PQR: Improving Dense Retrieval via Potential Query Modeling

Junfeng Kang[1], Rui Li[1], Qi Liu[1,2]*, Yanjiang Chen[1]
Zheng Zhang[1], Junzhe Jiang[1], Heng Yu[1], Yu Su [2,3]
[1]State Key Laboratory of Cognitive Intelligence, University of Science and Technology of China
[2]Institute of Artificial Intelligence, Hefei Comprehensive National Science Center
[3]School of Computer Science and Artificial Intelligence, Hefei Normal University
{kangjf, ruili2000, yjchen}@mail.ustc.edu.cn, qiliuql@ustc.edu.cn
{zhangzheng, jzjiang, yh112358_1321}@mail.ustc.edu.cn, yusu@hfnu.edu.cn

Abstract

Dense retrieval has now become the mainstream paradigm in information retrieval. The core idea of dense retrieval is to align document embeddings with their corresponding query embeddings by maximizing their dot product. The current training data is quite sparse, with each document typically associated with only one or a few labeled queries. However, a single document can be retrieved by multiple different queries. Aligning a document with just one or a limited number of labeled queries results in a loss of its semantic information. In this paper, we propose a training-free **P**otential **Q**uery **R**etrieval (**PQR**) framework to address this issue. Specifically, we use a Gaussian mixture distribution to model ***all*** potential queries for a document, aiming to capture its comprehensive semantic information. To obtain this distribution, we introduce three sampling strategies to sample a large number of potential queries for each document and encode them into a semantic space. Using these sampled queries, we employ the Expectation-Maximization algorithm to estimate parameters of the distribution. Finally, we also propose a method to calculate similarity scores between user queries and documents under the PQR framework. Extensive experiments demonstrate the effectiveness of the proposed method.[1]

1 Introduction

Dense retrieval aims to find the most relevant documents from a large corpus based on a user's natural language query. In this approach, both queries and documents are encoded into dense vectors, and their similarity score is calculated using dot product. It plays a key role in downstream applications such as Retrieval-Augmented Generation (Lewis et al., 2020) , Question Answering (Karpukhin et al., 2020; Sun et al., 2024) and Intelligent Education (Liu et al., 2021).

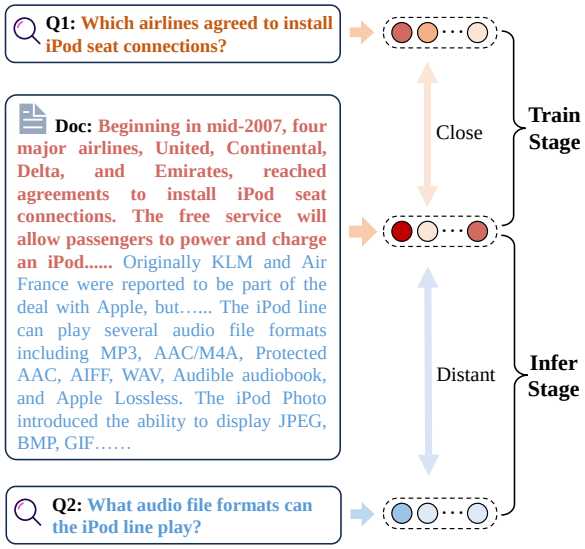

Figure 1: During the training phase, the document Doc is aligned semantically only with the query Q1. However, during the inference phase, queries like Q2, which should ideally retrieve the document Doc, end up having a greater semantic distance from it.

Dense retrieval heavily relies on annotated training data (Thakur et al., 2021). In existing datasets, a document is typically associated with only one or a few annotated queries. However, there is a many-to-one relationship (Zhang et al., 2022b) between queries and documents: different potential queries can retrieve the same document. Aligning a document with only one corresponding query can result in the loss of the document's semantic information (Luan et al., 2021). As illustrated in Figure 1, if the document *Doc* is aligned with query *Q1* during training, the information in the latter part of the document might be overlooked. At inference time, when the system encounters a potential user query like *Q2*, it might fail to retrieve the document *Doc* because *Doc* aligns more closely with the semantic information of *Q1*. This will inevitably result in

*Corresponding author.
[1]The source code is released at https://github.com/tojunfeng/PQR

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 13455–13469
July 27 - August 1, 2025

a drop in retrieval performance. If ***all*** potential queries of a document are available, they can be leveraged to better measure the similarity between the user query and the document. This is because all potential queries of a document can comprehensively capture its semantic information, thereby preventing any loss of document information.

In this paper, we take a fresh look at this problem from the perspective of the potential queries. A document can potentially correspond to an ***infinite*** number of queries. Then, the problem is ***how can we model the infinite potential queries of a document?*** The traditional approach to document modeling employs vector representations (Karpukhin et al., 2020), but it struggles to represent infinite queries. Probability distributions, on the other hand, naturally accommodate infinite data. This fundamental understanding leads us to model all potential queries through distribution.

Based on the idea of distribution modeling, the following two questions arise: how to ***determine*** the parameters of the distribution? And, how to ***measure*** the similarity between a user query and a document? To address these questions, we introduce the **P**otential **Q**uery **R**etrieval (**PQR**) framework, which models the potential queries using a Gaussian mixture distribution.

To determine distribution parameters, PQR employs the Expectation-Maximization algorithm to estimate the parameters of the distribution based on query samples. In order to acquire these query samples, we leverage a large language model to sample potential queries, and propose three sampling strategies: Zero-shot sampling, Sliding-window sampling, and Topic-aware sampling. For the question of measuring similarity between a user query and a document, PQR calculates the similarity score by taking the maximum dot product between the user query vector and the mean vector of each Gaussian distribution component in the document.

Compared to traditional dense retrieval methods, PQR offers the following advantages: **(1) Training-free Framework**: PQR is an unsupervised, training-free approach that eliminates the reliance on labeled data. **(2) Zero-shot Retrieval Capability**: PQR demonstrates strong domain adaptation, maintaining high performance in zero-shot retrieval scenarios.

We conduct extensive experiments on a range of zero-shot datasets within BEIR (Thakur et al., 2021), as well as large-scale web search datasets such as MSMARCO Document Ranking (Bajaj et al., 2016) and the TREC Deep Learning Track dataset (Craswell et al., 2020). The results demonstrate the effectiveness of the proposed PQR framework. Our main contributions can be summarized as follows:

- We propose a novel, training-free retrieval framework that models potential queries of a document as Gaussian mixture distribution, enabling it to comprehensively capture the document's semantic information.
- We introduce three sampling strategies for generating potential queries from documents.
- We propose a method to calculate similarity scores between user queries and documents under the PQR framework.
- We demonstrate the effectiveness of our approach via comprehensive experiments across various datasets.

