Exploring the Role of Reasoning Structures for Constructing Proofs in Multi-Step Natural Language Reasoning with Large Language Models

Zi'ou Zheng[†], **Christopher Malon**[‡], **Martin Renqiang Min**[‡], **Xiaodan Zhu**[†]
[†] Department of Electrical and Computer Engineering
& Ingenuity Labs Research Institute, Queen's University
[‡]NEC Laboratories America
{ziou.zheng,xiaodan.zhu}@queensu.ca
{malon,renqiang}@nec-labs.com

Abstract

When performing complex multi-step reasoning tasks, the ability of Large Language Models (LLMs) to derive structured intermediate proof steps is important for ensuring that the models truly perform the desired reasoning and for improving models' explainability. This paper is centred around a focused study: whether the current state-of-the-art generalist LLMs can leverage the structures in a few examples to better construct the proof structures with *in-context learning*. Our study specifically focuses on structure-aware demonstration and structure-aware pruning. We demonstrate that they both help improve performance. A detailed analysis is provided to help understand the results. [1]

1 Introduction

Large language models (LLMs) have played an essential role in a wide range of applications (Nori et al., 2023; Savelka et al., 2023; Wang et al., 2023; Qin et al., 2023) including intelligent agents (Liu et al., 2023; Cheng et al., 2022). Their ability to perform complex multi-step reasoning has become critical (Wei et al., 2022; Kojima et al., 2022; Yao et al., 2023; Besta et al., 2023; Lei et al., 2023; Dalvi et al., 2021; Ribeiro et al., 2023; Saparov and He, 2023). In complex multi-hop reasoning tasks, the proof steps often form a graph but not just a chain. The capability to construct correct, structured proofs is essential for ensuring that LLMs perform the desired reasoning and important for the explainability of the reasoning models (Dalvi et al., 2021; Ribeiro et al., 2023).

In this paper, we perform a focused study, providing evidence to understand whether the state-of-the-art LLMs can leverage a few examples to better construct the proof structure with *in-context learning*. Unlike previous work that fine-tunes the proof models (Hong et al., 2022; Yang et al., 2022; Dalvi et al., 2021), we focus on in-context learning (Brown et al., 2020) since in many applications, the number of available examples with proof structures is small. In general, an important goal of generalist models is solving different tasks with the built-in ability and without extensive fine-tuning.

In our research, we consider two key components that can utilize the known proof structures: (i) *demonstration*, and (ii) *proof path search and pruning*. We equip the state-of-the-art LLMs, e.g., `GPT-4` and `Llama-3-70B`, with structure-aware demonstration and structure-aware pruning. We set up our study with three benchmark datasets, `EntailmentBank` (Dalvi et al., 2021), `AR-LSAT` (Ribeiro et al., 2023) and `PrOntoQA` (Saparov and He, 2023). The experiment results show that both structure-aware demonstration and structure-aware pruning improve performance. We provide a detailed analysis to help understand the results.

2 Related Work

Recently, LLMs' reasoning ability has been significantly improved. Chain-of-thought (CoT) (Wei et al., 2022; Kojima et al., 2022) is arguably the simplest but effective way to elicit linear reasoning chains of LLMs. Tree-of-thought (ToT) (Yao et al., 2023) can further provide deeper insights into the model's reasoning structures. However, ToT has been applied to tasks such as game-of-24 and creative writing, but not to natural language entailment and reasoning tasks with complex proof structures. In this paper, we will compare our models to the CoT and ToT models.

Reasoning in natural language has been a central topic of artificial intelligence research since its inception, including the research in natural language inference (Dagan et al., 2005; Bowman et al., 2015; Chen et al., 2017, 2018; Feng et al., 2022). In addition to producing accurate results, another key

[1]Our code will be made publicly available at https://github.com/orianna-zzo/structure_reasoning

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 15299–15312

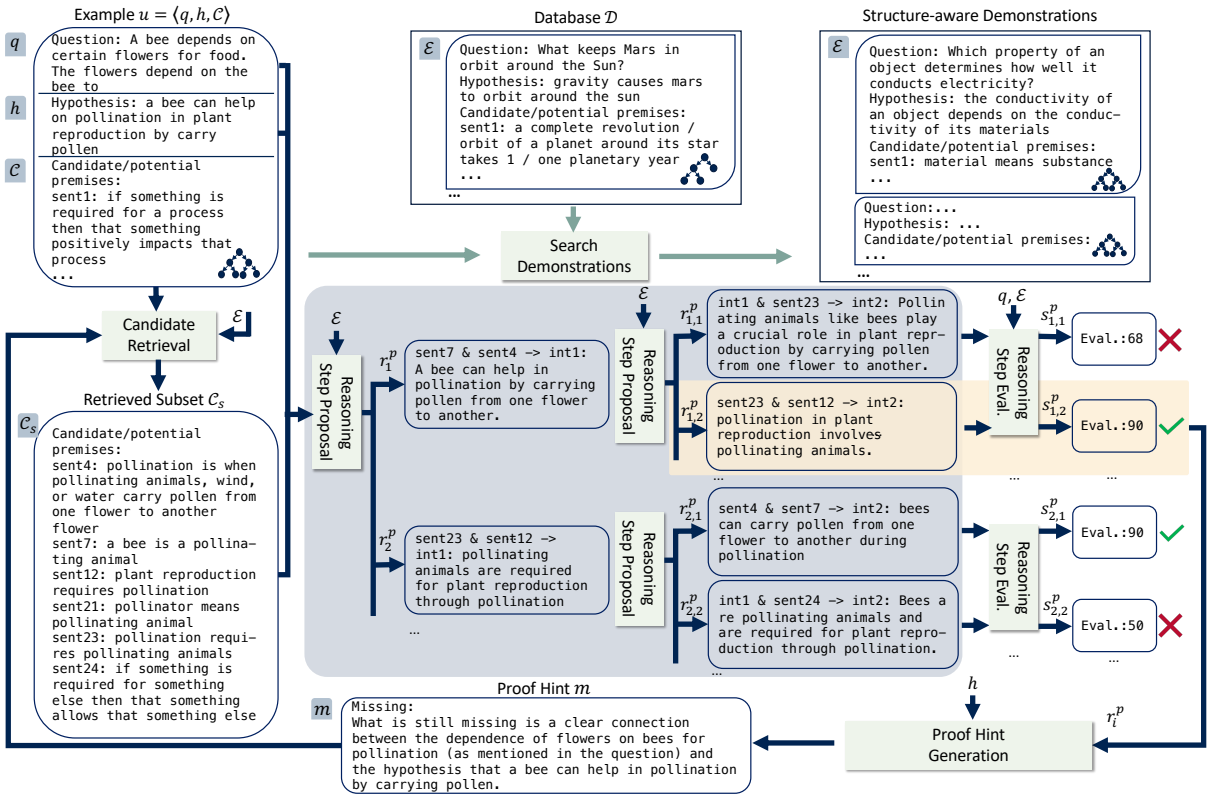

Figure 1: Overview of each module in our proposed framework. Green arrows indicate the process of searching structure-aware demonstrations, while glue arrows illustrate the proof construction process.

challenge is to improve the explainability of these black-box models, and a variety of recent work has been proposed to address this (Gurrapu et al., 2023; Nakamura et al., 2023; Zheng and Zhu, 2023; Liu et al., 2022b; Zini and Awad, 2022; Feng et al., 2020; Valentino et al., 2020). For complex multi-hop natural language reasoning tasks, it is only recently that researchers have begun to develop evaluation datasets to measure proof structure quality (Dalvi et al., 2021; Ribeiro et al., 2023; Yang et al., 2022; Hong et al., 2022). These papers fine-tune small models such as T5 (Raffel et al., 2020) or use older LLMs, lacking studies on structure-aware in-context learning.

