KIPE: Document-side Pseudo-query Indexing and Query-side Keyword-aware Reranking for Dense Retrieval

Anonymous ACL submission

Abstract

Dense retrieval usually represents each document with a single embedding and ranks documents by overall semantic similarity. This compact representation is efficient, but it may hide local document aspects and may over-rank documents that match peripheral query terms rather than the core query intent.

We propose **KIPE**, a training-free dense retrieval framework that improves query-document matching through two complementary paths: a document-side pseudo-query index that exposes local document aspects as query-like representations, and a query-side keyword-aware reranking module that emphasizes candidates covering important query semantics. Experiments on five BEIR datasets show that KIPE improves the average performance of multiple dense encoders and achieves strong results against unsupervised and pseudo-query-based baselines.

1 Introduction

Dense retrieval maps queries and documents into a shared vector space and ranks documents according to vector similarity (Karpukhin et al., 2020; Reimers and Gurevych, 2019; Thakur et al., 2021). Although this paradigm has become a standard choice for retrieval systems, representing each document with a single vector can be restrictive: a document often contains multiple aspects, while a user query may focus on only one of them. At the same time, words or phrases in a query do not contribute equally to the query intent. Matching a peripheral term can produce high semantic similarity, but it may still fail to satisfy the user's core information need.

Figure 1 illustrates these two issues. On the document side, a recommendation document may cover graph modeling, relationship modeling, and cold-start solutions, while a single embedding mainly captures the dominant topic and weakens the local cold-start aspect. On the query side, a document may match words such as "coffee" and "white" but miss the core intent of removing coffee stains from a white shirt. These cases suggest that retrieval should align query intent with relevant document aspects rather than rely only on holistic similarity.

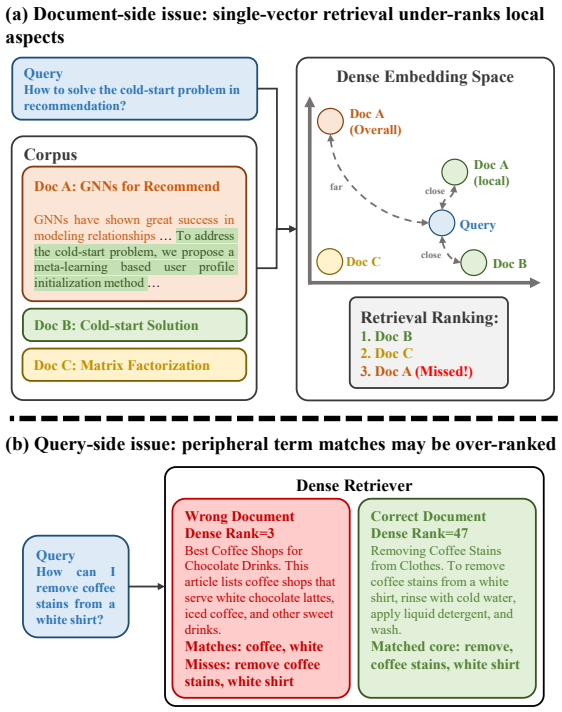

Figure 1: Motivation of KIPE. (a) Single-vector retrieval may under-rank documents for queries targeting local aspects. (b) Dense retrieval may over-rank documents that match peripheral terms rather than the core intent.

Based on this view, we propose **KIPE** (**K**eyword **I**mportance-aware **P**seudo-query **E**nhanced Dense Retrieval). KIPE performs aspect-aware matching with two lightweight, training-free operations: document-side pseudo queries expose local aspects as explicit access points, and query-side masked perturbation estimates which keywords are central to user intent. These signals are combined with the original dense score to improve ranking without

replacing the base retriever.

The main contributions of this paper are as follows:

1. We formulate dense retrieval enhancement as two-sided aspect-aware matching in a unified training-free framework.
2. We introduce independent pseudo-query indexing, allowing local document aspects to contribute while preserving original dense matching.
3. We estimate keyword importance with masked queries and convert it into a controlled keyword-coverage reward.
4. We evaluate KIPE on five BEIR datasets with multiple dense encoders, showing consistent average gains and strong performance against representative retrieval and pseudo-query-based baselines.

2 Related Work

2.1 Dense and Multi-vector Retrieval

Dense retrieval represents queries and documents in a shared embedding space and performs retrieval with vector similarity. Representative dual-encoder retrievers include DPR (Karpukhin et al., 2020), ANCE (Xiong et al., 2021), RocketQA/RocketQAv2 (Qu et al., 2021; Ren et al., 2021), Contriever (Izacard et al., 2022), and recent sentence embedding models (Ni et al., 2022; Chen et al., 2024). These methods are efficient, but a single document vector may not preserve all local aspects.

Multi-vector retrievers model finer-grained interactions. ColBERT and ColBERTv2 (Khattab and Zaharia, 2020; Santhanam et al., 2022) use token-level late interaction, while COIL (Gao et al., 2021) and CITADEL (Li et al., 2023) incorporate lexical-aware matching with efficient indexing. KIPE is complementary: it exposes document aspects through pseudo-query entries without changing the dense encoder, and further adds a query-side importance signal for reranking.

2.2 LLM-enhanced Retrieval

LLMs have been used to enhance retrieval through query-side and document-side augmentation.

Query-side augmentation. Query-side methods enrich or reinterpret the input query. HyDE (Gao et al., 2023) generates hypothetical documents and retrieves in the embedding space of the generated text, while Query2doc (Wang et al., 2023) expands queries with LLM-generated content. Other methods use LLMs as rerankers (Sun et al., 2023; Qin et al., 2024), improving relevance modeling but usually adding online inference cost.

Document-side augmentation. Document-side methods generate auxiliary text from documents to expose more possible user intents. doc2query and docT5query (Nogueira et al., 2019; Nogueira and Lin, 2019) append generated queries to documents, while InPars and Promptagator (Bonifacio et al., 2022; Dai et al., 2023) synthesize query-document pairs for training or adaptation. PQR (Kang et al., 2025) is closely related because it generates document-side potential queries and models their semantic distribution for dense retrieval, providing a strong pseudo-query-based comparison.

