Intrinsic Evaluation of RAG Systems for Deep-Logic Questions*

Junyi (Edward) Hu[a], You Zhou[b] and Jie Wang[c]
Miner School of Computer & Information Sciences, University of Massachusetts, Lowell, MA, U.S.A.
{junyi_hu, you_zhou1}@student.uml.edu, jie_wang@uml.edu

Keywords: Retrieval Augmented Generation, Logical-Relation Correctness Ratio, Overall Performance Index.

Abstract: We introduce the Overall Performance Index (OPI), an intrinsic metric to evaluate retrieval-augmented generation (RAG) mechanisms for applications involving deep-logic queries. OPI is computed as the harmonic mean of two key metrics: the Logical-Relation Correctness Ratio and the average of BERT embedding similarity scores between ground-truth and generated answers. We apply OPI to assess the performance of LangChain, a popular RAG tool, using a logical relations classifier fine-tuned from GPT-4o on the RAG-Dataset-12000 from Hugging Face. Our findings show a strong correlation between BERT embedding similarity scores and extrinsic evaluation scores. Among the commonly used retrievers, the cosine similarity retriever using BERT-based embeddings outperforms others, while the Euclidean distance-based retriever exhibits the weakest performance. Furthermore, we demonstrate that combining multiple retrievers, either algorithmically or by merging retrieved sentences, yields superior performance compared to using any single retriever alone.

1 INTRODUCTION

A RAG system typically consists of two major components: Indexing and Retrieval. The former is responsible for indexing a reference text document before any queries are made to it. The latter is responsible for retrieving relevant data from the indexed document in response to a query and passing that information, along with the query, to a large language model (LLM) to generate an answer. The Retrieval component is typically a framework that supports a variety of retrieval methods, each referred to as a retriever.

To assess the effectiveness of a retriever in uncovering the logical relationship for an answer to a query with respect to the reference document, we introduce the Overall Performance Index (OPI). This metric measures both the correctness of the answers generated by an LLM and the accuracy of the logical relations produced by a classifier. The OPI is calculated as the harmonic mean of the BERT embedding similarity between ground-truth and generated answers, and the logical-relation correctness ratio.

To demonstrate the effectiveness of the OPI metric, we use the RAG-Dataset-12000 provided by Hugging Face (D.H., 2024) as the training and testing dataset. We fine-tune GPT-4o to construct a classifier to generate logical relations between a query and an answer, with respect to the reference document. We then evaluate LangChain (LangChain, 2024), a popular RAG tool, with seven common retrievers, extracting relevant sentences from the reference document for each query. Using GPT-4o as the underlying LLM, we generate an answer to the query and use the fine-tuned GPT-4o classifier to generate a logical relation.

To rank retrievers, we calculate the average OPI score across all 13 logical relations provided in RAG-Dataset-12000. We then use OPI to analyze the strengths and weaknesses of individual retrievers. Moreover, we demonstrate that several variations of combining multiple retrievers, either algorithmically or by merging retrieved sentences, outperform a single retriever alone.

2 PRELIMINARIES

The technique of RAG was introduced by Lewis et al. (2020) (Lewis et al., 2020) a few years before the widespread adoption of LLMs. The performance of a RAG system relies on the quality of the underlying retriever and the ability of the underlying LLM.

[a] https://orcid.org/0000-0001-8524-0123
[b] https://orcid.org/0009-0005-0919-5793
[c] https://orcid.org/0000-0003-1483-2783
* This work was supported in part by Librum Technologies, Inc.

Hu, J., Zhou, Y. and Wang, J.
Intrinsic Evaluation of RAG Systems for Deep-Logic Questions.
DOI: 10.5220/0013070300003838

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 489-496
ISBN: 978-989-758-716-0; ISSN: 2184-3228

LangChain is a popular RAG tool, which divides a reference document into overlapping text chunks of equal size. The suffix of each chunk overlaps with the prefix of the next.

To the best of our knowledge, no previous research has comprehensively evaluated the performance of RAG systems in the context of deep-logic question answering.

Given below are seven common sentence retrievers supported by LangChain:

DPS (dot-product similarity) converts a query and a text chunk as BERT-based (Devlin et al., 2018) embedding vectors and compute their dot product as a similarity score. It returns k chunks with the highest scores to the query. (DPS in LangChain is referred to as Cosine Similarity.)

kNN (k-Nearest Neighbors) in LangChain is the normalized dot-product similarity by the L2-norm, which is widely referred to as the cosine similarity. It returns k chunks with the highest cosine similarity scores to the query.