3 Method

Given a hypothesis h and context $\mathcal{C}$ consisting of evidence sentences, the objective of the task is to provide a proof graph $\mathcal{G}$ to prove h based on some of the evidence sentences in context $\mathcal{C}$, *if* h can be proven. In entailment or multiple-choice question-answering tasks, h is often a concatenation of a question q and a candidate answer, or sometimes a paraphrase of such a concatenation. Since we will use both q and h to retrieve demonstrations and propose reasoning steps, both q and h will be included in our input. As a result, each instance u in the task is a tuple $u = \langle q, h, \mathcal{C} \rangle$. A proof database $\mathcal{D}$, containing examples of structured proofs, is provided for searching demonstrations that will be used in in-context learning. These demonstrations are exemplars provided to LLMs to help them understand the task requirements and output format.

Formally, we denote p_θ to be a pre-trained language model with parameter θ. Suppose $x = (x_1, \ldots, x_n)$ is a language sequence with n tokens, the probabilistic language model can be written as $p_\theta(x) = \prod_{i=1}^{n} p_\theta(x_i \mid x_{1,\ldots,i-1})$. Following Yao et al. (2023), we use $p_\theta^{\text{prompt}}(y|x)$ to represent $p_\theta(y|\text{prompt}(x))$, where $\text{prompt}(x)$ is the input sentences x wrapped with the prompt instructions and templates; y is the output.

In this paper, we hypothesize that the proof structures of similar examples can help LLMs to construct a structured proof for the target problem. In particular, we consider two key components that can utilize the known proof structures: *demonstration* and *proof-path pruning*. The overall architecture of our model is depicted in Figure 1 and Algorithm 1.

At the high level, to support our study, this paper

proposes a comprehensive framework consisting of six building blocks: *Structure-aware Demonstration*, *Candidate Retrieval*, *Reasoning Step Proposal*, *Reasoning Step Evaluation*, *Proof Hint Generation*, and *Structure-aware Pruning*, which enables us to perform structure-aware proof construction, and based on that, to conduct a deep study on the advantages and limitations of this construction process within the in-context learning setup.

As shown in Figure 1, given an example u with a question q, a hypothesis h, and context $\mathcal{C}$, we first derive suitable demonstrations from the provided database $\mathcal{D}$ in the *structure-aware demonstration* stage. Subsequently, the LLM is prompted to retrieve $\mathcal{C}_s$, the subset of the context $\mathcal{C}$, that will be used in the reasoning (*Candidate Retrieval*). Specifically in the example in Figure 1, the LLM returns sentences 4, 7, 12, 21, 23, and 24. With these retrieved sentences, the LLM is asked to provide the next potential reasoning step (*Reasoning Step Proposal*). This can be performed in multiple iterations. For the first iteration, the LLM provides two proof steps: $\text{sent}7\ \&\ \text{sent}4$, $\text{sent}23\ \&\ \text{sent}12$, and in the second iteration, the LLM proposes two additional steps for each branch. We then evaluate each reasoning step (*Reasoning Step Evaluation*). Next, the model performs structure-guided path pruning and selection (*Structure-aware Pruning*). We note that for each retained branch, the LLM is required to generate a proof hint to guide the next iteration (*Proof Hint Generation*). In the example, we keep two branches and show the generated proof hint for the proposed step $r^p_{1,2}$.

In the following subsections, we will discuss each component in detail and describe how the known proof structures are utilized.

Structure-aware Demonstration. Given an example $u = \langle q, h, \mathcal{C} \rangle$ and a database $\mathcal{D}$ where instances feature structured proofs, the search for most similar demonstrations $\mathcal{E}$ can be expressed as $\mathcal{E} = \mathcal{S}(u, \mathcal{D})$. Usually, $\mathcal{S}$ is defined as manually selecting several fixed demonstrations (Wei et al., 2022; Yao et al., 2023) or choosing the top k demonstrations with the example u based on the similarity (Fu et al., 2022; Liu et al., 2022a). At the initial stage, we prompt LLMs to provide a guessed proof graph $\mathcal{G}^a_u$ of the example u which is used to find the most similar examples as the demonstrations. As the proof moves forward, the partially constructed proof tree will be simply merged into the guessed tree.

This process follows the idea of the EM algorithm. We first obtain the guessed structure through LLMs, and then generate one reasoning step based on the demonstrations with the highest similarity after encoding the example with this estimated structure. Subsequently, we update the guessed structure based on the generated reasoning step. The generation of reasoning steps and the updating of the guess structure are performed iteratively.

Specifically, we use the graph attention network (GATv2) (Brody et al., 2022) and calculate the similarity between the proof graph $\mathbf{E}^a_u$ and each candidate demonstration v's proof graph $\mathbf{E}_v$, which considers both the structure and content of the graphs. We choose the candidates with the higher similarity scores as the demonstrations.

Candidate Retrieval. Given $u = \langle q, h, \mathcal{C} \rangle$, a proof hint m (discussed below), and a set of selected demonstrations $\mathcal{E}$, the candidate retrieval component aims to retrieve a set of most relevant sentences $\mathcal{C}_s$:

$$\mathcal{C}_s = \{o(z_i)\}_{i=1}^{k} \tag{1}$$

$$z_i \sim p_\theta^{\text{Retrieve}}(z|q, h, \mathcal{C}, m, \mathcal{E}) \tag{2}$$

where z_i represents the generated output, which is sampled from the generative language model p_θ that takes in the retrieval prompt. Because z_i contains the needed sentence *id*, we need to extract the *id* from it; the $o(\cdot)$ represents that extraction process. For detailed examples of prompts, refer to Appendix J.1. As a result, $\mathcal{C}_s$ represents a set of retrieved sentences after running the retrieval k times. The proof hint m measures the difference between the current proof status and the hypothesis, which will be discussed later in the *proof hint generation* subsection. Note that the retrieval models can be replaced by a search engine, but we focus more on reasoning itself.

Reasoning Step Proposal. We then prompt LLMs themselves to provide the most plausible proposal for the next reasoning steps. Formally, given $\langle q, h, \mathcal{C}_s, \mathcal{E} \rangle$, the output is reasoning candidates r for the subsequent reasoning step.

$$r_i \sim p_\theta^{\text{Propose}}(r|q, h, \mathcal{C}_s, \mathcal{E}) \tag{3}$$

Then we obtained a set of reasoning steps: $\mathcal{P} = \{r_i\}_{i=1}^{k'}$. The output r_i is parsed to transform the output text into a structured step r^p_i such as $\text{sent}\ i\ \&\ \text{sent}\ j \rightarrow \text{int}\ k$. In Figure 1, we can see one such step is $\text{sent}7\ \&\ \text{sent}4 \rightarrow \text{int}1$, meaning

intermediate conclusion int1 is drawn from sent7 and sent4.

Reasoning Step Evaluation. Given the current structured reasoning step candidate r_i^p and selected demonstrations $\mathcal{E}$, an LLM measures how likely this reasoning step can reach the final hypothesis with a score s_i^p.

$$s_i \sim p_\theta^{\text{Eval}}(s_i | r_i^p, \mathcal{E}) \tag{4}$$

where s_i is the language model output from which the score s_i^p is extracted.

Proof Hint Generation. This component asks LLMs to compare the intermediate conclusion r_i^p with the target hypothesis h to provide *proof hint*:

$$m_i \sim p_\theta^{\text{Compare}}(m_i | h, r_i^p) \tag{5}$$

An example is shown at the bottom of Figure 1. As discussed above, this will be used to guide the model to find the most relevant evidence.

Search Algorithm and Structure-aware Pruning. During the forward proving process, we combine the typical breadth-first search (BFS) with beam search. We maintain b beams of candidates, selecting those with the highest evaluation score from the *Reasoning Evaluation* for each exploration. Furthermore, we delve into the utilization of the problem's structure in this stage. To explore the effect of structure-guiding path selection, we conducted different experiments on how the structures may be used. In our probing experiment (Appendix A) on the dev set of EntailmentBank, we found that models benefit from selecting diverse candidate proof steps; *i.e.*, the models perform better when they are encouraged to select more diverse candidates. That is, two pieces of evidence located on different subtrees are regarded as more diverse than those on the same subtree.