2 Related Work

2.1 Dense Retrieval

Dense retrieval involves encoding both queries and documents into dense vectors, then calculating their relevance using methods like dot product or cosine similarity. Pre-training followed by fine-tuning is the mainstream paradigm for dense retrieval. To bridge the gap between pre-training tasks (Zhang et al., 2022a) and retrieval tasks, various approaches have been employed, such as weak decoders (Lu et al., 2021), masked autoencoders (Wang et al., 2023a; Xiao et al., 2022), and information bottleneck techniques (Gao and Callan, 2021).

During the fine-tuning phase, methods like negative sample mining (Xiong et al., 2020), knowledge distillation (Ren et al., 2021) and multi-vector retrieval methods (Santhanam et al., 2022; Zhang et al., 2022b) have also been extensively studied. However, these dense retrieval methods rely heavily on labeled data for training and face challenges in zero-shot retrieval scenarios.

In contrast, the PQR approach adopts a radically different method that not only eliminates the need for any labeled data but also demonstrates strong zero-shot retrieval performance.

2.2 LLM-assisted Retrieval

With the advancement of large language models (Mao et al., 2024), interest grows in using their reasoning and generation capabilities for retrieval

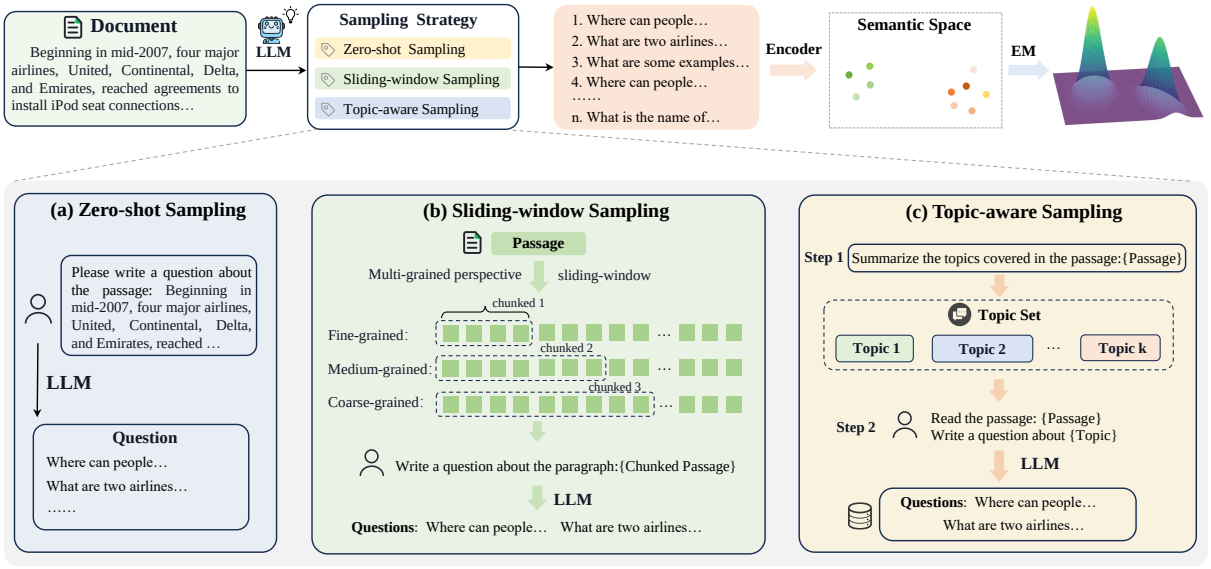

Figure 2: An overall diagram of the PQR framework. The process involves using a Gaussian mixture distribution to model a document's potential queries. First, sampling algorithms are employed to generate potential queries for the document, which are then encoded into a semantic space. Next, the EM algorithm is used to estimate the parameters of the distribution.

tasks. Given a query, some approaches (Sun et al., 2023; Qin et al., 2024) use large language models to rerank a small set of candidate documents. some researchers (Sachan et al., 2022; Muennighoff, 2022) directly utilize the logits of the language model to estimate the probability that a document can answer a given query. Other methods, such as HyDE (Gao et al., 2023) and GRM (Mackie et al., 2023), employ large language models to generate pseudo-documents based on user queries, using these pseudo-documents to assist in the retrieval process. Additionally, some researchers (Lewis et al., 2021; Bonifacio et al., 2022; Wang et al., 2023b; Dai et al., 2022) utilize synthetic data generated by language models to assist in training retrieval models.

Although the use of query generation from documents and synthetic data methods has been explored, they often overlook the many-to-one relationship between queries and documents, which is the central focus of our PQR method.

3 Methodology

In this section, we first formally define the problem of dense retrieval. Then, we provide a detailed introduction to the design of the PQR method.

3.1 Preliminaries

The goal of dense retrieval is to find the top-k documents most relevant to a given query from a large document corpus. The relevance score between a query and a document is typically based on dot product or cosine similarity. For a given query q and document d, a query encoder enc_q and a document encoder enc_d are used to encode the query and document into dense vectors v_q and v_d respectively. The relevance score between q and d is then calculated using the dot product of v_q and v_d :

$$\text{sim}(q, d) = \langle v_q, v_d \rangle$$

The core of dense retrieval is to align the semantics of queries and documents. However, since queries often focus on only parts of a document, this can lead to a loss of semantic information during the alignment process. Most current methods rely on training data where documents have only one or a few labeled queries, which makes it difficult to avoid this loss of information. In the PQR framework, we propose to model all potential queries of a document using a Gaussian mixture distribution, which helps prevent the issue of information loss.

3.2 Model Overview

PQR assumes that the probability distribution of the semantic vectors of potential queries for a document in the d-dimensional semantic space follows a Gaussian mixture distribution $P(\mathbf{x})$. As shown in Figure 2, to obtain the distribution of potential queries $P(\mathbf{x})$ for a document, we first em-

ploy a large language model to sample N potential queries. Next, we use a text similarity model to encode these queries into the semantic space. Finally, we apply the Expectation-Maximization algorithm to estimate the parameters of the Gaussian mixture distribution $P(\mathbf{x})$. We will provide a detailed introduction to the PQR method from three aspects: potential query sampling algorithms, distribution estimation, and similarity calculation methods.

