Highlights

Bridging the Question-Answer Gap in Retrieval-Augmented Generation:
Hypothetical Prompt Embeddings

Domen Vake, Jernej Vičič, Aleksandar Tošić

- Precomputing hypothetical prompts for improved retrieval
- Consistent gains in retrieval context precision and claim recall
- Compatibility with advanced RAG techniques

Bridging the Question-Answer Gap in Retrieval-Augmented Generation: Hypothetical Prompt Embeddings

Domen Vake[a,b], Jernej Vičič[a,c], Aleksandar Tošić[a,b]

[a] *University of Primorska, Faculty of Mathematics, Natural Sciences and Information Technologies*
[b] *InnoRenew CoE*
[c] *Research Centre of the Slovenian Academy of Sciences and Arts, The Fran Ramovš Institute*

Abstract

Retrieval-Augmented Generation (RAG) systems synergize retrieval mechanisms with generative language models to enhance the accuracy and relevance of responses. However, bridging the style gap between user queries and relevant information in document text remains a persistent challenge in retrieval-augmented systems, often addressed by runtime solutions (e.g., Hypothetical Document Embeddings (HyDE)) that attempt to improve alignment but introduce extra computational overhead at query time. To address these challenges, we propose Hypothetical Prompt Embeddings (HyPE), a framework that shifts the generation of hypothetical content from query time to the indexing phase. By precomputing multiple hypothetical prompts for each data chunk and embedding the chunk in place of the prompt, HyPE transforms retrieval into a question-question matching task, bypassing the need for runtime synthetic answer generation. This approach does not introduce latency but also strengthens the alignment between queries and relevant context. Our experimental results on six common datasets show that HyPE can improve retrieval context precision by up to 42 percentage points and claim recall by up to 45 percentage points, compared to standard approaches, while remaining compatible with re-ranking, multi-vector retrieval, and other RAG advancements.

Keywords: Retrieval-Augmented Generation, Large Language Models, Hypothetical prompt embedding, Dense Retrieval

1. Introduction

Retrieval-Augmented Generation (RAG) systems have emerged as a powerful paradigm in natural language processing, combining the strengths of retrieval-based approaches with the generative capabilities of large language models (LLMs) [**?**]. By leveraging external knowledge sources, RAG systems enhance the factual accuracy and relevance of generated responses, addressing the limitations of standalone generative models, such as outdated knowledge or limited access to restricted information. Despite their success, existing RAG implementations often struggle with aligning retrieval and generation in a way that efficiently bridges the gap between user queries and relevant document content.

To optimize retrieval performance, researchers have explored a variety of strategies, ranging from efficient chunking (splitting texts into coherent subunits)[**?** **?**] to re-ranking methods that refine initial retrieval results via cross-encoders or boosted similarity scoring[**?**]. Advanced frameworks such as GraphRAG exploit graph structures to capture cross-document relationships for more nuanced multi-hop or contextual queries [**?**]. Meanwhile, domain adaptation techniques focus on tailoring retrieval to specialized topics, ensuring that queries can be matched effectively, even in areas where language models might otherwise lack expertise.

Despite these efforts, a persistent hurdle in RAG remains the mismatch between user queries, which typically adopt an interrogative style, and corpus content, which is usually expository or declarative in nature. This style difference hampers the alignment of query embeddings with document embeddings, occasionally allowing key information to go unretrieved. A notable solution to this problem is Hypothetical Document Embeddings (HyDE) [**?**], which prompts an LLM at query time to generate a synthetic answer, then uses that short text as the query for retrieval.

In this paper, we introduce Hypothetical Prompt Embeddings (HyPE), a new approach that tackles query–document style mismatch without adding overhead to every user request. Rather than generating synthetic answers at inference, HyPE precomputes multiple hypothetical questions for each corpus chunk at indexing time. These question-like prompts are embedded and stored, so that query matching effectively becomes a question–question retrieval problem. By shifting hypothetical generation offline, HyPE avoids additional runtime LLM calls.

To evaluate HyPE's effectiveness, we compare it against a naive RAG

implementation and HyDE across multiple datasets and evaluation metrics, including precision, recall, and faithfulness. Our results demonstrate that HyPE offers substantial improvements in retrieval efficiency, reducing the computational burden while achieving comparable or better retrieval accuracy and contextual relevance.