BM25 (Robertson and Zaragoza, 2009) is a probabilistic information retrieval model that ranks documents based on the term frequency in a chunk and the inverse chunk frequency in the reference document. Let q be a query, T a chunk of text, $f(t_i,T)$ the frequency of term t_i in T, $|T|$ the size of T, avgTL the average chunk length, N the total number of chunks, and $n(t_i)$ the number of chunks that contain t_i. Then $\mathrm{BM25}(q,T)$ is defined by

$$\mathrm{BM25}(q,T) = \sum_{i=1}^{n} \ln\left(\frac{N-n(t_i)+0.5}{n(t_i)+0.5}+1\right)\cdot \frac{(f(t_i,T)\cdot(\kappa+1))}{f(t_i,T)+\kappa\cdot(1-b+b\cdot\frac{|T|}{\mathrm{avgTL}})},$$

where κ and b are parameters. Return k chunks of text with the highest BM25 scores to the query.

SVM (Support Vector Machine) (Cortes and Vapnik, 1995) is a supervised learning model that finds the hyperplane that best separates data points in a dataset. To use SVM as a retriever, first represent each chunk of text as a feature vector. This can be done using word embeddings, TF-IDF, or any other vectorization method. Then use the labeled dataset to train an SVM model. Convert the query into the same feature vector space as the chunks. Apply the SVM model to the query vector to produce a score that indicates how similar the query is to each chunk. Extract k chunks with the highest scores.

TF-IDF (Sammut and Webb, 2011) measures the importance of a word in a chunk of text relative to the set of chunks in the reference document, combining term frequency and inverse chunk frequency. In particular,

$$\text{TF-IDF}(t,T) = \mathrm{TF}(t,T)\times\mathrm{IDF}(t),$$

where t is a term, T is a chunk, and $\mathrm{IDF}(t)$ is the inverse chunk frequency of t. Given a query q, select k chunks with the highest $\text{TF-IDF}(q,T)$ values.

MMR (Carbonell and Goldstein, 1998) is a retrieval algorithm that balances relevance and diversity in the selection of k chunks. It iteratively selects chunks that are both relevant to the query and minimally redundant with respect to the chunks already selected.

EDI (Euclidean Distance) (Bishop, 2006) measures the straight-line distance between a query and a chunk, represented in bag-of-words vectors. Return k chunks with the shortest distance to the query.

A data point in RAG-Database-12000 contains the following attributes: 'context', 'question', 'answer', 'retrieved_sentences', 'logical_relation', where 'context' is the reference document. There are thirteen categories of logical reasoning in the dataset. Their names, abbreviations, descriptions, and the distribution of counts are presented in Table 1. All but the last category involve deep logical reasoning, meaning that arriving at the correct answer requires complex, multi-step processes involving multiple concepts, facts, or events extracted from the content. The table includes eleven specific types of deep reasoning, with an additional category for general deep reasoning, referred to as multi-hop reasoning.

3 OVERALL PERFORMANCE INDEX

Let A and LR denote, respectively, the ground-truth answer and logical relation to the question with respect to the question Q, the context C, and the retrieved sentences S. Let A' and LR' denote, respectively, the answer and the logical relation generated by a RAG system with an LLM. We represent A and A' using BERT embeddings and compute the cosine similarity of the embeddings.

For a given dataset D with respect to a particular logical relation LR, let $\mathrm{BERTSim}_D$ denote the average BERT similarity scores of all (A,A') pairs and LRCR_D (logical-relation correctness ratio) denote the proportion of data points where the predicted logical relation matches LR. Namely,

$$\mathrm{LRCR}_D = \frac{|\{d\in D \mid LR = LR'\}|}{|D|} \quad (1)$$

The OPI for dataset D is defined by the following parameterized harmonic mean of $\mathrm{BERTSim}_D$ and LRCR_D, similar to defining the F-measure (Lewis and Gale, 1994).

Table 1: Information of logical relations.

Logical Relation	Description	Count	Total
Adversarial (ADV)	The answer involves opposing perspectives or arguments.	156	5,080
Analogical (ANA)	The answer is based on similarities between different things or situations.	116	
Causal (CAU)	The answer is based on cause-and-effect relationships.	2,477	
Comparative (COM)	The answer compares and contrasts different items.	129	
Conditional (CON)	The answer is based on conditions or if-then statements.	161	
Deductive (DED)	The answer is a conclusion of multiple statements	140	
Fuzzy (FUZ)	The answer is a generalizing statement.	106	
Inductive (IND)	The answer involves uncertain or imprecise information.	152	
Multi-hop (MUH)	The answer involves multiple steps or connections.	198	
Predictive (PRE)	The answer makes predictions.	151	
Spatial (SPA)	The answer involves relationships based on locations.	1,077	
Temporal (TEM)	The answer involves relationships based on time.	217	
Direct Matching	The answer is based on a straightforward match of the extracted content.	6,920	6,920

$$\mathrm{OPI}(\beta)_D = \frac{(1+\beta^2)\cdot \mathrm{BERTSim}_D \cdot \mathrm{LRCR}_D}{(\beta^2 \cdot \mathrm{BERTSim}_D) + \mathrm{LRCR}_D}. \quad (2)$$

$\mathrm{OPI}(1)_D$ weighs answer accuracy and logical relation accuracy equally. $\mathrm{OPI}(\beta)_D$ weighs answer accuracy more heavily when $\beta > 1$ (e.g., $\beta = 2$), and weighs logical relation accuracy more heavily when $\beta < 1$ (e.g., $\beta = 0.5$).

When there is no confusion in the context, the subscript D is omitted. Denote $\mathrm{OPI}(1)$ as OPI-1, $\mathrm{OPI}(2)$ as OPI-2, and $\mathrm{OPI}(0.5)$ as OPI-0.5.

In addition to BERTSim, other metrics may be used to measure the similarity between the generated answer and the ground-truth answer, such as Hugging Face's MoverScore, as applied in the study of content significance distributions of text blocks in a document (Zhou and Wang, 2023). We choose BERTSim because MoverScore uses IDF to compute word weights, which is better suited for extractive answers but less appropriate for generative answers produced by LLMs.

Experimental results show that the BERTSim metric aligns well with the outcomes of extrinsic comparisons of the ground-truth answers with the generated answers (see Section 4.2 for details).

In what follows, we will use OPI-1 as the default intrinsic measure to study the performance of RAG systems for answering deep-logic questions.

4 EVALUATION

As seen in Table 1, the data points in RAG-Dataset-12000 are unevenly distributed across the 13 logical relations, with significant disparities, such as only 106 data points in Fuzzy Reasoning compared to 6,920 data points in Direct Matching. To fine-tune GPT-4o and construct a classifier for identifying logical relations, a balanced dataset is preferred. To achieve this, we randomly select 100 data points from each logical relation category, forming a new dataset called RAG-QA-1300 that consists of 1,300 data points. This dataset is then split with an 80-20 ratio to create a training set and a test set.

Fine-tuning was performed by combining the context, question, and answer from each data point into a cohesive input text, labeled with its corresponding logical relation. The process involved approximately 800 training steps, resulting in a validation loss of 10^{-4}. This specific checkpoint was selected for its optimal performance.

The fine-tuned GPT-4o classifier for logical relations significantly improves the accuracy to 75.77% on the test set, compared to 49.23% when using GPT-4o out-of-the-box without fine-tuning.

We used LangChain with the seven common retrievers mentioned in Section 2. We used GPT-4o to generate answers and the fine-tuned GPT-4o classifier to generate logical relations. LangChain supports a wide range of retrievers and allows for the seamless integration of pre-trained LLMs.

4.1 Intrinsic Evaluation

We set the chunk size to 100 (words) with a chunk overlap of 20 % in the setting of LangChain, where paragraph breaks, line breaks, periods, question marks, and exclamation marks are set to be the separators. These settings were fed into the LangChain function `RecursiveCharacterTextSplitter` to split a reference document into chunks, where each chunk contains up to 100 words, ending at a specified separator to break naturally such that the chunk is as large as possible, and adjacent chunks have a 20%

Table 2: Intrinsic comparisons across all logical relations, where "Retr" is an abbreviation of Retriever, "B" stands for BERTSim, "L" for LRCR, and "O-1" for OPI-1.

																Rank		
Retr	**Metrics**	**ADV**	**ANA**	**CAU**	**COM**	**CON**	**DED**	**DIM**	**FUZ**	**IND**	**MUH**	**PRE**	**SPA**	**TEM**	**Avg**	**B**	**L**	**O-1**
DPS	BERSim	0.78	0.79	0.85	0.80	0.82	0.84	0.81	0.67	0.84	0.82	0.83	0.87	0.84	0.8113	2		
	LRCR	0.45	0.70	0.60	0.75	0.80	0.70	0.75	0.60	0.60	0.60	0.70	0.75	0.75	0.6731		2	
	OPI-1	0.57	0.74	0.70	0.78	0.81	0.76	0.75	0.63	0.70	0.69	0.76	0.81	0.79	0.7358			2
kNN	BERTSim	0.77	0.79	0.83	0.81	0.82	0.81	0.83	0.73	0.85	0.82	0.84	0.90	0.80	0.8162	**1**		
	LRCR	0.55	0.70	0.60	0.70	0.65	0.70	0.80	0.60	0.60	0.65	0.75	0.80	0.70	0.6769		**1**	
	OPI-1	0.64	0.74	0.70	0.75	0.73	0.75	0.81	0.66	0.70	0.72	0.79	0.85	0.75	0.7401			**1**
BM25	BERTSim	0.80	0.80	0.82	0.81	0.78	0.81	0.79	0.75	0.82	0.78	0.81	0.88	0.76	0.8020	6		
	LRCR	0.60	0.60	0.60	0.70	0.60	0.65	0.70	0.65	0.60	0.65	0.65	0.75	0.75	0.6538		4	
	OPI-1	0.69	0.69	0.69	0.75	0.68	0.72	0.74	0.70	0.69	0.71	0.72	0.81	0.76	0.7204			4
SVM	BERTSim	0.76	0.74	0.83	0.82	0.85	0.80	0.80	0.67	0.84	0.82	0.79	0.90	0.84	0.8039	5		
	LRCR	0.58	0.61	0.64	0.63	0.69	0.61	0.68	0.62	0.62	0.60	0.63	0.72	0.71	0.6418		6	
	OPI-1	0.66	0.67	0.72	0.71	0.76	0.69	0.74	0.65	0.71	0.69	0.70	0.80	0.77	0.7137			6
TF-IDF	BERTSim	0.80	0.83	0.81	0.80	0.82	0.80	0.79	0.73	0.82	0.80	0.77	0.90	0.83	0.8069	4		
	LRCR	0.65	0.80	0.80	0.65	0.60	0.70	0.75	0.55	0.65	0.65	0.45	0.60	0.70	0.6577		3	
	OPI-1	0.72	0.81	0.80	0.72	0.69	0.75	0.77	0.63	0.72	0.72	0.57	0.72	0.76	0.7247			3
MMR	BERTSim	0.75	0.81	0.83	0.84	0.82	0.82	0.85	0.71	0.84	0.79	0.83	0.85	0.76	0.8074	3		
	LRCR	0.58	0.63	0.63	0.62	0.67	0.63	0.72	0.64	0.61	0.61	0.64	0.69	0.66	0.6420		5	
	OPI-1	0.66	0.71	0.72	0.71	0.74	0.71	0.78	0.67	0.71	0.69	0.73	0.76	0.71	0.7153			5
ED1	BERTSim	0.74	0.71	0.83	0.79	0.77	0.82	0.81	0.70	0.84	0.77	0.76	0.66	0.74	0.7646	7		
	LRCR	0.50	0.55	0.70	0.70	0.60	0.70	0.75	0.55	0.65	0.60	0.70	0.45	0.55	0.6154		7	
	OPI-1	0.60	0.62	0.76	0.74	0.68	0.76	0.78	0.62	0.73	0.67	0.73	0.53	0.63	0.6819			7

overlap.

We used the default settings for each retriever to return four chunks in the context with the best scores—highest for similarity and ranking measures, smallest for distance measures—from the underlying retriever as the most relevant to the query. We then converted the four chunks extracted by the retriever back into complete sentences as they appeared in the original article. These sentences and the query were then fed to GPT-4o to generate an answer. Moreover, we instructed GPT-4o to determine the logical relationship for the answer with respect to the input text.

We consider the accuracy of the generated answers and logical relations to be equally important. Table 2 presents the evaluation results of OPI-1 on the test data of RAG-QA-1300. The OPI-1 score with respect to each retriever is calculated for each set of data points of the same logical relation. The average OPI-1 score for each retriever across all logical relations is calculated by

$$\text{OPI-1} = \frac{2/|L| \cdot \sum_{\ell \in L} \text{BERTSim}_\ell \cdot \sum_{\ell \in L} \text{LRCR}_\ell}{\sum_{\ell \in L} \text{BERTSim}_\ell + \sum_{\ell \in L} \text{LRCR}_\ell}, \quad (3)$$

where L is the set of the 13 logical relations, and BERTSim_ℓ and LRCR_ℓ denote, respectively, the corresponding BERTSim score and LRCR value for the logical relation ℓ.

An alternative is to calculate the average OPI-1 score across all logical relations. While this differs slightly from Formula (3), the difference is minimal. We prefer Formula (3) for practical efficiency, as it bypasses the need to compute individual OPI-1 scores for each logical relation when these scores are not needed in applications, streamlining the process and reducing unnecessary computations.

4.2 Extrinsic Evaluation

The extrinsic evaluation uses a 0-3-7, 3-point scoring system to score A' for each pair (A, A'), where A is the ground-truth answer and A' is the answer generated directly by GPT-4o based on the question Q and the extracted sentences in the corresponding data point. The scoring system works as follows:

- A' receives 7 points if it exactly matches the meaning of A.
- A' receives 3 points if it partially matches the meaning of A, with or without extra information not contained in A.
- A' receives 0 points if there is no meaningful match.

This scoring system was designed to simplify extrinsic evaluation while widening the gaps between perfect, partial, and no matches, providing a more nuanced assessment compared to the traditional 1-2-3, 3-point scoring system. The intention is to create a ranking with greater discrepancy between scores, enabling clearer differentiation of performance levels and allowing for more meaningful comparisons.

Given below are examples of each scenario.

- 7 points: A' receives 7 points if A and A' have exactly the same meaning, with neither additional nor missing information. For example, the following A' using DPS is a 7-point answer.
  - Q: What is the debate about what Desmond said in the context?
  - A: The debate is about what Desmond said to Locke, specifically whether he said "snowman" or "silhouette".
  - A': The debate is about whether Desmond said "snowman" or "silhouette".
- 3 points. A' receives 3 points in either of the following two cases:

  Case 1: Extra information. A' matches the meaning of A, but includes additional information not present in A. For example, the following A' using MMR receives 3 points for the reason that A' contains extra information that the Ford GT40 was produced in limited runs from 1964 to 1969.
  - Q: What was the reason behind the creation of the Ford GT40?
  - A: The Ford GT40 was created as a result of a grudge between Henry Ford II and Enzo Ferrari. After a failed attempt to acquire Ferrari, Henry Ford II decided to compete against Ferrari in the prestigious 24 Hours of Le Mans race. To break Ferrari's dominance in the race, Ford's chief engineer, Roy Lunn, came up with the concept of the GT40.
  - A': The Ford GT40 was born out of a rivalry between Henry Ford II and Enzo Ferrari. After a failed attempt to acquire Ferrari, Henry Ford II resolved to challenge Ferrari in the prestigious 24 Hours of Le Mans race. To disrupt Ferrari's dominance, Ford's chief engineer, Roy Lunn, conceived the idea for the GT40. It was produced in limited runs from 1964 to 1969.

  Case 2: Partial match. The meaning of A' partially overlaps with the meaning of A, but not fully. For example, the following A' generated using BM25 receives 3 points. Reason: A' clearly leaves out information that the Ford GT40 was conceived by Ford's chief engineer, Roy Lunn.
  - Q: What was the reason behind the creation of the Ford GT40?
  - A: See Case 1 above.
  - A': The reason behind the creation of the Ford GT40 was to compete against Ferrari in racing events, as evidenced by Ford's continued efforts to improve the GT40 and best the Italians.
- 0 points: A' receive 0 points if A and A' are distinct from each other with no overlap in meaning. For example, the following A' generated through EDI receives 0 points.
  - Q: What was the reason behind the creation of the Ford GT40?
  - A: See Case 1 above.
  - A': The Ford GT40 was created to take full advantage of the benefits associated with a mid-engine design, including a slinky aerodynamic shape and benign handling characteristics."

Table 3 shows the average scores of comparing answers by freelance annotators as well as the corresponding BERTSim scores. The integers in the row below the row of evaluation scores represent the respective rankings.

Table 3: Evaluation scores by extrinsic evaluation and intrinsic BERTSim metric with rankings, where "Extr" stands for "extrinsic evaluation" and "Intr" for "intrisic evaluation".

	DPS	kNN	BM25	SVM	TF-IDF	MMR	EDI
Extr	2.8654	2.8654	2.7923	2.8615	2.8654	2.9077	2.6038
	2	2	6	5	2	1	7
Intr	0.8113	0.8162	0.8020	0.8039	0.8069	0.8074	0.7646
	2	1	6	5	4	3	7

It is evident that the extrinsic evaluation scores align well with the BERTSim scores, demonstrating consistency in ranking. In particular, both evaluations are in complete agreement for the 2nd, 5th, 6th, and 7th places, with only minor variations in the other rankings. For instance, MMR is ranked 1st by extrinsic evaluation and 3rd by BERTSim, which is quite close. Similarly, TF-IDF is ranked 2nd by extrinsic evaluation and 4th by BERTSim. Notably, DPS, kNN, and TF-IDF all share the 2nd rank in extrinsic evaluation, likely due to the coarseness of human annotation. Since DPS and kNN are essentially the same measures, they should logically be ranked closer to each other than to TF-IDF. Therefore, the extrinsic rank of TF-IDF, while differing slightly from BERTSim, can still be considered reasonably aligned. Overall, this suggests a strong correlation between the two evaluation methods.

4.3 Combining Multiple Retrievers

LangChain supports combining multiple retrievers into a new retriever. We use the default setting to return four chunks for each combination. This approach diversifies the retrieved content from the reference document, potentially improving overall performance.

Table 4: Evaluation results of various combinations of retrievers and sentences.

																Rank		
Retr	Metrics	ADV	ANA	CAU	COM	CON	DED	DIM	FUZ	IND	MUH	PRE	SPA	TEM	Avg	B	L	O-1
A-Seven	BERSim	0.81	0.81	0.86	0.85	0.85	0.84	0.86	0.66	0.86	0.84	0.81	0.91	0.85	0.8325	**1**		
	LRCR	0.65	0.70	0.75	0.70	0.75	0.70	0.75	0.55	0.70	0.70	0.65	0.75	0.70	0.6962		**1**	
	OPI-1	0.72	0.75	0.80	0.77	0.80	0.76	0.75	0.60	0.77	0.76	0.72	0.82	0.77	0.7583			**1**
A-Four	BERTSim	0.81	0.81	0.86	0.84	0.85	0.82	0.82	0.72	0.86	0.84	0.82	0.90	0.80	0.8276	2		
	LRCR	0.50	0.70	0.75	0.75	0.70	0.75	0.70	0.60	0.70	0.70	0.65	0.75	0.70	0.6885		2	
	OPI-1	0.62	0.75	0.80	0.79	0.77	0.78	0.76	0.65	0.77	0.76	0.72	0.82	0.75	0.7516			2
A-Two	BERTSim	0.76	0.79	0.83	0.80	0.84	0.82	0.76	0.73	0.86	0.82	0.83	0.89	0.80	0.8099	3		
	LRCR	0.50	0.70	0.60	0.70	0.80	0.65	0.75	0.65	0.60	0.65	0.70	0.75	0.70	0.6731		3	
	OPI-1	0.60	0.74	0.70	0.75	0.82	0.72	0.76	0.69	0.71	0.73	0.76	0.81	0.75	0.7352			3
S-Seven	BERSim	0.81	0.84	0.87	0.89	0.83	0.85	0.82	0.68	0.88	0.85	0.77	0.90	0.80	0.8310	**1**		
	LRCR	0.65	0.70	0.75	0.75	0.70	0.65	0.75	0.55	0.65	0.75	0.65	0.75	0.75	0.6962		**1**	
	OPI-1	0.72	0.76	0.81	0.81	0.76	0.74	0.75	0.61	0.75	0.80	0.70	0.82	0.78	0.7576			**1**
S-Four	BERTSim	0.82	0.81	0.87	0.85	0.86	0.83	0.83	0.73	0.87	0.84	0.82	0.91	0.80	0.8330	2		
	LRCR	0.50	0.75	0.75	0.70	0.70	0.75	0.70	0.60	0.70	0.75	0.65	0.75	0.70	0.6923		2	
	OPI-1	0.62	0.78	0.80	0.77	0.77	0.79	0.76	0.66	0.78	0.79	0.73	0.82	0.75	0.7562			2
S-Two	BERTSim	0.76	0.79	0.83	0.81	0.84	0.82	0.76	0.73	0.86	0.82	0.83	0.89	0.80	0.8120	3		
	LRCR	0.50	0.70	0.60	0.70	0.80	0.70	0.75	0.65	0.60	0.65	0.65	0.75	0.70	0.6731		3	
	OPI-1	0.60	0.74	0.70	0.75	0.82	0.75	0.76	0.69	0.71	0.73	0.73	0.81	0.75	0.7360			3

As examples, we combine all seven retrievers, denoted as A-Seven; three retrievers with the highest OPI-1 scores: kNN, DPS, and TF-IDF, plus MMR for its strength in balancing relevance and diversity, denoted as A-Four; and two retrievers with the highest OPI-1 scores: kNN and DPS, denoted as A-Two.

We may also combine the sentences retrieved by individual retrievers, removing any duplicates, and use the remaining set of sentences with the corresponding questions to generate answers and logical relations. Let S-Seven, S-Four, and S-Two denote the sets of sentences obtained this way by the corresponding retrievers as in A-Seven, A-Four, and A-Two.