3.3 Sampling Algorithm

3.3.1 Zero-shot Sampling

During the decoding process of large language models, sampling randomly from the distribution of logits allows the model to generate a diverse range of possible outputs. This inherent randomness in the decoding process enables us to introduce the Zero-shot Sampling strategy. Specifically, as illustrated in Figure 2(a), given a prompt that instructs the language model to infer potential queries from a document, the model is independently run N times to produce N distinct sampled queries. Formally, for a given prompt $Prompt_d$, the model generates a sequence of tokens $T_i = \{t_1, t_2, \ldots, t_k\}$ for $i = 1, 2, \ldots, N$, where each sequence T_i is sampled from the probability distribution $P(t_j | t_{<j}, Prompt_d)$. By repeating this process N times, we obtain a diverse set of potential query samples that reflect the model's understanding of the document.

3.3.2 Sliding-window Sampling

The semantic information in a document is spread across different parts of the text, and the length of text associated with various semantic elements can vary greatly. To ensure that language models focus on different sections of the document, we propose a multi-granularity sliding window sampling algorithm. As shown in Figure 2(b), we utilize three distinct window sizes (fine-grained, medium-grained, and coarse-grained) to segment the document. This method enables us to capture semantic information at multiple levels of granularity. Using the sliding window technique, we extract fragments of the document and apply zero-shot sampling to generate potential queries corresponding to each fragment. This process is repeated iteratively until we obtain N potential query samples. The sliding window sampling algorithm is specifically described as in Algorithm 1.

This strategy ensures that semantic information is captured comprehensively at different scales, and the generated queries effectively represent the document's content.

Algorithm 1 Sliding-Window Sampling Algorithm

Require: Document D (a sequence of sentences), sliding window steps $\{S_f, S_m, S_c\}$, number of queries to generate N
Ensure: Potential queries $\{q_1, q_2, \ldots, q_N\}$

```
Initialize query set Q ← ∅
for S ∈ {S_f, S_m, S_c} do
    Sliding window size: W ← max(|D|/S, 5)
    Get fragment set F by applying a sliding window of size W over document D
    Queries per fragment: k ← ⌈N / (3×|F|)⌉
    for f ∈ F do
        Get potential queries {q_1, q_2, ..., q_k} for fragment f using zero-shot sampling
        Q ← Q ∪ {q_1, q_2, ..., q_k}
    end for
end for
Randomly sample N queries from Q to form the final query set {q_1, q_2, ..., q_N}
return {q_1, q_2, ..., q_N}
```

3.3.3 Topic-aware Sampling

Large language models are highly proficient at reasoning, making them well-suited for tasks such as understanding a document and identifying its key topics. As shown in Figure 2(c), given a document D, the language model can analyze its content and extract a set of topics $T = \{t_1, t_2, \ldots, t_k\}$. Once the topics are identified, we can select a specific topic $t_i \in T$ and use the original document D as context to generate a set of potential queries $Q_i = \{q_{i1}, q_{i2}, \ldots, q_{im}\}$ related to t_i. This process can be iteratively repeated for multiple topics until we have accumulated N query samples across all topics:

$$Q = \bigcup_{i=1}^{k} Q_i, \quad \text{where } |Q| = N.$$

This approach leverages the reasoning capabilities of large language models to systematically generate diverse and relevant queries based on the document's content.

3.4 Distribution Estimation

We start by assuming that the potential queries for a document D follow a Gaussian mixture distribution $P(\mathbf{x})$. For a given document D, we combine

the samples obtained from the three sampling algorithms mentioned above, resulting in M samples. Each of these M potential query samples is then represented as a d-dimensional dense vector x. The probability of x belonging to the k-th Gaussian distribution is given by $P(x|k) = \mathcal{N}(x; \mu_k, \Sigma_k)$, where μ_k and Σ_k are the mean and covariance of the k-th Gaussian distribution, respectively. The overall distribution of the document's potential queries is modeled as a weighted mixture:

$$P(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x; \mu_k, \Sigma_k),$$

where π_k represents the mixture weight of the k-th Gaussian distribution.

After obtaining a set of sample points from the distribution, we use the Expectation-Maximization algorithm (Moon, 1996) to estimate the mean μ_k and covariance Σ_k by maximizing the log-likelihood of these samples. To determine the optimal number of components K in the Gaussian mixture distribution, we applied the Bayesian Information Criterion (BIC). BIC (Schwarz, 1978) balances model complexity and goodness of fit by penalizing overly complex models while rewarding better fit.

3.5 Similarity Calculation

Unlike traditional methods, PQR models a document as a Gaussian mixture distribution based on the distribution of its potential queries. Therefore, we cannot directly compute the similarity between a query and a document using the dot product of their vectors. In a Gaussian mixture distribution, the k-th Gaussian distribution represents the k-th type of semantic information. We use the mean vector of the k-th Gaussian distribution, denoted as $\mathbf{v}_k$, as a representation of the k-th semantic type. The similarity between the query and the k-th semantic type of the document is then computed as the dot product between the query vector $\mathbf{v}_q$ and $\mathbf{v}_k$. Finally, the overall similarity between the query and the document is defined as the maximum similarity across all K semantic components:

$$\text{sim}(q, d) = \max_{1 \le k \le K} \langle \mathbf{v}_q, \mathbf{v}_k \rangle,$$

where $\langle \mathbf{v}_q, \mathbf{v}_k \rangle$ denotes the dot product between the query vector $\mathbf{v}_q$ and the k-th Gaussian distribution mean vector $\mathbf{v}_k$.