KIPE also uses document-side LLM-generated pseudo queries, but keeps them as independent index entries rather than only modeling a document-level distribution. It further introduces query-side masked-query keyword importance, combining document-side aspect exposure and query-side intent focusing in a training-free framework.

3 Methodology

This section introduces **KIPE**, a training-free dense retrieval framework that improves alignment between query intent and document aspects. As shown in Figure 2, KIPE contains a document-side path that exposes possible user-query perspectives through pseudo-query indexing, and a query-side path that estimates important query semantics for reranking.

3.1 Overview

The goal of dense retrieval is to identify documents whose semantics best match the user's information need. Let $E(\cdot)$ denote a dense encoder that maps any text input x to a vector representation $\mathbf{v}_x = E(x)$. For a query q and a document d, the encoder produces $\mathbf{v}_q = E(q)$ and $\mathbf{v}_d = E(d)$. We compute vector similarity with cosine similarity, denoted by $\mathrm{sim}(\mathbf{v}_a, \mathbf{v}_b)$. The original document-level dense matching score is therefore $s_{\mathrm{doc}}(q, d) = \mathrm{sim}(\mathbf{v}_q, \mathbf{v}_d)$. Throughout the paper, scores such as $s_{\mathrm{doc}}(q, d)$ are written as functions of text inputs, but their similarities are always computed between the corresponding dense vectors.

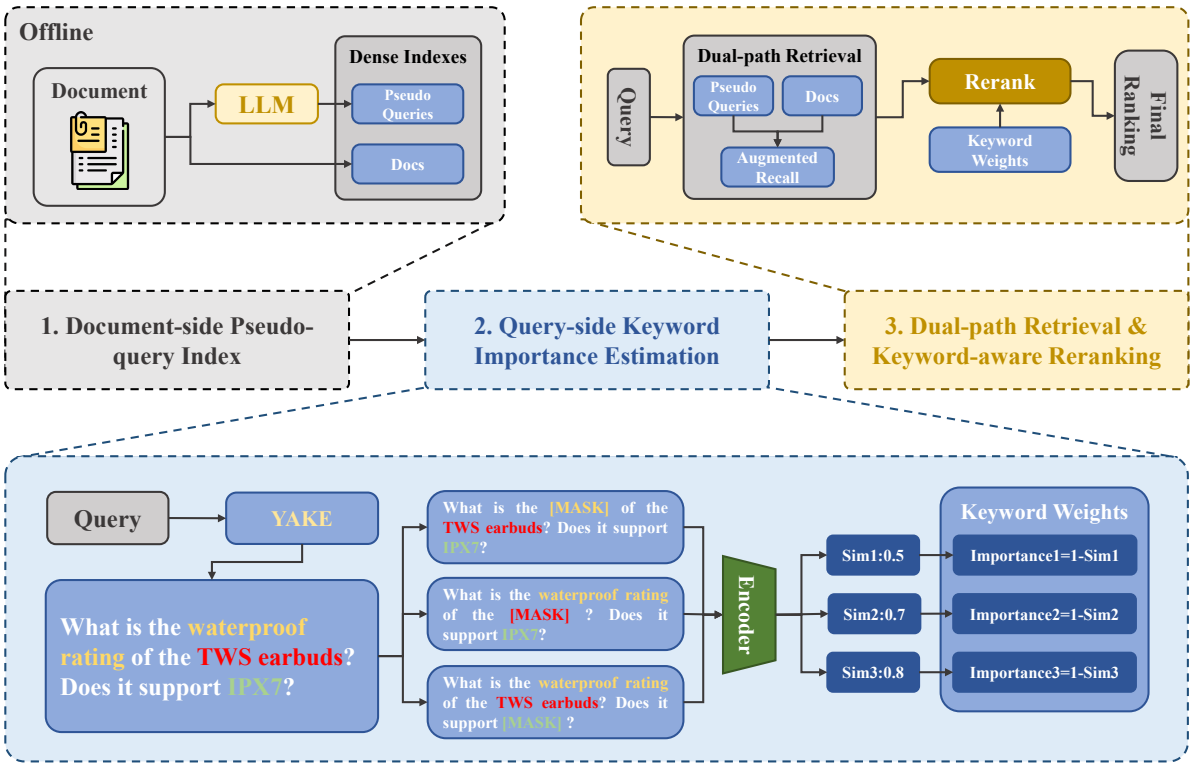

Figure 2: Framework of KIPE. KIPE retrieves documents through two complementary paths under the same aspect-aware matching goal. The document-side path builds both a document index and a pseudo-query index, so documents can be matched through holistic representations and query-like local aspect representations. The query-side path estimates raw keyword importance with masked-query similarity, normalizes it into keyword weights, and uses important keyword coverage to rerank recalled candidates.

KIPE aims to improve this basic dense retrieval objective by better aligning query intent with relevant document aspects. Instead of replacing the original dense matching signal, KIPE augments it with two complementary signals. First, document-side pseudo-query indexing makes local document aspects directly retrievable. Second, query-side keyword-aware reranking rewards documents that cover important keyword phrases in the query. Together, these two signals help KIPE promote documents that match not only the overall query semantics, but also the core intent and specific aspects expressed in the query.

3.2 Document-side Pseudo-query Index

For each document d, KIPE uses a large language model to generate a set of pseudo queries, denoted as $\mathcal{P}_d = \{p_{d,i}\}_{i=1}^{n_d}$, where $n_d \leq k$ and k is the maximum number of pseudo queries. Each pseudo query describes the document from a possible user-query perspective. Since pseudo queries are closer to real user queries than the original document text, they can amplify local semantics that may be weakened in the holistic document embedding.

After generating pseudo queries, KIPE builds two indexes. The document index stores the original document vector $\mathbf{v}_d = E(d)$ for each document, while the pseudo-query index stores the pseudo-query vectors $\mathbf{v}_{p_{d,i}} = E(p_{d,i})$ for all generated pseudo queries. Each pseudo-query entry keeps the identifier of its source document, so that a retrieved pseudo query can be mapped back to the corresponding document. Unlike document expansion methods that concatenate generated queries with the document text, KIPE treats pseudo queries as independent retrieval entries, separating holistic document matching from local aspect matching while allowing both signals to contribute to the final ranking.

