Improving Scientific Document Retrieval with Concept Coverage-based Query Set Generation

SeongKu Kang
Korea University
Seoul, South Korea
seongku@korea.ac.kr

Bowen Jin
University of Illinois at Urbana-Champaign
Champaign, United States
bowenj4@illinois.edu

Wonbin Kweon
Pohang University of Science and Technology
Pohang, South Korea
kwb4453@postech.ac.kr

Yu Zhang
University of Illinois at Urbana-Champaign
Champaign, United States
yuz9@illinois.edu

Dongha Lee
Yonsei University
Seoul, South Korea
donalee@yonsei.ac.kr

Jiawei Han
University of Illinois at Urbana-Champaign
Champaign, United States
hanj@illinois.edu

Hwanjo Yu*
Pohang University of Science and Technology
Pohang, South Korea
hwanjoyu@postech.ac.kr

Abstract

In specialized fields like the scientific domain, constructing large-scale human-annotated datasets poses a significant challenge due to the need for domain expertise. Recent methods have employed large language models to generate synthetic queries, which serve as proxies for actual user queries. However, they lack control over the content generated, often resulting in incomplete coverage of academic concepts in documents. We introduce **C**oncept **C**overage-based **Q**uery set **Gen**eration (**CCQGen**) framework, designed to generate a set of queries with comprehensive coverage of the document's concepts. A key distinction of CCQGen is that it adaptively adjusts the generation process based on the previously generated queries. We identify concepts not sufficiently covered by previous queries, and leverage them as conditions for subsequent query generation. This approach guides each new query to complement the previous ones, aiding in a thorough understanding of the document. Extensive experiments demonstrate that CCQGen significantly enhances query quality and retrieval performance.

CCS Concepts

• **Information systems** → **Information retrieval**; **Content analysis and feature selection**; *Information retrieval query processing*.

Keywords

Information retrieval; Query generation; Scientific document search

ACM Reference Format:
SeongKu Kang, Bowen Jin, Wonbin Kweon, Yu Zhang, Dongha Lee, Jiawei Han, and Hwanjo Yu. 2025. Improving Scientific Document Retrieval with Concept Coverage-based Query Set Generation. In *Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining (WSDM '25), March 10–14, 2025, Hannover, Germany*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3701551.3703544

*Corresponding author

WSDM '25, March 10–14, 2025, Hannover, Germany

ACM ISBN 979-8-4007-1329-3/25/03
https://doi.org/10.1145/3701551.3703544

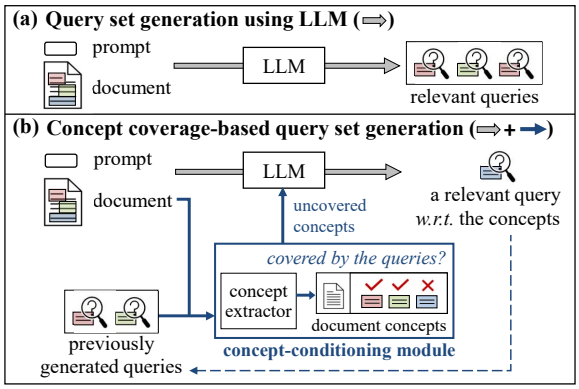

Figure 1: A conceptual comparison of (a) the existing approach for query set generation and (b) our concept coverage-based query set generation. Best viewed in color.

1 Introduction

Scientific document retrieval is a fundamental task that accelerates scientific innovations and access to technical solutions [17]. Recently, pre-trained language models (PLMs) have largely enhanced various ad-hoc searches [12, 18]. PLM-based retrievers are initially pre-trained on massive textual corpora to develop language understanding. They are then fine-tuned using vast datasets of annotated query-document pairs, enabling the models to accurately assess the relevance between queries and documents. However, in specialized domains like scientific document retrieval, constructing large-scale annotated datasets is challenging due to the need for domain expertise [4, 14, 22]. While there are a few general domain datasets (e.g., web search [2, 19]), they often fail to generalize to specialized domains [4, 47]. This remains a major obstacle for applications.

Recently, large language models (LLMs) [5, 40, 48, 51] have been actively utilized to generate synthetic data. Given a document and a prompt including an instruction such as "*generate five relevant queries to the document*" [9, 41], LLMs generate synthetic queries for each document (Figure 1a). The generated queries serve as proxies for actual user queries. Recent developments in prompting schemes have largely improved the quality of these queries. [4, 9] show that incorporating a few examples of actual query-document pairs in the prompt leads to the generation of queries with similar

Table 1: An example of synthetic queries. The queries are generated sequentially from q^1 to q^3. CCQGen is applied after generating q^1. Repeated keywords are denoted in red, while newly covered concepts are denoted in blue. Details of the generation process are illustrated in Figure 2.

Document	
	Automated music playlist generation is a specific form of music recommendation. Collaborative filtering methods can be used to ... However, the scarcity of thoroughly curated playlists and the bias towards popular songs ... we propose an alternative model based on a song-to-playlist classifier, ... while leveraging song features derived from audio, ... robust performance when recommending rare and out-of-set songs. ...
Generated queries for the document	
q^1	How can song-to-playlist classifiers enhance automated music playlist generation?
q^2	How can automated playlist creation be boosted through song-to-playlist classification and **feature exploitation**? (*w/o any condition*)
q^3	How does song-to-playlist classifier differ from traditional **collaborative filtering** for music recommendation? (*w/o any condition*)
$q^{2'}$	What techniques can be used to **overcome filter bubbles** and **recommend out-of-set songs**? (*w/ CCQGen*)
$q^{3'}$	How to **leverage mel-spectrogram features** to **mitigate the cold-start problem** in playlist recommendation? (*w/ CCQGen*)

distributions (e.g., expression styles) to actual queries. The state-of-the-art method [6] employs a *pair-wise* generation that instructs LLMs to generate relevant queries first and then relatively less relevant ones. These less relevant queries serve as natural 'hard negatives', further improving the efficacy of fine-tuning [6].

Though effective in generating plausible queries, the existing methods lack control over the content generated, which can lead to incomplete coverage of the academic concepts in a document. Academic concepts refer to fundamental ideas, theories, and methodologies that form the contents of scientific documents. A scientific document typically explores various concepts. For example, in Table 1, the document addresses the primary task of music playlist recommendation, along with the design of classification-based models, solutions for popularity biases and data scarcity, and the utilization of audio features. For a thorough understanding of the document, training queries should comprehensively cover these concepts.

However, in the absence of control over the content generated, the queries often repeatedly cover similar aspects of the document, showing high redundancy. For example, in Table 1, the generated queries (q^2, q^3) repeat keywords such as 'automated playlist creation' and 'song-to-playlist classification' already present in the previous query (q^1). While these concepts are undoubtedly relevant to the document, such redundant queries cannot effectively bring new training signals. Furthermore, the queries exhibit a particularly higher lexical overlap with the document, compared to the actual user queries (§4.2.1). We observe that the queries tend to repeat only a few terms extracted from the document. Given that users express the same concepts using various expressions in their queries, this limited term usage may not effectively simulate actual queries, reducing the efficacy of fine-tuning. As a naive solution, one might consider adding more instructions to the prompt, such as "*use various terms and reduce redundancy among the queries*". However, this still lacks systematic control over the generation and fails to bring consistent improvements (§4.1.1); the improved term usage often appears in common expressions (e.g., advance, enhance, and reinforce), not necessarily enhancing concept coverage.