4 Experiments

4.1 Setup

Datasets and Evaluation: We evaluated the effectiveness of our approach on the BEIR (Thakur et al., 2021) benchmark. Specifically, following prior studies (Wang et al., 2022; Gao et al., 2023; Shen et al., 2023), we selected six low-resource datasets from BEIR for our experiments: SciFact (scientific paper abstracts), FiQA (financial articles), TREC-COVID (COVID-19 scientific papers), NFCorpus (medical information retrieval), ArguAna (argument retrieval), and SciDocs (scientific document retrieval). Additionally, we assessed the performance of PQR on MS MARCO document dataset (Bajaj et al., 2016) and the TREC Deep Learning 2019 and 2020 (Craswell et al., 2020) test set (referred to as TREC-DL'19 and TREC-DL'20). MS MARCO is a popular document retrieval dataset containing approximately 3.2 million documents and around 5k queries in its validation set. TREC-DL'19 and TREC-DL'20 serve as test sets for the MS MARCO document ranking task, containing 43 and 45 queries respectively, with more detailed annotations. We report MRR@10, Recall@100, and Recall@1K on the MS MARCO document development set, as well as nDCG@10 on TREC-DL'19 and TREC-DL'20. For the BEIR benchmark datasets, we focus on NDCG@10 as the evaluation metric.

Implementation Details: In our sampling algorithm, We use the SGLang (Zheng et al., 2024) framework to deploy the Llama-3.1 8B model (Dubey et al., 2024) on four RTX 4090 GPUs, serving as the base large language model for query sampling. The model's temperature is set to 1.2, and the maximum token length for each query is limited to 28. Using the three sampling algorithms mentioned above, we generate 100 potential query samples for each algorithm, resulting in a total of 300 potential queries. For encoding these queries, we use three models: Sentence-T5-base (Ni et al., 2021), BGE-m3 (Chen et al., 2024), and all-MiniLM-L12-v2 (Reimers and Gurevych, 2019). In our experiments, we refer to these as $\text{PQR}_{\text{Sentence-T5}}$, $\text{PQR}_{\text{BGE-m3}}$, and $\text{PQR}_{\text{MiniLM}}$, respectively. We use the EM algorithm in the "GaussianMixture" class from the sklearn [2] library to estimate the parameters of a distribution. When determining the optimal number of Gaussian distribution components for the query

[2] https://scikit-learn.org/stable/

Model	FiQA	ArguAna	SciDocs	SciFact	NFCorpus	TREC-COVID	Avg.
BM25	0.236	0.315	0.158	0.665	0.325	0.656	0.393
docT5query	0.291	0.349	0.162	0.675	0.328	0.713	0.420
DPR	0.224	0.323	0.103	0.479	0.244	0.604	0.330
ANCE	0.295	0.415	0.122	0.507	0.237	0.654	0.372
RocketQAv2	0.302	0.451	0.131	0.568	0.293	0.675	0.403
SPLADE	0.289	0.445	0.149	0.633	0.322	0.661	0.417
ColBERTv2	0.356	0.463	0.154	0.693	0.338	0.738	0.457
COIL	0.313	0.294	0.155	0.707	0.331	0.668	0.412
CITADEL	0.332	0.490	0.147	0.695	0.337	0.680	0.447
$\text{HyDE}_{\text{Llama-8B}}$	0.220	0.299	0.133	0.639	0.312	0.604	0.368
Sentence-T5	0.348	0.448	0.142	0.458	0.286	0.407	0.348
BGE-m3	0.410	0.540	0.163	0.644	0.314	0.549	0.437
MiniLM	0.373	0.471	0.218	0.626	0.323	0.508	0.420
$\textbf{PQR}_{\textbf{Sentence-T5}}$	0.360	0.504	0.149	0.534	0.272	0.575	$0.399^{\uparrow 0.051}$
$\textbf{PQR}_{\textbf{BGE-m3}}$	0.403	0.478	0.178	0.669	0.334	0.608	$0.445^{\uparrow 0.008}$
$\textbf{PQR}_{\textbf{MiniLM}}$	0.401	0.521	0.208	0.688	0.348	0.617	$0.464^{\uparrow 0.044}$

Table 1: The zero-shot retrieval results on the BEIR dataset, evaluated using the NDCG@10 metric, achieved by our method and baseline models.

distribution, we select the number of components with the lowest BIC value from a range of 4 to 10. Refer to the appendix A.1 for more details.

Baselines: Since PQR requires no training or labeled data, we primarily focus on its performance in zero-shot retrieval settings. Specifically, we compare it with retrieval methods that do not require training data, such as BM25 (Robertson et al., 1995) and docT5query (Nogueira et al., 2019). Additionally, we include HyDE (Gao et al., 2023), an advanced zero-shot retrieval approach based on large language models, for comparison. We utilized Llama-8B to reproduce HyDE(referred to as $\text{HyDE}_{\text{Llama-8B}}$). To further validate the effectiveness of our method, we benchmark it against several popular methods, including DPR (Karpukhin et al., 2020), ANCE (Xiong et al., 2020), RocketQAv2 (Ren et al., 2021), SPLADE (Formal et al., 2021), Sentence-T5-base (Ni et al., 2021) (referred to as Sentence-T5), BGE-m3 (Chen et al., 2024) and all-MiniLM-L12-v2 (Reimers and Gurevych, 2019) (referred to as MiniLM). Multi-vector approaches demonstrate strong zero-shot retrieval capabilities in dense retrieval. We therefore compare our approach with multi-vector models, including ColBERTv2 (Santhanam et al., 2022), COIL (Gao et al., 2021) and CITADEL (Li et al., 2023).

4.2 Retrieval Performance

We compare PQR with mainstream baseline retrieval models, as shown in Table 1. Overall, the PQR method achieves a higher average nDCG@10 score across six datasets in zero-shot retrieval tasks compared to other models. While multi-vector models demonstrate strong performance in zero-shot retrieval, PQR method performs significantly better on the ArguAna, FiQA, NFCorpus and SciDocs datasets and remains competitive on the others. Moreover, all three PQR methods employing different encoding models ($\text{PQR}_{\text{Sentence-T5}}$, $\text{PQR}_{\text{BGE-m3}}$, and $\text{PQR}_{\text{MiniLM}}$) outperform their respective base encoding models. This demonstrates both the effectiveness and generalizability of the PQR modeling approach.

