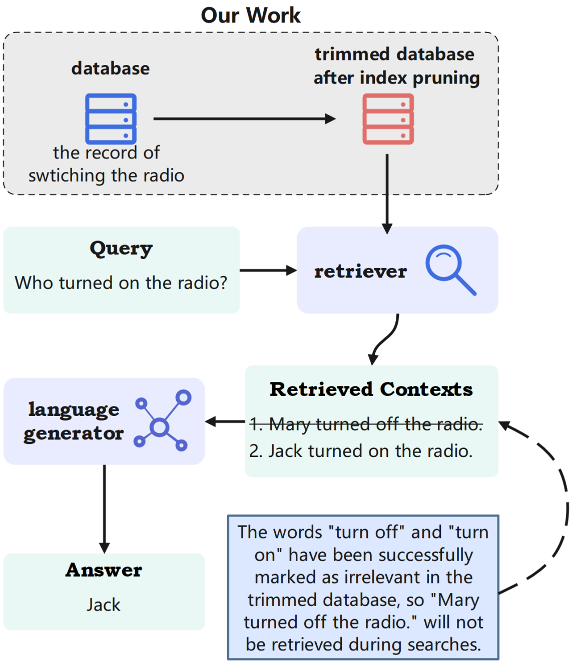
Context-Driven Index Trimming: A Data Quality Perspective to Enhancing Precision of RALMs

Kexin Ma[*1,2]**, Ruochun Jin**[*2]**,**
Haotian Wang[2], Xi Wang[1,2], Huan Chen[2], Yuhua Tang[1,2], Qian Wang[†3]
[1]Institute for Quantum Information & State Key Laboratory of High Performance Computing,
[2]College of Computer Science and Technology,
National University of Defense Technology, Changsha, China,
[3]Intelligent Game and Decision Lab, Academy of Military Science, Beijing, China
Correspondence: {makexin, jinrc}@nudt.edu.cn, wanqiannudt@sina.com

Abstract

Retrieval-Augmented Large Language Models (RALMs) have made significant strides in enhancing the accuracy of generated responses. However, existing research often overlooks the data quality issues within retrieval results, often caused by inaccurate existing vector-distance-based retrieval methods. We propose to boost the precision of RALMs' answers from a data quality perspective through the Context-Driven Index Trimming (CDIT) framework, where Context Matching Dependencies (CMDs) are employed as logical data quality rules to capture and regulate the consistency between retrieved contexts. Based on the semantic comprehension capabilities of Large Language Models (LLMs), CDIT can effectively identify and discard retrieval results that are inconsistent with the query context and further modify indexes in the database, thereby improving answer quality. Experiments demonstrate average improvement of 3.75% in accuracy on challenging question-answering tasks. Also, the flexibility of CDIT is verified through its compatibility with various language models and indexing methods, which offers a promising approach to bolster RALMs' data quality and retrieval precision jointly[1].


Our Work
database
trimmed database after index pruning
the record of swtiching the radio
Query
Who turned on the radio?
retriever
Retrieved Contexts
~~1. Mary turned off the radio.~~
2. Jack turned on the radio.
language generator
Answer
Jack

The words "turn off" and "turn on" have been successfully marked as irrelevant in the trimmed database, so "Mary turned off the radio." will not be retrieved during searches.

Figure 1: Improve data quality of database to enhance the accuracy of generated answers by RALMs.

1 Introduction

Retrieval-augmented large language models (RALMs) have drawn extensive attention, as they effectively ameliorate hallucination (Huang et al., 2023), update the knowledge required for LLMs with minimal cost (Lewis et al., 2020), and provide explanations for contents generated by LLMs (Gao et al., 2023). However, recent study has demonstrated that not all retrieved citations are useful for the generation result (Liu et al., 2023a; Wang et al., 2023), where retrieval may reduce the quality of generation. For example, if retrieval contains information conflicts, the generation quality may deteriorate, which leads to false answers to factual questions (Liu et al., 2023b).

In view of the challenges above, the NLP community mainly focus on enhancing the retrieval precision, e.g. ARR (Yu et al., 2023), REPLUG (Shi et al., 2023), and Atlas (Izacard et al., 2022), which learns to align the retriever outputs with the preferences of LLMs. However, these works pay little attention to the data quality of knowledge itself. More specifically, it is commonly assumed that data in the knowledge base (usually implemented as vectors in a vector database) is consistent, which is actually not the case in real-world applications. Some sentences appear contextually similar yet are actually opposite in reality. For ex-

[*]These authors contribute equally to this work.
[†]The corresponding author.
[1]Our code are available at https://github.com/makexine/CDIT.

Findings of the Association for Computational Linguistics: EMNLP 2024, pages 4886–4901
November 12-16, 2024 ©2024 Association for Computational Linguistics

ample, consider two sentences "He turned on the radio" and "He turned off the radio". It is obvious that the semantic meanings of actions in these two sentences are completely opposite. However, the distance between vector representations of the two tends to be small, as most text embedding models are trained to project sequences of tokens that frequently co-occur as neighboring vectors in the high dimensional semantic space (Li and Yang, 2018). Thus, existing vector-distance-based retrieval methods implemented in vector databases may treat these two sentences as “highly similar” knowledge, and provide such irrelevant or even conflicting sentences as referring knowledge to the down-stream language generator, which confuses the LLMs and deteriorates the quality of the generated answer(Lewis et al., 2020).

As shown in Figure 1, we approach the issue of retrieval quality from the perspective of data quality. Specifically, inspired by Matching Dependencies(MDs), a classical rule-based data quality management method in the database community (Fan et al., 2011), we propose **Context Matching Dependencies (CMDs)** that capture and regulate the consistency between the knowledge context and its vector representation. Then we establish a **Context-Driven Index Trimming (CDIT)** framework that mainly utilizes CMDs and LLM to improve the quality of RALMs answers by trimming the indexes of vector database. The CDIT framework starts with an initial retrieval by the retriever in RALMs. Then the preliminary retrieval results are sent to the CMDs where an LLM is employed to determine whether the retrieved knowledge conforms to the CMDs constraints. If the retrieval satisfies the CMDs, it will be passed to LLMs following conventional RALMs. Otherwise, the retrieval will be discarded, and the vector-search index related to this retrieval will be corrected such that future similar queries can avoid unrelated retrievals return by the vector database.

We experimentally verify the effectiveness of CDIT by open-domain question answering. In addition, we integrate CDIT with different language generation models and index construction methods, demonstrating the flexibility of our framework. CDIT surpasses the basic models with average accuracy improvements of 3.75% on vairous language models. It also boosts the model accuracy by 3.44%, 4.07%, and 3.75% over IndexFlatL2, IndexHNSWFlat, and IndexIVFFlat, respectively. Among them, the highest performance improvement can reach up to 15.21%.

Our main contributions are as follows:

- We propose Context Matching Dependencies (CMDs) that maintain consistency among vector data to address the challenge of poor retrieval in RALMs from the perspective of data quality management.
- We develop the Context-Driven Index Trimming (CDIT) framework based on CMDs and LLMs to improve the quality of RALMs answers by trimming the indexes of vector database, which is applicable to any RALM.
- We experimentally verify the effectiveness of CDIT, where the average and the most significant improvement can reach up to 3.75% and 15.21% respectively.

2 Related Work

Retrieval Improvements in RAG. Not all of the retrieved contexts benefit the final results(Liu et al., 2023a; Wang et al., 2023). In order to improve the retrieval quality, previous work mainly focuses on fine-tuning the retriever to align with the language model. For example, REPLUG(Shi et al., 2023) freezes the parameters of language model LLM and optimizes the retriever to adapt to the language model. Atlas(Izacard et al., 2022), by contrast, jointly trains the retriever and the language model. Additionally, other work explores improving strategies before and after retrieval. Specifically, document segmentation strategies (Touvron et al., 2023b) and embedding models (Karpukhin et al., 2020) can be improved before retrieval. Diversity Ranker in Haystack(Blagojevic) and LostInTheMiddleRanker(Liu et al., 2023a), on the other hand, investigates document re-ranking after retrieval. Different from previous methods, we approach the retrieval quality issue from the perspective of data quality management, where the vector index is trimmed based on data consistency captured by CMDs and LLMs.