Inspired by this, we discourage the model from using the intermediate conclusions which have been used in the previous steps, to avoid growing the tree from the evidence node that has just been generated. Specifically, when the *Reasoning Step Proposal* module proposes multiple one-step proofs (*e.g.* sent3 & sent4 $\rightarrow$ int2 or sent3 & int1 $\rightarrow$ int2), the pruning algorithm will consider the proof structure to encourage the newly proposed steps to grow the proof graph from different branches. If a newly proposed proof step grows the graph from the nodes that have just been generated in the previous time step, this proposal will be pruned at this time step (it may still be proposed and used in the future). We call this implementation the *diversity* (`div`) variant, which is used in our final model.

4 Experiment Set-Up

Datasets. We perform experiments on three benchmark datasets, `EntailmentBank` (Dalvi et al., 2021), `AR-LSAT` (Ribeiro et al., 2023) and `PrOntoQA` (Saparov and He, 2023). Details can be found in Appendix B.

Evaluation Metrics. We evaluate the predicted proof graph $\mathcal{G}_\text{p}$ against the golden graph $\mathcal{G}_\text{g}$ using the following metrics: F1 over evidence (`Ev-F`) (Dalvi et al., 2021), F1 over proof (`Pr-F`) (Dalvi et al., 2021), and reasoning `Graph Similarity` (`G Sim`) (Ribeiro et al., 2023). Details can be found in Appendix E.

Baselines. We compare the proposed method with CoT, self-consistency CoT (CoT-sc), ToT, and reasoning-via-planing (RAP) (Hao et al., 2023). Each model is prompted with three demonstrations. Details can be found in Appendix C.

5 Experiment Results

We conducted experiments on the representative closed-source (GPT-3.5/GPT-4) and open-source LLMs (Llama-2-70B/Llama-3-70B). Table 1 shows that our models outperform baseline models across the three datasets under different evaluation metrics. The improvements of the proposed model over baselines are statistically significant ($p < 0.05$) under one-tailed paired t-test. Note that the improvement is less in PrOntoQA, which is due to the fact that a larger percentage of data in PrOntoQA has linear reasoning patterns. Detailed results are included in Appendix F.

Effect of Proof Structure. To further understand the effect of proof structures of given examples, we conduct more experiments on EntailmentBank. Table 2 shows the effectiveness of different components of our model. Particularly, our focus is on the variants without structure-aware pruning ("`w/o prun.`") and without structure-aware demonstration ("`w/o demon.`"). We can see that the structure information contributes to the performance (`Ev-F` and `G Sim` scores dropped without them.). The comparison involving other variants of our model, specifically concerning the hint module and pruning strategies, is detailed in Table 12 in the Appendix. The bottom part of Table 2 focuses on

Dataset	Method	GPT-3.5			GPT-4			Llama-2-70B			Llama-3-70B		
		Ev-F	Pr-F	G Sim	Ev-F	Pr-F	G Sim	Ev-F	Pr-F	G Sim	Ev-F	Pr-F	G Sim
EntBank	CoT	.204	.059	.037	.295	.128	.105	.196	.055	.035	.281	.120	.100
	CoT-sc	.210	.062	.038	.303	.138	.112	.200	.059	.037	.288	.127	.110
	ToT	.220	.064	.051	.318	.150	.140	.215	.062	.050	.306	.143	.129
	RAP	.218	.063	.050	.315	.145	.135	.211	.059	.039	.304	.139	.122
	Ours	**.289**	**.100**	**.097**	**.355**	**.181**	**.162**	**.261**	**.085**	**.071**	**.334**	**.170**	**.145**
AR-LSAT	CoT	.472	.054	.007	.507	.078	.008	.470	.045	.006	.493	.078	.008
	CoT-sc	.479	.055	.007	.524	.082	.008	.482	.046	.006	.516	.081	.008
	ToT	.522	.058	.008	.535	.080	.008	.488	.050	.006	.525	.080	.008
	RAP	.519	.057	.008	.532	.074	.007	.482	.046	.006	.527	.075	.007
	Ours	**.585**	**.079**	**.009**	**.595**	**.093**	**.010**	**.515**	**.058**	**.007**	**.585**	**.089**	**.010**
PrOntoQA	CoT	.792	.760	.447	.827	.806	.528	.789	.754	.446	.818	.794	.518
	CoT-sc	.796	.765	.449	.832	.813	.530	.789	.754	.446	.828	.805	.528
	ToT	.814	.779	.482	.837	.812	.530	.793	.755	.447	.829	.804	.528
	RAP	.825	.786	.487	.843	.820	.532	.802	.760	.448	.834	.810	.530
	Ours	**.837**	**.798**	**.504**	**.852**	**.826**	**.533**	**.807**	**.767**	**.450**	**.844**	**.817**	**.531**

Table 1: Performance of different models on three benchmark datasets.

Method	Ev-P	Ev-R	Ev-F	Pr-P	Pr-R	Pr-F	G Sim
Ours	.388	.327	.355	.204	.162	.181	.162
- w/o prun.	.382	.311	.343	.192	.159	.174	.158
- w/o demon.	.341	.257	.293	.145	.103	.120	.110
- w/o hint	.339	.223	.269	.140	.088	.108	.093
- w/o retrieval	.331	.201	.250	.121	.057	.077	.075
Ours (w/o prun.)	.382	.311	.343	.192	.159	.174	.158
Ours_{sim} (w/o prun.)	.367	.258	.303	.149	.121	.134	.100
$\text{Ours}_{\text{oracle}}$ (w/o prun.)	.419	.333	.371	.240	.195	.215	.205

Table 2: Ablation analysis on GPT-4.

	Sequential						Non-sequential					
Dep.	CoT		ToT		Ours		CoT		ToT		Ours	
	Ev-F	Pr-F	Ev-F	Pr-F	Ev-F	Pr-F	Ev-F	Pr-F	Ev-F	Pr-F	Ev-F	Pr-F
3	.333	.150	.356	.157	.357	.157	.250	.121	.266	.149	.297	.151
4	.195	.145	.242	.128	.242	.129	.141	.074	.160	.091	.181	.133
5	.102	.010	.133	.015	.135	.019	.057	.005	.075	.006	.100	.007
6	.013	.001	.055	.005	.059	.005	.011	.001	.043	.003	.050	.004
7	.002	.000	.005	.002	.006	.002	.002	.000	.004	.001	.005	.001

Table 3: Results of sequential /non-sequential reasoning.

evaluating the impact of structure-aware demonstration. We compare the structure-aware demonstration (Ours) vs. regular structure-unaware simple demonstration (Ours_{sim}). We can see that our model is better under GPT-4. The `oracle` model means we suppose that we know in advance the proof structure of the question under study and use that to select the most similar demonstrations. We can see that our model is effective as its gap from the `oracle` is not large.

Analysis on Sequential and Non-sequential Reasoning. The EntailmentBank dataset consists of reasoning problems that only involve sequential reasoning (the ground-truth proof paths of these problems are chains), as well as non-sequential problems. Table 3 depicts the detailed analysis of these two sub-types in the testset with GPT-4. We can see that our method and ToT outperform CoT in both sequential and non-sequential reasoning. Between our model and ToT, they have comparable performance on the sequential subset, while our model performs better than ToT on the non-sequential subset. Our model also outperforms ToT at different depths. Specifically, we conducted a one-tailed t-test over different depths and the differences are statistically significant at depth 3-5 ($p < 0.05$). For depth 6 and 7, the numbers of available samples are too limited to draw any conclusion. In general, we can see that non-sequential reasoning is more challenging than sequential reasoning for all models, due to its higher demands on proof planning and development. The models not only need to explore new potential premises during reasoning but also ensure that the reasoning process remains coherent. Also, the performances of all models decrease on both sequential and non-sequential problems when the depth increases.