In Table 2, for a fair comparison, we selected models that were not trained on the MS MARCO document dataset. The results show that our PQR method outperforms other approaches on the dev set in terms of MRR@10, Recall@100, and Recall@1K. On the TREC-DL dataset, our PQR method also demonstrates competitive performance. More importantly, all three PQR methods using different encoder models ($\text{PQR}_{\text{Sentence-T5}}$, $\text{PQR}_{\text{BGE-m3}}$, and $\text{PQR}_{\text{MiniLM}}$) show significant improvements over their respective base encoders

Model	MS MARCO DEV			TREC-DL'19	TREC-DL'20
	MRR@10	Recall@100	Recall@1K	NDCG@10	NDCG@10
BM25	0.230	0.808	0.886	0.519	0.506
docT5query	0.288	0.861	0.926	0.597	0.582
$\text{HyDE}_{\text{Llama-8B}}$	0.213	0.745	0.911	0.612	0.519
Sentence-T5	0.200	0.718	0.882	0.399	0.403
BGE-m3	0.317	0.807	0.905	0.601	0.527
MiniLM	0.295	0.814	0.922	0.587	0.537
$\textbf{PQR}_{\textbf{Sentence-T5}}$	$0.247^{\uparrow 0.047}$	$0.775^{\uparrow 0.057}$	$0.905^{\uparrow 0.023}$	$0.500^{\uparrow 0.101}$	$0.480^{\uparrow 0.077}$
$\textbf{PQR}_{\textbf{BGE-m3}}$	$0.301^{\downarrow 0.016}$	$0.858^{\uparrow 0.051}$	$0.951^{\uparrow 0.046}$	$0.606^{\uparrow 0.005}$	$0.546^{\uparrow 0.019}$
$\textbf{PQR}_{\textbf{MiniLM}}$	$0.308^{\uparrow 0.013}$	$0.865^{\uparrow 0.051}$	$0.951^{\uparrow 0.029}$	$0.617^{\uparrow 0.030}$	$0.566^{\uparrow 0.029}$

Table 2: Our method's results on the MS MARCO document retrieval task are compared with those of baseline models that require no supervised training.

Model	SciDocs	FiQA	ArguAna
docT5query	0.162	0.291	0.349
docT5query_{300}	0.140	0.293	0.470
PQR	**0.208**	**0.401**	**0.521**

Table 3: The results compare the PQR with approach that does not use distribution modeling, evaluated using NDCG@10. docT5query_{300} refers to the use of 300 queries to enhance the docT5query approach. PQR uses MiniLM as its encoder. The highest value is highlighted in bold.

Model	SciDocs	FiQA	ArguAna
PQR_{zs}	0.207	**0.401**	0.509
PQR_{sw}	0.203	0.388	0.517
PQR_{ta}	**0.209**	0.396	0.512
PQR	0.208	**0.401**	**0.521**

Table 4: The impact of three query sampling algorithms on experimental results, evaluated using NDCG@10. All PQR models use MiniLM as its encoder. The highest value is highlighted in bold.

across all metrics, except for $\text{PQR}_{\text{BGE-m3}}$, which exhibits a slight decrease in MRR@10. These results further demonstrate the effectiveness and generalizability of our proposed PQR method.

4.3 Ablation Study

Effectiveness of Distribution Modeling: To compare PQR with document expansion methods like docT5query, we enhanced the original document representation by appending 300 sampled queries to their corresponding documents. These queries were generated using our proposed sampling algorithm. We then performed retrieval using the BM25 method, and the results are shown in Table 3. From the results, it is clear that the PQR method, which models potential queries as a Gaussian mixture distribution and uses sampled queries to estimate the distribution parameters, significantly outperforms the approach of directly enhancing document representation with generated queries. This demonstrates the effectiveness of our proposed PQR method.

Analysis of Sampling Algorithms: We explored the retrieval performance of three different sampling algorithms, each generating 300 potential query samples, as well as the performance of combining these three algorithms to obtain a mixed set of 300 query samples. As shown in Table 4, the results of the zero-shot sampling method, sliding window sampling method, and topic-aware sampling method are denoted as PQR_{zs}, PQR_{sw}, and PQR_{ta}, respectively. From the experimental results, it can be observed that each sampling method demonstrates a certain degree of effectiveness. However, when the three sampling algorithms are combined, the retrieval performance is improved. We believe this is likely because the combined approach generates a more comprehensive and diverse set of potential queries by leveraging the strengths of each individual sampling method.

5 Further Analysis

In this section, we provide a more detailed analysis of the PQR method. In all experiments, the encoder used for the PQR method is MiniLM.

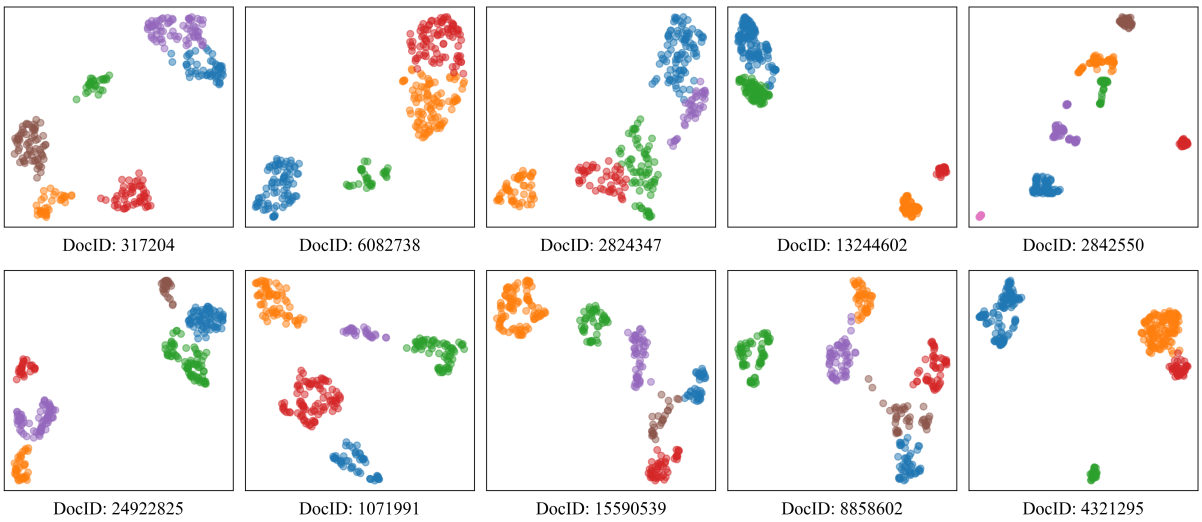

Figure 3: Visualization of the potential queries for 10 randomly selected documents from the SciFact dataset. Points with the same color represent samples belonging to the same Gaussian distribution component.

