0.1813</b>	<b>0.1272</b>	<b>0.1647</b>	<b>0.1326</b>	<b>0.6476</b>	<b>0.2854</b>
p-val	$2.26e^{-9}$	$1.56e^{-10}$	$2.22e^{-7}$	$1.55e^{-9}$	$1.04e^{-9}$	$9.96e^{-6}$	$7.62e^{-8}$	$1.97e^{-7}$	$1.5e^{-13}$	$4.63e^{-6}$

**Table 6: Statistical information of datasets.**

Dataset	# Nodes	# Edges	# Classes
ogb-arxiv	169,343	1,166,243	40
PubMed	19,717	44,338	3
Cora	25,120	182,280	70

- GraphGPT outperforms various state-of-the art baselines in both supervised and zero-shot scenarios
- GNN-based models work well on supervised task. However, when transferred to new datasets, their performance declines. In contrast, GraphGPT achieves a remarkable increase in accuracy in the zero-shot graph learning scenario.
- Without first stage, the models tends to overfit on the specific dataset.
- Benefits with COT distillation: COT enhance the performance on complex graph tasks.

# ➤ Evaluation

**Table 1: Performance comparison of various methods on node classification under both supervised and zero-shot settings.**

Dataset	Arxiv-Arxiv		Arxiv-PubMed		Arxiv-Cora		(Arxiv+PubMed)-Cora		(Arxiv+PubMed)-Arxiv	
Model	Accuracy	Macro-F1	acc	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1
MLP	0.5179	0.2536	0.3940	0.1885	0.0258	0.0037	0.0220	0.0006	0.2127	0.0145
GraphSAGE	0.5480	0.3290	0.3950	0.1939	0.0328	0.0132	0.0132	0.0029	0.1281	0.0129
GCN	0.5267	0.3202	0.3940	0.1884	0.0214	0.0088	0.0187	0.0032	0.0122	0.0008
GAT	0.5332	0.3118	0.3940	0.1884	0.0167	0.0110	0.0161	0.0057	0.1707	0.0285
RevGNN	0.5474	0.3240	0.4440	0.3046	0.0272	0.0101	0.0217	0.0016	0.1309	0.0126
DGI	0.5059	0.2787	0.3991	0.1905	0.0205	0.0011	0.0205	0.0011	0.5059	0.2787
GKD	0.5570	0.1595	0.3645	0.2561	0.0470	0.0093	0.0406	0.0037	0.2089	0.0179
GLNN	0.6088	0.3757	0.4298	0.3182	0.0267	0.0115	0.0182	0.0092	0.3373	0.1115
NodeFormer	0.5922	0.3328	0.2064	0.1678	0.0152	0.0065	0.0144	0.0053	0.2713	0.0855
DiffFormer	0.5986	0.3355	0.2959	0.2503	0.0161	0.0094	0.0100	0.0007	0.1637	0.0234
baichuan-7B	0.0946	0.0363	0.4642	0.3876	0.0405	0.0469	0.0405	0.0469	0.0946	0.0363
vicuna-7B-v1.1	0.2657	0.1375	0.5251	0.4831	0.1090	0.0970	0.1090	0.0970	0.2657	0.1375
vicuna-7B-v1.5	0.4962	0.1853	0.6351	0.5231	0.1489	0.1213	0.1489	0.1213	0.4962	0.1853
<b>GraphGPT-7B-v1.1-cot</b>	0.4913	0.1728	0.6103	0.5982	0.1145	0.1016	0.1250	0.0962	0.4853	0.2102
<b>GraphGPT-7B-v1.5-stage2</b>	<b>0.7511</b>	<b>0.5600</b>	0.6484	0.5634	0.0813	0.0713	0.0934	0.0978	0.6278	0.2538
<b>GraphGPT-7B-v1.5-std</b>	0.6258	0.2622	<b>0.7011</b>	<b>0.6491</b>	0.1256	0.0819	0.1501	0.0936	0.6390	0.2652
<b>GraphGPT-7B-v1.5-cot</b>	0.5759	0.2276	0.5213	0.4816	<b>0.1813</b>	<b>0.1272</b>	<b>0.1647</b>	<b>0.1326</b>	<b>0.6476</b>	<b>0.2854</b>
p-val	$2.26e^{-9}$	$1.56e^{-10}$	$2.22e^{-7}$	$1.55e^{-9}$	$1.04e^{-9}$	$9.96e^{-6}$	$7.62e^{-8}$	$1.97e^{-7}$	$1.5e^{-13}$	$4.63e^{-6}$