We propose **C**oncept **C**overage-based **Q**uery set **Gen**eration (CCQGen) framework to meet two desiderata for training queries: (1) The queries should cover complementary aspects, enabling comprehensive coverage of the document's concepts, and (2) The queries should articulate the concepts in various related terms, rather than merely echoing a few phrases from the document. A key distinction of CCQGen is that it adaptively adjusts the generation process based on the concept coverage of previously generated queries (Figure 1b). We introduce a *concept extractor* to (1) identify the core concepts of each text and (2) uncover concept-related terms not explicitly mentioned in the document. Using this information, we discern the concepts not sufficiently covered by previous queries, and leverage them as *conditions* for the subsequent query generation. Table 1 shows that the queries generated with CCQGen ($q^{2'}, q^{3'}$) cover complementary concepts using more various related terms. Furthermore, we introduce new techniques to filter out low-quality queries and enhance retrieval accuracy using the obtained concept information. Our primary contributions are:

- We show that existing query generation methods often fail to comprehensively cover academic concepts in documents, leading to suboptimal training and retrieval performance.
- We propose CCQGen framework, which adaptively imposes conditions for subsequent generation based on the concept coverage. CCQGen can be flexibly integrated with existing prompting schemes to enhance concept coverage of generated queries.
- We validate the effectiveness of CCQGen by extensive experiments. CCQGen brings significant improvements in query quality and retrieval performance over existing prompting schemes.

2 Preliminaries

2.1 Fine-tuning Retrieval Model

To perform retrieval on a new corpus, a PLM-based retriever is fine-tuned using a training set of annotated query-document pairs. For each query q, the contrastive learning loss is typically applied:

$$\mathcal{L} = -\log \frac{e^{s_{text}(q,\, d^+)}}{e^{s_{text}(q,\, d^+)} + \sum_{d^-} e^{s_{text}(q,\, d^-)}}, \tag{1}$$

where d^+ and d^- denote the relevant and irrelevant documents. $s_{text}(q, d)$ represents the similarity score between the query and a document, computed by the retriever. For effective fine-tuning, a substantial amount of training data is required. However, in specialized domains such as scientific document search, constructing vast human-annotated datasets is challenging due to the need for domain expertise, which remains an obstacle for applications [14, 22].

2.2 Prompt-based Query Generation

Several attempts have been made to generate synthetic queries using LLMs. Recent advancements have centered on advancing prompting schemes to enhance the quality of these queries. We summarize recent methods in terms of their prompting schemes.

Few-shot examples. Several methods [4, 6, 7, 9, 13, 38] incorporate a few examples of relevant query-document pairs in the prompt. The prompt comprises the following components: P =

$\{inst, (d_i, q_i)_{i=1}^k, d_t\}$, where *inst* is the textual instruction[1], $(d_i, q_i)_{i=1}^k$ denotes k examples of the document and its relevant query, and d_t is the new document we want to generate queries for. By providing actual examples of the desired outputs, this technique effectively generates queries with distributions similar to actual queries (e.g., expression styles and lengths) [9]. It is worth noting that this technique is also utilized in subsequent prompting schemes.

Label-conditioning. Relevance label l (e.g., relevant and irrelevant) has been utilized to enhance query generation [4, 7, 38]. The prompt comprises $P = \{inst, (l_i, d_i, q_i)_{i=1}^k, (l_t, d_t)\}$, where k label-document-query triplets are provided as examples. l_i represents the relevance label for the document d_i and its associated query q_i. To generate queries, the prompt takes the desired relevance label l_t along with the document d_t. This technique incorporates knowledge of different relevance, which aids in improving query quality and allows for generating both relevant and irrelevant queries [7].

Pair-wise generation. To further enhance the query quality, the state-of-the-art method [6] introduces a *pair-wise* generation of relevant and irrelevant queries. It instructs LLMs to first generate relevant queries and then generate relatively less relevant ones. The prompt comprises $P = \{inst, (d_i, q_i, q_i^-)_{i=1}^k, d_t\}$, where q_i and q_i^- denote relevant and irrelevant query for d_i, respectively. The generation of irrelevant queries is conditioned on the previously generated relevant ones, allowing for generating thematically similar rather than completely unrelated queries. These queries can serve as natural 'hard negative' samples for training [6].

Remarks. Though effective in generating plausible queries, there remains substantial room for improvement. We observe that existing techniques often generate queries with limited coverage of the document's concepts. That is, the queries frequently cover similar aspects of the document, exhibiting high redundancy and failing to add new training signals. Furthermore, the queries show a high lexical overlap with the document, often repeating a few keywords from the document (§4.2.1). Considering that the same concepts are expressed using diverse terms in actual user queries, merely repeating a few keywords may limit the efficacy of fine-tuning.

3 Methodology

We present **C**oncept **C**overage-based **Q**uery set **Gen**eration (CCQGen) framework, designed to meet two desiderata of training queries: (1) The concepts covered by queries should be complementary to each other, enabling a comprehensive coverage of the document's concepts. (2) The queries should articulate the document concepts in various related terms, rather than merely repeating phrases from the document. CCQGen consists of two major stages:

- **Concept identification and enrichment (§3.1):** We first identify the core academic concepts of each document. Then, we enrich the identified concepts by assessing their importance and adding related concepts not explicitly mentioned in the document. This information serves as the basis for generating queries.
- **Concept coverage-based query generation (§3.2):** Given the previously generated queries $Q_d^{m-1} = \{q_d^1, ..., q_d^{m-1}\}$, we compare the concepts of the document d with those covered by Q_d^{m-1} to identify uncovered concepts. These uncovered concepts are then leveraged as conditions for generating the subsequent query q_d^m, allowing q_d^m to cover complementary aspects of Q_d^{m-1}.

Moreover, we propose a new technique, concept similarity-enhanced retrieval (CSR), that leverages the obtained concept information for **filtering out low-quality queries** and for **improving retrieval accuracy (§3.2.3)**. Figure 2 provides an overview of CCQGen.

3.1 Concept Identification and Enrichment

To measure concept coverage, we first identify the core academic concepts of each document. We represent the concepts using a combination of two different granularities: topic and phrase levels (Figure 2a). Topic level provides broader categorizations of research, such as 'collaborative filtering' or 'machine learning', while phrase level includes specific terms in the document, such as 'playlist continuation' or 'song-to-playlist classifier', complementarily revealing the document concepts.

A tempting way to obtain these topics and phrases is to simply instruct LLMs to find them in each document. However, this approach has several limitations: the results may contain concepts not covered by the document, and there is always a potential risk of hallucination. As a solution, we propose a new approach that first constructs a candidate set, and then uses LLMs to pinpoint the most relevant ones from the given candidates, instead of directly generating them. By doing so, the output space is restricted to the predefined candidate space, greatly reducing the risk of hallucinations while effectively leveraging the language-understanding capability of LLMs.

3.1.1 ***Core topics identification.*** To identify the core topics of documents, we propose using an *academic topic taxonomy* [44]. In the scientific domain, academic taxonomies are widely used for categorizing studies in various institutions and can be easily obtained from the web.[2] A taxonomy refers to a hierarchical tree structure outlining academic topics (Figure 2a). Each node represents a topic, with child nodes corresponding to its sub-topics. Leveraging taxonomy allows for exploiting domain knowledge of topic hierarchy and reflecting researchers' tendency to classify studies.