6 Conclusion

Enabling LLMs to generate their proof structure is critical for the reliability and explainability of such models. By incorporating structure-aware components into the state-of-the-art LLMs, we demonstrate that LLMs can benefit from utilizing the given proof structures of similar examples. We find that measuring the gap between the intermediate steps and the final hypothesis can help narrow down the search space and enhance the performance. Further analysis of sequential and non-sequential reasoning reveals that our model offers greater advantages in the more complex task of non-sequential reasoning.

Limitations

Our proposed method is primarily designed for the natural language reasoning task, especially the task requiring multi-step proof to obtain the final conclusion. We do not test our method on other types of reasoning, *e.g*. mathematical reasoning and our method is only tested on the English reasoning dataset.

One limitation, as mentioned in the paper, is the increased token usage with the potential reasoning branches exploration since the system uses LLM-as-a-service API. Although we apply the beam search strategy over the graph which needs less exploration compared to the naive breadth-first search, the overall cost is still high. We also leverage LLM in several modules in the system, which increases the total API calls as well.

Another limitation is that the current system does not consider the negation proof or the conclusion that cannot be reached. The goal of the current system is to design a system that provides better proof. Proof by negation and other kinds of reasoning, *e.g*. conjunction, disjunction and conditionals, could be extended in future work.

References

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Michał Podstawski, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. 2023. Graph of Thoughts: Solving Elaborate Problems with Large Language Models.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Shaked Brody, Uri Alon, and Eran Yahav. 2022. How attentive are graph attention networks? In *International Conference on Learning Representations*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Diana Inkpen, and Si Wei. 2018. Neural natural language inference models enhanced with external knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2406–2417, Melbourne, Australia. Association for Computational Linguistics.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced LSTM for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1657–1668, Vancouver, Canada. Association for Computational Linguistics.

Kanzhi Cheng, Zheng Ma, Shi Zong, Jianbing Zhang, Xinyu Dai, and Jiajun Chen. 2022. Ads-cap: A framework for accurate and diverse stylized captioning with unpaired stylistic corpora. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 736–748. Springer.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Proceedings of the First International Conference on Machine Learning Challenges: Evaluating Predictive Uncertainty Visual Object Classification, and Recognizing Textual Entailment*, MLCW'05, page 177–190, Berlin, Heidelberg. Springer-Verlag.

Bhavana Dalvi, Peter Jansen, Oyvind Tafjord, Zhengnan Xie, Hannah Smith, Leighanna Pipatanangkura, and Peter Clark. 2021. Explaining answers with entailment trees. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Yufei Feng, Xiaoyu Yang, Xiaodan Zhu, and Michael Greenspan. 2022. Neuro-symbolic natural logic with introspective revision for natural language inference. *Transactions of the Association for Computational Linguistics*, 10:240–256.

Yufei Feng, Zi'ou Zheng, Quan Liu, Michael Greenspan, Xiaodan Zhu, et al. 2020. Exploring end-to-end differentiable natural logic modeling. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1172–1185.

Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. 2022. Complexity-based prompting for multi-step reasoning. *arXiv preprint arXiv:2210.00720*.

Sai Gurrapu, Ajay Kulkarni, Lifu Huang, Ismini Lourentzou, and Feras A Batarseh. 2023. Rationalization for explainable nlp: a survey. *Frontiers in Artificial Intelligence*, 6:1225093.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Hong, Zhen Wang, Daisy Wang, and Zhiting Hu. 2023. Reasoning with language model is planning with world model. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 8154–8173, Singapore. Association for Computational Linguistics.

Ruixin Hong, Hongming Zhang, Xintong Yu, and Changshui Zhang. 2022. METGEN: A module-based entailment tree generation framework for answer explanation. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1887–1905, Seattle, United States. Association for Computational Linguistics.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Bin Lei, Chunhua Liao, Caiwen Ding, et al. 2023. Boosting logical reasoning in large language models through a new framework: The graph of thought. *arXiv preprint arXiv:2308.08614*.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2022a. What makes good in-context examples for GPT-3? In *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 100–114, Dublin, Ireland and Online. Association for Computational Linguistics.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. 2023. Agentbench: Evaluating llms as agents. *arXiv preprint arXiv: 2308.03688*.

Zhixuan Liu, Zihao Wang, Yuan Lin, and Hang Li. 2022b. A neural-symbolic approach to natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2022*.

Mutsumi Nakamura, Santosh Mashetty, Mihir Parmar, Neeraj Varshney, and Chitta Baral. 2023. Logicattack: Adversarial attacks for evaluating logical consistency of natural language inference. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13322–13334.

Harsha Nori, Nicholas King, Scott Mayer McKinney, Dean Carignan, and Eric Horvitz. 2023. Capabilities of gpt-4 on medical challenge problems. *arXiv preprint arXiv:2303.13375*.

Chengwei Qin, Aston Zhang, Zhuosheng Zhang, Jiaao Chen, Michihiro Yasunaga, and Diyi Yang. 2023. Is ChatGPT a general-purpose natural language processing task solver? In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1339–1384, Singapore. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Danilo Neves Ribeiro, Shen Wang, Xiaofei Ma, Henghui Zhu, Rui Dong, Deguang Kong, Juliette Burger, Anjelica Ramos, zhiheng huang, William Yang Wang, George Karypis, Bing Xiang, and Dan Roth. 2023. STREET: A MULTI-TASK STRUCTURED REASONING AND EXPLANATION BENCHMARK. In *International Conference on Learning Representations*.

Abulhair Saparov and He He. 2023. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. In *The Eleventh International Conference on Learning Representations*.

Jaromir Savelka, Kevin D Ashley, Morgan A Gray, Hannes Westermann, and Huihui Xu. 2023. Explaining legal concepts with augmented large language models (gpt-4). *arXiv preprint arXiv:2306.09525*.

Marco Valentino, Mokanarangan Thayaparan, and André Freitas. 2020. Explainable natural language reasoning via conceptual unification. *arXiv preprint arXiv:2009.14539*.

Yue Wang, Hung Le, Akhilesh Gotmare, Nghi Bui, Junnan Li, and Steven Hoi. 2023. CodeT5+: Open code large language models for code understanding and generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1069–1088, Singapore. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Kaiyu Yang, Jia Deng, and Danqi Chen. 2022. Generating natural language proofs with verifier-guided search. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 89–105, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of Thoughts: Deliberate problem solving with large language models.

Zi'ou Zheng and Xiaodan Zhu. 2023. Natlogattack: A framework for attacking natural language inference models with natural logic. In *Proceedings of the 61st*

Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 9960–9976.

Julia El Zini and Mariette Awad. 2022. On the explainability of natural language processing deep models. *ACM Computing Surveys*, 55(5):1–31.

A Preliminary Experiments

We conduct two preliminary experiments on the dev set of EntailmentBank with GPT-3.5. For the *Preliminary Experiment I*, we provide all other proofs except for randomly deleting two pieces of evidence. We conduct three deletion strategies: two missing pieces of evidence are in the same subtree and the same reasoning step, in the same subtree but not the same reasoning step, or in a different subtree. Here, we set the depth of the subtree to 2. Specifically, "the same subtree and the same reasoning step" means the two missing pieces of evidence can together form an intermediate conclusion in the proof tree, while "the same subtree but different reasoning step" means that the intermediate conclusion from one missing piece of evidence could be combined with the other missing evidence to obtain another intermediate conclusion. "A different subtree" means the two missing pieces of evidence are not in the same 2-depth subtree. Results in Table 4 show that it is easier for the model to find evidence when they are located in a different proving subtree. We further mimic the practical searching scenario in the *Preliminary Experiment II*, where given one chosen reasoning step, *e.g.* sent_4 & $\text{sent}_5 \rightarrow \text{int}_1$, and missed two different reasoning step among which one is based on the given intermediate conclusion (`reuse_ic`) and the other (`div`) is not, *e.g.* sent_3 & $\text{int}_1 \rightarrow \text{int}_2$ and sent_1 & $\text{sent}_2 \rightarrow \text{int}_3$, we ask the model to provide the prediction of the reasoning step. Table 5 shows that `div` model outperforms `reuse_ic` and thus we apply `div` in the main experiment.