3.3 Query-side Keyword Importance

Given a query q, KIPE first extracts a set of keyword phrases with YAKE (Campos et al., 2020), denoted as $\mathcal{W}_q = \{w_i\}_{i=1}^{m}$, where m is the number of extracted phrases. For each keyword phrase w_i, KIPE replaces all its surface occurrences in q with

"[MASK]" to obtain a masked query $\widehat{q}_i$. The encoder then maps the masked query to $\mathbf{v}_{\widehat{q}_i} = E(\widehat{q}_i)$. We measure the semantic change caused by masking w_i through the vector similarity between the original query representation $\mathbf{v}_q$ and the masked-query representation $\mathbf{v}_{\widehat{q}_i}$. The intuition is that removing an important keyword phrase should cause a larger semantic shift in the query representation.

We compute the raw importance score of w_i as

$$\widetilde{I}(w_i) = 1 - \mathrm{sim}(\mathbf{v}_q, \mathbf{v}_{\widehat{q}_i}).$$

The final keyword importance is normalized within the extracted keyword set:

$$I(w_i) = \frac{\widetilde{I}(w_i)}{\sum_{w_j \in \mathcal{W}_q} \widetilde{I}(w_j)}.$$

A larger $I(w_i)$ indicates that the keyword phrase contributes more to the query semantics. This normalized mask-based estimation is unsupervised and does not require keyword-level annotations.

3.4 Retrieval and Reranking

At retrieval time, KIPE first computes document-level pseudo-query-enhanced scores over the corpus. For each document d, the pseudo-query matching score is the maximum vector similarity between the query and the document's pseudo queries:

$$s_{\mathrm{pq}}(q, d) = \max_{p_{d,i} \in \mathcal{P}_d} \mathrm{sim}(\mathbf{v}_q, \mathbf{v}_{p_{d,i}}).$$

The maximum operator reflects the document-side aspect-exposure goal: a document may be relevant as long as one of its pseudo queries captures the user's information need.

Generated pseudo queries may not be completely reliable, since LLM-generated text can contain hallucinated or unfaithful content (Huang et al., 2023; Maynez et al., 2020). Therefore, KIPE uses pseudo queries as a controlled reward rather than directly replacing the original document score. The pseudo-query reward is introduced only when the pseudo-query path provides stronger matching evidence than the original document path:

$$R_{\mathrm{pq}}(q, d) = \max(0, s_{\mathrm{pq}}(q, d) - s_{\mathrm{doc}}(q, d)).$$

The initial retrieval score is

$$S_{\mathrm{init}}(q, d) = s_{\mathrm{doc}}(q, d) + \alpha R_{\mathrm{pq}}(q, d),$$

where α controls the strength of document-side pseudo-query enhancement. In our offline evaluation, we compute S_{init} for all documents, recall the top L documents according to S_{init}, and then apply keyword-aware reranking only within this recalled set. Here, L denotes the reranking depth; its value is provided in Appendix D.

KIPE further rewards documents that cover important query keyword phrases. Keyword matching is performed by lowercased surface-form matching between each unique extracted keyword phrase and the document text. A keyword phrase is considered matched only when the full phrase appears in the document. The keyword reward is

$$R_{\mathrm{kw}}(q, d) = \sum_{w_i \in \mathcal{W}_q} I(w_i) \cdot \delta(w_i, d),$$

where $\delta(w_i, d) = 1$ if document d contains keyword phrase w_i, and 0 otherwise. The final ranking score adds a controlled keyword-coverage reward to the semantic score:

$$S_{\mathrm{final}}(q, d) = S_{\mathrm{init}}(q, d) + \beta R_{\mathrm{kw}}(q, d),$$

where β controls the strength of query-side keyword-aware reranking. Documents in the recalled set are finally ranked by $S_{\mathrm{final}}(q, d)$.

4 Experiments

4.1 Setup

Datasets and Evaluation. We use five zero-shot BEIR datasets (Thakur et al., 2021): FiQA (Maia et al., 2018) for financial question answering, ArguAna (Wachsmuth et al., 2018) for argument and counterargument retrieval, SciDocs (Cohan et al., 2020) for scientific document retrieval, SciFact (Wadden et al., 2020) for scientific claim verification over abstracts, and NFCorpus (Boteva et al., 2016) for medical information retrieval. Following previous work (Thakur et al., 2021), we report nDCG@10 (Järvelin and Kekäläinen, 2002) and compute the average over the five datasets.

Baselines. We compare KIPE with several categories of baselines. Sparse retrieval baselines include BM25 (Robertson and Zaragoza, 2009), docT5query (Nogueira and Lin, 2019), and SPLADE (Formal et al., 2021). Dense retrieval baselines include DPR (Karpukhin et al., 2020), ANCE (Xiong et al., 2021), RocketQAv2 (Ren et al., 2021), Sentence-T5 (Ni et al., 2022), BGE-m3 (Chen et al., 2024), and MiniLM (Wang et al., 2020). Multi-vector retrieval baselines include ColBERTv2 (Santhanam et al., 2022), COIL (Gao

et al., 2021), and CITADEL (Li et al., 2023). LLM-assisted and pseudo-query-based baselines include HyDE (Gao et al., 2023) and PQR (Kang et al., 2025). PQR is closely related to our setting because it also uses LLM-generated document-side potential queries, while KIPE additionally introduces query-side keyword importance. To evaluate whether KIPE generalizes across encoders, we instantiate it with Sentence-T5, BGE-m3, and MiniLM. In our implementation, MiniLM refers to the Sentence-Transformers all-MiniLM-L12-v2 encoder (Reimers and Gurevych, 2019; sentence-transformers, 2021). Additional details about the source of baseline results are provided in Appendix A.5.

Implementation Details. For each document, KIPE uses Llama-3.1-8B (Grattafiori et al., 2024) to generate pseudo queries offline. We then build two dense indexes: one for original document embeddings and the other for pseudo-query embeddings. Unless otherwise specified, we use a fixed default configuration with at most $k = 100$ pseudo queries, pseudo-query reward coefficient $\alpha = 0.5$, and keyword-aware reranking coefficient $\beta = 0.05$. These values are kept unchanged across all datasets and encoders in the main experiments. Section 5.1 reports a sensitivity analysis of these hyperparameters after the main evaluation, and the searched ranges are listed in Appendix D. For keyword importance estimation, we use YAKE (Campos et al., 2020) to extract the top 5 query keyword phrases and compute their normalized importance scores by comparing the original query embedding with the corresponding masked-query embeddings. Further pseudo-query generation and keyword processing details are provided in Appendix A.1 and Appendix A.3.

4.2 Retrieval Performance

Table 1 shows the zero-shot retrieval results on BEIR. Superscript arrows indicate the difference from the corresponding original dense encoder. For KIPE, a bold arrow indicates that its change relative to the original encoder is better than that of the corresponding pseudo-query variant under the same encoder, including either a larger improvement or a smaller decrease.

KIPE improves the average performance of all three base dense retrievers. With Sentence-T5, KIPE improves the average nDCG@10 from 0.336 to 0.379. With BGE-m3, KIPE improves the average score from 0.414 to 0.435 and outperforms $\text{PQR}_{\text{BGE-m3}}$ on all five datasets. With MiniLM, KIPE achieves the best average score among all compared methods, improving the average nDCG@10 from 0.402 to 0.445 and outperforming $\text{PQR}_{\text{MiniLM}}$.

Pseudo-query-based results provide the closest comparison because PQR also relies on document-side LLM-generated potential queries. KIPE uses fewer generated queries (at most 100 vs. 300 in PQR) yet achieves higher averages for all three encoders, improving over PQR by 0.015, 0.023, and 0.012 nDCG@10 with Sentence-T5, BGE-m3, and MiniLM. This suggests that independent pseudo-query indexing plus query-side importance can be more effective than adding more document-side queries alone.

The results also show that KIPE is less stable on ArguAna than on the other datasets. ArguAna focuses on argument retrieval, where relevance can depend on stance and counterargument relations rather than only topical or surface keyword coverage. This suggests that the current surface-form keyword reward may be less suitable for stance-sensitive retrieval, and explains why KIPE may underperform the pseudo-query baseline or the base encoder on this dataset while still improving the overall average.

4.3 Ablation Study

To investigate whether the two paths in KIPE provide complementary benefits, we conduct an ablation study using all-MiniLM-L12-v2 as the backbone encoder. We compare four settings: Dense Only, which uses only the original dense retrieval score; $\text{KIPE}_{\text{w/o PQ}}$, which removes the document-side pseudo-query path; $\text{KIPE}_{\text{w/o KW}}$, which removes the query-side keyword-aware path; and KIPE, which uses both paths. Here, PQ denotes pseudo-query enhancement and KW denotes keyword-aware reranking. All KIPE variants use at most $k = 100$ pseudo queries. Details of the ablation implementation are provided in Appendix A.4.

As shown in Table 2, both paths improve the Dense Only setting. $\text{KIPE}_{\text{w/o PQ}}$ improves the average nDCG@10 from 0.402 to 0.418, indicating that query-side keyword importance can adjust rankings toward documents that cover important query terms. $\text{KIPE}_{\text{w/o KW}}$ brings a larger improvement, increasing the average score from 0.402 to 0.440, which confirms that pseudo-query indexing provides effective document-side aspect evidence.

Type	Model	FiQA	ArguAna	SciDocs	SciFact	NFCorpus	Avg.
Sparse	BM25	0.236	0.315	0.158	0.665	0.325	0.340
Sparse	docT5query	0.291	0.349	0.162	0.675	0.328	0.361
Dense	DPR	0.224	0.323	0.103	0.479	0.244	0.275
Dense	ANCE	0.295	0.415	0.122	0.507	0.237	0.315
Dense	RocketQAv2	0.302	0.451	0.131	0.568	0.293	0.349
Sparse	SPLADE	0.289	0.445	0.149	0.633	0.322	0.368
Multi-vector	ColBERTv2	0.356	0.463	0.154	0.693	0.338	0.401
Multi-vector	COIL	0.313	0.294	0.155	0.707	0.331	0.360
Multi-vector	CITADEL	0.332	0.490	0.147	0.695	0.337	0.400
LLM-assisted	$\text{HyDE}_{\text{Llama-8B}}$	0.220	0.299	0.133	0.639	0.312	0.321
Dense	Sentence-T5	0.348	0.448	0.142	0.458	0.286	0.336
Pseudo-query	$\text{PQR}_{\text{Sentence-T5}}$	$0.360^{\uparrow 0.012}$	$0.504^{\uparrow 0.056}$	$0.149^{\uparrow 0.007}$	$0.534^{\uparrow 0.076}$	$0.272^{\downarrow 0.014}$	$0.364^{\uparrow 0.028}$
Ours	$\text{KIPE}_{\text{Sentence-T5}}$	$0.373^{\uparrow \mathbf{0.025}}$	$0.445^{\downarrow 0.003}$	$0.162^{\uparrow \mathbf{0.020}}$	$0.578^{\uparrow \mathbf{0.120}}$	$0.340^{\uparrow \mathbf{0.054}}$	$0.379^{\uparrow \mathbf{0.043}}$
Dense	BGE-m3	0.410	0.540	0.163	0.644	0.314	0.414
Pseudo-query	$\text{PQR}_{\text{BGE-m3}}$	$0.403^{\downarrow 0.007}$	$0.478^{\downarrow 0.062}$	$0.178^{\uparrow 0.015}$	$0.669^{\uparrow 0.025}$	$0.334^{\uparrow 0.020}$	$0.412^{\downarrow 0.002}$
Ours	$\text{KIPE}_{\text{BGE-m3}}$	$0.412^{\uparrow \mathbf{0.002}}$	$0.519^{\downarrow \mathbf{0.021}}$	$0.180^{\uparrow \mathbf{0.017}}$	$0.720^{\uparrow \mathbf{0.076}}$	$0.347^{\uparrow \mathbf{0.033}}$	$0.435^{\uparrow \mathbf{0.021}}$
Dense	MiniLM	0.373	0.471	0.218	0.626	0.323	0.402
Pseudo-query	$\text{PQR}_{\text{MiniLM}}$	$0.401^{\uparrow 0.028}$	$0.521^{\uparrow 0.050}$	$0.208^{\downarrow 0.010}$	$0.688^{\uparrow 0.062}$	$0.348^{\uparrow 0.025}$	$0.433^{\uparrow 0.031}$
Ours	$\text{KIPE}_{\text{MiniLM}}$	$0.427^{\uparrow \mathbf{0.054}}$	$0.496^{\uparrow 0.025}$	$0.226^{\uparrow \mathbf{0.008}}$	$0.721^{\uparrow \mathbf{0.095}}$	$0.356^{\uparrow \mathbf{0.033}}$	$0.445^{\uparrow \mathbf{0.043}}$

Table 1: Zero-shot BEIR results evaluated by nDCG@10 (Thakur et al., 2021; Järvelin and Kekäläinen, 2002). KIPE uses at most $k = 100$ pseudo queries, $\alpha = 0.5$, and $\beta = 0.05$, while PQR uses 300 generated potential queries per document. Arrows show changes from the corresponding dense encoder; bold arrows on KIPE rows mark changes better than the corresponding pseudo-query baseline under the same encoder.

Variant	FQA	Arg	SDoc	SFact	NFC	Avg.
Dense Only	0.373	0.471	0.218	0.626	0.323	0.402
$\text{KIPE}_{\text{w/o PQ}}$	0.404	0.478	0.221	0.660	0.329	0.418
$\text{KIPE}_{\text{w/o KW}}$	0.425	0.490	**0.226**	0.712	0.348	0.440
KIPE	**0.427**	**0.496**	**0.226**	**0.721**	**0.356**	**0.445**

Table 2: Ablation study of KIPE using all-MiniLM-L12-v2 (Wang et al., 2020; sentence-transformers, 2021). FQA, Arg, SDoc, SFact, and NFC denote FiQA, ArguAna, SciDocs, SciFact, and NFCorpus, respectively. PQ denotes pseudo-query enhancement and KW denotes keyword-aware reranking. Each ablated variant removes one module from the full KIPE framework. The best result in each column is bolded.

The full KIPE model further improves the average score to 0.445. Compared with $\text{KIPE}_{\text{w/o KW}}$, adding keyword-aware reranking brings an additional improvement of 0.005. These results show that the two components are not redundant: pseudo-query indexing increases the chance of recalling and promoting documents through local aspect matches, while keyword-aware reranking refines the ranking by focusing on core query semantics.

5 Model Analysis

This section analyzes how KIPE's key components and hyperparameters affect retrieval performance. To avoid conclusions tied to a single encoder, parameter sensitivity analysis averages results over Sentence-T5, BGE-m3, and all-MiniLM-L12-v2. Unless otherwise specified, when one hyperparameter is varied, the others are fixed to the main default configuration: $k = 100$, $\alpha = 0.5$, and $\beta = 0.05$. Detailed ranges are given in Appendix D.

5.1 Parameter Sensitivity Analysis

We analyze the number of pseudo queries k, the pseudo-query reward coefficient α, and the keyword-aware reward coefficient β. This analysis is intended to show how KIPE behaves under different settings. It is not used to select dataset-specific or encoder-specific hyperparameters for the main results.

Number of pseudo queries. Figure 3(a) shows that increasing k improves retrieval performance, from 0.403 at $k = 5$ to 0.420 at $k = 100$. Most gains appear before $k = 30$, suggesting that a moderate number of pseudo queries already covers many local aspects.

Pseudo-query reward strength. Figure 3(b) shows that increasing α from 0 to 0.5 improves average nDCG@10 from 0.397 to 0.420, but larger values hurt performance, dropping to 0.380 at $\alpha = 1.0$. A moderate reward corrects under-ranked local aspects, whereas an excessive reward may over-amplify generated queries.

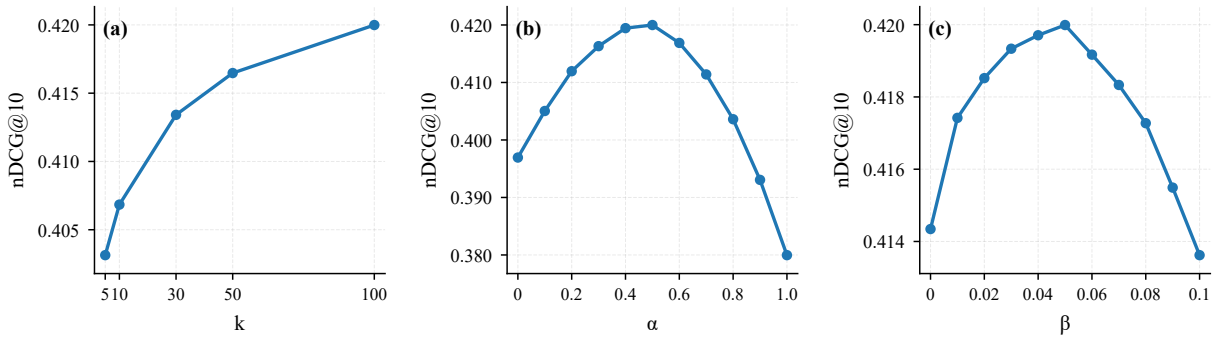

Figure 3: Parameter sensitivity analysis of KIPE. Results are averaged over Sentence-T5, BGE-m3, and all-MiniLM-L12-v2. Each plot varies one hyperparameter and keeps the others at their main default values.

LLM Generator	SciFact	NFCorpus	Avg.
Qwen3-8B	0.692	0.357	0.525
Qwen3-14B	0.716	0.364	0.540
Qwen3-32B	0.734	0.377	0.555

Table 3: Effect of LLM generator size. The dense encoder is fixed to all-MiniLM-L12-v2, and the Avg. score is computed over the two reported datasets.

Keyword-aware reward strength. Figure 3(c) shows that $\beta = 0.05$ improves average nDCG@10 from 0.414 to 0.420. Since top dense-retrieval candidates often have small score gaps, a lightweight keyword reward can adjust local ordering, while too large a reward overemphasizes exact matches.

5.2 Effect of LLM Generator Size

We also examine whether stronger LLM generators produce better pseudo queries. We fix the dense encoder to all-MiniLM-L12-v2 and vary the pseudo-query generator among Qwen3 models (Qwen Team, 2025). Details of the pseudo-query generation setup are described in Appendix A.1.

As shown in Table 3, larger generators improve average nDCG@10 from 0.525 with Qwen3-8B to 0.555 with Qwen3-32B, suggesting that stronger generators produce pseudo queries that better expose document-side aspects. This cost is incurred offline and does not require online LLM calls.

5.3 Case Study

The following cases illustrate how the two paths in KIPE serve the same aspect-aware matching objective. The first case shows how pseudo-query indexing exposes a local document aspect that is hidden in the holistic embedding. The second case shows how keyword-aware reranking corrects a candidate order when the initial semantic score favors a document that misses the core query phrase.

Query: Trade? Buy and hold? Or both?
Document snippet: "You mentioned three concepts: (1) trading (2) diversification (3) buy and hold. ... **Buy and hold is the only reasonable approach to a portfolio for someone who is not interested in spending a lot of time managing it.**"
Best matched pseudo query: Trading vs buy and hold investment approach
Rank change (dense → pseudo-query enhanced → final): 1913 → 9 → 4 **Scores:** $s_{\text{doc}} = 0.300$, $s_{\text{pq}} = 0.796$, $S_{\text{init}} = 0.548$, $S_{\text{final}} = 0.582$

Table 4: Case study of pseudo-query enhanced retrieval. The pseudo query captures the local "trading vs. buy-and-hold" semantics of a multi-aspect document.

Pseudo-query enhanced retrieval. For the query "Trade? Buy and hold? Or both?", the relevant document discusses multiple investment aspects, including trading, diversification, and buy-and-hold strategies. The original dense retriever ranks it only at 1913 because the query-related local aspect is weakened in the whole-document embedding. KIPE generates a pseudo query, "Trading vs buy and hold investment approach", which directly matches the user query. As a result, the document is promoted to rank 9 after pseudo-query enhanced retrieval and finally ranks 4.

The main promotion in this case comes from the pseudo-query path rather than keyword matching alone: the generated pseudo query explicitly represents a local aspect of the document that the holistic embedding underestimates.

Keyword-aware reranking. For the query "Is investing exclusively in a small-cap index fund a wise investment?", Candidate A has a higher initial rank but does not cover the key phrase "small-cap index fund". Candidate B is initially ranked lower,

Query: Is investing exclusively in a small-cap index fund a wise investment?
Candidate A (non-relevant, initial rank 11 → final rank 16): "Like an S&P 500 ETF? ... you have to buy and sell 500 different stocks ..." **Covered keywords:** none; $R_{\text{kw}} = 0.000$
Candidate B (relevant, initial rank 46 → final rank 13): "The market will not reward you for owning a **small-cap index fund** when you can purchase a globally diversified ... **index fund** at no additional cost." **Covered keywords:** small-cap index fund, small-cap index, index fund; $R_{\text{kw}} = 0.757$
Score change: Candidate A $0.659 \rightarrow 0.659$; Candidate B $0.623 \rightarrow 0.661$

Table 5: Case study of keyword-aware reranking. Candidate B is promoted because it covers important query keywords, while Candidate A only has a higher initial semantic score.

but directly discusses small-cap index funds and receives a larger keyword reward. KIPE therefore promotes Candidate B from rank 46 to rank 13, while Candidate A is demoted from rank 11 to rank 16.

This case shows the role of the query-side path: the keyword reward does not replace semantic retrieval, but corrects the ordering of candidates when the initial score favors a document that matches peripheral semantics while missing the core query phrase. Together, the two cases illustrate why KIPE uses both paths to align query intent with document aspects.

6 Conclusion

We propose KIPE, a training-free dense retrieval framework that improves query-document matching from both sides. Under a unified aspect-aware view, KIPE uses document-side pseudo-query indexing to expose local document aspects and query-side keyword-aware reranking to emphasize candidates covering core query semantics.

Experiments on five BEIR datasets show that KIPE improves multiple dense encoders on average and outperforms strong pseudo-query-based baselines while using fewer generated queries. Further analysis confirms that pseudo-query matching and keyword-aware reranking provide complementary benefits, and that both should be used as controlled auxiliary signals rather than replacements for the original dense matching score.

Limitations

KIPE requires generating pseudo queries for each document, introducing additional offline preprocessing and indexing cost. Since this cost is incurred before retrieval and KIPE uses fewer generated queries than comparable pseudo-query methods, we expect it to remain practical as LLM inference and indexing techniques continue to improve.

KIPE also depends on the quality of generated pseudo queries and extracted keywords. Noisy pseudo queries or surface-form keyword matching may introduce imperfect signals. However, KIPE uses pseudo queries only as a controlled reward and treats keyword matching as a lightweight reranking signal, which helps keep the framework efficient and interpretable.

References

Luiz Henrique Bonifacio, Hugo Queiroz Abonizio, Marzieh Fadaee, and Rodrigo Nogueira. 2022. InPars: Data augmentation for information retrieval using large language models. *arXiv preprint arXiv:2202.05144*.

Vera Boteva, Demian Gholipour Ghalandari, Artem Sokolov, and Stefan Riezler. 2016. A full-text learning to rank dataset for medical information retrieval. In *Advances in Information Retrieval*, volume 9626 of *Lecture Notes in Computer Science*, pages 716–722. Springer.

Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Mário Jorge, Célia Nunes, and Adam Jatowt. 2020. YAKE! keyword extraction from single documents using multiple local features. *Information Sciences*, 509:257–289.

Jianlyu Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. M3-embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. In *Findings of the Association for Computational Linguistics: ACL 2024*. Association for Computational Linguistics.

Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel Weld. 2020. SPECTER: Document-level representation learning using citation-informed transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2270–2282. Association for Computational Linguistics.

Zhuyun Dai, Vincent Y. Zhao, Ji Ma, Yi Luan, Jianmo Ni, Jing Lu, Anton Bakalov, Kelvin Guu, Keith B. Hall, and Ming-Wei Chang. 2023. Promptagator: Few-shot dense retrieval from 8 examples. In *International Conference on Learning Representations*.

Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. 2021. SPLADE: Sparse lexical and expansion model for first stage ranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2288–2292.

Luyu Gao, Zhuyun Dai, and Jamie Callan. 2021. COIL: Revisit exact lexical match in information retrieval with contextualized inverted list. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2023. Precise zero-shot dense retrieval without relevance labels. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, pages 1762–1777. Association for Computational Linguistics.

Aaron Grattafiori and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

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 preprint arXiv:2311.05232*.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research*.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, 20(4):422–446.

Junfeng Kang, Rui Li, Qi Liu, Yanjiang Chen, Zheng Zhang, Junzhe Jiang, Heng Yu, and Yu Su. 2025. PQR: Improving dense retrieval via potential query modeling. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, pages 13455–13469. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 6769–6781. Association for Computational Linguistics.

Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and effective passage search via contextualized late interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 39–48.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pages 611–626.

Minghan Li, Sheng-Chieh Lin, Barlas Oguz, Asish Ghoshal, Jimmy Lin, Yashar Mehdad, Wen-tau Yih, and Xilun Chen. 2023. CITADEL: Conditional token interaction via dynamic lexical routing for efficient and effective multi-vector retrieval. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Macedo Maia, Siegfried Handschuh, André Freitas, Brian Davis, Ross McDermott, Manel Zarrouk, and Alexandra Balahur. 2018. WWW'18 open challenge: Financial opinion mining and question answering. In *Companion Proceedings of the The Web Conference 2018*, pages 1941–1942.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919.

Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. 2022. Sentence-T5: Scalable sentence encoders from pre-trained text-to-text models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1864–1874. Association for Computational Linguistics.

Rodrigo Nogueira and Jimmy Lin. 2019. From doc2query to doctttttquery. Online manuscript.

Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375*.

Zhen Qin, Rolf Jagerman, Kai Hui, Honglei Zhuang, Junru Wu, Le Yan, Jiaming Shen, Tianqi Liu, Jialu Liu, Donald Metzler, Xuanhui Wang, and Michael Bendersky. 2024. Large language models are effective text rankers with pairwise ranking prompting. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1504–1518. Association for Computational Linguistics.

Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Wayne Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. 2021. RocketQA: An optimized training approach to dense passage retrieval for open-domain question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 5835–5847. Association for Computational Linguistics.

Qwen Team. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 3982–3992. Association for Computational Linguistics.

Ruiyang Ren, Yingqi Qu, Jing Liu, Wayne Xin Zhao, Qiaoqiao She, Hua Wu, Haifeng Wang, and Ji-Rong Wen. 2021. RocketQAv2: A joint training method for dense passage retrieval and passage re-ranking. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2825–2835. Association for Computational Linguistics.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389.

Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 2022. ColBERTv2: Effective and efficient retrieval via lightweight late interaction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics.

sentence-transformers. 2021. all-MiniLM-L12-v2. Hugging Face model card. Accessed 2026-05-23.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. 2023. Is ChatGPT good at search? investigating large language models as re-ranking agents. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14918–14937. Association for Computational Linguistics.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Henning Wachsmuth, Shahbaz Syed, and Benno Stein. 2018. Retrieval of the best counterargument without prior topic knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 241–251. Association for Computational Linguistics.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. 2020. Fact or fiction: Verifying scientific claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 7534–7550. Association for Computational Linguistics.

Liang Wang, Nan Yang, and Furu Wei. 2023. Query2doc: Query expansion with large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9414–9423. Association for Computational Linguistics.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Advances in Neural Information Processing Systems*, volume 33, pages 5776–5788.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *International Conference on Learning Representations*.

A Implementation Details

A.1 Pseudo-query Generation

For pseudo-query generation, we deploy Llama-3.1-8B with vLLM (Kwon et al., 2023) on two Tesla V100 GPUs, each with 32GB memory. The decoding temperature is set to 0. For each document, we generate pseudo queries offline before retrieval. If a document exceeds the generator input limit, we directly truncate the input to the maximum allowed length.

When constructing the text input for pseudo-query generation and retrieval, we concatenate the document title and body as `title\nbody`. After generation, we remove empty lines and duplicate pseudo queries. If the model does not output exactly `[K]` valid lines, where `[K]` corresponds to the maximum pseudo-query number k, we use all valid pseudo queries after cleaning. Therefore, the actual number of pseudo queries used by KIPE can be smaller than the specified maximum, suggesting that stronger output compliance or generation control may further improve performance.

A.2 Similarity Computation

Unless otherwise specified, all similarity scores in KIPE are instantiated with cosine similarity between dense vectors. Under each encoder setting, we use the same encoder model $E(\cdot)$ for all embedding-based computations, including document-query dense retrieval, pseudo-query-query dense retrieval, and masked-query similarity for keyword importance estimation. For example, under the all-MiniLM-L12-v2 setting, $s_{\mathrm{doc}}(q, d)$, $s_{\mathrm{pq}}(q, d)$, and $\widetilde{I}(w_i)$ are all computed with all-MiniLM-L12-v2 embeddings.

A.3 Keyword Extraction, Importance, and Matching

For each query, we use YAKE (Campos et al., 2020) with `language=en` and maximum n-gram size 3 to extract the top 5 keyword phrases. For each extracted keyword phrase w_i, we replace all its surface occurrences in the query with "[MASK]" to obtain $\widehat{q}_i$, encode both q and $\widehat{q}_i$ with the dense encoder $E(\cdot)$, compute the raw importance score $\widetilde{I}(w_i) = 1 - \mathrm{sim}(\mathbf{v}_q, \mathbf{v}_{\widehat{q}_i})$, and normalize the raw scores within the query. These normalized scores are used for keyword-aware reranking.

For keyword coverage computation, we use lowercased direct surface-form matching between each unique extracted keyword phrase and document text. A keyword phrase is considered matched only when the full extracted phrase appears in the document. We treat extracted keywords as a set during coverage computation, so repeated or overlapping keyword phrases are not double counted.

A.4 Ablation Implementation

The ablation variants in Table 2 are implemented by disabling the corresponding scoring modules. Dense Only removes both the pseudo-query enhancement module and the keyword-aware reranking module. $\text{KIPE}_{\text{w/o PQ}}$ disables pseudo-query enhancement while keeping keyword-aware reranking. $\text{KIPE}_{\text{w/o KW}}$ disables keyword-aware reranking while keeping pseudo-query enhancement. In implementation, these variants are controlled by setting the corresponding coefficients α and β to zero.

A.5 Baseline Results

The comparison baseline results in Table 1 are taken from the cited papers rather than reproduced by us. In particular, the HyDE results are taken from PQR (Kang et al., 2025). For KIPE and our dense encoder variants, pseudo-query generation and retrieval use the title-body concatenation format described in Appendix A.1.

B Effect of Keyword Extraction Methods

We further compare different keyword extraction methods to analyze their effect on KIPE. Except for the keyword extraction method, all other configurations are kept the same as the main experimental setting. The dense encoder is fixed to all-MiniLM-L12-v2. The comparison is conducted on SciFact. For each method, we extract the top 5 keywords.

Keyword Extractor	SciFact
Llama-3.1-8B	0.72082
YAKE	0.72082
TF-IDF	0.71795

Table 6: Effect of different keyword extraction methods on SciFact. All other configurations are the same as the main experimental setting, and the dense encoder is fixed to all-MiniLM-L12-v2.

We evaluate three keyword extraction methods: Llama-3.1-8B, which uses Llama-3.1-8B to extract the keywords from the query; YAKE, which uses the same configuration as the main experiments; and TF-IDF, which uses TF-IDF with unigram features.

As shown in Table 6, Llama-3.1-8B and YAKE achieve the same best performance on SciFact, while TF-IDF performs slightly worse. Considering both computational cost and retrieval performance, we choose YAKE as the keyword extraction method in the main experiments.

For Llama-3.1-8B keyword extraction, we use the following prompt. The placeholder `[K]` is replaced by 5, and `[SENTENCE]` is replaced by the input query.

Keyword Extraction Prompt

Extract keywords from the sentence, identify the top `[K]` most important keywords/entities.
Return ONLY a JSON array (list).
Rules:
* No explanation
* Only JSON array
* Each item must be exact text from the sentence
Example:
Input:
"Find cheap Italian restaurants in New York open after 10pm this weekend under 50 dollars"
Output:
["Italian restaurants",
"New York",
"under 50 dollars",
"open after 10pm",
"this weekend"]
Sentence:
`[SENTENCE]`

C Prompting Template

We use the following prompt to generate pseudo queries for each document. The placeholder `[K]` is replaced by the number of pseudo queries, corresponding to k in the main text, and `[DOCUMENT]` is replaced by the document text.

Pseudo-query Generation Prompt

You are an expert in information retrieval and query generation.
Your task is to generate search queries for a given document.
Goal:
The generated queries should be optimized for retrieving the given document from a large corpus. Each query should be realistic, diverse, and reflect how real users might search for this document.
Requirements:
1. Generate exactly [K] queries.
2. Queries must be written in natural, fluent English.
3. Each query should capture a different aspect, phrasing, or intent related to the document.
4. Include a mix of:
   * keyword-style queries
   * natural question-style queries
   * paraphrases and semantic variations
5. Avoid copying full sentences directly from the document.
6. Queries should vary in length (short, medium, long).
7. Focus on retrieval effectiveness: queries should be likely to match this document better than others.
STRICT OUTPUT RULES (CRITICAL):
* Return ONLY the queries.
* Do NOT include any explanations, comments, numbering, or additional text.
* Output exactly [K] lines.
* Each line must contain one query.
* Separate each query with a newline character.
Document:
[DOCUMENT]

D Hyperparameter Settings

The main experiments use a fixed default configuration with $k = 100$, $\alpha = 0.5$, and $\beta = 0.05$. These hyperparameters are kept the same for all datasets and encoders in the main results. The following ranges are used only for sensitivity analysis and are not used to select dataset-specific or encoder-specific settings.

The pseudo-query number analysis uses $k \in \{5, 10, 30, 50, 100\}$.

The pseudo-query reward analysis uses $\alpha \in \{0, 0.1, 0.2, \ldots, 1.0\}$.

The keyword-aware reward analysis uses $\beta \in \{0, 0.01, 0.02, \ldots, 0.10\}$.

The reranking depth is set to 100 in the main experiments. Specifically, we compute pseudo-query-enhanced scores over the corpus, recall the top 100 documents according to S_{init}, and then apply keyword-aware reranking to these documents.

E Example Display

We provide examples from the NFCorpus dataset to illustrate the generated pseudo queries and the extracted keyword importance used by KIPE. The pseudo queries are generated using the setting described in Appendix A.1, and we display the first 5 generated pseudo queries. The keyword importance scores are computed with all-MiniLM-L12-v2.

Document ID: MED-306

Sensitivity of Continuous Performance Test (CPT) at Age 14 Years to Developmental Methylmercury Exposure
Hit Reaction Time latencies (HRT) in the Continuous Performance Test (CPT) measure the speed of visual information processing. Prenatal methylmercury exposure is associated with increased reaction time latencies. A total of 878 adolescents completed the CPT. The results suggest that sustained attention is particularly vulnerable to developmental methylmercury exposure, indicating probable underlying dysfunction of the frontal lobes.

Generated Pseudo Queries

1. CPT sensitivity to methylmercury exposure at age 14
2. How does prenatal methylmercury affect CPT reaction times?
3. Association between methylmercury levels and HRT in adolescents
4. Impact of developmental methylmercury on sustained attention in children
5. Neurotoxic effects of methylmercury on frontal lobe function as measured by CPT

Query and Keyword Importance

Query ID: PLAIN-23
Query: How to Reduce Exposure to Alkylphenols Through Your Diet
Extracted keywords and importance scores:
1. Exposure to Alkylphenols 0.337
2. Diet 0.206
3. Reduce Exposure 0.185
4. Exposure 0.175
5. Reduce 0.097

F Reproducibility Notes

Pseudo queries are generated before retrieval. KIPE does not require online LLM calls during inference. The online stage consists of dense retrieval over the document and pseudo-query indexes, followed by lightweight keyword-aware reranking.