Candidate set construction. One challenge in finding candidate topics is that the taxonomy obtained from the web is often very large and contains many irrelevant topics.To effectively narrow down the candidates, we employ a *top-down traversal* technique that *recursively visits* the child nodes with the highest similarities at each level. For each document, we start from the root node and compute its similarity to each child node. We then visit child nodes with the highest similarities.[3] This process recurs until every path reaches leaf nodes, and *all visited nodes* are regarded as candidates.

The document-topic similarity $s(d, c)$ can be defined in various ways. As a topic encompasses its subtopics, we collectively consider the subtopic information for each topic node. Let $\mathcal{N}_c$ denote the set of nodes in the sub-tree having c as a root node. We compute the similarity as: $s(d, c) = \frac{1}{|\mathcal{N}_c|} \sum_{j \in \mathcal{N}_c} \cos(\mathbf{e}_d, \mathbf{e}_j)$, where $\mathbf{e}_d$ and $\mathbf{e}_j$

[1] For example, "*Given a document, generate five search queries for which the document can be a perfect answer*". The instructions vary slightly across methods, typically in terms of word choice. In this work, we follow the instructions used in [6].

[2] E.g., IEEE Taxonomy (link), ACM Computing Classification System (link).

[3] We visit multiple child nodes and create multiple paths, as a document usually covers various topics. For a node at level l, we visit $l+2$ nodes to reflect the increasing number of nodes at deeper levels of the taxonomy. The root node is level 0.

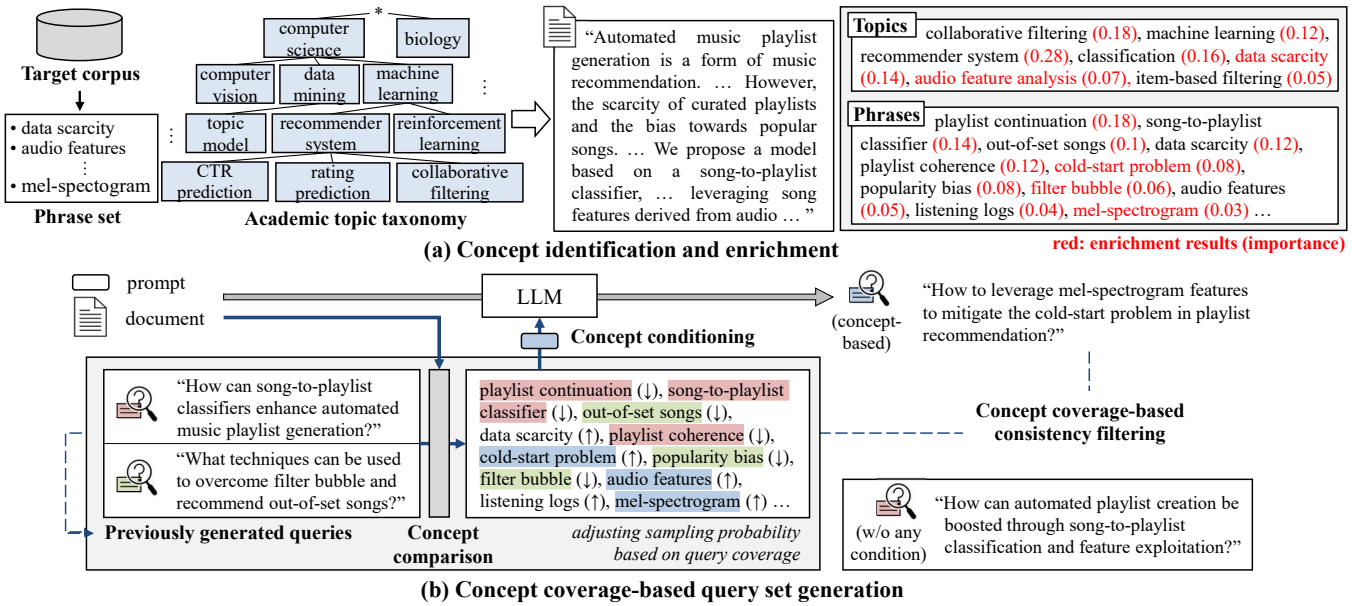

Figure 2: The overview of Concept Coverage-based Query set Generation (CCQGen) framework. Best viewed in color.

denote representations from PLM for a document d and the topic name of node j, respectively.[4]

Core topic selection. We instruct LLMs to select the most relevant topics from the candidates. An example of an input prompt is:

> You will receive a document along with a set of candidate topics. Your task is to select the topics that best align with the core theme of the document. Exclude topics that are too broad or less relevant. You may list up to $[k^t]$ topics, using only the topic names in the candidate set. **Document**: [DOCUMENT], **Candidate topic set**: [CANDIDATES]

In this work, we set $k^t = 10$. For each document d, we obtain core topics as $\mathbf{y}_d^t \in \{0,1\}^{|\mathcal{T}|}$, where $y_{di}^t = 1$ indicates i is a core topic of d, otherwise 0. $\mathcal{T}$ denotes the topic set obtained from the taxonomy.

3.1.2 ***Core phrases identification.*** From each document, we identify core phrases used to describe its concepts. These phrases offer fine-grained details not captured at the topic level. We note that not all phrases in the document are equally important. Core phrases should describe concepts strongly relevant to the document but *not frequently covered* by other documents with similar topics. For example, among documents about 'recommender system' topic, the phrase 'user-item interaction' is very commonly used, and less likely to represent the most important concepts of the document.

Candidate set construction. Given the phrase set $\mathcal{P}$ of the corpus[5], we measure the distinctiveness of phrase p in document d. Inspired by recent phrase mining methods [20, 46], we compute the distinctiveness as: $\exp(\text{BM25}(p,d)) / (1+\sum_{d' \in \mathcal{D}_d} \exp(\text{BM25}(p,d')))$. This quantifies the relative relevance of p to the document d compared to other topically similar documents $\mathcal{D}_d$. $\mathcal{D}_d$ is simply retrieved using Jaccard similarity of core topics $\mathbf{y}_d^t$. We set $|\mathcal{D}_d| = 100$. We select phrases with top-20% distinctiveness score as candidates.

Core phrase selection. We instruct LLMs to select the most relevant phrases (up to k^p phrases) from the candidates, using the same instruction format used for the topic selection. We set $k^p = 15$. The core phrases are denoted by $\mathbf{y}_d^p \in \{0,1\}^{|\mathcal{P}|}$, where $y_{dj}^p = 1$ indicates j is a core phrase of d, otherwise 0.

3.1.3 ***Enriching concept information.*** We have identified core topics and phrases representing each document's concepts. We further enrich this information by (1) measuring their relative importance, and (2) incorporating strongly related concepts (i.e., topics and phrases) not explicitly revealed in the document. This enriched information serves as the basis for generating queries.

Concept extractor. We employ a small model called a *concept extractor*. For a document d, the model is trained to predict its core topics $\mathbf{y}_d^t$ and phrases $\mathbf{y}_d^p$ from the PLM representation $\mathbf{e}_d$. We formulate this as a two-level classification task: topic and phrase levels.