**Table 6: Statistical information of datasets.**

Dataset	# Nodes	# Edges	# Classes
ogb-arxiv	169,343	1,166,243	40
PubMed	19,717	44,338	3
Cora	25,120	182,280	70

- More data boost model transfer ability: Arxiv → Cora, Arxiv+pubmed → Cora:
  - The performance of GNN-based models deteriorates.
- More data yet no forgetting : Arxiv → Arxiv, Arxiv+pubmed → Arxiv
  - GNN-based model approaches experience a significant decline in performance because the catastrophic forgetting.

## ➤ Evaluation

**Table 2: Performance comparison of various instruction mixtures in supervised learning on the Arxiv dataset and the zero-shot setting on the Cora dataset for node classification.**

Dataset	Supervision. on Arxiv		Zero Shot on Cora	
Model	Acc	Macro-F1	Acc	Macro-F1
MLP	0.5179	0.2536	0.0220	0.0006
GraphSAGE	0.5480	0.3290	0.0132	0.0029
GCN	0.5267	0.3202	0.0187	0.0032
GAT	0.5332	0.3118	0.0161	0.0057
RvGNN	0.5474	0.3240	0.0217	0.0016
DGI	0.5059	0.2787	0.0205	0.0011
GKD	0.5570	0.1595	0.0406	0.0037
GLNN	0.6088	0.3757	0.0182	0.0092
NodeFormer	0.5922	0.3328	0.0144	0.0053
DIFFormer	0.5986	0.3355	0.0100	0.0007
baichuan-7b	0.0946	0.0363	0.0405	0.0469
vicuna-7B-v1.1	0.2657	0.1375	0.1090	0.0970
vicuna-7B-v1.5	0.4962	0.1853	0.1489	0.1213
Arxiv-std + PubMed-std	0.6390	0.2652	0.1501	0.0936
Arxiv-cot + PubMed-cot	0.6476	0.2854	0.1647	0.1326
Arxiv-mix + PubMed-mix	0.6139	0.2772	0.1544	0.1048
Arxiv-std + PubMed-std + Link	0.5931	0.2238	<b>0.1847</b>	<b>0.1579</b>
Arxiv-mix + Pubmed-mix + Link	<b>0.6874</b>	<b>0.3761</b>	0.1836	0.1494

- Recent studies on instruction tuning suggest that mixing different instruction tuning data can further enhance the performance of LLMs.



# Summary

- The paper proposes a framework which aligns LLMs with graph structural knowledge with a graph instruction tuning paradigm.
- Incorporates **a text-graph grounding component** to establish a connection between textual information and graph structures.
- Proposes **a dual-stage instruction tuning** paradigm with a lightweight graph-text alignment projector.
  - Self-supervised instruction tuning: **graph matching**
  - Task-specific instruction tuning: **node classification** and **link prediction**
- The experimental results indicate that the GraphGPT work well in supervised and zero-shot node classification tasks.



# Survey: LLMs + GNNs

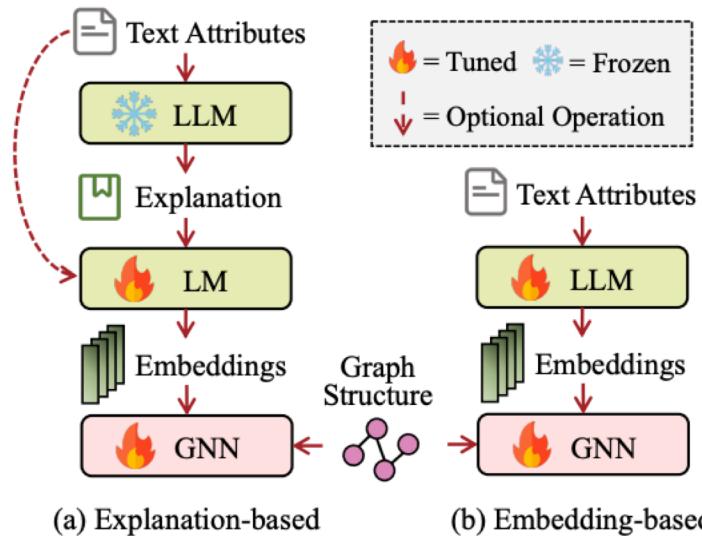


Figure 3: The illustration of LLM-as-enhancer approaches: **a)** **explanation-based enhancement**, which uses LLMs to generate explanations of text attributes to enhance text embeddings; **b)** **Embedding-based enhancement**, which directly obtains text embeddings by LLMs as initial node embeddings.

## LLM as Enhancer

LLMs are used to enhance the performance of GNNs

A Survey of Graph Meets Large Language Model: Progress and Future Directions(arXiv 2023)

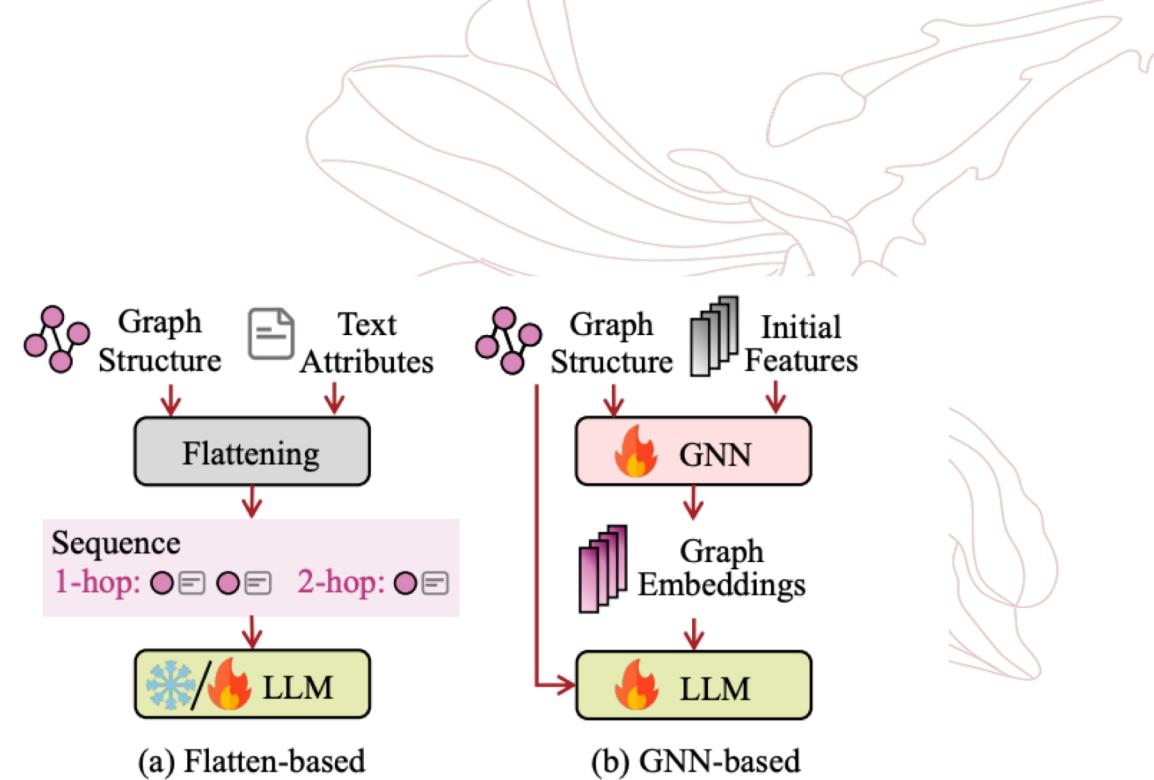


Figure 4: The illustration of LLM-as-predictor approaches: **a)** **Flatten-based prediction**, which incorporates graph structure with LLMs via different flattening strategies; **b)** **GNN-based prediction**, utilizing GNNs to capture structural information for LLMs.

## LLM as Predictor

LLMs utilize the input graph structure information to make predictions

# Survey: LLMs + GNNs

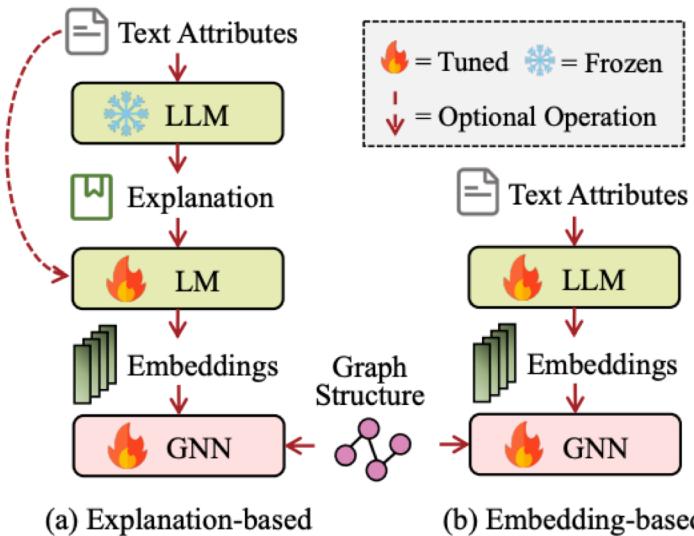


Figure 3: The illustration of LLM-as-enhancer approaches: **a)** **explanation-based enhancement**, which uses LLMs to generate explanations of text attributes to enhance text embeddings; **b)** **Embedding-based enhancement**, which directly obtains text embeddings by LLMs as initial node embeddings.

## LLM as Enhancer

LLMs are used to enhance the performance of GNNs

A Survey of Graph Meets Large Language Model: Progress and Future Directions(arXiv 2023)

# GraphGPT

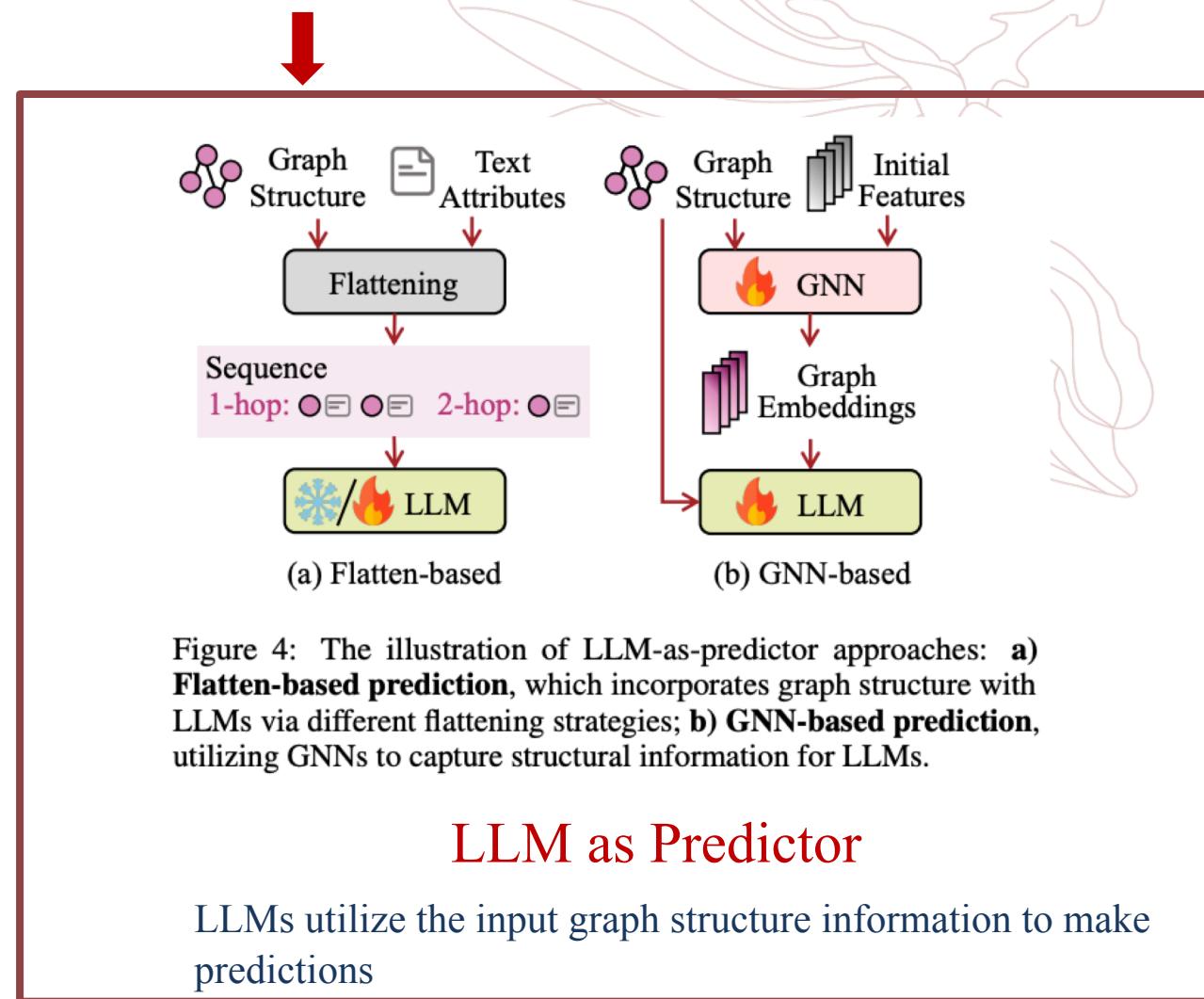


Figure 4: The illustration of LLM-as-predictor approaches: **a)** **Flatten-based prediction**, which incorporates graph structure with LLMs via different flattening strategies; **b)** **GNN-based prediction**, utilizing GNNs to capture structural information for LLMs.

## LLM as Predictor

LLMs utilize the input graph structure information to make predictions

# Survey: LLMs + GNNs

	<b>Model</b>	<b>GNN</b>	<b>LLM</b>	<b>Predictor</b>	<b>Fine-tuning</b>	<b>Prompting</b>	<b>Domain</b>	<b>Task</b>	<b>Code</b>
LLM as Enhancer	GIANT [Chien et al., 2021]	SAGE, RevGAT, etc.	BERT	GNN	✗	✗	Citation, Co-purchase	Node	Link
	GaLM [Xie et al., 2023]	RGCN, RGAT	BERT	GNN	✓	✗	E-Commerce, Recommendation	Node, Link	-
	TAPE [He et al., 2023]	RevGAT	ChatGPT	GNN	✗	✓	Citation	Node	Link
	Chen et al. [Chen et al., 2023a]	GCN, GAT	ChatGPT	GNN	✗	✓	Citation, Co-purchase	Node	-
	LLM4Mol [Qian et al., 2023]	-	ChatGPT	LM	✗	✗	Molecular	Graph	Link
	SimTeG [Duan et al., 2023]	SAGE, RevGAT, SEAL	allMiniLM-L6-v2, etc.	GNN	✓ $\heartsuit$	✗	Citation, Co-purchase	Node, Link	Link
	G-Prompt [Huang et al., 2023b]	SAGE, RevGAT	RoBERTa-Large	GNN	✓	✓	Citation, Social	Node	-
	TouchUp-G [Zhu et al., 2023]	SAGE, MB-GCN, etc.	BERT	GNN	✓	✗	Citation, Co-purchase, Recommendation	Node, Link	-
	OFA [Liu et al., 2023a]	R-GCN	Sentence-BERT	GNN	✗	✓	Citation, Web link, Knowledge, Molecular	Node, Link, Graph	Link
	LLMRe [Wei et al., 2023]	LightGCN	ChatGPT	GNN	✗	✓	Recommendation	Recommendation	Link
	WalkLM [Tan et al., 2023b]	-	DistilRoBERTa	MLP	✓	✗	Knowledge	Node, Link	Link
LLM as Predictor	NLGraph [Wang et al., 2023b]	-	Text-davinci-003	LLM	✗	✓	-	Reasoning	Link
	GPT4Graph [Guo et al., 2023]	-	Text-davinci-003	LLM	✗	✓	-	Reasoning, Node, Graph	Link
	GIMLET [Zhao et al., 2023a]	-	T5	LLM	✓/✗	✓	Molecular	Graph	Link
	Chen et al. [Chen et al., 2023a]	-	ChatGPT	LLM	✗	✓	Citation	Node	Link
	GIT-Mol [Liu et al., 2023c]	GIN	MolT5	LLM	✓ $\heartsuit$	✓	Molecular	Graph, Captioning	-
	InstructGLM [Ye et al., 2023]	-	FLAN-T5/LLaMA-v1	LLM	✓ $\heartsuit$	✓	Citation	Node	Link
	Liu et al. [Liu and Wu, 2023]	-	GPT-4, etc.	LLM	✗	✓	-	Reasoning	Link
	Huang et al. [Huang et al., 2023a]	-	ChatGPT	LLM	✗	✓	Citation, Co-purchase	Node	Link
	GraphText [Zhao et al., 2023c]	-	ChatGPT/GPT-4	LLM	✗	✓	Citation, Web link	Node	-
	Fatemi et al. [Fatemi et al., 2023]	-	PaLM/PaLM 2	LLM	✗	✓	-	Reasoning	-
GNN-LLM Alignment	GraphLLM [Chai et al., 2023]	Graph Transformer	LLaMA-v2	LLM	✓ $\heartsuit$	✓	-	Reasoning	Link
	Hu et al. [Hu et al., 2023]	-	ChatGPT/GPT-4	LLM	✗	✓	Citation, Knowledge, Social	Node, Link, Graph	-
	MolCA [Liu et al., 2023d]	GINE	Galactica/MoLT5	LLM	✓ $\heartsuit$	✓	Molecular	Graph, Retrieval, Captioning	Link
	GraphGPT [Tang et al., 2023]	Graph Transformer	Vicuna	LLM	✓ $\heartsuit$	✓	Citation	Node	Link
	RecLM [Sun et al., 2023]	TAC, OCN	Vicuna/ChatGPT	LLM	✗	✓	Molecular	Reaction Prediction	Link
	LLM4DyG [Zhang et al., 2023c]	-	Vicuna/LLaMA-v2/ChatGPT	LLM	✗	✓	-	Reasoning	-
	DGTL [Qin et al., 2023]	Disentangled GNN	LLaMA-v2	LLM	✓	✓	Citation, E-Commerce	Node	-
	SAFER [Chandra et al., 2020]	GCN, GAT, etc.	RoBERTa	Linear	✓	✗	News	Node	Link
	GraphFormers [Yang et al., 2021]	Graph Transformer	UniLM	LLM	✓	✗	Citation, E-Commerce, Knowledge	Link	Link
	Text2Mol [Edwards et al., 2021]	GCN	SciBERT	GNN/LLM	✓	✗	Molecular	Retrieval	Link
Others	MoMu [Su et al., 2022]	GIN	BERT	GNN/LLM	✓	✗	Molecular	Graph, Retrieval	Link
	MoleculeSTM [Lin et al., 2022]	GIN	BERT	GNN/LLM	✓	✗	Molecular	Graph, Retrieval	Link
	GLEM [Zhao et al., 2022]	SAGE, RevGAT, etc.	DeBERTa	GNN/LLM	✓	✗	Citation, Co-purchase	Node	Link
	GrAD [Mavromatis et al., 2023]	SAGE	SciBERT/DeBERTa	LLM	✓	✗	Citation, Co-purchase	Node	Link
	G2P2 [Wen and Fang, 2023]	GCN	Transformer	GNN/LLM	✓	✓	Citation, Recommendation	Node	Link
	Patton [Jin et al., 2023]	Graph Transformer	BERT/SciBERT	Linear/LLM	✓	✗	Citation, E-Commerce	Node, Link, Retrieval, Reranking	Link
	ConGrAT [Brannon et al., 2023]	GAT	all-mppnet-base-v2/DeBERTa	GNN/LLM	✓	✗	Citation, Knowledge, Social	Node, Link	Link
	GRENADE [Li et al., 2023b]	SAGE, RevGAT-KD, etc.	BERT	GNN/MLP	✓	✗	Citation, Co-purchase	Node, Link	Link
	LLM-GNN [Chen et al., 2023b]	GCN, SAGE	GPT3.5	GNN	✗	✓	Citation, Co-purchase	Node	Link
	GPT4GNAS [Wang et al., 2023a]	GCN, GIN, etc.	GPT-4	GNN	✗	✓	Citation	Node	-
	ENG [Yu et al., 2023]	GCN, GAT	ChatGPT	GNN	✗	✓	Citation	Node	-

Table 1: A summary of models that leverage LLMs to assist graph-related tasks in literature, ordered by their release time. **Fine-tuning** denotes whether it is necessary to fine-tune the parameters of LLMs, and  $\heartsuit$  indicates that models employ parameter-efficient fine-tuning (PEFT) strategies, such as LoRA and prefix tuning. **Prompting** indicates the use of text-formatted prompts in LLMs, done manually or automatically. Acronyms in **Task**: Node refers to node-level tasks; Link refers to link-level tasks; Graph refers to graph-level tasks; Reasoning refers to Graph Reasoning; Retrieval refers to Graph-Text Retrieval; Captioning refers to Graph Captioning.



**Thank You!**