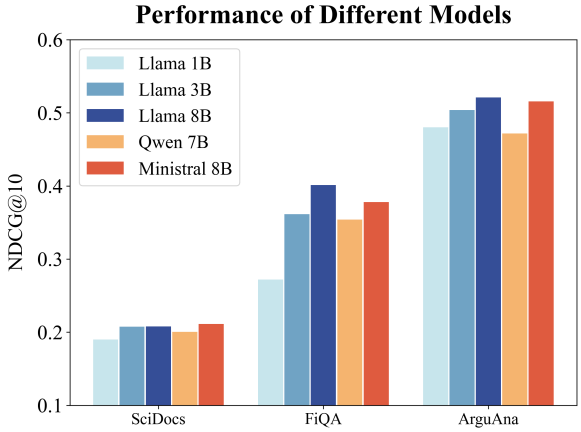

Figure 4: The impact of using different generative models for potential query sampling on model performance.

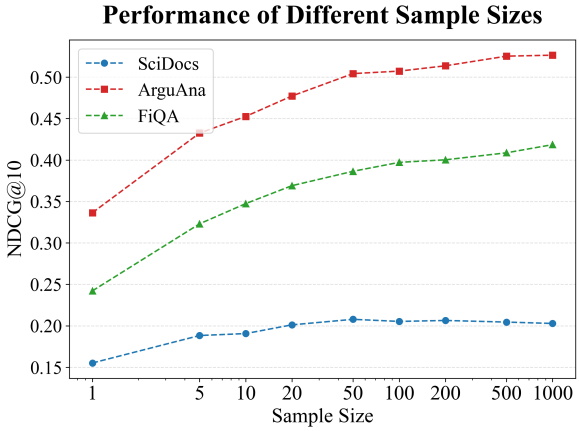

Figure 5: The impact of the number of potential query samples on model performance.

5.1 Visualization

As shown in Figure 3, we randomly selected 10 documents from the SciFact dataset to visualize the distribution of their corresponding potential queries in the semantic space. The encoded queries were dimensionally reduced using the UMAP (McInnes et al., 2018) method and then plotted on a 2D plane, as shown in the figure above. For each document, we used different colors to represent the various semantic queries associated with it. From the visualization, we can observe that the potential queries corresponding to a single document tend to cluster at various locations within the semantic space. This demonstrates PQR's ability to model different semantic information within document.

5.2 Generative Models Analysis

We used different sizes of the Llama 3 (Dubey et al., 2024) series models (1B, 3B, and 8B) as generative models for potential query sampling. Additionally, we compared them with other open-source models, including Qwen2.5-7B (Yang et al., 2024) and Ministral-8B (Mistral, 2024). Our experiments were conducted on the FiQA, SciDocs, and ArguAna datasets. As shown in Figure 4, the results indicate that the retrieval performance of PQR improves significantly as the size of the Llama model increases. Moreover, using different types of large language models of similar scale as the generative model for potential query sampling also affects the final PQR performance. These findings suggest that the generative capability of large language models has a notable impact on PQR's performance. Em-

ploying more capable large language models can significantly enhance PQR's effectiveness. This also indicates that as large language models continue to evolve, the performance of PQR is likely to improve further.

5.3 Analysis of Sample Size

As shown in Figure 5, we studied the impact of the number of sampled potential queries on PQR retrieval performance across three datasets: SciDocs, ArguAna, and FiQA. The experimental results indicate that when the number of sampled queries is fewer than 50, the overall performance of PQR decreases significantly. However, when the number of sampled queries exceeds 200, further increasing the sample size continues to improve PQR performance, but the improvements begin to level off. In practical applications, to balance performance and sampling efficiency, it is recommended to keep the sample size within the range of 50 to 500 for optimal results.

6 Conclusion

In this paper, we propose a novel potential query retrieval framework. Specifically, we first assume that the potential queries of documents follow a Gaussian mixture distribution in the semantic space. Then, we leverage large language models to generate potential query samples for documents using our proposed query sampling algorithms and encode them into the semantic space. Finally, we estimate the parameters of the distribution using the Expectation-Maximization algorithm. We conduct experiments on zero-shot retrieval datasets and the MS MARCO document dataset to validate the effectiveness of the proposed PQR method.

Limitations

Our PQR method requires generating hundreds of query data points for each document, making it highly dependent on computing power. However, with the decreasing cost of hardware and advancements in model inference techniques, we remain optimistic about the time efficiency of PQR, especially since it is used during the offline preprocessing stage. Additionally, the storage space required for PQR depends on the number of Gaussian distribution components, which presents a storage efficiency challenge similar to many existing multi-vector methods. Fortunately, ongoing research on improving storage efficiency for multi-vector retrieval methods has made this issue increasingly manageable. For a detailed analysis, please see the appendix A.2.

Acknowledgments

This research was supported by grants from the National Key Research and Development Program of China (Grant No. 2024YFC3308200), the National Natural Science Foundation of China (62337001), the Key Technologies R & D Program of Anhui Province (No. 202423k09020039) and the Fundamental Research Funds for the Central Universities.

References

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Luiz Bonifacio, Hugo Abonizio, Marzieh Fadaee, and Rodrigo Nogueira. 2022. Inpars: Unsupervised dataset generation for information retrieval. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2387–2392.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv preprint arXiv:2402.03216*.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Ellen M Voorhees. 2020. Overview of the trec 2019 deep learning track. *arXiv preprint arXiv:2003.07820*.

Zhuyun Dai, Vincent Y Zhao, Ji Ma, Yi Luan, Jianmo Ni, Jing Lu, Anton Bakalov, Kelvin Guu, Keith B Hall, and Ming-Wei Chang. 2022. Promptagator: Few-shot dense retrieval from 8 examples. *arXiv preprint arXiv:2209.11755*.

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

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 and Jamie Callan. 2021. Condenser: a pre-training architecture for dense retrieval. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 981–993.

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: Human Language Technologies*, pages 3030–3042.

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 (Volume 1: Long Papers)*, pages 1762–1777.

Vladimir Karpukhin, Barlas Oguz, 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 (EMNLP)*, pages 6769–6781.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Patrick Lewis, Yuxiang Wu, Linqing Liu, Pasquale Minervini, Heinrich Küttler, Aleksandra Piktus, Pontus Stenetorp, and Sebastian Riedel. 2021. Paq: 65 million probably-asked questions and what you can do with them. *Transactions of the Association for Computational Linguistics*, 9:1098–1115.