The experimental results of both types of combinations are shown in Table 4.

5 ANALYSIS

We first analyze the performance of individual retrievers, followed by examining the combinations of retrievers and the sentences retrieved by multiple retrievers.

5.1 Individual Retrievers

For each retriever, we first analyze the performance for each logical relation individually and then assess the overall performance across all logical relations.

5.1.1 Individual Logical Relation

We use the OPI-1 scores to help identify the strengths and weaknesses of individual retrievers across the 13 logical relations. For example, as seen in Table 2, almost all retrievers tend to perform the worst on adversarial reasoning, followed by fuzzy reasoning. For other logical relations, the performance of retrievers varies, indicating that certain retrievers may be more suited to specific types of reasoning tasks while struggling with others. For example, even for the worst-performing retriever, EDI, which consistently ranks the lowest in both extrinsic and intrinsic evaluations of answer accuracy as seen in Table 3, it still performs best on deductive reasoning. This suggests that while EDI may generally be less effective across various logical relations, it has a particular strength in handling tasks that involve deductive reasoning. This example highlights the nuanced performance of retrievers, where even a generally weaker retriever can excel in specific logical tasks. This variability in performance highlights the importance of selecting the appropriate retriever.

5.1.2 Across all Logical Relations

The average OPI-1 scores provide a means to identify, across all 13 logical relations, which retrievers are more suitable for specific tasks and which retrievers should be avoided. For example, as shown in Table 2, EDI has the lowest and SVM the second-lowest average OPI-1 scores, indicating they should generally be avoided. This is likely due to the limitations of the underlying features used to compute SVM scores

and the coarseness of L2-norms when representing text chunks as bag-of-word vectors, which may fail to capture the nuanced relationships required for deep logical reasoning tasks.

On the other hand, kNN has the highest and DPS the second-highest average OPI-1 scores, indicating that these retrievers would be the best choices for answering deep-logic questions. kNN (cosine similarity) and DPS are similar measures, with kNN being a normalized version of DPS, which explains their comparable performance. However, kNN takes slightly more time to compute than DPS, as DPS is the fastest among all seven retrievers—dot products are the simplest and quickest to compute compared to the operations used by other retrievers.

The MMR retriever allows GPT-4o to generate better answers across all logical relations, as shown in Table 3 . However, it does not perform as well in producing the correct logical relations. This discrepancy may be attributed to MMR's focus on balancing relevance and diversity in retrieved content, which improves answer quality but doesn't necessarily align with capturing accurate logical relations.

BM25 is in general more effective for retrieving longer documents in a document corpus with the default parameter values for κ and b. However, to retrieve sentences from an article, it was shown that BM25 would should use different parameter values (Zhang et al., 2021). This explains why BM25 is the second worse for generating answers as shown in Table 3 by both extrinsic and intrinsic evaluations. It is not clear, however, why it produces a relatively higher LRCR value.

TF-IDF's performance falls in the middle range, which is expected. As a frequency-based approach, it may struggle to capture deeper semantic information, but it remains relatively effective because it retains lexical information, ensuring that important terms are still emphasized in the retrieval process.

5.2 Performance of Various Combinations

We first analyze the performance of combinations of retrievers versus individual retrievers, followed by an analysis of combining retrievers algorithmically versus combining sentences retrieved by individual retrievers within the combination.

5.2.1 Combinations vs. Individuals

It can be seen from Table 4 that A-Seven outperforms A-Four, which in turn outperforms A-Two. A similar ranking is observed with S-Seven, S-Four, and S-Two. Moreover, both A-Seven and A-Four are substantially better than the top performer, kNN, when only a single retriever is used (see both Tables 2 and 4). A similar result is observed with S-Seven and S-Four, where combining more retrieved sentences from different retrievers also enhances performance, reinforcing the benefits of increased diversity in the retrieval process. These results all confirm the early suggestion that combining more retrievers generally enhances performance in both algorithmic and sentence-based combinations, supporting the idea that diverse retrieval methods contribute positively to the overall effectiveness of the RAG system.

However, we also observe that some combinations of retrievers may actually lead to poorer performance compared to using the individual retrievers alone. This is evident in the case of A-Two and S-Two, the algorithmic and sentence combinations of kNN and DPS, both result in slightly lower average OPI-1 scores than kNN alone. This is probably due to the fact that kNN and DPS are very similar measures, and combining them doesn't significantly increase diversity. Worse, the extra information provided through their combination seems to have led to diminishing returns, negating the potential benefits of combining retrievers to improve performance. This phenomenon warrants further investigation.

Nevertheless, combining retrievers based on different retrieval methodologies could help increase diversity and, consequently, improve overall performance. This is evident in the case of A-Seven and S-Seven, which combine retrievers utilizing diverse retrieval methods, as well as in A-Four and S-Four, where MMR—a retrieval method that balances relevance and diversity—complements kNN. By leveraging varied retrieval techniques, we can ensure that a broader range of relevant content is retrieved, potentially leading to greater accuracy and more robust logical reasoning in the generated answers.

5.2.2 Combining Algorithms vs. Combining Sentences

We compare the outcomes of combining retrievers at the algorithm level versus the sentence level. Combining retrievers at the algorithm level is a feature supported by LangChain, which returns the same default number of chunks before sentences are extracted. In contrast, combining retrievers at the sentence level involves merging sentences retrieved by individual retrievers, which may include more sentences than the algorithmic combination, and so should lead to a slightly better performance. This is evident when comparing A-Four with S-Four and A-Two with S-Two (see Table 4).

However, having more sentences may not always lead to improvement, as it can introduce conflicting information. This is evident when comparing A-Seven with S-Seven, where S-Seven has a lower average OPI-1 score than A-Seven. This is likely because A-Seven has already saturated the useful sentences, while S-Seven introduces additional sentences that negatively impact the average OPI-1 score.

In summary, these analyses suggest that, when combining appropriate retrievers, both algorithmic and sentence-level approaches offer performance improvements, with each method providing distinct advantages in terms of retrieval diversity and the quality of generated answers. Selecting appropriate retrievers requires a deeper understanding of the underlying retrieval mechanisms, making this an interesting topic for further investigation.

6 FINAL REMARKS

This paper presents an effective intrinsic evaluate method for the performance of RAG systems in connection to question-answering involving deep logical reasoning.

LangChain supports a wide range of retrievers and allows users to integrate custom retrievers. Additionally, there are numerous large language models (LLMs) such as the Gemini series (Google, 2024), LlaMA series (Meta, 2024), and Claude series (Claude AI, 2024), among others, as well as various retrieval-augmented generation (RAG) tools like LLAMAINDEX (LlamaIndex, 2024), HayStack (Deepset, 2024), EmbedChain (EmbedChain, 2024), and RAGatouille (AnswerDotAI, 2024). Evaluating the performance of these models and tools, particularly for answering deep-logic questions where identifying logical relations is essential, represents an intriguing direction for future research.

Regularly reporting the findings of such investigations would significantly contribute to the advancement of RAG technologies. Furthermore, we aim to develop a tool that quantitatively assesses the depth of logical relations in question-answering systems relative to the underlying context. This effort would necessitate the creation of a new dataset that annotates the depth of each logical relation for every triple consisting of a question, an answer, and a set of reference sentences.

REFERENCES

AnswerDotAI (2024). Ragatouille. Accessed: 2024-09-06.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.

Carbonell, J. and Goldstein, J. (1998). The use of mmr, diversity-based reranking for reordering documents and producing summaries. *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 335–336.

Claude AI (2024). Claude. Accessed: 2024-09-06.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

Deepset (2024). Haystack - deepset. Accessed: 2024-09-06.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

D.H., C. (2024). Rag dataset 12000.

EmbedChain (2024). Embedchain. Accessed: 2024-09-06.

Google (2024). Gemini - google. Accessed: 2024-09-06.

LangChain (2024). Langchain official website.

Lewis, D. D. and Gale, W. A. (1994). A study of f-measure in information retrieval. In *Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval (SDAIR'94)*, pages 187–199. Citeseer.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., tau Yih, W., Rocktäschel, T., et al. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. *arXiv preprint arXiv:2005.11401*.

LlamaIndex (2024). Retrieval-augmented generation (rag) - llamaindex. Accessed: 2024-09-06.

Meta (2024). Llama - meta. Accessed: 2024-09-06.

Robertson, S. and Zaragoza, H. (2009). The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389.

Sammut, C. and Webb, G. I. (2011). *TF–IDF*. Springer.

Zhang, H., Zhou, Y., and Wang, J. (2021). Contextual networks and unsupervised ranking of sentences. In *Proceedings of the 33rd IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2021)*.

Zhou, Y. and Wang, J. (2023). Content significance distributions of sub-text blocks in articles and its application to article-organization assessment. In *Proceedings of the 15th Knowledge Discovery and Information Retrieval (KDIR 2023)*.