The contributions of this paper are as follows:

- We introduce the concept of precomputed hypothetical prompt embeddings to optimize retrieval efficiency in RAG systems.
- We provide a comprehensive performance comparison between a Naive RAG implementation, HyDE, and HyPE.
- We present experimental results showcasing the trade-offs in retrieval quality and the effect of retrieval approach on generation across various datasets.

By shifting the hypothetical generation process from runtime to indexing, HyPE represents a scalable and efficient alternative for RAG systems, offering practical benefits for real-world applications requiring fast and reliable retrieval-augmented text generation.

2. Related Works

Early RAG pipelines [**?** **?**] simply embed the user queries and retrieve text chunks via approximate nearest neighbour (ANN) search over a vector index. However, a persistent challenge remains: user queries are often phrased in question form, whereas documents or chunks are stored in an expository or statement-oriented style, creating a semantic or "lexical–conceptual" gap [**?** **?**]. This mismatch can degrade the retrieval's accuracy and ultimately weaken the generative model's faithfulness.

One direction of research attempts to alleviate this mismatch by expanding documents with likely queries. For instance, Doc2Query [**?**] uses a sequence-to-sequence model (often T5) to generate synthetic questions for each document, appending them to the text so that a bag-of-words ranker like BM25 [**?**] can match real user queries more easily. While Doc2Query often boosts first-stage recall, subsequent studies noted that the generation process can hallucinate irrelevant expansions, leading to index bloat. Among

them, Doc2Query– (Minus-Minus) [?] addresses this by filtering out low-quality expansions, improving accuracy and reducing index size for BM25-based retrieval. However, these methods predominantly operate in a lexical retrieval space rather than dense embeddings, and they store expansions as text appended to each document and, as such, act more like an enrichment of the chunks.

Another related direction focuses on generating synthetic training data to train or fine-tune dense retrievers in new domains. Ma et al. [?] propose using a question-generation model, trained on general-purpose Question-Answer(QA) pairs, to generate "pseudo-queries" for domain-specific passages. A dual-encoder retrieval model is then trained on these synthetic (question, passage) pairs—effectively learning domain adaptation in a zero-shot setting. While powerful for building domain-specialized retrievers, the method requires re-training or fine-tuning a dense model on large-scale synthetic data. By contrast, our approach bypasses model re-training at retrieval time and, instead, alters how we store the passages (i.e., their hypothetical question embeddings).

More recently, HyDE [?] addresses query–document mismatch by generating a hypothetical answer or short passage at query time. Instead of embedding the user's question directly, HyDE prompts an LLM to produce an approximate response, then embeds that synthetic text. This is used to retrieve relevant real documents from a vector index. While HyDE can improve retrieval accuracy for zero-shot question answering, it incurs an extra inference cost per user query. Additionally, the method may struggle, where the prompt queries for niche domain knowledge, where the model may not have sufficient knowledge to produce a representative sample.

3. Methodology

HyPE addresses the challenge of aligning user queries and relevant content by pre-computing hypothetical prompts at the indexing stage, contrasting with HyDE's runtime generation of synthetic answers. This shift avoids additional inference overhead per query and improves retrieval precision by ensuring that both user queries and stored embeddings share a question-like form.

The method begins by splitting the corpus D into coherent chunks $C_1, C_2, \ldots, C_n$, where each chunk provides a self-contained unit of information. For each chunk C_i, an LLM G generates multiple hypothetical prompts

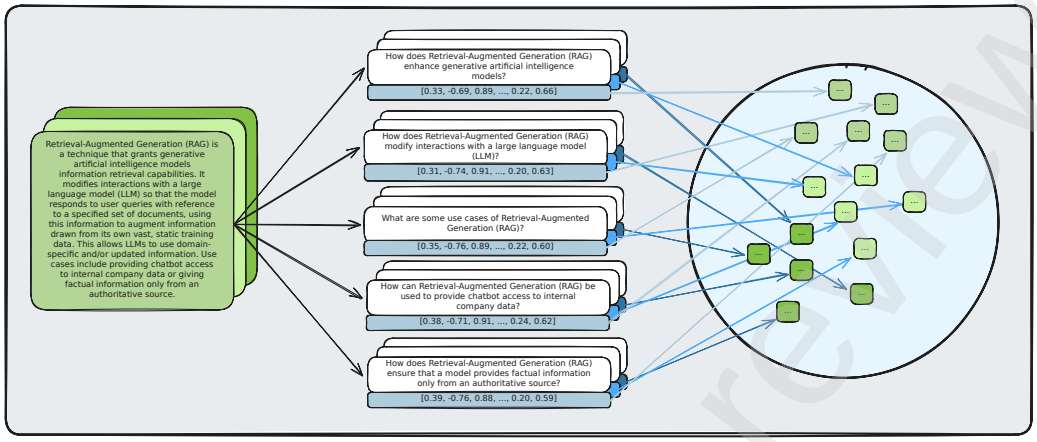

Figure 1: Illustration of the Hypothetical Prompt Embeddings (HyPE) framework, showcasing the process of precomputing hypothetical questions during indexing to optimize retrieval efficiency in Retrieval-Augmented Generation (RAG) systems.

$Q_i = q_{i1}, q_{i2}, \ldots, q_{ik}$, simulating possible user queries that the chunk might answer. This offline step does not introduce any additional computational cost at query time, as no new prompts need to be generated for each user request.

Each hypothetical prompt q_{ij} is then mapped to an embedding $v_{ij} = f(q_{ij}) \in \mathbb{R}^d$ using a pre-trained dense retrieval model f. Rather than storing these prompt embeddings separately, we associate each v_{ij} with the original chunk C_i, thus building an index of vector–chunk pairs:

$$E = \{(v_{11}, C_1), (v_{12}, C_1), \ldots, (v_{nk}, C_n)\}$$

Each chunk is effectively represented multiple times, once for each hypothetical prompt. This extends the coverage of how queries may be phrased and matched.

The retrieval process at runtime follows a standard approximate nearest-neighbor (ANN) search in the vector space. When a user query q arrives, it is embedded into $q = f(q)$. The system then locates the nearest v_{ij} vectors within E, and retrieves the associated chunks for final answer generation by an LLM. Although the pipeline remains structurally similar to a Naive RAG, the key difference is that HyPE matches questions against questions, rather than questions against chunk text.

This question–question alignment increases the probability of finding the correct chunks for two main reasons. First, many embedding models exhibit style-based clustering [**?**]. Texts of similar form (e.g., interrogative sentences) often lie closer in the vector space. As a result, a user's real-world query naturally aligns more closely with the hypothetical prompts that share its interrogative style. Second, generating multiple hypothetical queries per chunk broadens the "semantic reach," covering a wider range of possible question formulations. Even if a user query is phrased in a slightly different way, there is a higher chance that at least one of the chunk's hypothetical questions closely corresponds to it. While HyPE still requires LLM calls to generate hypothetical prompts, this process is performed entirely before indexing and remains fixed to the dataset size, rather than scaling with query volume. This means that all computational overhead is incurred up front, making inference-time retrieval cheaper and more efficient compared to methods that require dynamic LLM generation per query.

Another advantage of HyPE lies in how it addresses the inherent chunking tradeoff in retrieval systems. If chunks are too large, their embeddings become less precise because they encode a mix of multiple concepts, making vector-based similarity less reliable [**?**]. Conversely, reducing chunk size improves embedding specificity but risks losing crucial surrounding context. HyPE mitigates this issue by ensuring that each stored vector represents a specific piece of information within a chunk, while retrieval still returns the entire chunk with its broader context. This allows the system to retain the benefits of detailed, fine-grained embeddings without sacrificing the context for accurate retrieval.

4. Experiments

We evaluated three RAG pipelines to assess retrieval and generation performance as presented in Table 1

Retriever Pipeline	Augmentation stage	Context Space
Naive RAG	/	prompt-to-document
HyDE	Inference	document-to-document
HyPE	Indexing	prompt-to-prompt

Table 1: Table presents the key differences of compared pipelines

The comparison isolates the impact of runtime document generation versus precomputed prompt generation approaches on efficiency and accuracy, with Naive RAG providing a baseline reference.

For evaluating the RAG pipelines, we used the RAGChecker framework [**?**], a comprehensive evaluation toolkit developed by Amazon Science for assessing both retrieval and generation performance in RAG systems. It provides structured metrics to analyse retrieval effectiveness through context precision, which measures how many retrieved passages are relevant, and claim recall, which assesses whether all necessary information is retrieved.

For generation, RAGChecker evaluates faithfulness, ensuring the generated text remains grounded in the retrieved passages, along with the hallucination rate, which identifies unsupported claims, and context utilization, which measures how effectively retrieved passages contribute to responses. Additionally, it assesses robustness through noise sensitivity, testing the system's response to query variations, and self-knowledge, which quantifies the model's ability to recognize when it lacks sufficient information. Additionally, it computes precision, recall, and F1 scores, providing an overall measure of retrieval and response accuracy [**?**].

We evaluated our approach on six datasets, chosen to test distinct aspects of RAG systems. *MS MARCO* [**?**], a large-scale question-answering benchmark, and *Ragas-WikiQA*[1] evaluate general-purpose retrieval in real-world scenarios. *RAG-dataset-12000*[2] and *MultiHopRAG* [**?**] emphasize multi-hop reasoning, requiring systems to synthesize information across multiple documents. *RAGBench* [**?**] tests hybrid tasks demanding both precise retrieval and coherent generation. Finally, *Single-Topic RAG* dataset[3] focuses on narrow domains, assessing precision in specialized contexts.

All datasets already come pre-segmented into chunks, except for *RAG-dataset-12000* that contains a single context block, and *MultiHopRAG*, which contains multiple larger documents. For these two cases, we manually split the source text into segments of 500 tokens, overlapping each segment by 50 tokens to preserve cross-boundary coherence. This approach, while straightforward, may not be optimal as the choice of chunking strategy can significantly impact retrieval effectiveness and quality of generation. Accordingly,

[1] https://huggingface.co/datasets/explodinggradients/ragas-wikiqa

[2] https://huggingface.co/datasets/neural-bridge/rag-dataset-12000

[3] https://www.kaggle.com/datasets/samuelmatsuoharris/single-topic-rag-evaluation-dataset

we apply the same chunking procedure across all three pipelines for consistency in our experiments, but we note that different pipelines might benefit from tailored chunking strategies. We leave an in-depth exploration of chunk size, overlap, and other segmentation heuristics as a direction for future research.

> Analyze the input text and generate essential questions that, when answered, capture the main points and core meaning of the text. The questions should be exhaustive and understandable without context. When possible, named entities should be referenced by their full name. Only answer with questions where each question should be written in its own line (separated by newline) with no prefix.

Figure 2: Prompt used to generate hypothetical prompts

For all pipelines, we tested retrieval depths $k \in \{1, 3, 5, 10\}$. At $k = 5$, we additionally compared cosine similarity and Euclidean distance functions to assess their impact on chunk relevance ranking. The embedding model chosen for all pipelines was *bge-m3* [**?**] for dense vector representations. The generator LLM used was *Mistral-NeMo*[4].

5. Analysis

In Table 2, we compare the three retrieval methods across six datasets and varying numbers of retrieved chunks (k). Each cell reports context precision (how many of the retrieved chunks directly match the query's needs) and claim recall (how many relevant pieces of information are captured). Overall, HyPE improves recall by about 16 percentage points and precision by about 20 percentage points compared to Naive RAG, on average. The difference can be even larger on specific datasets, such as *Single-Topic RAG* at $k = 1$ where HyPE surpasses Naive RAG by more than 40 percentage points in precision and at $k = 10$ surpasses by 44.6 percentage points in recall. Although HyPE performs slightly below Naive RAG on *MS MARCO* at $k = 1$, it catches up or exceeds Naive RAG at deeper retrieval. For *RAGBench* and *Ragas-WikiQA*, HyPE also achieves strong gains, especially at lower k values, indicating its ability to retrieve the correct chunks accurately.

[4]https://mistral.ai/news/mistral-nemo/

Dataset@k	Naive		HyDE		HyPE	
	Precision	*Recall*	*Precision*	*Recall*	*Precision*	*Recall*
RAG-12000@1	55.8	34.7	55.1	33.3	**82.6**	**63.6**
RAG-12000@3	36.5	44.2	36.6	42.8	**73.0**	**76.2**
RAG-12000@5	31.3	49.0	32.0	47.9	**66.9**	**80.1**
RAG-12000@10	26.4	56.1	27.4	55.6	**56.1**	**84.6**
MS MARCO@1	**73.6**	**56.2**	69.8	52.4	68.7	50.2
MS MARCO@3	**70.2**	**74.5**	67.2	70.0	66.9	70.2
MS MARCO@5	**67.6**	**80.6**	65.5	76.8	65.6	77.3
MS MARCO@10	61.5	**85.7**	60.7	82.8	**62.5**	84.0
MultiHop@1	36.2	21.9	35.6	21.9	**43.7**	**27.2**
MultiHop@3	32.7	35.3	31.9	34.4	**42.2**	**43.2**
MultiHop@5	30.3	39.9	30.3	39.8	**40.1**	**50.6**
MultiHop@10	25.8	46.2	26.8	47.6	**37.5**	**59.2**
RAGBench@1	65.0	38.7	58.7	33.9	**65.7**	**39.5**
RAGBench@3	62.8	54.2	56.5	48.5	**63.0**	**58.0**
RAGBench@5	60.0	60.7	54.3	55.3	**60.7**	**65.6**
RAGBench@10	55.1	68.8	49.9	63.8	**57.1**	**74.6**
Single-Topic@1	28.7	15.6	22.5	8.5	**68.8**	**32.9**
Single-Topic@3	27.5	22.9	20.8	18.2	**64.6**	**63.0**
Single-Topic@5	25.2	26.8	24.5	28.6	**64.5**	**69.0**
Single-Topic@10	21.2	36.8	19.6	36.4	**63.0**	**81.4**
WikiQA@1	51.7	32.5	53.0	31.7	**86.6**	**61.1**
WikiQA@3	47.4	57.1	48.7	56.7	**84.2**	**77.9**
WikiQA@5	40.1	65.0	43.8	67.6	**84.7**	**85.6**
WikiQA@10	33.4	79.1	38.2	78.9	**81.9**	**90.4**

Table 2: Performance comparison of retrieval methods across datasets using context precision and claim recall metrics (bigger is better) with varying numbers of retrieved context chunks (k). Gradient intensity reflects metric strength (lighter to darker green indicates lower to higher values), with bold entries highlighting the best-performing method for each configuration.

Figure 3 compares Retriever Context Precision across different numbers of documents retrieved (k) for the Naive, HyDE, and HyPE methods revealing significant improvement in precision, using HyPE. As the number of retrieved documents increases from 1 to 10, HyPE consistently demonstrates higher precision. Its precision is notably superior to both Naive and HyDE methods. This suggests that HyPE's approach of precomputing hypothetical questions during the indexing phase effectively aligns retrieved content

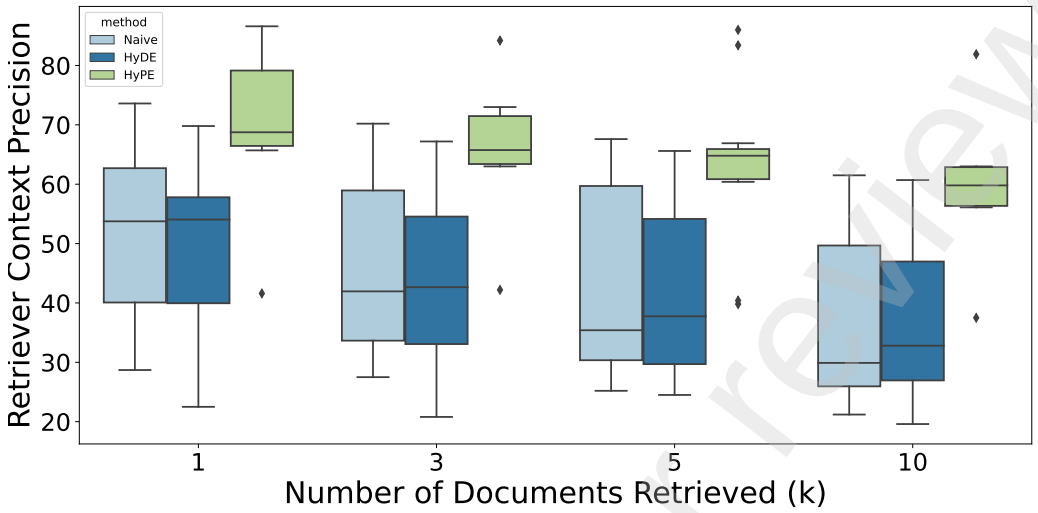

Figure 3: Box plot comparison of Retriever Context Precision across different numbers of documents retrieved (k) for three methods: Naive, HyDE, and HyPE. The plot illustrates the distribution and variability of precision scores for each method and retrieval depth.

with user queries, reducing the semantic mismatch often encountered in traditional methods. The narrower interquartile ranges for HyPE further indicate its consistency in retrieving relevant information across varying retrieval depths.

Additionally, the figure 4 of claim recall complements these findings. Claim recall measures the proportion of relevant information successfully retrieved from the documents. The balanced performance in both metrics highlights HyPE's effectiveness in bridging the gap between user queries and relevant document content.

The comparison in Figure 5 shows that for all three methods the choice between Euclidean and Cosine distance metrics does not significantly impact their effectiveness.

In Figure 6, we compare generator metrics, including context utilization, noise sensitivity, hallucination, self-knowledge, and faithfulness. While these metrics primarily evaluate the large language model, the only variable across our tests is the retrieval pipeline. Therefore, any observed differences in generation performance can be attributed to variations in retrieved content rather than differences in the LLM itself.

HyPE consistently achieves higher context utilization and faithfulness,

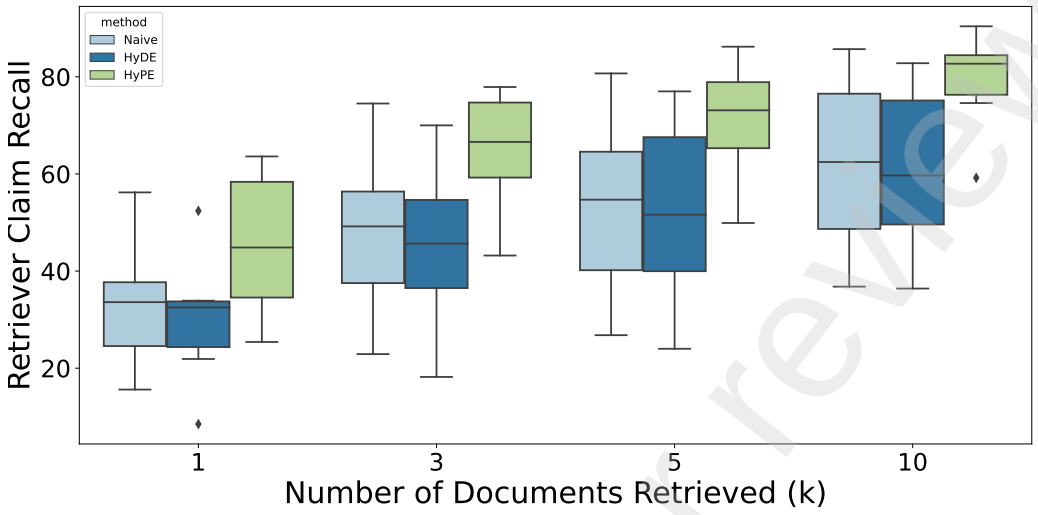

Figure 4: Box plot comparison of Retriever Claim Recall across different numbers of documents retrieved (k) for three methods: Naive, HyDE, and HyPE. The plot illustrates the distribution and variability of precision scores for each method and retrieval depth.

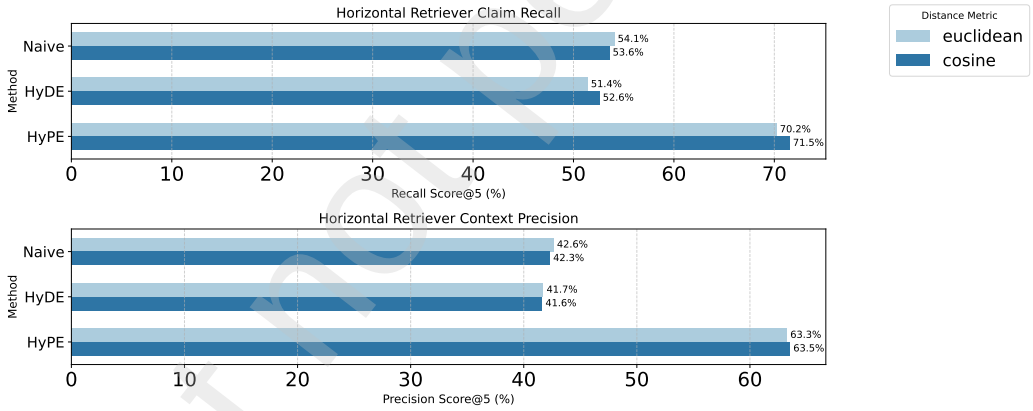

Figure 5: Comparison of Retriever Claim Recall and Context Precision across three retrieval methods using two distance metrics: Euclidean and Cosine. Performance measured at $k = 5$.

indicating that its retrieval strategy provides more relevant and coherent supporting text for generation. A particularly interesting observation is HyPE's higher noise sensitivity in relevant contexts. While increased noise sensitivity in irrelevant contexts typically suggests a model's susceptibility to distracting information, higher sensitivity in relevant contexts may indicate a stronger

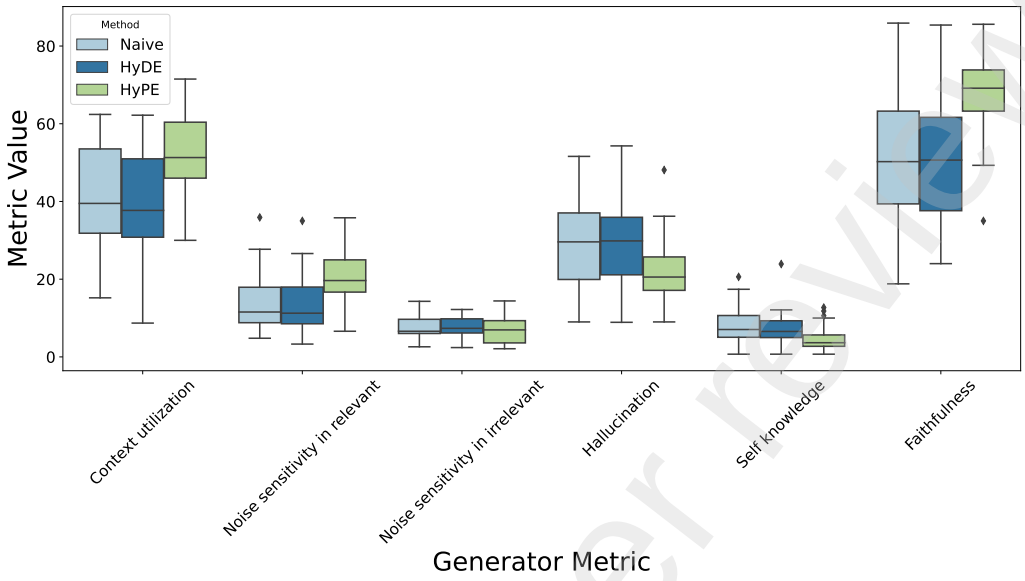

Figure 6: Comparative box plot analysis of various generator metrics when using one of the three retrieval methods, highlighting the performance differences among the methods in terms of their effectiveness and reliability in generating accurate and contextually relevant responses.

ability to distinguish between subtle variations in retrieved documents. This suggests that HyPE enables the model to engage more precisely with relevant details, potentially leading to more nuanced and contextually appropriate responses.

HyPE also exhibits lower hallucination rates compared to Naive RAG and HyDE, reinforcing the idea that better-aligned retrieval reduces the likelihood of introducing incorrect or unsupported claims. Although these results are based on a single LLM, the trend is likely to generalize across different models, as improvements in retrieval typically translate to improved generation performance. However, the exact degree of impact may vary depending on the LLM's retrieval dependence and sensitivity to context quality. The combination of improved context grounding, reduced hallucinations, and stronger response alignment highlights HyPE's potential for enhancing the reliability of RAG systems.

Figure 7 compares the overall F1 scores for retrievers and generators of the pipelines through the datasets. With the exception of the *MS MARCO* dataset where all three methods show comparable performance, the F1 scores

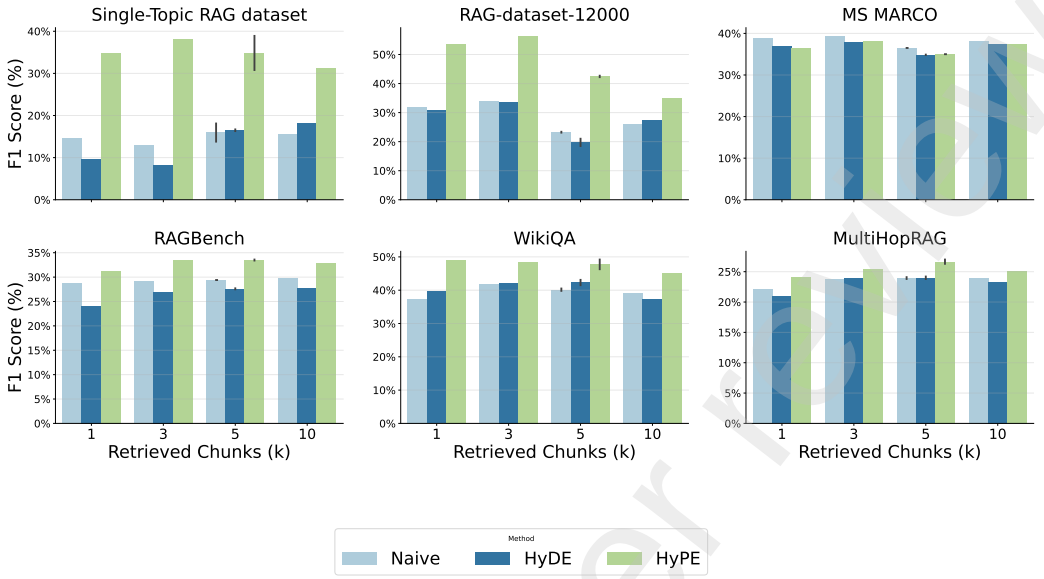

Figure 7: Bar chart comparison of F1 scores across six different datasets for three retrieval methods. Each subplot represents a dataset and shows the F1 scores for varying numbers of retrieved chunks (k = 1, 3, 5, 10).

show, that HyPE consistently outperforms the other two methods, particularly in datasets like *Single Topic RAG* and *RAG-dataset-12000*, where the complexity and specificity of queries demand precise retrieval.

6. Conclusion

The paper presents Hypothetical Prompt Embeddings (HyPE), a framework that pre-computes hypothetical prompts at indexing time to reshape retrieval in RAG pipelines into a prompt-to-prompt matching process. Our experimental findings show that HyPE surpasses both Naive RAG and HyDE on multiple datasets and metrics, with notable gains in precision and recall. By eliminating the need for query-time synthetic answer generation and instead relying on strategically generated questions offline, HyPE improves efficiency and provides stronger alignment between user queries and relevant content.

Although HyPE may not outperform every specialized RAG variant in all domains, it offers a flexible and modular upgrade to existing pipelines. Swapping in pre-computed question embeddings remains compatible with

advances in chunking, re-ranking, multi-vector retrieval, and fine-tuning large language models. HyPE also integrates smoothly into agent-based systems, where prompt-level alignment can help specialized retrieval sub-agents handle distinct query types more effectively.

Looking ahead, combining HyPE with GraphRAG, which maps documents or chunks as graph nodes, may further enhance multi-hop reasoning and retrieval accuracy in complex scenarios. Such a hybrid approach could be especially valuable for practitioners building robust and context-aware RAG systems.

Another direction for future research involves investigating the chunking tradeoff in the context of expanding LLM context windows. As language models evolve to accommodate larger input lengths, it becomes increasingly feasible to supply bigger chunks as prompts. However, larger chunks can dilute semantic specificity in their embeddings, resulting in less precise vector matching. This tension between maintaining detailed embeddings and preserving broader context may become more pronounced as context windows expand. Further testing of chunk size and indexing depth with HyPE would help clarify how embedding precision balances with retrieval breadth.

Overall, HyPE demonstrates that shifting from question-to-document to question-to-question alignment leads to tangible gains in retrieval accuracy and cost-effectiveness. As RAG solutions continue to evolve, offline prompt generation strategies like HyPE can serve as a foundation for more efficient generation.

7. Acknowledgements

This project received funding from the European Union's Horizon 2020 grant #101135012