Model	Ev-P	Ev-R	Ev-F
same subtree and same reasoning step	0.62	0.59	0.61
same subtree but different reasoning step	0.62	0.58	0.60
different subtree	0.63	0.60	0.62

Table 4: Result of Preliminary Experiment I

Model	Ev-P	Ev-R	Ev-F	Pr-P	Pr-R	Pr-F
reuse_ic	0.57	0.42	0.49	0.35	0.19	0.25
div	0.59	0.45	0.51	0.36	0.19	0.25

Table 5: Result of Preliminary Experiment II

B Dataset

EntailmentBank (Dalvi et al., 2021) not only lists the supporting textural evidence but also offers a hierarchical tree structure showing how the evidence organized to lead to the hypothesis. In the entailment tree, the supporting evidence is the leaf node, the hypothesis is the root node, and the intermediate conclusions are the internal nods. EntailmentBank is also included in the STREET benchmark (Ribeiro et al., 2023). We exclude the cases which only need one reasoning step, *i.e.*, proof depth and length equal to 1.

AR-LSAT is the Analytical Reasoning -Law School Admission Test task from the STREET benchmark (Ribeiro et al., 2023). STREET benchmark is a unified multi-task and multi-domain natural language reasoning and explanation benchmark. Unlike other existing question-answering (QA) datasets, models are expected to not only answer questions but also produce step-by-step structured explanations describing how premises in the question are used to produce intermediate conclusions that can prove the correctness of a certain answer. We only include AR-LSAT in addition to EntailmentBank because the other datasets in STREET focus on math problems or the sequence process prediction which needs different prompts, especially for the comparison module, with those regarding to logic reasoning in this paper. For QA datasets, we keep the question as the input q and append the question and correct answer as the input hypothesis h.

PrOntoQA (Saparov and He, 2023) is a synthetic question-answering dataset, where each example is generated from a synthetic world model represented in first-order logic. The rules applied during the synthetic generation endow it with extractable structural information. We applied a similar process on this QA dataset as AR-LSAT except that some examples reasoned by negative deduction are removed in this version.

C Implementation Details

We retrieve 5 times independently and take the union set as the result of the retrieval component. For each step, we propose 3 potential reasoning steps at each node and we keep the beam size as 3 in the breadth-first search. The number of demon-

strations is set to 3 for all few-shot models. The max iteration number is set to 5 times of the max reasoning depth for each dataset. We conduct the experiments on gpt-3.5-turbo-0613 version of GPT-3.5 and gpt-4-0125-preview version of GPT-4. For GATv2, we train the model with the training set of EntailmentBank.

Details for ToT and RAP We made some modifications for ToT and RAP to better adapt the baselines to the structure-aware natural language reasoning task in this paper. For ToT, we make the thought generator output the potential reasoning step and apply the depth-first-search strategy. RAP needs one fact as the original state to perform the reasoning, which does not work on EntailmentBank and AR-LSAT, and thus we changed the initial state to the void sentence on those datasets.

D Algorithms

The overall algorithm for the proposed method is described in Algorithm 1, and searching structure-aware demonstration and structure-aware pruning can be found in Algorithm 2 and Algorithm 3 respectively. Specifically in Algorithm 2, we use GATv2 as the graph encoder for $\mathrm{GNN}(\cdot)$ and compute cosine similarity score for $\mathrm{sim}(\cdot)$.

E Evaluation Metrics

We evaluate the predicted proof graph $\mathcal{G}_{\text{pred}}$ against the golden graph $\mathcal{G}_{\text{gold}}$ with three metrics, describing evidence, proof and graph similarity. Unlike previous work, we target the model's ability to provide correct proofs more than the true or false result.

Evidence. Following (Dalvi et al., 2021), we perform an evaluation over the chosen evidence to check whether the predicted proof graph uses the correct evidence. Suppose E_{pred} and E_{gold} are the selected evidence set for the predicted proof graph $\mathcal{G}_{\text{pred}}$ and the golden graph $\mathcal{G}_{\text{gold}}$ respectively. We compute precision (`Ev-P`), recall (`Ev-R`) and F1 (`Ev-F`) score by comparing E_{pred} and E_{gold} and taking average over the examples.

Proof Following (Dalvi et al., 2021), we evaluate over individual reasoning steps to check whether the predicted proof graph is structurally correct. Suppose P_{pred} and P_{gold} are the reasoning step set for the predicted proof graph $\mathcal{G}_{\text{pred}}$ and the golden graph $\mathcal{G}_{\text{gold}}$ respectively. We compute precision (`Pr-P`), recall (`Pr-R`) and F1 (`Pr-F`) score by comparing P_{pred} and P_{gold} and taking average over the examples.

Graph Similarity. Following (Ribeiro et al., 2023), we compute the reasoning graph similarity (`G Sim`) $\mathrm{sim}\left(\mathcal{G}_p, \mathcal{G}_g\right)$ by comparing the predicted and the golden reasoning graphs through $\delta\left(\mathcal{G}_p, \mathcal{G}_g\right)$

Algorithm 1: Overview

Input: An example u consisting of a question q, a hypothesis h and a context $\mathcal{C}$: $u = \langle q, h, \mathcal{C}\rangle$ and a database $\mathcal{D}$
Output: Proof graph $\mathcal{G}_p$

Init $\mathcal{G}_p = \varnothing$
Init *Proof hint* $m = \varnothing$
Init $states = \varnothing$
Init $iter = 0$
do
  $\mathcal{E} = \mathrm{SearchDemonstration}(u, \mathcal{D}, \mathcal{G}_p)$ // Obtain structure-aware demonstrations
  $\mathcal{C}_s = \mathrm{CandidateRetrieval}(q, h, \mathcal{C}, m, \mathcal{E})$ // Retrieve most relevant sentences
  $\{r\} = \mathrm{ReasoningProposal}(q, h, \mathcal{C}_s, \mathcal{E})$ // Obtain reasoning step proposals
  foreach $r_i \in \{r\}$ **do**
    $s_i^p = \mathrm{Evaluate}(r_i, \mathcal{E})$ // Obtain evaluation score from *Reasoning Step Evaluation*
  end
  $\{\langle r, s^p\rangle\}_{\text{pruned}} = \mathrm{Prune}(\{\langle r, s^p\rangle\}, \mathcal{G}_p, states)$
  $iter+ = 1$
  foreach $\langle r_i, s_i^p\rangle \in \{\langle r, s^p\rangle\}_{pruned}$ **do**
    $m_i = \mathrm{GenerateHint}(h, r_i)$ // Generate hint for the next iteration
    $\mathcal{G}_p' = \mathcal{G}_p \cup \{\langle r_i, s_i\rangle\}$
    if $\mathcal{G}_p'$ *reaches* h **then**
      Return: $\mathcal{G}_p'$
    else if $iter >$ *the maximum iteration number* **then**
      Continue
    else
      Add $\langle \mathcal{G}_p', m_i, r_i, s_i^p, iter\rangle$ to $states$
    end
  end
  if $states == \varnothing$ **then**
    Break
  end
  Go to next state in $states$ and update $(\mathcal{G}_p, m, iter)$ with the saved value $(\mathcal{G}_p', m_i, iter)$ in $states$
while *True*

Algorithm 2: SearchDemonstration

Input: An example $u = \langle q, h, \mathcal{C}\rangle$, a database $\mathcal{D}$ and current obtained proof graph $\mathcal{G}_p$
Output: Demonstrations $\mathcal{E}$

$\mathcal{G}_u^a = \mathrm{GuessGraph}(\mathcal{G}_{\mathrm{p}})$ // Obtain initial Guessed Proof graph $\mathcal{G}_u^a$
$\mathbf{E}_u^a = \mathrm{GNN}(\mathcal{G}_u^a)$ // Encode graph $\mathbf{E}_u^a$ with GNN
foreach $d_i \in \mathcal{D}$ **do**
  $\mathbf{E}_{di} = \mathrm{GNN}(\mathcal{G}_{di})$ // Encode $\mathcal{G}_{di}$ with GNN
  $s_i = \mathrm{Sim}(\mathbf{E}_{di}, \mathbf{E}_u^a)$ // Compute similarity between $\mathbf{E}_{di}$ and $\mathbf{E}_u^a$
end
Return: Demonstrations $\{d_i\}$ with the top k s_i

Algorithm 3: Prune

Input: Obtained reasoning steps $\{\langle r, s^p\rangle\}$, current obtained proof graph $\mathcal{G}_p$, and active states $states$

Output: Pruned reasoning steps $\{\langle r, s^p\rangle\}_{\text{pruned}}$

Keep $\langle r_i, s_i^p\rangle$ where $s_i^p \in \text{top}_k\{s^p\}$

if $|\mathcal{G}_p| == 1$ **then**

  Delete $\langle r_i, s_i^p\rangle$ where r_i generates conclusion based on intermediate conclusion

end

Return: $\{\langle r, s^p\rangle\}_{\text{pruned}}$

where δ is a graph edit distance function using insertion, deletion and substitution as elementary edit operator over nodes and edges. This can be computed as

$$\text{sim}\left(\mathcal{G}_p, \mathcal{G}_g\right) = 1 - \left[\frac{\delta\left(\mathcal{G}_p, \mathcal{G}_g\right)}{\max\left(|N_p| + |E_p|, |N_g| + |E_g|\right)}\right] \quad (6)$$

F Additional Results

Table 6 and Table 7 shows the additional results of precision (`Ev-P/Pr-P`), recall (`Ev-R/Pr-R`) of evidence and proof on GPT-3.5, GPT-4, Llama-2-70B and Llama-3-70B. The improvements of the proposed model over baselines are statistically significant ($p < 0.05$) under one-tailed paired t-test. For example, the p-values of Ev-F and Pr-F (our method vs RAP on ProntoQA) are 0.0247 and 0.0312 for GPT-3.5, while the values are 0.0404 and 0.0388 for GPT-4. Table 8 and Table 9 shows the additional ablation results on GPT-3.5. In Table 8 and Table 2, in `w/o prun.`, we do not prune the 'proof steps' based on the structure, while in `w/o demon.`, we use three fixed demonstrations instead. In Table 9 and Table 2, the 'sim' variant searches demonstrations only based on the context similarity, without being aware of the structure. Table 10 shows the sequential/non-sequential reasoning on GPT-4. Table 11 shows the proportion of examples where the gold proof is a subset of the predicted proof steps, *i.e.*, the proportion of examples where the per-proof recall is 1. This is a stricter metric than the `Pr-F1`, but it is valuable as it provides insight into the ability of generalist LLMs to produce human-thought correct proofs under different models.

Regarding the guessed structure, we manually examined 20 randomly selected examples (The randomly selected cases are 15, 23, 36, 97, 114, 118, 139, 142, 154, 165, 172, 210, 213, 223, 247, 271, 306, 336, 339, 353 in the EntailmentBank). We found that 8 examples could provide the correct structure in the first guess and the correct number was improved to 12 in the last round of guessing. In comparison to our proposed method, the 'sim' variant in Table 2 and Table 9 searches demonstrations only based on the context similarity, without being aware of the structure.

G Other Variants

Table 12 shows the analysis with other variants of our model. The `reuse_ic` variant requires the model to reuse the intermediate conclusion generated in the previous iteration in the 2nd iteration's reasoning, while `div` variant forces the model to explore the reasoning step from the untouched premises. The `w/o hint` includes all modules except the *proof hint generation* module. We modify the prompt in this module into asking the model what is the next step of reasoning in `what's next`. Our findings indicate that the `div` variant has higher performance than the `reuse_ic` and `w/o pruning` variant, showcasing the effectiveness of the structure-aware pruning.

H Examples

Proof Hint Generation. Table 13 shows two examples with generated hints in the first iteration, and we conduct a comparison between the model with or without the *proof hint generation* module. In the first example, both models could make the correct reasoning in the first iteration and the intermediate conclusion finds out that carbon dioxide is required photosynthesis process. Without the *proof hint generation* module, the model could not retrieve the wanted sentences, while with the *proof hint generation* module, the model succeeds in focusing on the missing relationship with 'step'. Similarly, in the second example, both models could correctly retrieve sent6. However, with the *proof hint generation* module, the model cares more about what the Earth revolve around, not the moon. The examples show that the *proof hint generation* module explicitly asks the model to think about the missing part between the current intermediate conclusion and the final goal and the model could retrieve relevant information based on this action.

Structure-aware Demonstration. Table 14 shows the example with structure-aware demonstrations. For the page limit, we only show the proof structure of one demonstration in the table. We observe that the model is prone to providing the proof that is structurally similar

Dataset	Model	GPT-3.5							GPT-4						
		Ev-P	Ev-R	Ev-F	Pr-P	Pr-R	Pr-F	G Sim	Ev-P	Ev-R	Ev-F	Pr-P	Pr-R	Pr-F	G Sim
EntBank	CoT	.283	.160	.204	.092	.043	.059	.037	.326	.270	.295	.152	.110	.128	.105
	CoT-sc	.289	.165	.210	.098	.045	.062	.038	.332	.279	.303	.161	.121	.138	.112
	ToT	.302	.173	.220	.104	.046	.064	.051	.347	.293	.318	.174	.132	.150	.140
	RAP	.303	.170	.218	.100	.046	.063	.050	.351	.285	.315	.168	.128	.145	.135
	Ours	**.374**	**.236**	**.289**	**.118**	**.087**	**.100**	**.097**	**.388**	**.327**	**.355**	**.204**	**.162**	**.181**	**.162**
AR-LSAT	CoT	.482	.462	.472	.077	.042	.054	.007	.523	.492	.507	.092	.068	.078	.008
	CoT-sc	.490	.468	.479	.079	.042	.055	.007	.541	.508	.524	.100	.070	.082	.008
	ToT	.537	.507	.522	.083	.045	.058	.008	.562	.510	.535	.111	.063	.080	.008
	RAP	.539	.501	.519	.079	.045	.057	.008	.571	.498	.532	.098	.060	.074	.007
	Ours	**.595**	**.576**	**.585**	**.086**	**.073**	**.079**	**.009**	**.602**	**.588**	**.595**	**.122**	**.075**	**.093**	**.010**
PrOntoQA	CoT	.802	.782	.792	.782	.740	.760	.447	.843	.811	.827	.812	.800	.806	.528
	CoT-sc	.803	.790	.796	.782	.748	.765	.449	.848	.816	.832	.820	.806	.813	.530
	ToT	.828	.801	.814	.802	.758	.779	.482	.849	.825	.837	.825	.800	.812	.530
	RAP	.840	.811	.825	.811	.762	.786	.487	.860	.827	.843	.827	.814	.820	.532
	Ours	**.857**	**.817**	**.837**	**.821**	**.776**	**.798**	**.504**	**.866**	**.838**	**.852**	**.831**	**.821**	**.826**	**.533**

Table 6: Performance of different models on GPT-3.5 and GPT-4.

Dataset	Model	Llama-2-70B							Llama-3-70B						
		Ev-P	Ev-R	Ev-F	Pr-P	Pr-R	Pr-F	G Sim	Ev-P	Ev-R	Ev-F	Pr-P	Pr-R	Pr-F	G Sim
EntBank	CoT	.272	.153	.196	.087	.040	.055	.035	.318	.252	.281	.143	.103	.120	.100
	CoT-sc	.281	.155	.200	.092	.043	.059	.037	.322	.261	.288	.149	.110	.127	.110
	ToT	.293	.170	.215	.100	.045	.062	.050	.335	.281	.306	.166	.125	.143	.129
	RAP	.288	.167	.211	.094	.043	.059	.039	.335	.279	.304	.162	.121	.139	.122
	Ours	**.339**	**.212**	**.261**	**.109**	**.069**	**.085**	**.071**	**.361**	**.311**	**.334**	**.194**	**.151**	**.170**	**.145**
AR-LSAT	CoT	.501	.443	.470	.056	.037	.045	.006	.507	.479	.493	.090	.069	.078	.008
	CoT-sc	.508	.458	.482	.059	.038	.046	.006	.535	.499	.516	.101	.068	.081	.008
	ToT	.510	.467	.488	.063	.042	.050	.006	.552	.501	.525	.105	.065	.080	.008
	RAP	.506	.458	.482	.059	.038	.046	.006	.557	.500	.527	.096	.061	.075	.007
	Ours	**.538**	**.493**	**.515**	**.066**	**.051**	**.058**	**.007**	**.590**	**.581**	**.585**	**.114**	**.073**	**.089**	**.010**
PrOntoQA	CoT	.805	.773	.789	.780	.729	.754	.446	.833	.803	.818	.798	.791	.794	.518
	CoT-sc	.805	.773	.789	.780	.729	.754	.446	.847	.809	.828	.807	.804	.805	.528
	ToT	.807	.779	.793	.780	.732	.755	.447	.845	.814	.829	.810	.799	.804	.528
	RAP	.811	.793	.802	.783	.738	.760	.448	0.85	.819	.834	.818	.802	.810	.530
	Ours	**.813**	**.802**	**.807**	**.785**	**.749**	**.767**	**.450**	**.862**	**.827**	**.844**	**.825**	**.810**	**.817**	**.531**

Table 7: Performance of different models on Llama.

Model	Ev-P	Ev-R	Ev-F	Pr-P	Pr-R	Pr-F	G Sim
Ours	.374	.236	.289	.118	.087	.100	.097
- w/o prun.	.372	.230	.284	.117	.087	.100	.097
- w/o demon.	.332	.182	.235	.107	.053	.071	.067
- w/o hint	.313	.167	.218	.103	.049	.066	.064
- w/o retrieval	.311	.166	.216	.092	.047	.062	.058

Table 8: Cumulative ablation analysis.

Model	Ev-P	Ev-R	Ev-F	Pr-P	Pr-R	Pr-F	G Sim
Ours (w/o prun.)	.372	.230	.284	.117	.087	.100	.097
Ours_{sim} (w/o prun.)	.358	.211	.266	.112	.069	.085	.077
$\text{Ours}_{\text{oracle}}$ (w/o prun.)	.392	.259	.312	.153	.132	.142	.138

Table 9: Ablation of demonstration methods.

to the proofs given in the demonstration and we attribute the performance improvement brought by structure-aware demonstrations to this observation.

Dep.	Sequential						Non-sequential					
	CoT		ToT		Ours		CoT		ToT		Ours	
	Ev-F	Pr-F	Ev-F	Pr-F	Ev-F	Pr-F	Ev-F	Pr-F	Ev-F	Pr-F	Ev-F	Pr-F
3	.328	.138	.330	.143	.330	.144	.238	.108	.257	.129	.282	.135
4	.189	.070	.202	.104	.202	.113	.132	.068	.149	.077	.175	.102
5	.082	.003	.123	.007	.125	.007	.049	.002	.069	.005	.093	.006
6	.012	.000	.047	.004	.047	.004	.010	.000	.038	.003	.045	.004
7	.002	.000	.005	.001	.006	.001	.002	.000	.004	.001	.005	.001

Table 10: Results of sequential reasoning /non-sequential reasoning.

I Computation Cost

We observe that the cost of experimenting is higher than the baselines. We leverage the language model in several different modules and apply the beam search strategy in the breadth-first search. We keep a most promising states per step and b beams of candidates with the highest evaluation score for

Dataset	Method	GPT-3.5	GPT-4
EntailmentBank	CoT	.018	.030
	ToT	.024	.041
	Ours	.041	.071
AR-LSAT	CoT	.008	.013
	ToT	.010	.015
	Ours	.018	.030
PrOntoQA	CoT	.210	.226
	ToT	.229	.236
	Ours	.238	.252

Table 11: Proportion of the examples where the per-proof recall is 1.

Model	Ev-P	Ev-R	Ev-F	Pr-P	Pr-R	Pr-F	G Sim
GPT-3.5							
Ours (w/o hint)	.359	.220	.273	.100	.057	.073	.072
Ours (what's next)	.363	.221	.275	.108	.077	.090	.089
Ours (w/o pruning)	.372	.230	.284	.117	.087	.100	.097
Ours (reuse_ic)	.363	.231	.282	.117	.082	.096	.095
Ours (div)	.374	.236	.289	.118	.087	.100	.097
GPT-4							
Ours (w/o hint)	.371	.247	.297	.136	.102	.117	.101
Ours (what's next)	.379	.253	.303	.158	.121	.137	.121
Ours (w/o pruning)	.382	.311	.343	.192	.159	.174	.158
Ours (reuse_ic)	.380	.309	.341	.192	.157	.173	.158
Ours (div)	.388	.327	.355	.204	.162	.181	.162

Table 12: Ablation analysis on EntailmentBank.

each exploration in the beam search strategy. Although we cut down the total number of explored cases of n reasoning iterations to $a+(n-1)\times b\times a$ from $a+a^2+a^3+\cdots+a^n$ because of the beam search over the tree, it is still higher than CoT (1) and ToT ($n \times a$). Table 3 shows our benefits on non-sequential reasoning but similar performance with ToT on sequential reasoning. Considering the computation cost, our model might not be a good choice if most data belongs to sequential reasoning.

J Example Prompts

We provide three demonstrations in all few-shot models, but we only show one in the example in this section.

J.1 Prompt for Candidate Retrieval

System: Below, you are given a question, a hypothesis and a set of candidate premises. You are required to select a small set of candidates (at least provide 3 sentences) to deduce the hypothesis. Please only filter out the sentences that you are sure of.

[example]
Question: What keeps Mars in orbit around the Sun?
Hypothesis: gravity causes mars to orbit around the sun
Candidate/potential premises:
sent1: a complete revolution / orbit of a planet around its star takes 1 / one planetary year
sent2: our sun is located at the center of our solar system
sent3: celestial objects are located in outer space
sent4: gravity causes orbits
sent5: orbit is a kind of characteristic
sent6: a star usually is larger than a planet
sent7: revolving around something means orbiting that something
sent8: a satellite orbits a planet
sent9: uranus is a kind of planet
sent10: planets are found in space
sent11: gravity means gravitational pull / gravitational energy / gravitational force / gravitational attraction
sent12: as mass of a planet / of a celestial body increases, the force of gravity on that planet will increase
sent13: the sun is the strongest source of gravity in the solar system
sent14: a galaxy is made of stars
sent15: orbit means orbital path
sent16: can be means able to be
sent17: celestial bodies / celestial objects are found in space
sent18: satellites are found in space
sent19: proxima centauri is a kind of star
sent20: planets in the solar system orbit the sun
sent21: mars is a kind of planet
sent22: venus is a kind of planet
sent23: mars is located in the solar system
sent24: isaac newton discovered the theory of gravity
sent25: a comet is a kind of celestial body

Retrieval sentences (at least 3): sent4, sent20, sent21, sent23
Proof: sent20 & sent4 -> int1: gravity causes the planets in the solar system to orbit the sun; sent21 & sent23 -> int2: mars is a planet in the solar system; int1 & int2 -> hypothesis;
—-
[Question]
Question: A bee depends on certain flowers for food. The flowers depend on the bee to
Hypothesis: a bee can help on pollination in plant reproduction by carry pollen

Model	Previous Intermediate Conclusion	Retrieved Premises
	Case 1: Hypothesis: taking in carbon dioxide is a step in the photosynthesis process sent2: when carbon dioxide in the atmosphere is absorbed by plants , the amount of carbon dioxide in the atmosphere is reduced in the atmosphere sent4: if something is required in a process then obtaining that something is a step in that process sent8: taking something in is a kind of method for obtaining that something sent17: a step in a process means a stage in a process sent25: taking in and releasing means exchanging Generated hint: What is still missing is a direct connection or evidence that "taking in carbon dioxide" is indeed a step or part of the process of photosynthesis, rather than just a raw material involved in it. Expected: sent4 & (sent9 & sent19)	
w/o hint	sent9 & sent19 -> int1: Carbon dioxide is a required raw material in the photosynthesis process.	sent2, sent12, sent17, sent25
w/ hint	sent9 & sent19 -> int1: Carbon dioxide is required as a raw material in the photosynthesis process.	sent2, **sent4**, sent8, sent17
	Case 2: Hypothesis: the difference between the earth and the moon is that the moon revolves around a planet sent1: celestial bodies / celestial objects are found in space sent3: earth is a kind of planet sent4: moons / comets / planets are part of the solar system sent6: the earth revolves around the sun sent9: the sun is a kind of star sent10: a moon is a kind of satellite sent11: revolving around means orbiting sent23: the moon is earth 's moon sent24: a celestial body travelling around another celestial body means that celestial body completes a cycle around that other celestial body Generated hint: What is still missing is evidence that explicitly states that the Earth does not revolve around another planet. Expected: (sent6 & sent9) & ((sent25 & sent3)& sent11)	
w/o hint	sent3 & sent25 -> int1: The Earth and the Moon are both planets, but the Moon orbits the Earth.	sent1, sent4, **sent6**, sent10, sent23, sent24
w/ hint	sent3 & sent25 -> int1: Earth is a planet and the Moon orbits it.	sent3, **sent6**, **sent9**, sent10, **sent11**

Table 13: 2nd iteration of reasoning examples for w/ and w/o proof hint generation module

Model	Demonstration Proof	Final Proof
	Hypothesis: wood boards are a kind of building material that is made of a renewable natural resource sent3: wood boards are made of wood sent7: wood is a renewable resource sent8: a renewable resource is a kind of natural resource sent17: wood boards can be used to build houses sent19: a house is a kind of building sent23: building materials are used to build buildings Expected: ((sent19 & sent23) & sent17) & ((sent7 & sent8) & sent3)	
Text-aware Demonstration	(sent25 & sent3) & sent2	((sent7 & sent8) & sent17)
Structure-aware Demonstration	((sent26 & sent3) & sent1) & ((sent7 & sent9) & sent10)	((sent19 & sent23) & sent17) & ((sent7 & sent8) & sent3)

Table 14: Final proof for structure-aware demonstration and demonstration with the most similar context

Candidate/potential premises:
sent1: if something is required for a process then that something positively impacts that process
sent2: pollinated means after pollination
sent3: pollinating is a kind of function
sent4: pollination is when pollinating animals, wind, or water carry pollen from one flower to another flower
sent5: if something causes a process then that something is required for that process
sent6: seed dispersal has a positive impact on a plant / a plant's reproduction sent7: a bee is a pollinating animal
sent8: flowers sometimes become fruits after pollination
sent9: if a living thing requires something then that something has a positive impact on that living thing
sent10: flowers are a source of fruit
sent11: if something is required then that something must be provided
sent12: plant reproduction requires pollination
sent13: needing something means depending on that something
sent14: to be used for something means to be required by that something
sent15: flowers often have a sweet smell to attract pollinators
sent16: to carry means to transport
sent17: a bird is a pollinating animal
sent18: a flower's purpose is to produce seeds
sent19: when pollen sticks to a hummingbird, that pollen will move to where the hummingbird moves
sent20: plant requires seed dispersal for reproduction
sent21: pollinator means pollinating animal
sent22: seed dispersal is a kind of method of sexual reproduction
sent23: pollination requires pollinating animals
sent24: if something is required for something else then that something allows that something else
sent25: requiring something means needing that something

Retrieval sentences (at least 3):

J.2 Prompt for Reasoning Step Proposal

System: Provide me several sentences with the sentence number and one intermediate conclusion that are possible to be used in the next step in this small set. If the deduction reaches the hypothesis, tell me 'Finish'; otherwise please provide the (intermediate) conclusion.

[example]
Question: What keeps Mars in orbit around the Sun?
Hypothesis: gravity causes mars to orbit around the sun
Candidate/potential premises:
sent4: gravity causes orbits
sent5: orbit is a kind of characteristic
sent12: as mass of a planet / of a celestial body increases, the force of gravity on that planet will increase
sent20: planets in the solar system orbit the sun
sent21: mars is a kind of planet
sent22: venus is a kind of planet
sent23: mars is located in the solar system
sent24: isaac newton discovered the theory of gravity

Possible Next Reasoning: sent20 & sent4 -> int1: gravity causes the planets in the solar system to orbit the sun
—
[Question]
Question: A bee depends on certain flowers for food. The flowers depend on the bee to
Hypothesis: a bee can help on pollination in plant reproduction by carry pollen
Candidate/potential premises:
sent4: pollination is when pollinating animals, wind, or water carry pollen from one flower to another flower
sent7: a bee is a pollinating animal
sent12: plant reproduction requires pollination
sent21: pollinator means pollinating animal
sent23: pollination requires pollinating animals
sent24: if something is required for something else then that something allows that something else

Possible Next Reasoning:

J.3 Prompt for Reasoning Step Evaluation

System: Evaluate whether these intermediate conclusions could reach the hypothesis with candidates. Provide me the number of possibilities (0-99) of these intermediate conclusions: Surely: 85-99, Likely: 50-84, Impossible: 0-49

[example]
Question: What keeps Mars in orbit around the Sun?
Hypothesis: gravity causes mars to orbit around the sun
Candidate/potential premises:
sent4: gravity causes orbits
sent5: orbit is a kind of characteristic
sent12: as mass of a planet / of a celestial body increases , the force of gravity on that planet will increase
sent20: planets in the solar system orbit the sun
sent21: mars is a kind of planet
sent22: venus is a kind of planet
sent23: mars is located in the solar system
sent24: isaac newton discovered the theory of gravity

Possible Next Reasoning: sent20 & sent4 -> int1: gravity causes the planets in the solar system to orbit the sun
Evaluate: 99
—
[Question]
Question: The body of a fish is covered by scales for
Hypothesis: scales are used for protection by fish
Candidate/potential premises:
sent1: a fish is a kind of scaled animal
sent8: scales are a covering around the body of a scaled animal
sent12: scales are used for protection by scaled animals
sent15: protecting is a kind of function

J.4 Prompt for Proof Hint Generation

System: Compare the intermediate conclusion with the hypothesis and the question, and provide me one sentence of what is still missing.

[example]
Question: What keeps Mars in orbit around the Sun?
Hypothesis: gravity causes mars to orbit around the sun
Intermediate Conclusion: int1: gravity causes the planets in the solar system to orbit the sun
Missing: What is missing is to specifically state that Mars is one of the planets in the solar system.
—
[Question]
Question: The body of a fish is covered by scales for
Hypothesis: scales are used for protection by fish
Intermediate Conclusion: int1: scales cover the body of a fish
Missing: