Can Graph Learning Improve Planning in LLM-based Agents?

Xixi Wu[1] Yifei Shen[2] Caihua Shan[2] Kaitao Song[2] Siwei Wang[2] Bohang Zhang[3] Jiarui Feng[4] Hong Cheng[1] Wei Chen[2] Yun Xiong[5] Dongsheng Li[2]

[1]The Chinese University of Hong Kong [2]Microsoft Research Asia [3]Peking University [4]Washington University, Saint Louis [5]Fudan University

xxwu@se.cuhk.edu.hk https://wxxshirley.github.io/

香港中文大學 The Chinese University of Hong Kong

arXiv GitHub 知乎

NEURAL INFORMATION PROCESSING SYSTEMS

Graph Formulation of Planning

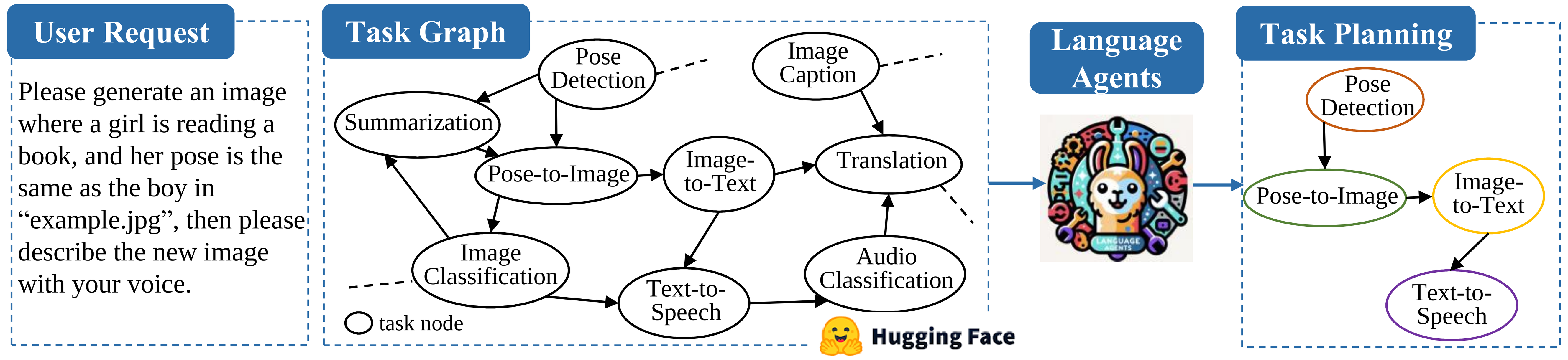
Planning is a fundamental component of **human intelligence**

- Travel Planning
- Math Problem Solving
- Research "Divide-and-Conquer"

Planning Capabilities of LLM-based Agents are Crucial for Achieving AGI

- Tool Agent HuggingGPT
- Game Agent Voyager
- Research Agent Chemical Research

Planning in LLM-based Agents (e.g., HuggingGPT) as a Graph Decision-Making Problem



Available tasks form a task graph where each node represents a task and edges indicate their dependencies. Task planning can be formulated as selecting a connected path or subgraph within this graph to fulfill the user's request.

Theoretical **Can LLMs Effectively Solve Graph Decision-making Problems?** **Empirical**

Graph decision-making problems are often solved by dynamic programming (DP). We investigate the expressive of Transformers to simulate DP.

Theorem 1. *(Inductive bias of language hinders expressiveness)*

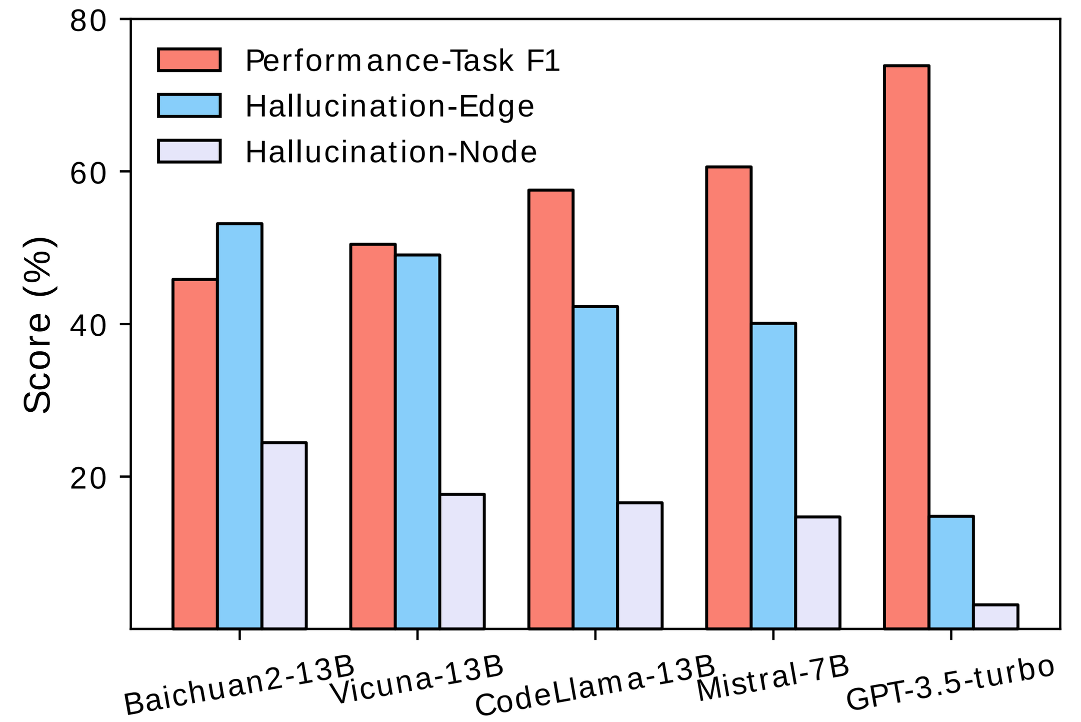
Transformers can simulate DP based on in-context graph input. But language pretrained Transformers with sparse attention cannot.

Theorem 2. *(Spurious correlations of auto-regressive loss)*

The graph decision-making is a RL problem while next-token-prediction is imitation learning, which introduces **spurious correlations.**

Theorem 3. *(GNNs are dynamic programmers* Dudzik & Veličković, 2022*)*

LLMs' **Performance** and **Hallucinations** on HuggingGPT



- **Limited Task Graph Understanding**, i.e., LLMs exhibit certain hallucinations

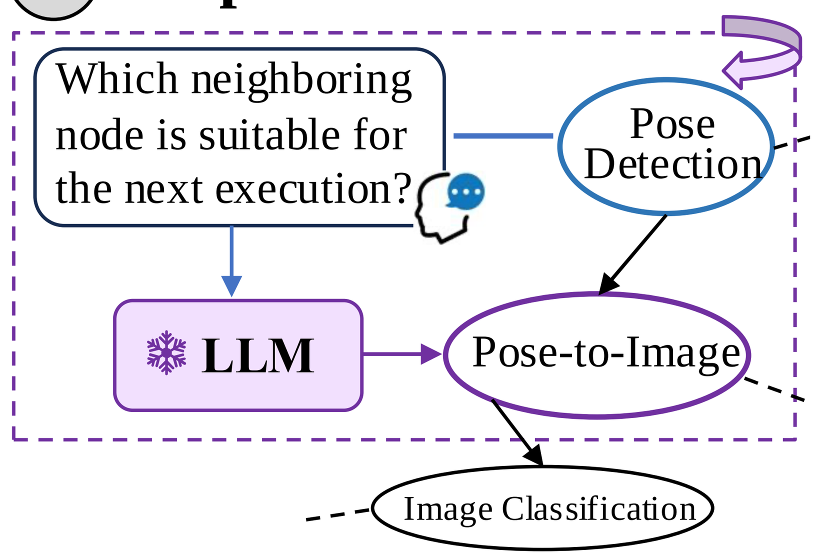
Graph Learning Improves Planning

1 Graph Prompt

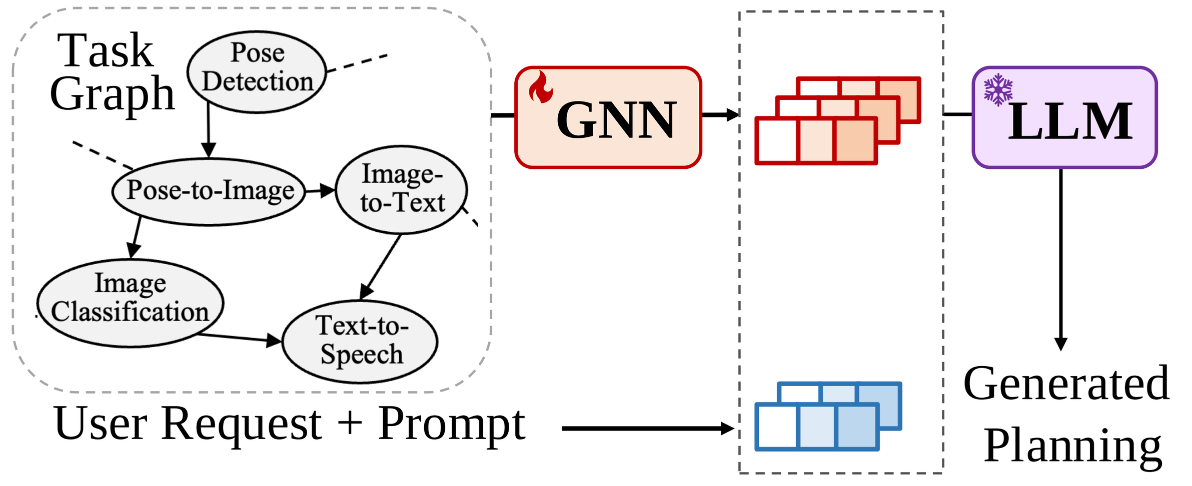
"Plan Like a Graph"

Please carefully understand the above task graph's relations … You can first think about this graph and then produce planning results …

2 GraphSearch

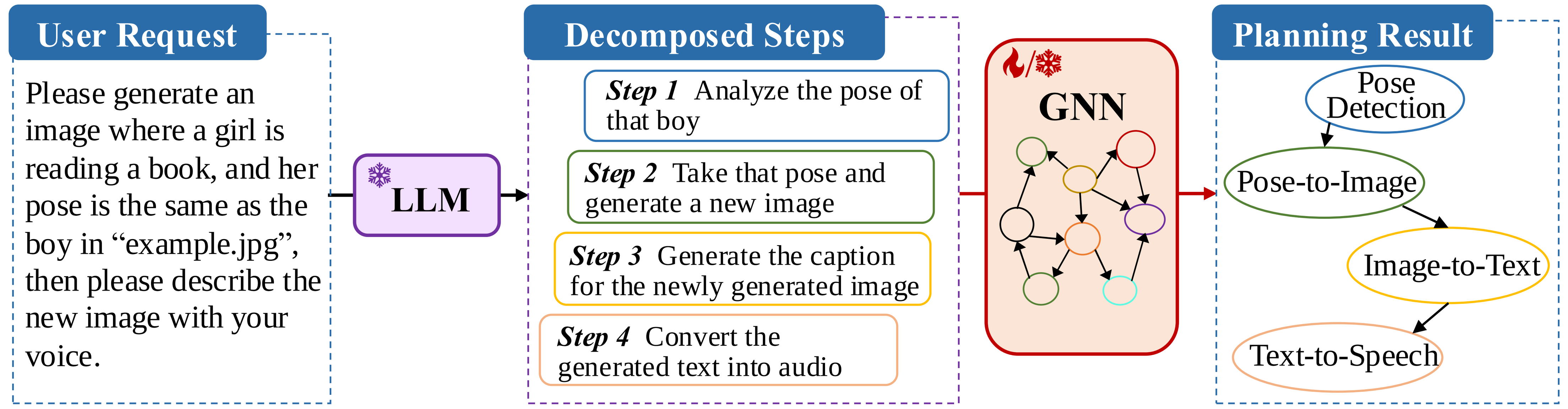


3 Graph Language Model



4 (Ours) GNN for Task Retrieval

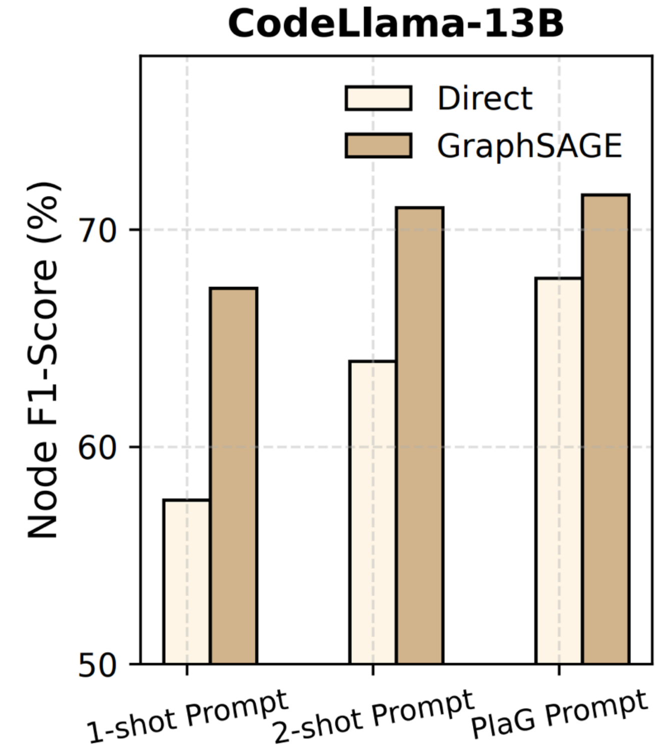
- \+ Overcome Theoretical Limitations
- \+ Avoid Hallucinations
- \+ Scalability to Large Task Graphs



Takeaways: (1) All methods improve the performance (2) GNN is the best

Performance on the UltraTool Benchmark

LLM	Method	0-shot n-F1 ↑	0-shot l-F1 ↑	0-shot Acc ↑	0-shot # Tok ↓	1-shot n-F1 ↑	1-shot l-F1 ↑	1-shot Acc ↑	1-shot # Tok ↓
CodeLlama-13B	Direct	38.88	16.42	13.58	10,535	57.64	30.44	26.25	10,737
	BeamSearch	49.71	22.51	17.08	26,008	64.93	36.23	33.47	23,023
	SGC	61.07	37.61	25.31	10,456	71.64	44.00	39.68	10,658
	GraphSAGE	**63.78**	**39.91**	**27.98**	**10,456**	**72.81**	**45.26**	**43.49**	**10,658**
GPT-3.5-turbo	Direct	54.35	21.35	18.33	8,462	63.58	30.85	25.00	8,614
	BeamSearch	55.40	28.02	19.76	21,979	63.41	34.05	26.28	20,813
	SGC	59.80	37.82	25.87	8,352	64.96	37.96	29.70	8,504
	GraphSAGE	**63.97**	**42.26**	**30.35**	**8,352**	**70.49**	**47.79**	**39.74**	**8,504**
GPT-4-turbo	Direct	68.63	40.01	27.20	8,513	69.54	41.79	28.17	8,693
	BeamSearch	**71.29**	**43.99**	30.40	18,793	**71.99**	44.54	31.62	20,515
	SGC	70.87	44.01	31.60	8,346	70.46	44.82	33.00	8,504
	GraphSAGE	70.67	43.83	**34.40**	**8,346**	70.75	**47.68**	**37.22**	**8,504**



- \+ **SOTA Performance** Outperform existing methods in performance and efficiency
- \+ **Flexible Options** Available in both **training-free** and training-required variants
- \+ **Compatibility** Compatible with both open-sourced and **close-sourced** LLMs
- \+ **Orthogonal Direction** Orthogonal to prompt design and LLMs' fine-tuning

Our paper features a 40-page analysis of theoretical results and empirical evaluations!