Data Quality Rules. Various logic-based rules and dependencies have been proposed for data quality management. For instance, Functional Dependencies (FDs) (Codd, 1971) were first introduced in the 1970s to represent integrity constraints and relationships among data. Based on FDs, Conditional Functional Dependencies (CFDs) (Fan et al., 2008) have been proposed for data cleaning purposes. They use conditions to specify the subset of tuples on which a dependency holds. Subsequently, Matching Dependencies (MDs) (Fan et al., 2011) have been proposed to identify records representing the same real-world entity. Approximate Functional Dependencies (AFDs) (Karegar et al., 2021) have been proposed to tolerate partial violation tuples to handle noisy datasets better. In addition, Association Rules (ARs), which were first used to capture item relationships in transaction data, have also been widely studied for data repair and association analysis on relationship data. Meanwhile, the mining of data dependencies can be referred to research in (Song and Chen, 2009; Schirmer et al., 2020; Fan et al., 2010; Santhya et al., 2014). Similar rules have been applied on graphs (Galárraga et al., 2013; Cao et al., 2023; Fan et al., 2022), to analyze social networks by extracting relations (Erlandsson et al., 2016; Cagliero and Fiori, 2013). Graph Association Rules (GARs) (Fan et al., 2015, 2016, 2020) have defined association rules directly on graphs, for graph data analysis (Fang et al., 2016; Song et al., 2016) and knowledge graph search (Namaki et al., 2017). However, all these data quality rules are designed for relations or graphs, which can hardly support vector data quality management tailored to RALMs.

Vector Indexing Strategies. Vector databases facilitate efficient similarity search using specialized indexing structures such as KD-trees(Bentley, 1975), R-trees(Guttman, 1984), and HNSW (Malkov and Yashunin, 2016). In Faiss (Douze et al., 2024), there are numerous indexing implementations available, including IndexFlatL2, IndexHNSWFlat, IndexIVFFlat and so on. Although IndexFlatL2 is relatively slow and memory-intensive, it achieves the highest in precision(Douze et al., 2024). By contrast, IndexHNSWFlat is fast during searches, at the cost of long index building time and large memory space(Malkov et al., 2014). Moreover, all these methods are inadequate for RAG since the ANN-based indexing can hardly distinguish statements that are literally similar but semantically different(Noonan, 2015). Several studies have noted the impact of indexing and resorted to re-ranking after retrieval, such as Diversity Ranker in Haystack(Blagojevic) and LostInTheMiddleRanker(Liu et al., 2023a). However, they failed to recognize that indexing method of the vector database supporting RAG is inherently unreliable.

Our research, different from previous studies, attends to data management of database, modifying the indexing structure to provide an ideal vector data source for search potentially.

3 Context Matching Dependencies

We first recall Matching Dependencies (MDs) (Fan et al., 2011) before introducing our methods. Given a relational schema R consisting of a set of attributes $attr(R)$, for each attribute $A \in attr(R)$, $dom(A)$ denotes the domain of A. Consider an instance r of R and a tuple $t \in r$, then for $\forall A \in attr(R), t[A] \in dom(A)$, where $t[A]$ represents the projection of t onto A. Matching Dependencies (MDs) are defined to match the attributes of different tuples as follows.

Definition 1 *Matching Dependency*

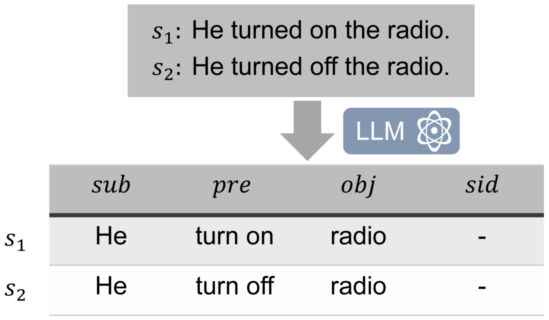
$$\bigwedge_{j\in[1,k]} (r_1[A_j] \approx_j r_2[B_j]) \to \bigwedge_{i\in[1,h]} (r_1[E_i] \rightleftharpoons r_2[F_i])$$

where for $\forall j \in [1, k], \forall i \in [1, h]$, A_j and E_i are attributes of r_1, B_j and F_i are attributes of r_2, $\approx$ is the similarity predicate which returns true if the two attributes are regarded as similar, $\rightleftharpoons$ is the matching operator which indicates that the attributes are identified (Fan et al., 2011).

Similar to relational database tuples that consist of multiple attributes, natural language sentences can be represented by their linguistic components such as subject, predicate and object(Stefanescu et al., 2014). Thus, inspired by MDs for entity resolution in relational databases, we can determine whether two sentences are semantically similar based on the similarity between their corresponding linguistic components. Let sub, pre and obj denote the subject, predicate and object of a sentence, respectively. We also define *semantic id* (sid) that denote the semantic meaning of a natural sentence. Similar to "id" as the main key in relational databases, sid identifies a sentence in the high-dimensional semantic space. If sentences s_1 and s_2 have similar semantics, then their semantic ids are consistent, denoted as $s_1[sid] \sim s_2[sid]$.

Example 1 *Consider the following two sentences.*
s_1 *: He turned on the radio.*
s_2 *: He turned off the radio.*
As shown in Figure 2, $s_1[sub]$,$s_1[pre]$ *and* $s_1[obj]$ *represents He, turn on and radio in sentence* s_1*, respectively, while* $s_2[sub]$,$s_2[pre]$ *and* $s_2[obj]$ *denotes He, turn off and radio in* s_2*, respectively.*

Inspired by MDs designed for entity resolution in databases, in order to serve RAG better, we



s_1: He turned on the radio.
s_2: He turned off the radio.

	sub	*pre*	*obj*	*sid*
s_1	He	turn on	radio	-
s_2	He	turn off	radio	-

Figure 2: Example relational data of natural language

symbols	notations
R, r	relational schema, instance
A, B, E, F	attribute
X, Y	component
s, s_1, s_2	natural sentences
$s[X]$	the corresponding words in sentence
$\sim, \not\sim$	consistency predicate
$\approx, \not\approx$	similarity predicate
$\rightleftharpoons$	the matching operator

Table 1: Summary of main symbols and notations

propose Context Matching Dependency (CMD) to manage data consistency in vector databases.

Definition 2 *Context Matching Dependency*

$$s_1[sid] \sim s_2[sid] \rightarrow \bigwedge_{i \in [1,k]} (s_1[X_i] \approx_i s_2[Y_i])$$

where $\approx$ means that the corresponding sentence components are similar, X_i, Y_i denote grammatical components in s_1, s_2. In addition, $\not\sim$ and $\not\approx$ denote sid inconsistency and dissimilarity of sentence components, respectively.

Example 2 *Consider the following CMD.*

$$\begin{aligned}\phi_1 : & s_1[sid] \sim s_2[sid] \rightarrow s_1[sub] \approx s_2[sub] \wedge \\ & s_1[pre] \approx s_2[pre] \wedge s_1[obj] \approx s_2[obj]\end{aligned}$$

ϕ_1 claims that if the semantic meanings of s_1 and s_2 are consistent, then their corresponding subjects, predicates, and objects should be similar in semantics. Similar to FDs and MDs (Codd, 1971; Fan et al., 2011), CMD can be applied to vector databases to check data consistency.

Table 1 summarizes symbols and notations.

4 Context Driven Index Trimming

4.1 Method Overview

As shown in Figure 3, CDIT starts with an initial retrieval (step ①) and the query will be checked that whether a similar query has been processed before (step ②). If a similar query is found by the determiner via semantic similarity search (Gao et al., 2023), the initial retrieval along with the query will be used to generate the final answer (step ③). Otherwise, CDIT employs an LLM to extract the main semantic components of the retrieved sentences and checks whether the retrieval data and the query conform with the CMDs (step ④). Retrieval results that are determined as consistent by the CMDs and the LLM will be preserved and passed to the following steps, while inconsistent results are discarded. Later in step ⑤, CDIT trims the vector index based on the LLM judgments, which enables the database to update its vector search index for better retrieval in the future. Key steps ④⑤ will be introduced in following sections.