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 (Volume 1: Long Papers)*, pages 11891–11907.

Qi Liu, Zhenya Huang, Yu Yin, Enhong Chen, Hui Xiong, Yu Su, and Guoping Hu. 2021. Ekt: Exercise-aware knowledge tracing for student performance prediction.

Shuqi Lu, Di He, Chenyan Xiong, Guolin Ke, Waleed Malik, Zhicheng Dou, Paul Bennett, Tie-Yan Liu, and Arnold Overwijk. 2021. Less is more: Pretrain a strong siamese encoder for dense text retrieval using a weak decoder. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2780–2791.

Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. 2021. Sparse, dense, and attentional representations for text retrieval. *Transactions of the Association for Computational Linguistics*, 9:329–345.

Iain Mackie, Ivan Sekulic, Shubham Chatterjee, Jeffrey Dalton, and Fabio Crestani. 2023. Grm: generative relevance modeling using relevance-aware sample estimation for document retrieval. *arXiv preprint arXiv:2306.09938*.

Qingyang Mao, Zhi Li, Qi Liu, Likang Wu, Hefu Zhang, and Enhong Chen. 2024. Promoting machine abilities of discovering and utilizing knowledge in a unified zero-shot learning paradigm. *ACM Transactions on Knowledge Discovery from Data*, 19(1):1–26.

Leland McInnes, John Healy, and James Melville. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Mistral. 2024. Un ministral, des ministraux. Accessed: 2025-02-07.

Todd K Moon. 1996. The expectation-maximization algorithm. *IEEE Signal processing magazine*, 13(6):47–60.

Niklas Muennighoff. 2022. Sgpt: Gpt sentence embeddings for semantic search. *arXiv preprint arXiv:2202.08904*.

Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith B Hall, Daniel Cer, and Yinfei Yang. 2021. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. *arXiv preprint arXiv:2108.08877*.

Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. 2019. From doc2query to doctttttquery. *Online preprint*, 6(2).

Zhen Qin, Rolf Jagerman, Kai Hui, Honglei Zhuang, Junru Wu, Le Yan, Jiaming Shen, Tianqi Liu, Jialu Liu, Donald Metzler, et al. 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.

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*. 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.

Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. 1995. Okapi at trec-3. *Nist Special Publication Sp*, 109:109.

Devendra Sachan, Mike Lewis, Mandar Joshi, Armen Aghajanyan, Wen-tau Yih, Joelle Pineau, and Luke Zettlemoyer. 2022. Improving passage retrieval with zero-shot question generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3781–3797.

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: Human Language Technologies*, pages 3715–3734.

Gideon Schwarz. 1978. Estimating the dimension of a model. *The annals of statistics*, pages 461–464.

Tao Shen, Guodong Long, Xiubo Geng, Chongyang Tao, Tianyi Zhou, and Daxin Jiang. 2023. Large language models are strong zero-shot retriever. *arXiv preprint arXiv:2304.14233*.

Ruijun Sun, Hanqin Tao, Yanmin Chen, and Qi Liu. 2024. Hacan: a hierarchical answer-aware and context-aware network for question generation. *Frontiers of Computer Science*, 18(5):185321.

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.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models. *arXiv preprint arXiv:2104.08663*.

Kexin Wang, Nandan Thakur, Nils Reimers, and Iryna Gurevych. 2022. Gpl: Generative pseudo labeling for unsupervised domain adaptation of dense retrieval. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2345–2360.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2023a. Simlm: Pre-training with representation bottleneck for dense passage retrieval. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2244–2258.

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

Shitao Xiao, Zheng Liu, Yingxia Shao, and Zhao Cao. 2022. Retromae: Pre-training retrieval-oriented language models via masked auto-encoder. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 538–548.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2020. Approximate nearest neighbor negative contrastive learning for dense text retrieval. *arXiv preprint arXiv:2007.00808*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Kai Zhang, Kun Zhang, Mengdi Zhang, Hongke Zhao, Qi Liu, Wei Wu, and Enhong Chen. 2022a. Incorporating dynamic semantics into pre-trained language model for aspect-based sentiment analysis. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3599–3610.

Shunyu Zhang, Yaobo Liang, Ming Gong, Daxin Jiang, and Nan Duan. 2022b. Multi-view document representation learning for open-domain dense retrieval. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5990–6000.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E Gonzalez, et al. 2024. Sglang: Efficient execution of structured language model programs. *arXiv preprint arXiv:2312.07104*.

A Appendix

A.1 Other Implementation Details

When the input document exceeds 6K tokens, it will be truncated. For the sliding window sampling algorithm, the number of steps for coarse-grained, medium-grained, and fine-grained sampling are 1, 2, and 4, respectively. Additionally, each window must contain at least 5 sentences. When estimating the parameters of the Gaussian mixture distribution using the sklearn library, we set the random seed to 42 and the maximum number of iterations for the EM algorithm to 50, while keeping all other settings at their default values.

A.2 Computational Overhead Analysis

Model	Tokens	Latency
HyDE	2755.3	1.798s
PQR	2380.9	1.623s

Table 5: Result of computational overhead analysis. Experiments were conducted on the NFCorpus dataset.

High computational costs remain a significant challenge when applying large language models to retrieval tasks. To address concerns regarding the computational efficiency of PQR, we conducted a comparative analysis with the influential HyDE (Gao et al., 2023) method. As presented in Table 5, PQR generates queries with an average length of 23.809 tokens during sampling. In practical scenarios where 100 queries are sampled per document—a setting empirically shown to be effective—this results in the generation of approximately 2,380.9 tokens per document. In contrast, HyDE generates 8 pseudo-documents per query, each averaging 344.407 tokens, amounting to a total of 2,755.256 tokens per query. Our experiments, conducted on a single A800 GPU using Llama-8B, demonstrate that HyDE requires 1.798 seconds per query for pseudo-document generation, whereas PQR only requires 1.623 seconds per document for query sampling. Notably, PQR supports offline document processing, thereby eliminating any additional latency during online retrieval.

A.3 Distribution Modeling Analysis

Model	SciDocs	FiQA	ArguAna
$PQR_{K\text{-means}}$	0.205	**0.403**	0.518
$PQR_{MeanScore}$	0.196	0.360	0.465
$PQR_{MeanPool}$	0.198	0.332	0.446
PQR	**0.208**	0.401	**0.521**

Table 6: The impact of different modeling methods on experimental results, evaluated using NDCG@10. All PQR models use MiniLM as its encoder. The highest value is highlighted in bold.

We explored different approaches for processing potential queries after encoding them into semantic space. $PQR_{K\text{-means}}$ replaces Gaussian Mixture Distribution modeling with K-means clustering; $PQR_{MeanScore}$ computes the similarity score by averaging the dot products between the user query vector and the mean vectors of each distribution in the document (whereas PQR uses the maximum value); $PQR_{MeanPool}$ directly applies mean pooling to all potential query vectors to obtain the final document representation. The experimental results are shown in Table 6. Both $PQR_{MeanScore}$ and $PQR_{MeanPool}$ exhibit significant performance drops, demonstrating the effectiveness of the original PQR modeling approach. $PQR_{K\text{-means}}$ performs slightly worse than PQR, which is expected since K-means is a special case of GMM with constrained covariance.

A.4 Example Display

We randomly selected a document from the NFCorpus dataset and present its potential query examples generated using three different sampling algorithms.

Document ID: MED-3457

Reactive oxygen species produced during vigorous exercise may permeate into cell nuclei and induce oxidative DNA damage, but the supporting evidence is still lacking. By using a 42 km marathon race as a model of massive aerobic exercise, we demonstrated a significant degree of unrepaired DNA base oxidation in peripheral immunocompetent cells, despite a concurrent increase in the urinary excretion of 8-hydroxy-2'-deoxyguanosine. Single cell gel electrophoresis with the incorpo-

ration of lesion-specific endonucleases further revealed that oxidized pyrimidines (endonuclease III-sensitive sites) contributed to most of the postexercise nucleotide oxidation. The oxidative DNA damage correlated significantly with plasma levels of creatinine kinase and lipid peroxidation metabolites, and lasted for more than 1 week following the race. This phenomenon may be one of the mechanisms behind the immune dysfunctions after exhaustive exercise.

Below are queries obtained using three different sampling algorithms. For each sampling algorithm, we randomly selected 4 queries for display.

Zero-shot Sampling

1. What is the physiological effect of 42 km marathon racing on human DNA.
2. Do intense periods of exercise correlate with long-term cellular damage.
3. What is the extent and duration of oxidative DNA damage in human peripheral leukocytes after a marathon?
4. Can massive aerobic exercise cause DNA damage in human peripheral leukocytes and what are the associated physiological effects?

Sliding-window Sampling

1. What types of cellular damage have been discovered in immunocompetent cells subject to extremely strenuous physical exercise?
2. How long did oxidative DNA damage from a marathon last after the event?
3. Does massive aerobic exercise cause unrepaired DNA base oxidation in peripheral leukocytes?
4. What types of cellular and molecular effects occur due to massive aerobic exercise in the human body and how long do these changes last?

Topic-aware Sampling

1. Does exhaustive physical activity have a measurable impact on the rate of DNA damage in immune cells after a prolonged duration?
2. What are the potential long-term health consequences associated with unrepaired oxidative DNA damage in athletes subjected to extreme physical activity?
3. What is the impact of a marathon race on oxidative DNA damage in human peripheral leukocytes?
4. What is a possible cause of immune dysfunction that is triggered by prolonged physical activity?

A.5 Prompting Template

The Prompting Templates for Zero-shot Sampling and Sliding-window Sampling are as follows:

NFCorpus

{PASSAGE}
Please read the above passage about medical information, and write a Question from a different perspective that dense retrieval model could use to find this Passage. Directly output the Question without any additional information.
Question:

SciFact

{PASSAGE}
Please read the above Passage about scientific claim, and write a Question from a different perspective that dense retrieval model could use to find this Passage. Directly output the Question without any additional information.
Question:

SciDocs

{PASSAGE}
Please read the above Passage about scientific claim, and write a Question from a different perspective that dense retrieval model could use to find this Passage. Directly output the Question without any additional information.
Question:

ArguAna

{PASSAGE}
Please read the above Passage, and write a Question from a different perspective that a dense retrieval model can use to find this passage. Directly output the Question without any additional information.
Question:

FiQA

{PASSAGE}
Please read the above Passage about financial information, and write a Question from a different perspective that dense retrieval model could use to find this Passage. Directly output the Question without any additional information.
Question:

TREC-COVID

{PASSAGE}
Please read the above Passage about biomedical information, and write a Question from a different perspective that dense retrieval model could use to find this Passage. Directly output the Question without any additional information.
Question:

MS MARCO

{PASSAGE}
Please read the above Passage, and write a Question from a different perspective that dense retrieval model could use to find this Passage. Directly output the Question without any additional information.
Question:

The prompting template for sampling the topics of all documents is as follows:

Topic Sampling Prompt Template

{PASSAGE}
Please read the above Passage and summarize a Topic it includes. Output the Topic directly without any additional information.
Topic:

The Prompting Templates for Topic-aware Sampling are as follows:

NFCorpus

{PASSAGE}
Please read the above Passage about medical information, and write a Question related to "{TOPIC}" from from a different perspective that dense retrieval model could use to find this Passage. Directly output the Question without any additional information.
Question:

SciFact

{PASSAGE}
Please read the above Passage about scientific claim, and write a Question related to "{TOPIC}" that dense retrieval model could use to find this Passage. Directly output the Question without any additional information.
Question:

SciDocs

{PASSAGE}
Please read the above Passage about scientific claim, and write a Question related to "{TOPIC}" that dense retrieval model could use to find this Passage. Directly output the Question without any additional information.
Question:

ArguAna

{PASSAGE}
Please read the above Passage, and write a Question related to "{TOPIC}" that a dense retrieval model can use to find this Passage. Directly output the Question without any additional information.
Question:

FiQA

{PASSAGE}
Please read the above Passage about financial information, and write a Question

related to "{TOPIC}" that dense retrieval model could use to find this Passage. Directly output the Question without any additional information.
Question:

TREC-COVID

{PASSAGE}
Please read the above Passage about biomedical information, and write a Question related to "{TOPIC}" that dense retrieval model could use to find this Passage. Directly output the Question without any additional information.
Question:

MS MARCO

{PASSAGE}
Please read the above Passage, and write a Question related to "{TOPIC}" from from a different perspective that dense retrieval model could use to find this Passage. Directly output the Question without any additional information.
Question: