

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/385173895>


EvidenceChat: A RAG Enhanced LLM Framework for Trustworthy and Evidential Response Generation

Conference Paper · October 2024

CITATIONS
0

READS
29

4 authors, including:



Longchao Da

Arizona State University

20 PUBLICATIONS 44 CITATIONS

SEE PROFILE




Parth Shah

Arizona State University

1 PUBLICATION 0 CITATIONS

SEE PROFILE



Hua Wei

New Jersey Institute of Technology

68 PUBLICATIONS 2,755 CITATIONS

SEE PROFILE

EvidenceChat: A RAG Enhanced LLM Framework for Trustworthy and Evidential Response Generation

Longchao Da, Parth Mitesh Shah, Ananya Singh, Hua Wei*

{longchao,pshah113,asing582,hua.wei}@asu.edu

Arizona State University

Tempe, Arizona, USA

Abstract

The Large Language Model has become an important assistant for many of human's decision-making. However, a side note almost comes along with every single LLM: 'LLMs can make mistakes. Be careful with important info'. This reveals the fact that not all of the information from LLMs is trustworthy, and people still need to judge by themselves. Yet, the hallucination is so powerful that a made-up conclusion could even come with a seemingly plausible reason, which brings intricate challenges and a trust crisis among users. This paper proposes EvidenceChat, a framework that tackles this issue in a retrieval-augmented fashion, specifically, when the user uploads a material document, an indexed vector space will be constructed based on text embeddings, which helps construct a retrieval-augmented agent, enhancing the agent's responses with additional knowledge beyond its training corpus. Then, we propose a three-step evidential retrieval method to accurately locate the evidence of a language agent's conclusion in the source context: Chain-of-Thought (CoT) logic generation, top-k relevant chunk vector matching, and a self-reflection-based precise sentence identification. We demonstrate that our method improves the existing models' performance in terms of identifying the exact evidence in a free-form context, providing a reliable way to examine the resources of LLM's conclusion and help with the judgment of the trustworthiness. The dataset and code will be released.

CCS Concepts

• **Computing methodologies** → **Natural language processing**; *Artificial intelligence*; • **Information systems** → *Web searching and information discovery*.

Keywords

Retrieval Augmented Generation, Evidential Response, LLMs

ACM Reference Format:

Longchao Da, Parth Mitesh Shah, Ananya Singh, Hua Wei*. 2024. EvidenceChat: A RAG Enhanced LLM Framework for Trustworthy and Evidential Response Generation. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM '24)*. ACM, New York, NY, USA, 5 pages. <https://doi.org/XXXXXXX.XXXXXXX>

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

CIKM '24, October 25, 2024, Boise, USA

© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 978-1-4503-XXXX-X/18/06

<https://doi.org/XXXXXXX.XXXXXXX>

1 Introduction

The Large Language Models (LLMs) have shown promising abilities in multi-round conversational chat, it could understand questions [5], provide responses, and even conduct reasoning or inference based on provided context [20]. The rapid development of LLMs has advanced many applications in diverse fields such as customer support [3, 11], virtual assistance [19], and even augmented agent with tool usage abilities [1].

However, even though the LLMs are trained on massive expert corpus covering a wide range of topics, they are not immune to generating incorrect or misleading information, as called 'hallucination' [7]. It is always suggested to users on LLM interface that, 'LLMs can make mistakes. Check important information carefully', which underscores the dilemma of seeking to benefit from LLMs' capabilities while contending with the challenge of the quality of the responses.

The solutions can be roughly divided into two aspects: One is to ground the LLMs' output to the factuality, either by fine-tuning [17] or information retrieval augmentation [16]. Fine-tuning on the LLMs to reduce hallucination takes more resources and can be computationally expensive, meanwhile, it is not applicable to those black-box commercial LLMs. And information-retrieval based methods take more steps for multi-resource factuality checking and grounding, such as [2, 6]. This will rely on the information source to vote for a trustworthiness of the generated responses, and requires more query times from other APIs or Knowledge bases. The other direction is to understand the confidence/uncertainty of LLMs' responses: such as the training uncertainty prediction layers, or applying post-hoc response level uncertainty quantification. These methods tend to quantify a measurement value for the LLMs, it provides a convenient way to compare the performance among LLMs, but not straightforward enough for users to decide whether to believe the LLMs' conclusion.

Standing on this shortcoming, some work started with the evidence matching of the generated responses. Essentially, when a user uploads a document and performs question asking to certain LLM, it tends to highlight the corresponding raw context from a document that is best relevant with the response, thus guide the user to understand where the conclusion is drawn from the original document. However, the current method such as [4, 9] use the direct LLM responded source of the evidence [15], when faced redundant response, it can only perform on the chunk-level resource highlight, which often gives a whole paragraph of relevant context without fine concentration. We also empirically verify that the LLMs' own source evidence reflection performance varies a lot on the LLMs' instruction-following ability, but sometimes for an arbitrary small language model that answers specific domain questions well, yet

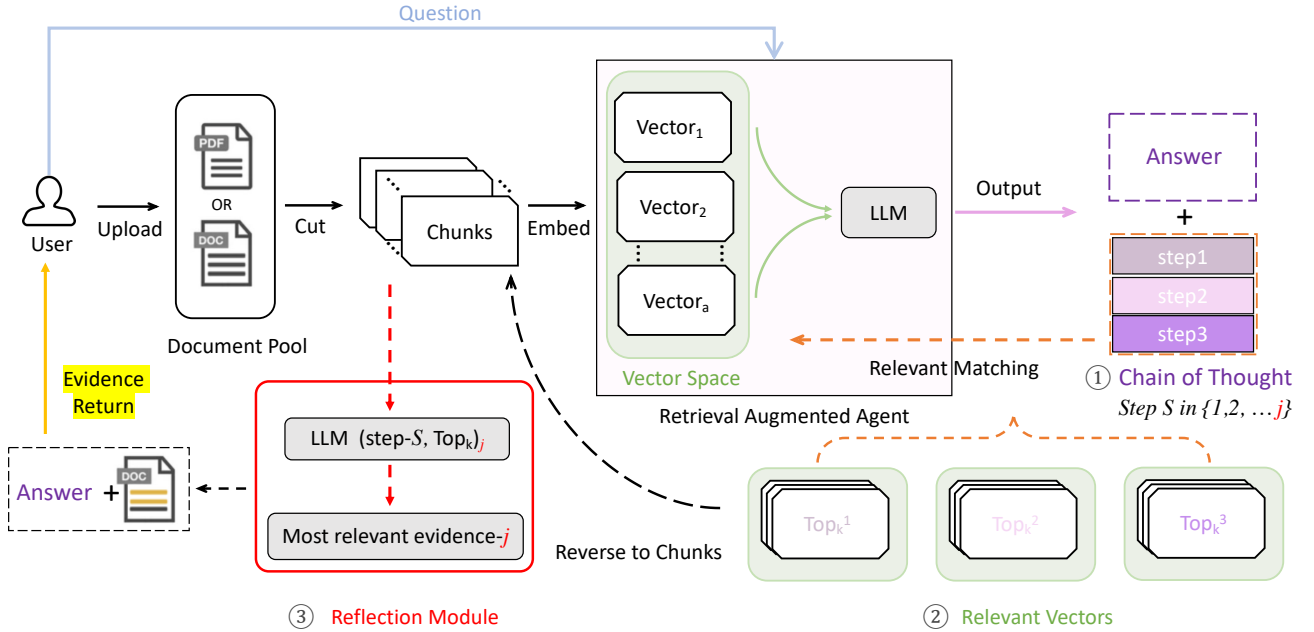


Figure 1: The Framework of EvidenceChat. At the beginning, the user could choose to upload a document type (either PDF or text DOC file), and the raw document is cut into a chunks, then they are transformed into vectors by embedding action, and form a Vector Space supporting the LLM’s response (together the combination is taken as a Retrieval Augmented Agent), when a user asks a question on the document, ① Chain of Thought (CoT) is provided along with the answer, serving as the raw reasoning steps, and ② the top k relevant chunk vectors are searched, traversed back to raw chunk corpus, and then fed into the LLM with a reflection template, then it extracts ③ the most relevant sentence level evidence from the raw text source. And the user is provided with the answer and the source evidence supporting the conclusion.

lacking the instruct fine-tuning, it is hard to perform highlighting for users to refer.

To resolve above questions, this paper proposes a framework named **EvidenceChat** that provides users the evidence of a LLM’s response in a more accurate and generalizable way. It worth noting that, different from existing work, this method not only pose hard-constraint on derived source that it must come from the raw context, besides, **EvidenceChat** provides sentence-level fine-grained identification to accurately mark out the evidence supporting the LLMs’ conclusion. What’s more, this framework can be applied to any LLM with outstanding evidence retrieval ability (even a small model without fine-tuning on instruction-following ability, our framework still helps to highlight the evidence of conclusion). We compare with the direct LLM source reflection of the response, and our method essentially improves on the evidence retrieval performance. A [demo video](#) and [dataset](#)¹ are provided for better understanding.

2 Approach

In this section, we will discuss the details of the **EvidenceChat** framework. Since this work builds upon the retrieval augmented generation agent (RAG agent), and then provide a universal paradigm for better evidential response generation, we will introduce the process of constructing the RAG agent, and then explain three components in our framework.

¹Click link to the demo video

RAG agent construction As shown in the upper-left part of Fig. 1, the retrieval augmented generation agent mainly incorporates three actions: Upload, Cut, Embed.

First, when a user start the interaction with the system, it is triggered by the uploading action of a document, this document could be the normal .TXT/.DOC file, or could also be the .PDF format, if the file is in .PDF, an extra action will be enforced by PyMuPDF [12], which is a parser for handling PDF files in Python, enabling to manipulate the metadata in the file efficiently.

Second, the meta data will be cut into the corpus chunks to temporarily store the file, in this step, the number of chunks is controlled by a parameter α , assume the original document contains \mathcal{W} tokens, with a simple calculation shows that each chunk will, on average, contain approximately $c_i = \mathcal{W}/\alpha$ tokens, where the c_i is the i_{th} chunk. The chunk often serves as a smallest unit for current prevalent vector indexing method to return the relevant corpus source, thus, the existing work [13] directly adopting the vector indexing method (such as LlamaIndex [10]), suffers from the tuning of the parameter α , and can not provide concise source evidence reflection and highlighting while returning a whole paragraph.

Third, with chunks $c_i \in C$, each chunk will be fed into the corresponding tokenizer to embed, which aligns with the following LLM that is used to conduct retrieval generation, to ensure the transformed vector space is the same as the LLM’s training space. And the derived $v_i \in \vec{V}$ will be used to support the LLM’s generation,

where $|\vec{V}| = |C| = \alpha$. The later Relevant Vector Matching will be performed on the \vec{V} space to ensure the obtained evidence is hard-constrained on the raw document, without any fabrication or rephrase to avoid hallucination.

Based on the above RAG agent, the following sections will focus the three key components: the Chain-of-Thought Reasoning Inducer of LLMs to generate reasoning steps for a conclusion, the Relevant Vector Matching to obtain k nearest origin-space vectors, and the Fine-grained LLM Reflection Module to produce sentence-level relevance matching.

2.1 Chain-of-Thought Reasoning Inducer

This section introduces the Chain-of-Thought (CoT) reasoning inducer, which serves as a primary step to derive the reasoning process. It is well acknowledged that majority of the LLMs can automatically performs CoT [18] to elicit the reasoning process, i.e., how they draw the conclusion step by step. We follow the same idea to induce the logic steps $Logic_steps\{step_1, step_2, \dots, step_n\}$ from the LLM given a question Q on a document. By designing a CoT template, we tend to achieve the following:

$$Answer, Logic_steps = CoT_template(Q, Doc) \quad (1)$$

The template is inspired by work [21], and is shown as below:

«INST»«SYS»
You are an agent to provide question answering tasks based on the provided document.
[Task]
Your task is to generate answers to the user's question, please think step-by-step for the conclusion, and provide your thinking steps behind the output.
[Output Format]
Answer: { [text] }
Thoughts: {1.[text] 2.[text] 3.[text] ... n.[text]}

This step is in correspondence with the the upper right part of the Fig. 1, procedure ①, where we take 3 steps as an example, and there is a limit in practice for maximum of 10 steps.

2.2 Relevant Vector Matching

When the $Logic_steps$ are generated, each of the steps reflects the LLM's reasoning details, it will be transformed back into the vector space by $v_s^j = \text{Embed}(S_j)$, where the step j reasoning sentence: S_j will be taken as input, and generate the v_s^j as output. Then we perform the vector searching based on the distance in vector space, the embedding searching is at the sentence level to best represent the closest semantic information. The distance d between a reasoning step embedding v_s^j to other vector v_i is calculated as following equation:

$$d = 1 - \frac{\vec{v}_s^j \cdot \vec{v}_i}{\|\vec{v}_s^j\| \cdot \|\vec{v}_i\|} \quad (2)$$

where the ' \cdot ' denotes the dot product of the two vectors, $\|\vec{v}_s^j\|$ and $\|\vec{v}_i\|$ are the Euclidean norms on reasoning step embedding and candidate vectors in vector space. We calculate the distance and

obtain the sentence S_i in corresponding to the vector v_i reflecting the minimal distance to the vector of reasoning step j :

$$Text_k = \underset{S_i}{\operatorname{argmin}} d(\vec{v}_s^j, \vec{v}_i | i \in |\vec{V}|) \quad (3)$$

where in this paper, we perform the experiment under the setting that $k = 3$.

As shown in the bottom right part of the Fig. 1, for each of the induced reasoning step description, we generate k closest vectors and indexed with the original sentence chunks (reverse back). And these sentence chunks will be applied for LLM reflection on the evidence in the following section.

2.3 Fine-grained LLM Reflection Module

As discussed in work [7], LLMs can steadily enhance the generation factuality and conciseness by self-examination. Thus, in this section, we tend to further increasing granularity and relevance by LLM's self-reflection ability. We designed the Fine_template to achieve that: $\text{Count}(\text{Evidence_sentences}) \leq \text{Count}(\text{Text}_k)$ on supporting the reasoning step S_j , the $\text{Count}(\cdot)$ indicates the sentence counting, note that here we use Text_k as the Top_k interchangeably:

$$\text{Evidence_sentences} = \text{Reflection_template}(S_j, \text{Text}_k) \quad (4)$$

Assume the current processed Text with $k = 1$, then the instance of Reflection_template is as below:

«INST»«SYS»
You are an agent helping with the relevance examination.
[Task]
Your task is to pick the sentence-level content that could be used as evidence supporting the reasoning step { S_j } given a text segment { Text_1 }.
[Do Not]
You should not make up for any sentence but directly extract the sentence from the segment { Text_1 }.
[Output Format]
Evidences:{ [text] }

We show this step in the procedure ③ in Fig. 1. It generates the most relevant evidence for each step of the reasoning process on the raw document basis. And all of the evidence will eventually be obtained for the whole document by going through the j steps. And the evidence is a part of the meta data in document, which will be applied to highlight in the preview of a document. The answer from LLM together with the returned evidence helps user to better judge on the trustworthiness of the LLM's response quality, and the raw context based evidence forbids the fabrication of LLMs.

3 Experiment

In this section, we conduct the experiment to demonstrate the performance of EvidenceChat. The work [4] directly applies the LLM's output source for evidence generation, since it is for commercial use, the most advanced GPT4o is adopted to achieve the superiority. However, in our scenario, we tend to provide a generalizable framework that help any LLMs accurately shows the evidence of

their answer, so we first evaluate the performance under the setting of [4] across multiple prevalent models.

3.1 Experiment Setup

Dataset Construction: Due to the lack of evidence sources in the existing work, We have constructed a set of dataset for evaluation on the quality of evidence generation of different methods. We have 10 categories of test sets, and each of the set will be proposed certain questions, and the correct answers as well as the evidence sentence in context is annotated by humans, in total, we perform evaluation on 200 cases. The detailed categories and the relevant match id of the experiment figure is shown in Table 1.

Table 1: Overview of PDF Categories in the Collection

Category	PDF File Details	Match
type1	Chemistry Book	DS0
type2	Story Book	DS1
type3	News Analysis	DS2
type4	Deep Learning Fundamentals	DS3
type5	History Book	DS4
type6	Culinary Recipes	DS5
type7	Thermodynamics Book	DS6
type8	Embedded Lock System Book	DS7
type9	Regulations and Standards	DS8
type10	Cell Biology Book	DS9

Evaluation Metric: We refer to the existing work [8] that uses the cosine similarity to evaluate the relevance of the generated text (generated evidence) with the correct text (correct evidence), and use the conciseness score [14] to quantify the LLM’s ability to find the corresponding evidence precisely. In general, we have the following calculation that combines the two aspects of evaluation on Evidence_{score}:

$$\text{Evidence}_{\text{score}} = \frac{1}{N} \sum_{i=1}^N \left[\cos(E_i, E_{gt_i}) \cdot \min \left(1, \frac{L_{gt_i}}{L_i} \right) \right] \quad (5)$$

where the E_i is the embedding of the generated evidence for question i , and E_{gt_i} is the groundtruth evidence. The first term measures the cosine similarity of two evidences, the larger the better, and L_{gt_i} is the length of text for groundtruth evidence while the L_i is the generated evidence, $\frac{L_{gt_i}}{L_i}$ measures how concise the generated evidence is given the relevance on meaning from cosine similarity.

3.2 Experiment Result

We first conducted experiment on the 11 models and analyze their result in the Table. 2. We could observe that the evidence retrieval ability of GPT4o is the best, while other models performs much worse, especially because too many words (w) generated that prevents a concise evidence presenting. Then we applied our method EvidenceChat to the models except for GPT4o, and the results is shown in the Fig. 2.

4 Conclusion

In this paper, we have presented EvidenceChat, a novel framework designed to address the critical issue of trustworthiness in LLMs.

Table 2: The evaluation on the prevalent models

Model	Evidence Score	Average Word Count
llama2	0.17	238w
llama2.1:3b	0.12	286w
<u>llama2-70b</u>	<u>0.65</u>	<u>68w</u>
llama3.1	0.29	190w
phi3	0.46	143w
phi3:medium	0.35	146w
germa2:2b	0.39	116w
germa2	0.29	255w
llava	0.20	169w
qwe2	0.29	255w
GPT4o	0.73	30w

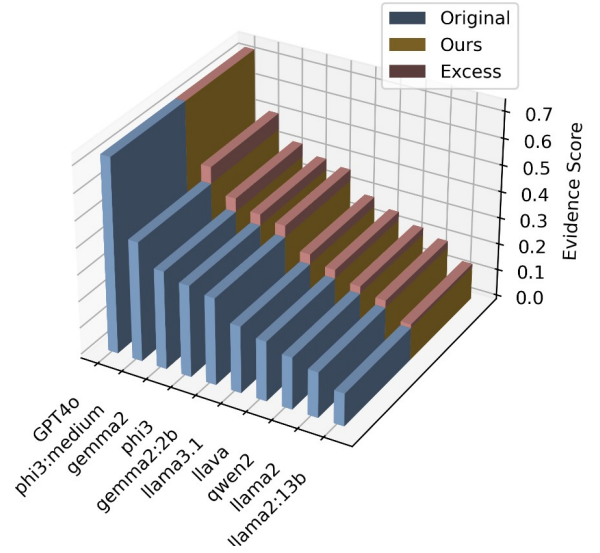


Figure 2: The performance comparison on the evidential generation between original LLM generation and EvidenceChat.

Our approach introduces a rigorous method for evidence retrieval and verification, significantly enhancing the reliability of LLM-generated responses. The implementation of hard constraints on source derivation, coupled with sentence-level highlight capabilities, has yielded remarkably promising results.

Our comprehensive evaluation, encompassing ten diverse LLMs, both open and closed-source—demonstrates the robustness and versatility of EvidenceChat. This extensive comparison not only validates the efficacy of our approach but also underscores its potential for broad applicability across various language models.

In conclusion, EvidenceChat represents a step forward in addressing the trust crisis surrounding LLMs. By providing a robust, flexible, and user-friendly framework for evidence retrieval and verification, we contribute to the broader goal of making AI systems more transparent, reliable, and trustworthy. As LLMs continue to play an increasingly crucial role in decision-making processes across various sectors, frameworks like EvidenceChat will be instrumental in ensuring their responsible and effective deployment.

References

- [1] Longchao Da, Kuanru Liou, Tiejun Chen, Xuesong Zhou, Xiangyong Luo, Yezhou Yang, and Hua Wei. 2024. Open-ti: Open traffic intelligence with augmented language model. *International Journal of Machine Learning and Cybernetics* (2024), 1–26.
- [2] Hanxing Ding, Liang Pang, Zihao Wei, Huawei Shen, and Xueqi Cheng. 2024. Retrieve only when it needs: Adaptive retrieval augmentation for hallucination mitigation in large language models. *arXiv preprint arXiv:2402.10612* (2024).
- [3] Asbjørn Følstad and Marita Skjuve. 2019. Chatbots for customer service: user experience and motivation. In *Proceedings of the 1st international conference on conversational user interfaces*. 1–9.
- [4] Rujun Han, Yuhao Zhang, Peng Qi, Yumo Xu, Jinyuan Wang, Lan Liu, William Yang Wang, Bonan Min, and Vittorio Castelli. 2024. RAG-QA Arena: Evaluating Domain Robustness for Long-form Retrieval Augmented Question Answering. *arXiv:2407.13998* [cs.CL] <https://arxiv.org/abs/2407.13998>
- [5] Wenbo Hu, Yifan Xu, Yi Li, Weiye Li, Zeyuan Chen, and Zhuowen Tu. 2024. Bliva: A simple multimodal llm for better handling of text-rich visual questions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 2256–2264.
- [6] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *Comput. Surveys* 55, 12 (2023), 1–38.
- [7] Ziwei Ji, Tiezheng Yu, Yan Xu, Nayeon Lee, Etsuko Ishii, and Pascale Fung. 2023. Towards mitigating LLM hallucination via self reflection. In *Findings of the Association for Computational Linguistics: EMNLP 2023*. 1827–1843.
- [8] Leonie aka helloiamleonie. 2024. *evaluatingrag1*. <https://blog.langchain.dev/evaluating-rag-pipelines-with-ragas-langsmith/>
- [9] Demiao Lin. 2024. Revolutionizing retrieval-augmented generation with enhanced PDF structure recognition. *arXiv preprint arXiv:2401.12599* (2024).
- [10] LlamaIndex Project Contributors. 2024. *LlamaIndex*. <https://docs.llamaindex.ai/en/stable/>
- [11] Chandran Nandkumar and Luka Peternel. 2024. Enhancing Supermarket Robot Interaction: A Multi-Level LLM Conversational Interface for Handling Diverse Customer Intents. *arXiv preprint arXiv:2406.11047* (2024).
- [12] PyMuPDF Project Contributors. 2024. *PyMuPDF Documentation*. <https://pymupdf.readthedocs.io/en/latest/> Accessed: 2024-08-18.
- [13] Quanteria Project Contributors. 2024. *quantera*. <https://www.quantera.ai/>
- [14] Ragas authors. 2024. *conciseness*. https://docs.ragas.io/en/stable/concepts/metrics/summarization_score.html
- [15] Jon Saad-Falcon, Joe Barrow, Alexa Siu, Ani Nenkova, Ryan A Rossi, and Franck Dernoncourt. 2023. PDFTriage: question answering over long, structured documents. *arXiv preprint arXiv:2309.08872* (2023).
- [16] Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation. *arXiv preprint arXiv:2104.07567* (2021).
- [17] Lei Wang, Jiabang He, Shenshen Li, Ning Liu, and Ee-Peng Lim. 2024. Mitigating fine-grained hallucination by fine-tuning large vision-language models with caption rewrites. In *International Conference on Multimedia Modeling*. Springer, 32–45.
- [18] 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 (2022), 24824–24837.
- [19] Rongxuan Wei, Kangkang Li, and Jiaming Lan. 2024. Improving Collaborative Learning Performance Based on LLM Virtual Assistant. In *2024 13th International Conference on Educational and Information Technology (ICEIT)*. IEEE, 1–6.
- [20] Lifan Yuan, Ganqu Cui, Hanbin Wang, Ning Ding, Xingyao Wang, Jia Deng, Boji Shan, Huimin Chen, Ruobing Xie, Yankai Lin, et al. 2024. Advancing llm reasoning generalists with preference trees. *arXiv preprint arXiv:2404.02078* (2024).
- [21] Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2022. Automatic chain of thought prompting in large language models. *arXiv preprint arXiv:2210.03493* (2022).

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009