4.2 Extracting and Comparing Components

An LLM is employed to extract and compare the subjects, predicates and objects of sentences and further judge whether the retrieval data and the query are consistent based on CMDs. Specifically, a prompt consisting of rules and instructions is designed for extraction, comparison and judgment (see Appendix B.1 for details). In the rule part of the prompt, we explain the meaning of CMDs to the LLM via natural language. In the instruction part, we ask the LLM to extract and compare the sentence components based on the CMD and decide whether the data is consistent. In our experiments, we adopt GPT-3.5-turbo as the extraction and comparison model, which provides accurate judgments and is easy to implement with good flexibility and reasonable price. Continue with Example 1, as shown in Figure 2, basic semantic components sub, pre, obj of s_1, s_2 are firstly extracted by GPT-3.5-turbo. After comparison, the LLM finds that $turn\ on$ and $turn\ off$ are dissimilar, which is denoted as $s_1[pre] \not\approx s_2[pre]$. Therefore, the CMD ϕ_1 is violated, and the LLM returns "False" which means that $s_1[sid] \not\sim s_2[sid]$.

In summary, the consistency of retrieved contexts and queries are checked in this step, where the LLM decomposes sentences into main components and serves as a comparator for matching.

4.3 Trimming indexes

We propose an index trimming algorithm (Algorithm 1) based on the **Witness Theorem** to prune incorrect indexes of the retrieved data, such that inconsistent contexts along with their corresponding

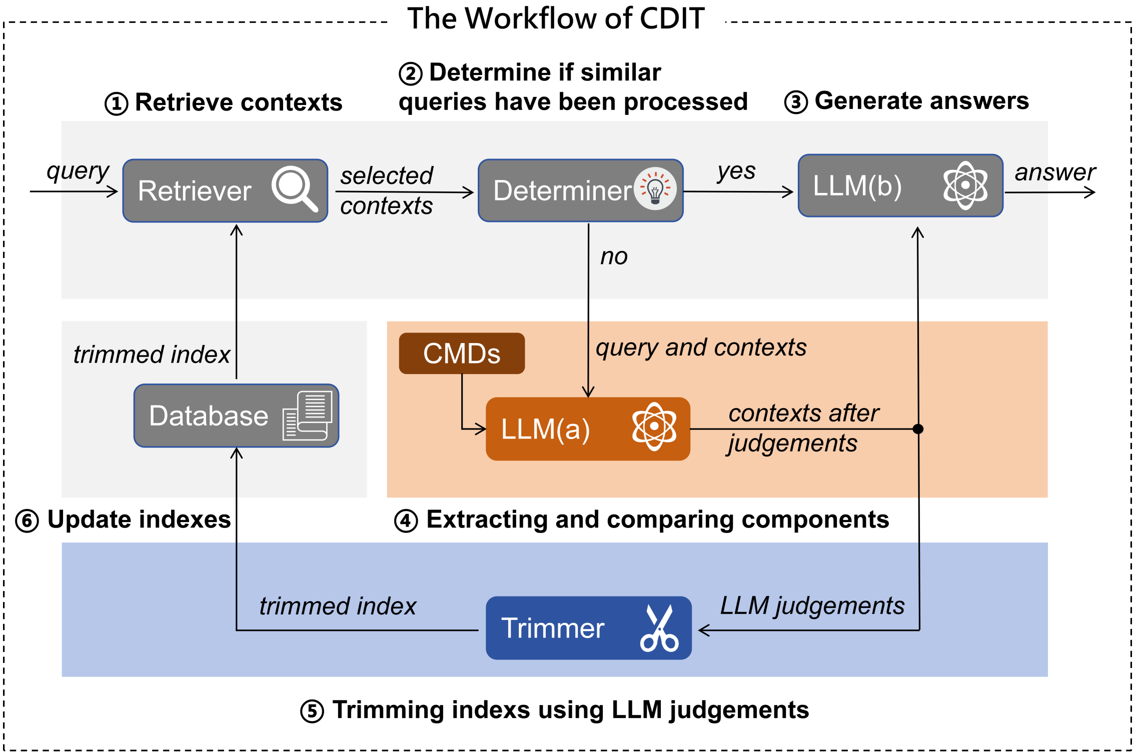


Figure 3: Overview of our mechanism. The LLM(a) represents the more advanced large-parameter language models currently, such as GPT-3.5-turbo; LLM(b) stands for LLMs with smaller parameters and easier deployment, such as Llama2-7b, playing the role of a language generator.

vectors no longer link together.

Witness Theorem identifies the contexts wrongly considered similar in the vector database.

Theorem 1 *Witness Theorem. Given a query q and sentences s_1, s_2. If $q[sid] \sim s_1[sid]$ and $q[sid] \not\sim s_2[sid]$, then the query q is a* ***witness*** *to the separation of the two sentences $s1$ and s_2.*

In other words, if the sid consistency judgment of q, s_1 and q, s_2 differs, then q witnesses the contradiction between s_1 and s_2. As a sufficient number of witnesses are collected, it can be determined that s_1 and s_2 are actually dissimilar. In that case, we modify the vector index by cutting the similarity linkage between s_1 and s_2. Algorithm 1 shows this process of trimming indexes.

To illustrate, we take IndexHNSWFlat in Faiss as an example. As shown in Figure 4, IndexHNSWFlat establishes a vector search index based on HNSW algorithm (Malkov and Yashunin, 2016), where data in the knowledge base is organized as a hierarchical similarity graph to facilitate efficient searching. From Figure 4(a), we can see that A and B are regarded as similar and connected by HNSW. However, as the LLM determines that q_1, q_2 are similar to A while dissimilar to B, and q_3 is similar to B but not A (Figure 4(b)), there has been adequate number of witnesses for A and B to separate. Thus, CDIT will cut the edge between A and B (the dashed line in Figure 4(c)) and the vector search index is trimmed.

Although the retrieved contexts are all considered similar by the vector search index, Witness Theorem assists in pruning incorrect similarity links of the retrieval in the search index. This way, the next time a similar query is encountered, the retrieved context will have better data consistency since the index has been previously modified.

5 Experiments

5.1 Experimental settings

Datasets. We verify the effectiveness of CDIT on a range of downstream tasks, including ARC-Challenge, PubHealth, PopQA and TriviaQA-unfiltered, which are well-accepted and challenging factual question benchmarks for RAG(Asai et al., 2023; Gao et al., 2023), evaluating the correctness of models. We clarify our tasks as the followings.

Multi-Choice. ARC-Challenge(Clark et al., 2018) is a multiple-choice reasoning dataset that requires far more powerful knowledge than previous

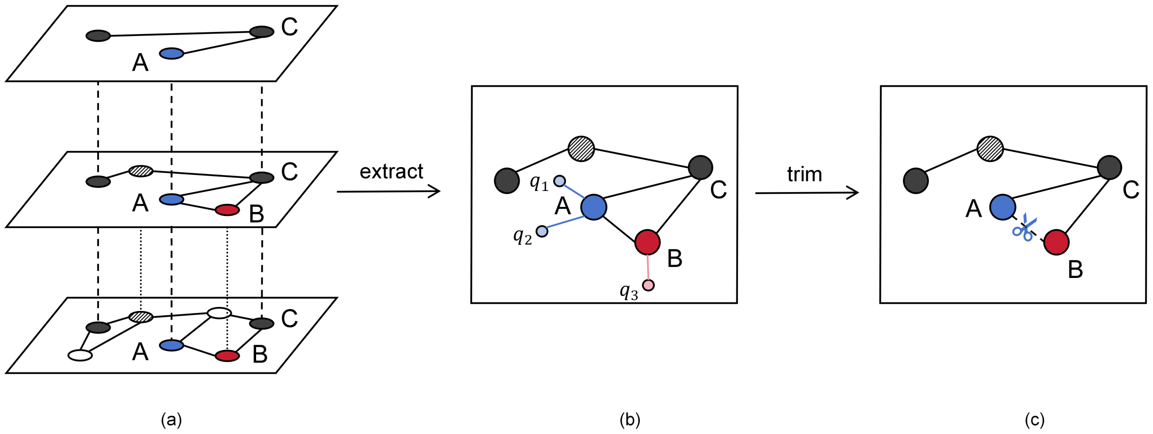


Figure 4: Diagram of HNSW indexing. A, B, and C denote the data vector and q_1, q_2, q_3 denote the query vector. (a) The diagram of HNSW structure. (b) A single-layer graph is extracted from the stereoscopic structure in (a). (c)After trimming the indexes, the relationship pointed to by the dashed line was successfully deleted.

```
Algorithm 1: Index trimming
Input: Query
for q in Query do
    $T$ ←new empty collection;
    $s_C$ ←new empty collection;
    $C$ ← Retrieve($q$);
    // Retrieve contexts similar
       with q.
    add $C$ to $s_C$;
    for s in C do
        $result$ ←Judge($s, q$);
        // Judge $sids$ with CMDs.
        if result=True then
            add $s$ to $T$;
            // Record similar
               sentences.
        else
            Accumulate($s, T$);
            // Accumulate the number
               of witnesses.
for $s_1$, $s_2$ in $s_C$ do
    if Witness($s_1, s_2$) then
        Cutoff($s_1, s_2$);
        // If the contexts have been
           witnessed a certain number
           of times, then cut off the
           link for $s_1$ and $s_2$.
```

tasks. It contains challenging questions that most retrieval-based algorithms can hardly answer correctly. Accuracy is utilized as the evaluation metric, calculated by the given ground-truth answers.

Fact-Checking. PubHealth is a fact-checking task about public health, containing 987 non-disputed factual and faked claims for evaluating the fact-check performance. Also, accuracy is utilized as the metric for the task.

Single-Hop includes two datasets. **PopQA**(Mallen et al., 2022) contains QA pairs whose questions are generated by converting a knowledge tuple (subject_entity,object_entity, relationship_type) retrieved from Wikidata. **TriviaQA-unfiltered**(Joshi et al., 2017) has complex and compositional questions, raising the need for more precise retrieval. We follow the rough matching in (Asai et al., 2023; Mallen et al., 2022) as the performance metric, where a generation is correct when the ground-truth answer is included.

Configurations in CDIT. We employ Contriever-MS MARCO(Izacard et al., 2021) as the retriever and Faiss(Douze et al., 2024) as the vector search interface. Meanwhile, zero-shot evaluations are conducted on our experiments, which describes tasks without few-shot demonstrations(Wei et al., 2021; Sanh et al., 2021). By default, the top-10 documents returned by the retriever are selected in CDIT and the official April 2018 English Wikipedia dump is used as the knowledge base. GPT-3.5-turbo is employed to extract and judge the consistency of $sids$. CMD rules ϕ_1 are used as the constraints. More details of the experiments can be found in Appendix A.

5.2 Baselines

Language models. We have tested CDIT on various baseline language models which serve as the answer generator in the RAG stage, in-

Basic Model	**Index**	**PopQA**(acc)		**TQA**(acc)		**ARC**(acc)		**Pub**(acc)		**Avg-Impro**
		orignal	CDIT	orignal	CDIT	orignal	CDIT	orignal	CDIT	
Llama-7b	IndexFlatL2	14.35	**19.02**	23.82	**32.48**	27.47	**32.94**	23.10	**27.03**	↑ 5.69
	IndexHNSWFlat	20.85	**23.94**	26.48	**32.48**	27.39	**30.97**	25.53	**29.10**	↑ 4.06
	IndexIVFFlat	20.43	**24.44**	26.74	**32.90**	32.93	**34.22**	23.71	**26.94**	↑ 3.67
Llama2-7b	IndexFlatL2	15.76	**17.85**	22.61	**30.52**	27.47	**29.35**	24.51	**26.24**	↑ 3.40
	IndexHNSWFlat	19.52	**27.61**	24.49	**32.33**	28.16	**29.27**	23.30	**26.13**	↑ 4.97
	IndexIVFFlat	20.26	**25.44**	26.45	**30.78**	30.80	**31.06**	26.34	**28.74**	↑ 3.05
Alpaca-7b	IndexFlatL2	21.85	**27.19**	33.57	**43.40**	26.45	**27.99**	56.34	**60.56**	↑ 5.23
	IndexHNSWFlat	30.44	**35.20**	37.84	**47.32**	28.16	**30.46**	56.53	**63.26**	↑ 5.82
	IndexIVFFlat	29.85	**34.02**	37.57	**44.14**	27.05	**32.00**	56.53	**63.21**	↑ 5.59
Llama3-8b	IndexFlatL2	**42.00**	41.87	39.03	**39.66**	**33.02**	32.08	46.30	**50.11**	↑ 0.80
	IndexHNSWFlat	41.79	**41.87**	38.99	**39.45**	31.22	**32.00**	43.26	**48.41**	↑ 1.62
	IndexIVFFlat	41.37	**42.20**	38.23	**39.15**	31.31	**33.53**	44.78	**49.02**	↑ 2.05
Mistral-7b	IndexFlatL2	40.12	**42.70**	60.21	**62.41**	55.29	**57.25**	21.48	**23.46**	↑ 2.18
	IndexHNSWFlat	31.11	**34.28**	57.58	**60.48**	53.13	**56.74**	21.58	**24.23**	↑ 3.08
	IndexIVFFlat	35.45	**41.87**	61.09	**62.59**	54.93	**56.48**	23.10	**29.21**	↑ 3.90
Bloomz-7b1	IndexFlatL2	24.86	**27.52**	49.71	**51.14**	40.10	**48.89**	56.84	**57.98**	↑ 3.51
	IndexHNSWFlat	23.52	**24.10**	47.71	**49.73**	43.51	**48.55**	53.19	**55.26**	↑ 2.43
	IndexIVFFlat	24.19	**29.53**	48.37	**53.29**	43.60	**48.63**	57.34	**59.21**	↑ 4.30
Falcon-7b	IndexFlatL2	28.69	**32.53**	40.02	**45.89**	20.04	**21.08**	25.32	**27.68**	↑ 3.28
	IndexHNSWFlat	20.60	**25.35**	27.47	**42.68**	21.50	**25.26**	23.91	**26.21**	↑ 6.51
	IndexIVFFlat	22.52	**29.94**	37.84	**41.71**	20.01	**21.59**	27.56	**29.54**	↑ 3.71

Table 2: Experiment results on different language models and index structure. Bold numbers indicate the best performance among models. "Avg-Impro" refers to the average improvement of CDIT of all types of datasets. **PopQA, TQA, ARC and Pub** refer to PopQA, TriviaQA-unfiltered, ARC-Challenge, and PubHealth, respectively.

cluding Llama-7b(Touvron et al., 2023a), Llama2-7b(Touvron et al., 2023b), Alpaca-7b(Dubois et al., 2023), Llama3-8b(Dubey et al., 2024), Mistral-7b(Jiang et al., 2023), Bloomz-7b1(Muennighoff et al., 2022) and Falcon-7b(Almazrouei et al., 2023) in consideration of their convenience, accessibility, and versatility. Bloomz is a Multitask Prompting Fine Tuned (MTF) version of the BLOOM(Le Scao et al., 2023), and Alpaca is replicated based on Llama, and the instruction-tuned LM adopts an official system prompt during training.

Vector search indexes. We have also tested CDIT on various representative vector similarity search indexes including IndexFlatL2, IndexHNSWFlat and IndexIVFFlat(Douze et al., 2024). Specifically, IndexFlatL2 performs Euclidean distance search on all vectors, which is the most accurate but slow in search and memory-intensive. IndexHNSWFlat is built on the Navigable Small World (NSW) graph, which provides extremely fast search at the cost of both long building time and large memory space for the index. IndexIVFFlat reduces the search space via clustering, which strikes a balance between search quality and speed. Unless otherwise specified, the default configuration for the experiment shall be IndexL2Flat.

5.3 Main Results

Table 2 compares the answer accuracy of the original RAG and CDIT, where we find the following:

CDIT works for various language models. CDIT surpasses the basic models with average accuracy improvements of 4.47%, 3.80%, 5.54%, 1.49%, 3.05%, 3.41% and 4.50% on Llama-7b, Llama2-7b, Alpaca-7b, Llama3-8b, Mistral-7b, Bloomz-7b1 and Falcon-7b, respectively, and the most significant increase reaches up to 15.21% when applying CDIT framework to Falcon-7b model on TriviaQA dataset with IndexHNSWFlat index. This is because retrieval information that is unrelated or inconsistent with the query is discarded by CDIT, which reduces the distracting inputs to LLMs.

CDIT works for different indexing methods. CDIT has on average boosted the model accuracy by 3.44%, 4.07%, and 3.75% over IndexFlatL2, IndexHNSWFlat, and IndexIVFFlat, respectively, which proves its effectiveness in modifying the original vector index. Moreover, CDIT achieves higher improvements on more coarse-grained indexing structures, such as IndexHNSWFlat.

5.4 Analysis

As CDIT is a method of pruning indexes and can be flexibly integrated with other RALMs as a functional module, it is actually an atomic component and is difficult to conduct ablation studies. Thus, we analyze the impact of hyper-parameters and CMDs on CDIT. Specific results are shown in Appendix C.

Effects of varying top-k. In order to analyze how the number of documents returned by the retriever affects the performance of CDIT, we vary the number of retrieved documents (top-k) from 5 to 10 and test on PopQA dataset for all three vector indexes with llama2-7b as the generator. As shown in Figure 5, CDIT shows improvements across various top-k, and the improvement is more significant under larger top-k. The main reason for this is that larger top-k may return more useless information, which deteriorates the performance and can be filtered out by CDIT, while smaller top-k provides little space for trimming where the performance gain is limited.

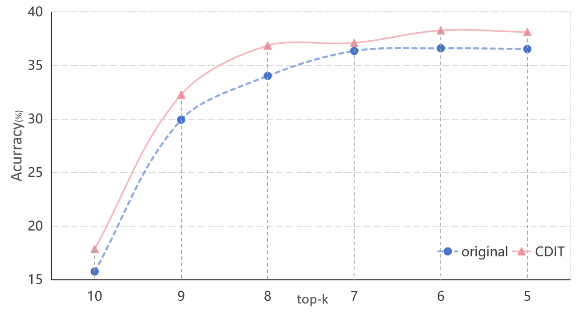


Figure 5: Top-k analysis on PopQA with Llama2-7b and IndexL2Flat index structure.

In particular, as for Llama3, a generator with strong language abilities, We test its performance with a much larger top-k to evaluate how CDIT helps resolve the retrieval information explosion. Previously, as shown in Table 2, the performance gain of applying CDIT to Llama3 is limited, where the possible reason may be that the strong language ability of Llama3 allows more contexts and better identifies irrelevant retrieval contexts independently without CDIT. In order to investigate Llama3's ability limit of processing retrieval information explosion and whether CDIT can still help to improve, we vary (top-k) from 10 to 50 and test on PopQA with Llama3 as the generator. As shown in Table 3, the performance gain of CDIT is significantly enhanced as top-k increases, which means that CDIT plays a better role when excessively large context information is fed to the LLM.

Top-k	10	20	30	40	50
original	42.20	41.03	41.20	39.86	16.51
CDIT	42.37	41.25	41.45	40.95	24.35
impro	↑ 0.17	↑ 0.22	↑ 0.25	↑ 1.09	↑ 7.84

Table 3: The performance of CDIT on Llama3 as top-k ranks from 10 to 50.

Effects of CMDs. The CMD ϕ_1 does not describe all constraints of retrieved data. In order to specify the constraints more accurately, we need to consider the relationship between other components of the sentences. For example, attributives and adverbials also constraint the consistency of the sentences, and the corresponding CMD can be written as below:

$$\begin{aligned}\phi_2 : & s_1[sid] \sim s_2[sid] \to \\ & s_1[att] \approx s_2[att] \wedge s_1[adv] \approx s_2[adv]\end{aligned}$$

where att, adv denotes the attributive and adverbial of the sentence. We add CMD ϕ_2 to the comparing step, such that the consistency of sid requires simultaneous satisfaction of both CMD ϕ_1 and CMD ϕ_2. As shown in Figure 6, different CMDs have an influence on the accuracy of CDIT. Thus, it is desirable to investigate the optimized combinations of various CMDs in future.

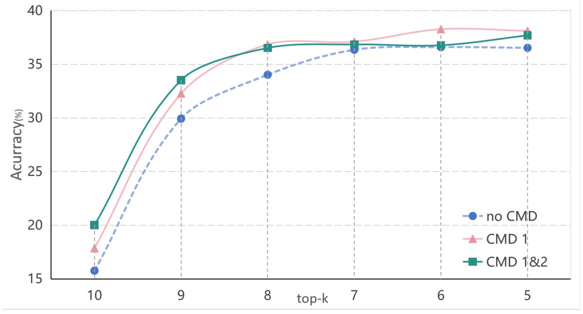


Figure 6: CMDs analysis on PopQA with Llama2-7b and IndexL2Flat index structure.

Integration with other RALMs. We analyze the performance of CDIT when integrated with enhanced RAG models. As CDIT directly modifies the indexes of database, it has strong flexibility and can be easily integrated with existing RAG models to improve the answer quality. We integrate CDIT with Self-RAG(Asai et al., 2023), which is a refined RAG model by improving knowledge retrieval, and evaluate the accuracy of the answers. Llama2-7b is used in this test, and other settings are unchanged. As shown in Figure 7, CDIT consistently improves the accuracy of answers after integrating with self-rag, where the average increase is 3.62%. This shows that based on existing state-of-the-art RAG models that mainly improve 12.3%, CDIT could further enhance the performance by refining data quality.

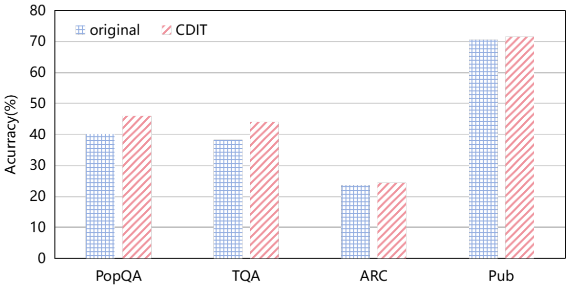


Figure 7: Self-RAG Integration analysis on IndexL2Flat with Llama2-7b.

LLM costs. We analyze the costs of utilizing LLMs in CDIT, which has been validated as acceptable. Time to call LLMs is considered to assess the cost with results as follows. We determine the average time expenditure for LLM invocations through multiple experimental measurements across various language models and indexes. As shown in Table 4, even with the maximum number of calls to LLMs, without considering repeated queries, the time did not exceed 6 minutes. Compared to the time required for the inference of language model itself, we consider this cost to be reasonable.

Dataset	PopQA	TQA	ARC	Pub
Time(s)	59.94	351.05	58.62	29.61

Table 4: Time to call LLMs in CDIT.

Case Study. We use the scenario depicted in Figure 1 as an example to explain the working principle of CDIT, as shown in Figure 8. In this scenario, the input query is:

q: "*Who turned on the radio?*",

and the two relevant retrieved contexts are:

s_1: "*Mary turned off the radio.*"

s_2: "*Jack turned on the radio.*"

In conventional approaches, the basic RALMs will answer "*Mary.*" due to the first retrieved contexts, while they actually give a wrong answer.

For CDIT, the query q and contexts s_1, s_2 are first sent to GPT-3.5-turbo. Given that $s_1[pre]$ refers to "*turn off*", $s_2[pre]$ refers to "*turn on*", and $s_1[pre] \not\approx s_2[pre]$, the LLM deduces that $s_1[sid] \not\sim s_2[sid]$ according to CMD ϕ_1. Hence these two retrieved contexts are inconsistent. As a result, context s_1 is discarded and s_2 is reserved. Meanwhile, the link between the vectors of s_1 and s_2 on the index structure is cut off. At this point, the language model only receives query and context s_2. According to context s_2, RALMs with CDIT can answer "*Jack.*" correctly. Similarly, next time when the model encounters a query:

q_1: "*Please tell me who turned on the radio.*"

which resembles q, the retrieved contexts will only be s_2 due to the trimmed index structure.

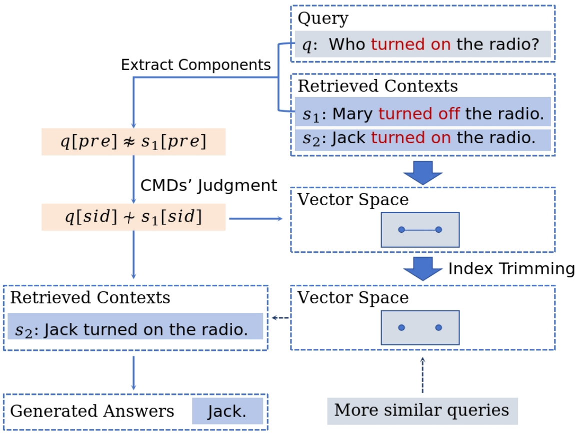

Query
q: Who turned on the radio?
Extract Components
Retrieved Contexts
s_1: Mary turned off the radio.
s_2: Jack turned on the radio.
$q[pre] \not\approx s_1[pre]$
CMDs' Judgment
$q[sid] \not\sim s_1[sid]$
Vector Space
Index Trimming
Retrieved Contexts
s_2: Jack turned on the radio.
Vector Space
Generated Answers
Jack.
More similar queries

Figure 8: Case study for CDIT.

6 Conclusion

Our study presents a Context-Driven Index Trimming (CDIT) framework, which enhances the accuracy of RALMs by focusing on data quality within vector databases. Experiments show an average 3.75% increase in accuracy, highlighting the robustness of CDIT across models and indexing methods. While challenges in long text handling and reliance on LLMs are noted, the adaptability and potential of CDIT suggest a bright future in NLP.

Acknowledgments

This work is partially supported by NSFC No. 62302503, NUDT Youth Independent Innovation Science Fund Project Grant No. ZK23-15, and the Open Research Fund from State Key Laboratory of High Performance Computing of China Grant No. 202401-09.

Limitations

Limitations still exist in our work. Firstly, long texts may cause a subpar performance of CDIT. A possible reason is the complexity of long texts, making it difficult to extract and compare basic semantic components. What's more, the basic components mentioned above may be incompetent to represent long text, resulting in error judgements. Secondly, the CMDs in this article are proposed manually based on our experience, so they may not be thorough and accurate enough to describe all the constraints of retrieved data. Regarding the two limitations, we plan to conduct further research on the mining of CMDs in the future to enable it to represent the constraints of various types of text more accurately. Finally, over-reliance on GPT is another potential problem. The extraction and comparison of components need online LLMs, which may be a trouble for a completely offline environment with considerable costs. We may subsequently consider employing proper lexical analyzers such as dependency parsing(Manning et al., 2014), Stanza(Qi et al., 2020), etc, to mitigate this issue.

References

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, et al. 2023. The falcon series of open language models. *arXiv preprint arXiv:2311.16867*.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2023. Self-RAG: Self-reflective retrieval augmented generation. In *NeurIPS 2023 Workshop on Instruction Tuning and Instruction Following*.

Jon Louis Bentley. 1975. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18:509–517.

Vladimir Blagojevic. Enhancing rag pipelines in haystack: Introducing diversityranker and lostinthemiddleranker. `https://haystack.deepset.ai/blog/enhancing-rag-pipelines-in-haystack`.

Luca Cagliero and Alessandro Fiori. 2013. Discovering generalized association rules from twitter. *Intell. Data Anal.*, 17:627–648.

Yang Cao, Wenfei Fan, Wenzhi Fu, Ruochun Jin, Weijie Ou, and Wenliang Yi. 2023. Extracting graphs properties with semantic joins. *2023 IEEE 39th International Conference on Data Engineering (ICDE)*, pages 2262–2275.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *ArXiv*, abs/1803.05457.

E. F. Codd. 1971. Further normalization of the data base relational model. *Research Report / RJ / IBM / San Jose, California*, RJ909.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. 2024. The faiss library.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Yann Dubois, Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy Liang, and Tatsunori Hashimoto. 2023. Alpacafarm: A simulation framework for methods that learn from human feedback. *ArXiv*, abs/2305.14387.

Fredrik Erlandsson, Piotr Bródka, Anton Borg, and Henric Johnson. 2016. Finding influential users in social media using association rule learning. *ArXiv*, abs/1604.08075.

Wenfei Fan, Hong Gao, Xibei Jia, Jianzhong Li, and Shuai Ma. 2011. Dynamic constraints for record matching. *The VLDB Journal*, 20:495–520.

Wenfei Fan, Floris Geerts, Xibei Jia, and Anastasios Kementsietsidis. 2008. Conditional functional dependencies for capturing data inconsistencies. *ACM Trans. Database Syst.*, 33:6:1–6:48.

Wenfei Fan, Floris Geerts, Jianzhong Li, and Ming Xiong. 2010. Discovering conditional functional dependencies. *IEEE Transactions on Knowledge and Data Engineering*, 23(5):683–698.

Wenfei Fan, Liang Geng, Ruochun Jin, Ping Lu, Resul Tugay, and Wenyuan Yu. 2022. Linking entities across relations and graphs. *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, pages 634–647.

Wenfei Fan, Ruochun Jin, Muyang Liu, Ping Lu, Chao Tian, and Jingren Zhou. 2020. Capturing associations in graphs. *Proceedings of the VLDB Endowment*, 13:1863 – 1876.

Wenfei Fan, Xin Wang, Yinghui Wu, and Jingbo Xu. 2015. Association rules with graph patterns. *Proc. VLDB Endow.*, 8:1502–1513.

Wenfei Fan, Yinghui Wu, and Jingbo Xu. 2016. Adding counting quantifiers to graph patterns. *Proceedings of the 2016 International Conference on Management of Data*.

Yixiang Fang, Reynold Cheng, Siqiang Luo, and Jiafeng Hu. 2016. Effective community search for large attributed graphs. *Proc. VLDB Endow.*, 9:1233–1244.

Jiazhan Feng, Chongyang Tao, Xiubo Geng, Tao Shen, Can Xu, Guodong Long, Dongyan Zhao, and Daxin Jiang. 2023. Synergistic interplay between search and large language models for information retrieval.

Luis Galárraga, Christina Teflioudi, Katja Hose, and Fabian M. Suchanek. 2013. Amie: association rule mining under incomplete evidence in ontological knowledge bases. *Proceedings of the 22nd international conference on World Wide Web*.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Qianyu Guo, Meng Wang, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *ArXiv*, abs/2312.10997.

Antonin Guttman. 1984. R-trees: a dynamic index structure for spatial searching. In *ACM SIGMOD Conference*.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2023. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ArXiv*, abs/2311.05232.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Unsupervised dense information retrieval with contrastive learning. *Trans. Mach. Learn. Res.*, 2022.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane A. Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2022. Few-shot learning with retrieval augmented language models. *ArXiv*, abs/2208.03299.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *ArXiv*, abs/1705.03551.

Reza Karegar, Parke Godfrey, Lukasz Golab, Mehdi Kargar, Divesh Srivastava, and Jaroslaw Szlichta. 2021. Efficient discovery of approximate order dependencies. In *Proceedings of the 24th International Conference on Extending Database Technology, EDBT 2021, Nicosia, Cyprus, March 23 - 26, 2021*, pages 427–432. OpenProceedings.org.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Yu Wu, Sergey Edunov, Danqi Chen, and Wen tau Yih. 2020. Dense passage retrieval for open-domain question answering. *ArXiv*, abs/2004.04906.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2023. Bloom: A 176b-parameter open-access multilingual language model.

Patrick Lewis, Ethan Perez, Aleksandara Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kuttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *ArXiv*, abs/2005.11401.

Yang Li and Tao Yang. 2018. Word embedding for understanding natural language: A survey.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023a. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Yi Liu, Lianzhe Huang, Shicheng Li, Sishuo Chen, Hao Zhou, Fandong Meng, Jie Zhou, and Xu Sun. 2023b. Recall: A benchmark for llms robustness against external counterfactual knowledge. *ArXiv*, abs/2311.08147.

Yury Malkov, Alexander Ponomarenko, Andrey Logvinov, and Vladimir Krylov. 2014. Approximate nearest neighbor algorithm based on navigable small world graphs. *Inf. Syst.*, 45:61–68.

Yury Malkov and Dmitry A. Yashunin. 2016. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42:824–836.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Hannaneh Hajishirzi, and Daniel Khashabi. 2022. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Annual Meeting of the Association for Computational Linguistics*.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Annual Meeting of the Association for Computational Linguistics*.

Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng-Xin Yong, Hailey Schoelkopf, et al. 2022. Crosslingual generalization through multitask finetuning. *arXiv preprint arXiv:2211.01786*.

Mohammad Hossein Namaki, Yinghui Wu, Qi Song, Peng Lin, and Tingjian Ge. 2017. Discovering graph temporal association rules. *Proceedings of the 2017*

ACM on Conference on Information and Knowledge Management.

Harold W. Noonan. 2015. Relative identity. *Philosophical Investigations*, 38(1-2):52 – 71. Cited by: 3.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2021. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*.

R Santhya, S Latha, S Balamurugan, and S Charanyaa. 2014. Further investigations on strategies developed for efficient discovery of matching dependencies. *Int. J. Innov. Res. Comput. Commun. Eng.(An ISO 3297: 2007 Certified Organization)*, 3:18998–19004.

Philipp Schirmer, Thorsten Papenbrock, Ioannis Koumarelas, and Felix Naumann. 2020. Efficient discovery of matching dependencies. *ACM Transactions on Database Systems (TODS)*, 45(3):1–33.

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Rich James, Mike Lewis, Luke Zettlemoyer, and Wen tau Yih. 2023. Replug: Retrieval-augmented black-box language models. *ArXiv*, abs/2301.12652.

Qi Song, Yinghui Wu, and Xin Dong. 2016. Mining summaries for knowledge graph search. *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 1215–1220.

Shaoxu Song and Lei Chen. 2009. Discovering matching dependencies. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 1421–1424.

Dan C. Stefanescu, Rajendra Banjade, and Vasile Rus. 2014. A sentence similarity method based on chunking and information content. In *Conference on Intelligent Text Processing and Computational Linguistics*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models. *ArXiv*, abs/2302.13971.

Hugo Touvron, Louis Martin, Kevin R. Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Daniel M. Bikel, Lukas Blecher, Cristian Cantón Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony S. Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel M. Kloumann, A. V. Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, R. Subramanian, Xia Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zhengxu Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models. *ArXiv*, abs/2307.09288.

Yile Wang, Peng Li, Maosong Sun, and Yang Liu. 2023. Self-knowledge guided retrieval augmentation for large language models. *ArXiv*, abs/2310.05002.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Zichun Yu, Chenyan Xiong, Shih Yuan Yu, and Zhiyuan Liu. 2023. Augmentation-adapted retriever improves generalization of language models as generic plug-in. In *Annual Meeting of the Association for Computational Linguistics*.

A Implementation

A.1 Datasets Details

For all the datasets, we set the maximum new token number to 100 tokens. For PopQA and ARC-Challenge, we retrieve top-k documents from the 2018 English Wikipedia. For TriviaQA, we additionally retrieve documents using Google Programmable Search.

A.2 Query Rewriting.

After the indexes have been trimmed, we rewrite the query to test the performance. We follow the approach of Feng et al., combining the original query and the top-1 retrieval document to form a new query. Therefore, the new query sent to the language generator is shown in Table 5.

A.3 Configuration for self-rag

We follow the method of Asai et al., employing the pre-trained weights[2]. The integration experiment uses CDIT first to modify indexes, then self-rag is employed to test the overall quality. The default configuration is employed for everything else.

[2] https://huggingface.co/selfrag/selfrag_llama2_7b

B Prompting

Based on time and performance considerations, we choose OpenAI[3] GPT-3.5-turbo API as the employed LLM. CDIT primarily utilizes LLM in two parts, with detailed prompts as follows.

B.1 Extraction and Comparison.

Prompting is used in GPT-3.5 to achieve the functionality of extracting components and utilizing the CMDs for comparison. Prompt 1 and 2 in Table 6 are utilized respectively for trimming with CMD ϕ_1 and CMD ϕ_1 & ϕ_2.

B.2 Prompt for Answers

After retrieval, we combine the retrieval contexts and other instructions to prompt the language generator for the final answers. For ARC-Challenge, we follow Asai et al., designing task instructions shown in Table 8. For other tasks, we do not design additional task instructions. The final prompts are shown in Table 7.

C Experiment Results

Table 9 shows the specific results of the top-k experiment. In this experiment, Llama2-7b, IndexL2Flat are employed as the language generator and the indexing structure. Top-k varies from 10 to 5. Table 10 shows the results of the cmd expended experiment. In this experiment, Llama2-7b, IndexL2Flat are employed as the language generator and the indexing structure. CMD ϕ_1, ϕ_2 are employed differently. Table 11 shows the results of the integration experiment between CDIT and Self-RAG(Asai et al., 2023). CDIT is employed firstly to enhance the data quality of retrieved contexts. Then, the model was trained and tested using the Self-RAG approach. Llama2-7b is employed as the language generator in this experiment, and top-k is 10.

[3] https://platform.openai.com/docs/api-reference

Original Query: What is Henry Feilden's occupation?
Retrieved Documents: [1] Henry Feilden (Conservative politician) Henry Master Feilden (21 February 1818 – 5 September 1875) was an English Conservative Party politician. [2] Henry Wemyss Feilden Colonel. Henry Wemyss Feilden, CB (6 October 1838 – 8 June 1921) was a British Army officer, Arctic explorer and naturalist.
New Query Structure: Given a question [original query] and its possible answering passages [top-1 retrieved documents], Now give a possible answer.
New Query: Given a question [What is Henry Feilden's occupation?] and its possible answering passages [Henry Feilden (Conservative politician) Henry Master Feilden (21 February 1818 – 5 September 1875) was an English Conservative Party politician.], Give a possible answer.

Table 5: Construction of new queries in query rewriting

CMD: CMD ϕ_1
Prompt 1: You are a cautious language assistant. ###[Rules] Here are some language rules: # If the two sentences can be identified as similar, then the subjects, predicates and objects of the two sentences are similar. Be especially mindful of predicate phrases that appear similar but actually have opposite meanings, which make sentences dissimilar. ###[Instructions] Are the following statements similar with the question? Just say True if they are; otherwise just say False. Only output one word.
Sentences: He turned on the radio. He turned off the radio.
Answer: False. ✔
CMD: CMD ϕ_1&ϕ_2
Prompt 2: You are a cautious language assistant. ###[Rules] Here are some language rules: # If the two sentences can be identified as similar, then the subjects, verbs and objects of the two sentences are similar. Be especially mindful of verb phrases that appear similar but actually have opposite meanings, which make sentences dissimilar. # If the two sentences can be identified as similar, then the adverbials and attributives of the two sentences are similar. ###[Instructions] Are the following statements similar with the question? Just say True if they are; otherwise, just say False. Only output one word.
Sentences: He turned on the radio at five. He turned on the radio at six.
Answer: False. ✔

Table 6: Prompts used in extraction and comparison

Query: What is Henry Feilden's occupation?
Retrieved Documents: [1] Henry Feilden (Conservative politician) Henry Master Feilden (21 February 1818 – 5 September 1875) was an English Conservative Party politician. [2] Henry Wemyss Feilden Colonel Henry Wemyss Feilden, CB (6 October 1838 – 8 June 1921) was a British Army officer, Arctic explorer and naturalist.

Prompt Structure:
###Background: {retrieved documents}
###Instruction: {query+task instructions}
###Response:

Prompt: ###Background: {[1] Henry Feilden (Conservative politician) Henry Master Feilden (21 February 1818 – 5 September 1875) was an English Conservative Party politician. [2] Henry Wemyss Feilden Colonel Henry Wemyss Feilden, CB (6 October 1838 – 8 June 1921) was a British Army officer, Arctic explorer and naturalist.} ###Instruction: {What is Henry Feilden's occupation?} ###Response:

Table 7: Prompts for generating answers.

Task Instruction
Given four answer candidates, A, B, C and D, choose the best answer choice. Please answer with the capitalized alphabet only, without adding any extra phrase or period.

Table 8: Task instruction for ARC-Challenge.

Index	Method	**Top-k**						**Avg-Impro**
		10	9	8	7	6	5	
IndexL2Flat	original	15.76	29.94	34.03	36.36	36.61	36.54	
	CDIT	**17.85**	**32.28**	**36.86**	**37.11**	**38.28**	**38.12**	↑ 1.88
	Impro.	↑ 2.09	↑ 2.34	↑ 2.83	↑ 0.75	↑ 1.67	↑ 1.58	
IndexHNSWFlat	original	19.52	30.52	34.61	34.52	35.69	34.19	
	CDIT	**27.61**	**34.62**	**36.03**	**36.61**	**36.11**	**35.11**	↑ 2.84
	Impro.	↑ 7.09	↑ 4.10	↑ 1.42	↑ 1.09	↑ 0.42	↑ 0.92	
IndexIVFFlat	original	20.26	31.35	34.78	35.94	**36.86**	**37.78**	
	CDIT	**25.44**	**35.03**	**35.94**	**37.86**	36.20	37.61	↑ 1.85
	Impro.	↑ 5.18	↑ 3.68	↑ 1.16	↑ 1.92	↓ 0.66	↓ 0.17	

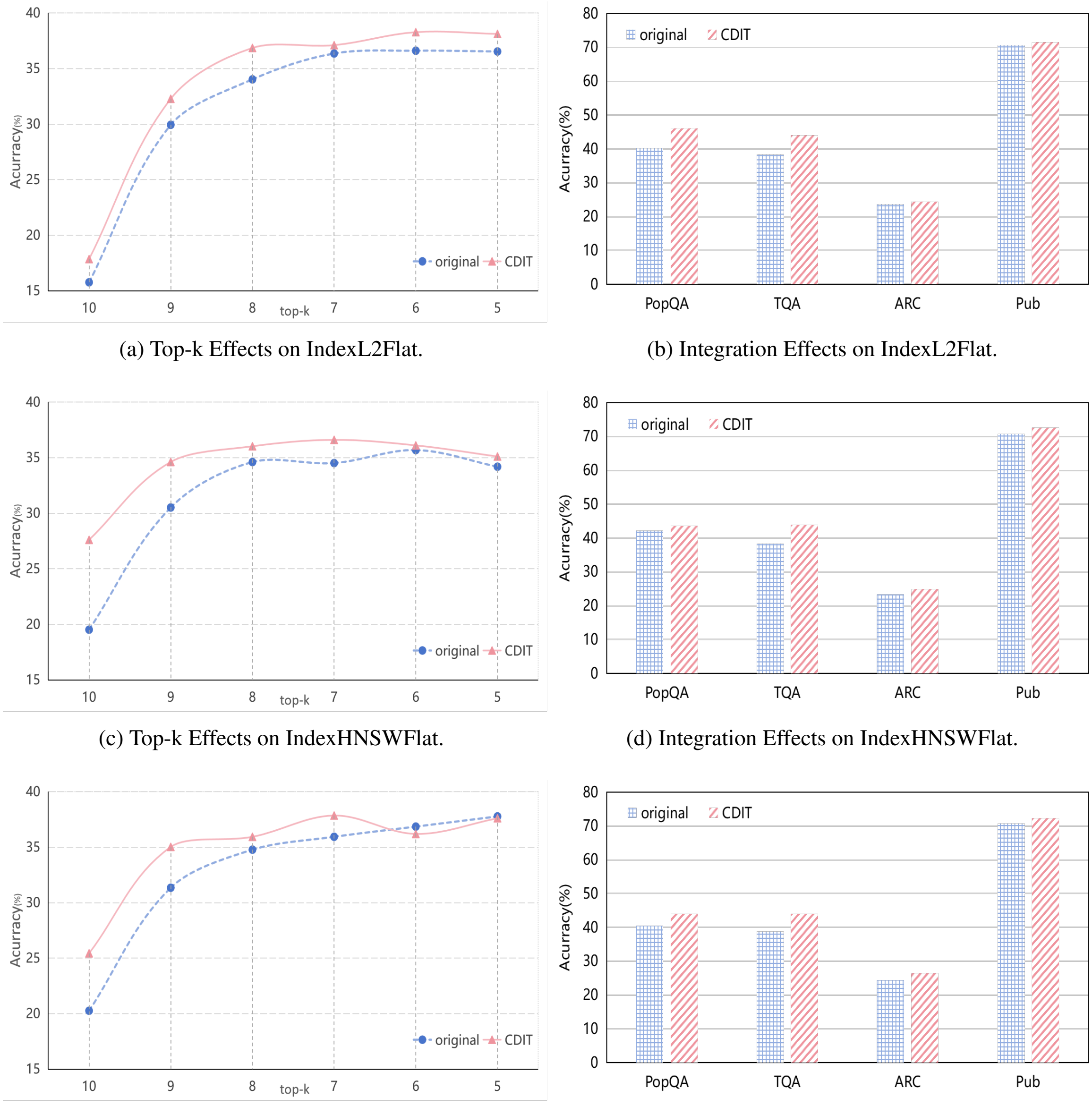
Table 9: Changes in accuracy for models with and without CDIT on PopQA as the top-k parameter varies.

top-k	10	9	8	7	6	5
no CMD	15.76	29.94	34.03	36.36	36.61	36.54
CMD ϕ_1	17.85	32.28	36.86	37.11	38.28	38.12
CMD ϕ_1&ϕ_2	20.02	33.53	36.53	36.86	36.78	37.70

Table 10: Accuracy of CDIT with different CMDs.

Index	Method	PopQA	TQA	ARC	Pub	*Avg.*
IndexL2Flat	original	40.12	38.30	23.63	70.62	43.17
	CDIT	**46.04**	**44.02**	**24.40**	**71.53**	**46.50** ↑ 3.33
IndexHNSWFlat	original	41.11	38.29	22.36	70.72	43.37
	CDIT	**43.54**	**43.80**	**24.91**	**72.68**	**46.24** ↑ 2.87
IndexIVFFlat	original	40.45	38.82	24.48	70.72	43.62
	CDIT	**44.04**	**44.04**	**26.37**	72.34	**46.70** ↑ 3.08

Table 11: The performance of CDIT integrated with self-rag on three datasets with top-k being 10.



(a) Top-k Effects on IndexL2Flat.

(b) Integration Effects on IndexL2Flat.

(c) Top-k Effects on IndexHNSWFlat.

(d) Integration Effects on IndexHNSWFlat.

(e) Top-k Effects on IndexIVFFlat.

(f) Integration Effects on IndexIVFFlat.

Figure 9: Additional experiments of top-k and integration with Self-RAG on Llama2-7b.