Topics and phrases represent concepts at different levels of granularity, and learning one task can aid the other by providing a complementary perspective. To exploit their complementarity, we employ a multi-task learning model with two heads [28]. Each head has a Softmax output layer, producing probabilities for topics $\hat{\mathbf{y}}_d^t$ and phrases $\hat{\mathbf{y}}_d^p$, respectively. The cross-entropy loss is then applied for classification learning: $-\sum_{i=1}^{|\mathcal{T}|} y_{di}^t \log \hat{y}_{di}^t - \sum_{j=1}^{|\mathcal{P}|} y_{dj}^p \log \hat{y}_{dj}^p$.

Concept enrichment. Using the trained concept extractor, we compute $\hat{\mathbf{y}}_d^t$ and $\hat{\mathbf{y}}_d^p$, which reveal their importance in describing the document's concepts. Also, we identify strongly related topics and phrases that are expressed differently or not explicitly mentioned, by incorporating those with the highest prediction probabilities. For example, in Figure 2, we identify phrases 'cold-start problem', 'filter bubble', and 'mel-spectrogram', which are strongly relevant to the document's concepts but not explicitly mentioned, along with their detailed importance. These phrases are used to aid in articulating the document's concepts in various related terms.

[4]We use BERT with mean pooling as the simplest choice.
[5]The phrase set is obtained using an off-the-shelf phrase mining tool [42].

We obtain $k^{t'}$ enriched topics and $k^{p'}$ enriched phrases for each document with their importance from $\hat{\mathbf{y}}_d^t$ and $\hat{\mathbf{y}}_d^p$. We set the probabilities for the remaining topics and phrases as 0, and normalize the probabilities for selected topics and phrases, denoted by $\bar{\mathbf{y}}_d^t$ and $\bar{\mathbf{y}}_d^p$.

3.2 Concept Coverage-based Query Generation

We present how we generate a set of queries that comprehensively cover the various concepts of a document. We first identify concepts insufficiently covered by the previously generated queries (§3.2.1) and leverage them as conditions for subsequent generation (§3.2.2). Then, a filtering step is applied to ensure the query quality (§3.2.3).

This process is repeated until a predefined number (M) of queries per document is achieved. M is empirically determined, considering available training resources such as GPU memory and training time. For the first query of each document, we impose no conditions, thus it is identical to the results obtained from existing methods.

3.2.1 ***Concept sampling based on query coverage.*** The enriched information $\bar{\mathbf{y}}_d$ reveals the core concepts and their importance within the document. Our key idea is to generate queries that align with this distribution to ensure comprehensive coverage of the document's concepts. Let $Q_d^{m-1} = \{q_d^1, ..., q_d^{m-1}\}$ denote the previously generated queries. Using the concept extractor, which is trained to predict core concepts from the text, we identify the concepts covered by the queries, i.e., $\bar{\mathbf{y}}_Q^t$ and $\bar{\mathbf{y}}_Q^p$. We use the concatenation of queries as input, denoted as Q. A high value in $\bar{\mathbf{y}}_d$ coupled with a low value in $\bar{\mathbf{y}}_Q$ indicates that the existing queries do not sufficiently cover the corresponding concepts.

Based on the concept coverage information, we identify concepts that need to be more emphasized in the subsequently generated query. We opt to leverage phrases as *explicit* conditions for generation, as topics reveal concepts at a broad level, making them less effective for explicit inclusion in the query. Note that topics are *implicitly* reflected in identifying and enriching core phrases. We define a probability distribution to identify less covered concepts as:

$$\boldsymbol{\pi} = \text{normalize}(\ \max(\bar{\mathbf{y}}_d^p - \bar{\mathbf{y}}_Q^p,\ \epsilon)\ ) \tag{2}$$

We set $\epsilon = 10^{-3}$ as a minimal value to the core phrases for numerical stability. We sample $\lfloor \frac{k^{p'}}{M} \rfloor$ different phrases from Multinomial($\boldsymbol{\pi}$), where M is the total number of queries per document. Note that $\bar{\mathbf{y}}_Q^p$ is dynamically adjusted during the construction of the query set.

3.2.2 ***Concept conditioning for query generation.*** The sampled phrases are leveraged as conditions for generating the next query q_d^m. There have been active studies to control the generation of LLMs for various tasks. Recent methods [23, 57] have specified conditions for the desired outputs, such as sentiment, keywords, and an outline, directly in the prompts. Following these studies, we impose a condition by adding a simple textual instruction C: "*Generate a relevant query based on the following keywords*: [Sampled phrases]". While more sophisticated instruction could be employed, we obtained satisfactory results with our choice.

The final prompt is constructed as $[P; C]$, where P is an existing prompting scheme discussed in §2.2. This integration allows us to inherit the benefits of existing techniques (e.g., few-shot examples), while generating queries that comprehensively cover the document's concepts. For example, in Figure 2, C includes phrases like 'cold-start problem' and 'audio features', which are not well covered by the previous queries. Based on this concept condition, we guide LLMs to generate a query that covers complementary aspects to the previous ones. It is important to note that C adds an *additional condition* for P; the query is still about playlist recommendation, the main task of the document.

3.2.3 ***Concept coverage-based consistency filtering.*** After generating a query, we apply a filtering step to ensure its quality. A critical criterion for this process is *round-trip consistency* [1]; a query should be answerable by the document from which it was generated. Existing work [7, 9] employs a retriever to assess this consistency. Given a generated pair (q_d, d), the retriever retrieves documents for q_d. Then, q_d is retained only if d ranks within the top-N results. The accuracy of the retriever is crucial in this step; a weak retriever may fail to filter out low-quality queries and also only retain queries that are *too easy* (e.g., high lexical overlap with the document), thereby limiting the effectiveness of training.

We note that relying on the existing retriever is insufficient for measuring relevance. While it is effective at capturing similarities of surface texts, the retriever often fails to match underlying concepts. For example, in Figure 2, the generated query includes phrases 'cold-start problem' and 'mel-spectrogram', which are highly pertinent to 'data scarcity' and 'audio features' discussed in the document. Nevertheless, as these phrases are not directly used in the document, the retriever struggles to assess the relevance and ranks the document low. Consequently, the query is considered unreliable and removed during the filtering process.

Concept similarity-enhanced retrieval (CSR). We propose a simple and effective technique to enhance retrieval by using concept information. For relevance prediction, we consider both textual similarity from the retriever $s_{text}(q, d)$, and concept similarity $s_{concept}(q, d)$. We measure concept similarity using core phrase distributions, i.e., $s_{concept}(q, d) = sim(\bar{\mathbf{y}}_q^p,\ \bar{\mathbf{y}}_d^p)$, which reveals related concepts at a fine-grained level.[6] $sim(\cdot, \cdot)$ is the similarity function, for which use inner-product. The relevance score is defined as:

$$rel_{CSR}(q, d) = f(s_{text}(q, d),\ s_{concept}(q, d)), \tag{3}$$

where $f(\cdot, \cdot)$ is a function that combines the two scores. We use a simple addition after rescaling them via z-score normalization. We denote this technique as Concept Similarity-enhanced Retrieval (CSR).

For **filtering process**, we assess the round-trip consistency using CSR. By directly matching underlying concepts not apparent from the surface text, we can more accurately measure relevance and distinguish low-quality queries. Additionally, for **search with test queries** (i.e., after fine-tuning using the generated data), CSR can be used as a supplementary technique to further enhance retrieval. It helps to understand test queries, which contain highly limited contexts and jargon not included in the training queries, by associating them with pre-organized concept information.

[6] Here, we compute the similarity for top-10% phrases (instead of $k^{p'}$) to consider concepts having a certain degree of relevance. We also tried using core topics. However, it proved less effective as topics reveal concepts only at a broad level.

4 Experiments

Datasets. We conduct a thorough review of the literature to find retrieval datasets in the scientific domain, specifically those where relevance has been assessed by skilled experts or annotators. We select two recently published datasets: **CSFCube** [31] and **DORIS-MAE** [49]. They offer test query collections annotated by human experts and LLMs, respectively, and embody two real-world search scenarios: query-by-example and human-written queries. For both datasets, we conduct retrieval from the entire corpus, including all candidate documents. CSFCube dataset consists of 50 test queries, with about 120 candidates per query drawn from approximately 800,000 papers in the S2ORC corpus [26]. DORIS-MAE dataset consists of 165,144 test queries, with candidates drawn similarly to CSFCube. We consider annotation scores above '2', which indicate documents are 'nearly identical or similar' (CSFCube) and 'directly answer all key components' (DORIS-MAE), as relevant. Note that training queries are not provided in both datasets.

Academic topic taxonomy. We utilize the field of study taxonomy from Microsoft Academic [44], which contains 431, 416 nodes with a maximum depth of 4. After the concept identification step (§3.1), we obtain 1, 164 topics and 18, 440 phrases for CSFCube, and 1, 498 topics and 34, 311 phrases for DORIS-MAE.

Metrics. Following [14, 29], we employ Recall@K (R@K) for a large retrieval size (K), and NDCG@K (N@K) and MAP@K (M@K) for a smaller K ($\leq$ 20). Recall@K measures the proportion of relevant documents in the top K results, while NDCG@K and MAP@K assign higher weights to relevant documents at higher ranks.

Backbone retrievers. We employ two representative models: (1) **Contriever-MS** [12] is a widely used retriever fine-tuned using vast labeled data from general domains (i.e., MS MARCO). (2) **SPECTER-v2** [45] is a PLM specifically developed for the scientific domain. It is trained using metadata (e.g., citation relations) of scientific papers. For both models, we use public checkpoints: `facebook/contriever-msmarco` and `allenai/specter2_base`.

Baselines. We compare various query generation methods. For all LLM-based methods, we use `gpt-3.5-turbo-0125`. Additionally, we explore the results with a smaller LLM (`Llama-3-8B`) in §4.2.3. For each document, we generate **five** relevant queries [47].

- **GenQ** [47] employs a specialized query generation model, trained with massive document-query pairs from the general domains. We use T5-base, trained using approximately 500, 000 pairs from MS MARCO dataset [32]: `BeIR/query-gen-msmarco-t5-base-v1`.

CCQGen can be flexibly integrated with existing LLM-based methods to enhance the concept coverage of the generated queries. We apply CCQGen to two recent approaches, discussed in §2.2.

- **Promptgator** [9] is a recent LLM-based query generation method that leverages **few-shot examples** within the prompt.
- **Pair-wise generation** [6] is the state-of-the-art method that generates relevant and irrelevant queries in a **pair-wise** manner.

Additionally, we devise a new competitor that adds more instruction in the prompt to enhance the quality of queries: **Promptgator_diverse** is a variant of Promptgator, where we add the instruction "*use various terms and reduce redundancy among the queries*".

Implementation details. We conduct all experiments using 4 NVIDIA RTX A5000 GPUs, 512 GB memory, and a single Intel Xeon Gold 6226R processor. For fine-tuning, we use top-50 BM25 hard negatives for each query [10]. We use 10% of training data as a validation set. The learning rate is set to 10^{-6} for Contriever-MS and 10^{-7} for SPECTER-v2, after searching among $\{10^{-7}, 10^{-6}, ..., 10^{-3}\}$. We set the batch size as 64 and the weight decay as 10^{-4}. We report the average performance over five independent runs. For all methods, we generate five synthetic queries for each document ($M = 5$). For the few-shot examples in the prompt, we randomly select five annotated examples, which are then excluded in the evaluation process [9]. We follow the textual instruction used in [6]. For other baseline-specific setups, we adhere to the configurations described in the original papers. For the concept extractor, we employ a multi-gate mixture of expert architecture [28], designed for multi-task learning. We use three experts, each being a two-layer MLP. For the consistency filtering, we set $N = 5$. We set the number of enriched topics and phrases to $k^{t'} = 15$ and $k^{p'} = 20$, respectively.

4.1 Performance Comparison

4.1.1 ***Effectiveness of CCQGen.*** Table 2 presents retrieval performance after fine-tuning with various query generation methods. CCQGen consistently outperforms all baselines, achieving significant improvements across various metrics with both backbone models. We observe that GenQ underperforms compared to LLM-based methods, showing the advantages of leveraging the text generation capability of LLMs. Also, existing methods often fail to improve the backbone model (i.e., no Fine-Tune), particularly Contriever-MS. As it is trained on labeled data from general domains, it already captures overall textual similarities well, making further improvements challenging. The consistent improvements by CCQGen support its efficacy in generating queries that effectively represent the scientific documents. Notably, Promptgator_diverse struggles to produce consistent improvements. We observe that it often generates redundant queries covering similar aspects, despite increased diversity in their expressions (further analysis provided in §4.2.1). This underscores the importance of proper control over generated content and supports the validity of our approach.

Impact of amount of training data. In Figure 3, we further explore the retrieval performance by limiting the amount of training data, using Contriever-MS as the backbone model. The existing LLM-based generation method (i.e., Pair-wise gen.) shows limited performance under restricted data conditions and fails to fully benefit from an increasing volume of training data. This supports our claim that the generated queries are often redundant and do not effectively introduce new training signals. Conversely, CCQGen consistently delivers considerable improvements, even with a limited number of queries. CCQGen guides each new query to complement the previous ones, allowing for reducing redundancy and fully leveraging the limited number of queries.

4.1.2 ***Effectiveness of CCQGen with CSR.*** In §3.2.3, we introduce CSR, designed to enhance retrieval using concept information from CCQGen. This technique aligns with the ongoing research direction of enhancing retrieval by integrating additional context not directly revealed from the queries and document [14, 29, 52, 56]. We compare CSR with two recent methods: (1) **GRF** [29] generates relevant contexts by LLMs. For a fair comparison, we generate both topics and keywords, as used in CCQGen. (2) **ToTER** [14] uses the

Table 2: Retrieval performance comparison after fine-tuning with the generated queries. Red color denotes results that fail to show improvements over no Fine-Tune. † and * indicate a statistically significant difference ($p < 0.05$) from no Fine-Tune (one-sample t-test) and the applied query generation method (paired t-test), respectively.

	Query generation	CSFCube						DORIS-MAE					
		N@10	N@20	M@10	M@20	R@50	R@100	N@10	N@20	M@10	M@20	R@50	R@100
Contriever-MS	no Fine-Tune	0.3313	0.3604	0.1525	0.1937	0.5783	0.7136	0.2603	0.2707	0.1177	0.1422	0.4509	0.5877
	GenQ	0.3401	0.3495	0.1476	0.1841	0.5571	0.6843	0.2496	0.2647	0.1152	0.1396	0.4598	0.5805
	Promptgator_diverse	0.3539	0.3771	0.1606	0.2029	0.5950	0.7132	0.2461	0.2690	0.1143	0.1406	0.4645	0.5951
	Promptgator	0.3441	0.3670	0.1538	0.1974	0.5928	0.7298	0.2526	0.2724	0.1161	0.1418	0.4718	0.5961
	w/ CCQGen (ours)	**0.3605**†	**0.3991**$^{\dagger *}$	**0.1614**†	**0.2194**$^{\dagger *}$	**0.6333**$^{\dagger *}$	**0.7467**†	**0.2697***	**0.2883**$^{\dagger *}$	**0.1267**$^{\dagger *}$	**0.1536**$^{\dagger *}$	**0.4983**$^{\dagger *}$	**0.6327**$^{\dagger *}$
	Pair-wise generation	0.3418	0.3686	0.1522	0.1971	0.5961	0.7225	0.2541	0.2753	0.1177	0.1445	0.4809	0.5947
	w/ CCQGen (ours)	**0.3670**$^{\dagger *}$	**0.4063**$^{\dagger *}$	**0.1656**$^{\dagger *}$	**0.2228**$^{\dagger *}$	**0.6362**$^{\dagger *}$	**0.7526**$^{\dagger *}$	**0.2783**$^{\dagger *}$	**0.2943**$^{\dagger *}$	**0.1308**$^{\dagger *}$	**0.1577**$^{\dagger *}$	**0.5089**$^{\dagger *}$	**0.6331**$^{\dagger *}$
SPECTER-v2	no Fine-Tune	0.3503	0.3579	0.1615	0.2043	0.5341	0.6859	0.2121	0.2283	0.0942	0.1147	0.4182	0.5441
	GenQ	0.3658	0.3659	0.1699	0.2083	0.5541	0.6836	0.2338	0.2525	0.1045	0.1287	0.4412	0.5613
	Promptgator_diverse	0.3672	0.3801	0.1721	0.2157	0.5687	0.6972	0.2469	0.2733	0.1121	0.1401	0.4843	0.6102
	Promptgator	0.3766	0.3886	0.1790	0.2245	0.5715	0.6962	0.2479	0.2713	0.1131	0.1398	0.4851	0.6064
	w/ CCQGen (ours)	**0.4105**$^{\dagger *}$	**0.4176**$^{\dagger *}$	**0.2085**$^{\dagger *}$	**0.2549**$^{\dagger *}$	**0.5886**†	**0.7355**$^{\dagger *}$	**0.2634**$^{\dagger *}$	**0.2891**$^{\dagger *}$	**0.1226**†	**0.1520**$^{\dagger *}$	**0.4988**†	**0.6265**†
	Pair-wise generation	0.3870	0.3999	0.1966	0.2423	0.5722	0.6972	0.2523	0.2782	0.1163	0.1442	0.4885	0.6148
	w/ CCQGen (ours)	**0.4031**$^{\dagger *}$	**0.4150**†	**0.2040**†	**0.2534**$^{\dagger *}$	**0.5844**†	**0.7333**$^{\dagger *}$	**0.2681**$^{\dagger *}$	**0.2932**$^{\dagger *}$	**0.1247**†	**0.1546**$^{\dagger *}$	**0.5064**†	**0.6304**†

Table 3: Retrieval performance comparison with various enhancement methods. * indicates a significant difference (paired t-test, $p < 0.05$) from the best baseline (i.e., the combination of the best existing query generation and enhancement methods).

Query generation	Retrieval enhancement	CSFCube						DORIS-MAE					
		N@10	N@20	M@10	M@20	R@50	R@100	N@10	N@20	M@10	M@20	R@50	R@100
Pair-wise generation	Retriever	0.3418	0.3686	0.1522	0.1971	0.5961	0.7225	0.2541	0.2753	0.1178	0.1445	0.4809	0.5947
	w/ GRF	0.3401	0.3713	0.1540	0.2008	0.5778	0.7151	0.2535	0.2753	0.1147	0.1416	0.4832	0.6159
	w/ ToTER	**0.3745**	**0.4072**	**0.1719**	**0.2267**	**0.6352**	**0.7606**	**0.2932**	**0.3138**	**0.1381**	**0.1680**	**0.5361**	**0.6579**
w/ CCQGen (ours)	Retriever	0.3670	0.4063	0.1656	0.2228	0.6362	0.7526	0.2783	0.2943	0.1308	0.1577	0.5089	0.6331
	w/ GRF	0.3741	0.4071	0.1715	0.2272	0.6288	0.7490	0.2709	0.2925	0.1262	0.1542	0.5138	0.6384
	w/ ToTER	0.4023	0.4205	0.1844	0.2403	**0.6441**	0.7698	0.2965	0.3159	0.1394	0.1697	0.5391	0.6635
	w/ CSR (ours)	**0.4244***	**0.4359***	**0.2029***	**0.2530***	0.6412	**0.7792***	**0.3034***	**0.3237***	**0.1438***	**0.1745***	**0.5588***	**0.6818***

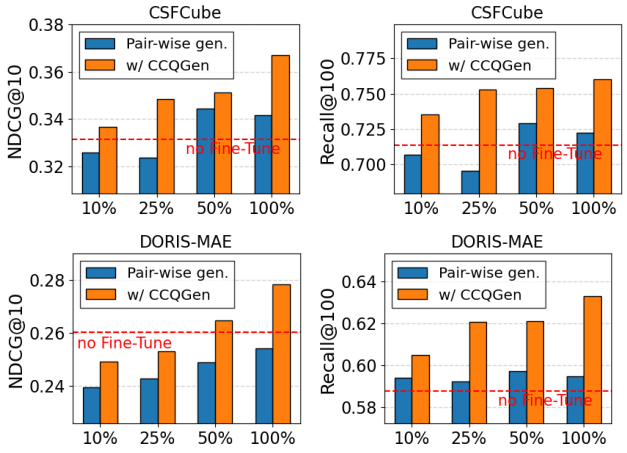

Figure 3: Results with varying amounts of training data. x% denotes setups using a random x% of generated queries.

topic distributions between queries and documents, with topics provided by the taxonomy. Contriever-MS is used as the backbone.

Table 3 presents the retrieval performance with various enhancement methods. CSR significantly improves the retrieval performance. Notably, the combination of proposed concept-based query generation (CCQGen) and enhancement (CSR) methods achieves significant improvements over the best existing solutions (i.e., Pair-wise gen. combined with ToTER). GRF often degrades performance because the LLM-generated contexts are not tailored to target documents; these contexts may be related but often not covered by documents in the corpus, potentially causing discrepancies in focused aspects. Lastly, ToTER only considers topic-level information, which may be insufficient for providing find-grained details necessary to distinguish between topically-similar documents.

4.2 Study of CCQGen

4.2.1 ***Analysis of generated queries.*** We analyze whether CCQGen indeed reduces redundancy among the queries and includes a variety of related terms. We introduce two criteria: (1) **redundancy**, measured as the average cosine similarity of term frequency vectors of queries.[7] A high redundancy indicates that queries tend to cover similar aspects of the document. (2) **lexical overlap**, measured as the average BM25 score between the queries and the document. A higher lexical overlap indicates that queries tend to reuse terms from the document.

In Table 4, the generated queries show higher lexical overlap with the document compared to the actual user queries. This shows that the generated queries tend to use a limited range of terms already present in the document, whereas actual user queries include a broader variety of terms. With the 'diverse condition' (i.e., Promptgator_diverse), the generated queries exhibit reduced lexical overlap and redundancy. However, this does not consistently

[7]We use CountVectorizer from the SciKit-Learn library.

Table 4: Analysis of generated queries. (a) Statistics of queries generated by different methods. (b) Retriever performance (SPECTER-v2 on NDCG@10) after fine-tuning using the queries. The average lexical overlap of actual queries is 13.32 for CSFCube and 20.42 for DORIS-MAE.

CSFCube

Query generation	(a) Query statistics		(b) Retriever performance
	redundancy (↓)	lexical overlap (↓)	
Promptgator	0.5072	31.51	0.3766
w/ diverse condition	0.4512 (-11.0%)	**24.05 (-23.7%)**	0.3672 (-2.6%)
w/ CCQGen	**0.3997 (-21.2%)**	24.41 (-22.5%)	**0.4105 (+9.0%)**

DORIS-MAE

Query generation	(a) Query statistics		(b) Retriever performance
	redundancy (↓)	lexical overlap (↓)	
Promptgator	0.4861	53.58	0.2479
w/ diverse condition	**0.3958 (-18.6%)**	41.56 (-22.4%)	0.2469 (-0.4%)
w/ CCQGen	0.3993 (-17.9%)	**40.54 (-24.3%)**	**0.2634 (+6.2%)**

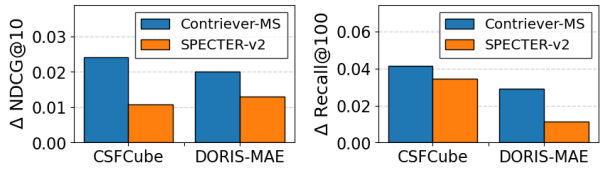

Figure 4: Improvements by concept coverage-based filtering.

lead to performance improvements. The improved term usage often appears in common expressions, not necessarily enhancing concept coverage. Conversely, CCQGen directly guides each new query to complement the previous ones. Also, CCQGen incorporate concept-related terms not explicitly mentioned in the document via enrichment step (§3.1.3). This provides more systematic controls over the generation, leading to consistent improvements.

4.2.2 ***Effectiveness of concept coverage-based filtering.*** Figure 4 presents the improvements achieved through the filtering step, which aims to remove low-quality queries that the document does not answer (§3.2.3). As shown in Table 3, CSR largely enhances retrieval accuracy by incorporating concept information. This enhanced accuracy helps to accurately measure round-trip consistency, effectively improving the effects of fine-tuning.

4.2.3 ***Results with a smaller LLM.*** In Table 5, we explore the effectiveness of the proposed approach using a smaller LLM, Llama-3-8B, with Contriever-MS as the backbone model. Consistent with the trends observed in Table 2 and Table 3, the proposed techniques (CCQGen and CSR) consistently improve the existing method. We expect CCQGen to be effective with existing LLMs that possess a certain degree of capability. Since comparing different LLMs is not the focus of this work, we leave further investigation on more various LLMs and their comparison for future study.

5 Related Work

We provide a detailed survey of LLM-based query generation in §2.2.

PLM-based retrieval models. The advancement of PLMs has led to significant progress in retrieval. Recent studies have introduced retrieval-targeted pre-training [11, 12], distillation from cross-encoders [54], and advanced negative mining methods [36, 53]

Table 5: Retrieval performance with Llama-3-8B. We report improvements over no Fine-Tune. * denotes $p < 0.05$ from paired t-test with pair-wise generation.

Dataset	Method	N@10	N@20	R@100
CSFCube	Pair-wise generation	+5.25%	+0.94%	- 0.21%
	w/ CCQGen	+6.55%	+7.82%*	+5.48%*
	w/ CCQGen + CSR	+27.92%*	+20.09%*	+9.01%*
DORIS-MAE	Pair-wise generation	+0.00%	+2.92%	+5.43%
	w/ CCQGen	+5.19%*	+9.57%*	+6.69%
	w/ CCQGen + CSR	+16.75%*	+20.87%*	+14.65%*

There is also an increasing emphasis on pre-training methods specifically designed for the scientific domain. In addition to pre-training on academic corpora [3], researchers have exploited metadata associated with scientific papers. [37] uses journal class, [8, 34] use citations, [30] uses co-citation contexts, and [25] utilizes venues, affiliations, and authors. [45, 55] devise multi-task learning of related tasks such as citation prediction and paper classification. Very recently, [14, 17] leverage corpus-structured knowledge (e.g., core topics and phrases) for academic concept matching.

Synthetic query generation. Earlier studies [24, 27, 32, 33, 50] have employed dedicated query generation models, trained using massive document-query pairs from general domains. Recently, there has been a shift towards replacing these generation models with LLMs [4, 6, 9, 13, 38, 39], as discussed in §2.2. On the other hand, many recent studies have focused on developing query generation tailored to specific retrieval domains. [41] focuses on entity search for virtual assistants, [43] improves the diversity of queries for news article searches guided by a knowledge graph, [35] focuses on enhancing the retrievability of entities on online content (e.g., Podcast) platforms. However, a dedicated method for scientific document retrieval has not been studied well in the literature.

6 Conclusion

We propose CCQGen framework to generate a set of queries that comprehensively cover the document concepts. CCQGen identifies concepts not sufficiently covered by previous queries, and leverages them as conditions for subsequent query generation. This approach guides each new query to complement the previous ones, aiding in a comprehensive understanding of the document. Extensive experiments show that CCQGen significantly improves both query quality and retrieval performance, even with limited training data. Future work may explore its applicability across various domains. In particular, e-commerce [15, 16, 21] presents a promising opportunity, as users often express multi-faceted needs involving desired attributes, characteristics, or specific use cases. We expect CCQGen to better simulate user queries in such scenarios and leave further investigation for future work.

Acknowledgements. This work was supported IITP grant funded by MSIT (No.2018-0-00584, No.2019-0-01906), NRF grant funded by the MSIT (No.RS-2023-00217286, No.2020R1A2B5B03097210). It was also in part by US DARPA INCAS Program No. HR0011-21-C0165 and BRIES Program No. HR0011-24-3-0325, National Science Foundation IIS-19-56151, the Molecule Maker Lab Institute: An AI Research Institutes program supported by NSF under Award No. 2019897, and the Institute for Geospatial Understanding through an Integrative Discovery Environment (I-GUIDE) by NSF under Award No. 2118329.

References

[1] Chris Alberti, Daniel Andor, Emily Pitler, Jacob Devlin, and Michael Collins. 2019. Synthetic QA Corpora Generation with Roundtrip Consistency. In *ACL*. 6168–6173.

[2] 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* (2016).

[3] Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: Pretrained Language Model for Scientific Text. In *EMNLP*. 3615–3620.

[4] Luiz Bonifacio, Hugo Abonizio, Marzieh Fadaee, and Rodrigo Nogueira. 2022. Inpars: Unsupervised dataset generation for information retrieval. In *SIGIR*. 2387–2392.

[5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *NeurIPS*, Vol. 33. 1877–1901.

[6] Aditi Chaudhary, Karthik Raman, and Michael Bendersky. 2024. It's All Relative!–A Synthetic Query Generation Approach for Improving Zero-Shot Relevance Prediction. In *Findings of NAACL*.

[7] Aditi Chaudhary, Karthik Raman, Krishna Srinivasan, Kazuma Hashimoto, Mike Bendersky, and Marc Najork. 2023. Exploring the viability of synthetic query generation for relevance prediction. In *The SIGIR 2023 Workshop on eCommerce*.

[8] Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel S. Weld. 2020. SPECTER: Document-level Representation Learning using Citation-informed Transformers. In *ACL*.

[9] 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 *ICLR*.

[10] Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. 2022. From distillation to hard negative sampling: Making sparse neural ir models more effective. In *SIGIR*. 2353–2359.

[11] Luyu Gao and Jamie Callan. 2022. Unsupervised Corpus Aware Language Model Pre-training for Dense Passage Retrieval. In *ACL*. 2843–2853.

[12] Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Unsupervised dense information retrieval with contrastive learning. *arXiv preprint arXiv:2112.09118* (2021).

[13] Vitor Jeronymo, Luiz Bonifacio, Hugo Abonizio, Marzieh Fadaee, Roberto Lotufo, Jakub Zavrel, and Rodrigo Nogueira. 2023. Inpars-v2: Large language models as efficient dataset generators for information retrieval. *arXiv preprint arXiv:2301.01820* (2023).

[14] SeongKu Kang, Shivam Agarwal, Bowen Jin, Dongha Lee, Hwanjo Yu, and Jiawei Han. 2024. Improving Retrieval in Theme-specific Applications using a Corpus Topical Taxonomy. In *WWW*. 1497–1508.

[15] SeongKu Kang, Junyoung Hwang, Wonbin Kweon, and Hwanjo Yu. 2020. DE-RRD: A Knowledge Distillation Framework for Recommender System. In *CIKM*.

[16] SeongKu Kang, Junyoung Hwang, Dongha Lee, and Hwanjo Yu. 2019. Semi-supervised learning for cross-domain recommendation to cold-start users. In *CIKM*.

[17] SeongKu Kang, Yunyi Zhang, Pengcheng Jiang, Dongha Lee, Jiawei Han, and Hwanjo Yu. 2024. Taxonomy-guided Semantic Indexing for Academic Paper Search. In *EMNLP*. 7169–7184.

[18] 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 *EMNLP*. 6769–6781.

[19] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics* 7 (2019), 453–466.

[20] Dongha Lee, Jiaming Shen, SeongKu Kang, Susik Yoon, Jiawei Han, and Hwanjo Yu. 2022. Taxocom: Topic taxonomy completion with hierarchical discovery of novel topic clusters. In *WWW*. 2819–2829.

[21] Youngjune Lee, Yeongjong Jeong, Keunchan Park, and SeongKu Kang. 2023. MvFS: Multi-view Feature Selection for Recommender System. In *CIKM*.

[22] Haitao Li, Qingyao Ai, Jia Chen, Qian Dong, Yueyue Wu, Yiqun Liu, Chong Chen, and Qi Tian. 2023. SAILER: Structure-aware Pre-trained Language Model for Legal Case Retrieval. In *SIGIR*.

[23] Yunzhe Li, Qian Chen, Weixiang Yan, Wen Wang, Qinglin Zhang, and Hari Sundaram. 2024. Advancing Precise Outline-Conditioned Text Generation with Task Duality and Explicit Outline Control. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics*. 2362–2377.

[24] Davis Liang, Peng Xu, Siamak Shakeri, Cicero Nogueira dos Santos, Ramesh Nallapati, Zhiheng Huang, and Bing Xiang. 2020. Embedding-based zero-shot retrieval through query generation. *arXiv preprint arXiv:2009.10270* (2020).

[25] Xiao Liu, Da Yin, Jingnan Zheng, Xingjian Zhang, Peng Zhang, Hongxia Yang, Yuxiao Dong, and Jie Tang. 2022. OAG-BERT: Towards a Unified Backbone Language Model for Academic Knowledge Services. In *KDD*. 3418–3428.

[26] Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel S Weld. 2020. S2ORC: The Semantic Scholar Open Research Corpus. In *ACL*. 4969–4983.

[27] Ji Ma, Ivan Korotkov, Yinfei Yang, Keith Hall, and Ryan McDonald. 2021. Zero-shot Neural Passage Retrieval via Domain-targeted Synthetic Question Generation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. 1075–1088.

[28] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H Chi. 2018. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *KDD*. 1930–1939.

[29] Iain Mackie, Shubham Chatterjee, and Jeffrey Dalton. 2023. Generative Relevance Feedback with Large Language Models. In *SIGIR*. 2026–2031.

[30] Sheshera Mysore, Arman Cohan, and Tom Hope. 2022. Multi-Vector Models with Textual Guidance for Fine-Grained Scientific Document Similarity. In *NAACL*. 4453–4470.

[31] Sheshera Mysore, Tim O'Gorman, Andrew McCallum, and Hamed Zamani. 2021. CSFCube-A Test Collection of Computer Science Research Articles for Faceted Query by Example. *NeurIPS 2021 Track on Datasets and Benchmarks* (2021).

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

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

[34] Malte Ostendorff, Nils Rethmeier, Isabelle Augenstein, Bela Gipp, and Georg Rehm. 2022. Neighborhood Contrastive Learning for Scientific Document Representations with Citation Embeddings. In *EMNLP*.

[35] Gustavo Penha, Enrico Palumbo, Maryam Aziz, Alice Wang, and Hugues Bouchard. 2023. Improving content retrievability in search with controllable query generation. In *WWW*. 3182–3192.

[36] 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 *NAACL-HLT*. 5835–5847.

[37] Anastasiia Razdaibiedina and Aleksandr Brechalov. 2023. MIReAD: Simple Method for Learning High-quality Representations from Scientific Documents. In *ACL*. 530–539.

[38] Jon Saad-Falcon, Omar Khattab, Keshav Santhanam, Radu Florian, Martin Franz, Salim Roukos, Avirup Sil, Md Sultan, and Christopher Potts. 2023. UDAPDR: Unsupervised Domain Adaptation via LLM Prompting and Distillation of Rerankers. In *EMNLP*. 11265–11279.

[39] 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 *EMNLP*. 3781–3797.

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

[41] Sonal Sannigrahi, Thiago Fraga-Silva, Youssef Oualil, and Christophe Van Gysel. 2024. Synthetic query generation using large language models for virtual assistants. In *SIGIR*. 2837–2841.

[42] Jingbo Shang, Jialu Liu, Meng Jiang, Xiang Ren, Clare R Voss, and Jiawei Han. 2018. Automated phrase mining from massive text corpora. *IEEE Transactions on Knowledge and Data Engineering* 30, 10 (2018), 1825–1837.

[43] Xinyao Shen, Jiangjie Chen, Jiaze Chen, Chun Zeng, and Yanghua Xiao. 2022. Diversified query generation guided by knowledge graph. In *WSDM*. 897–907.

[44] Zhihong Shen, Hao Ma, and Kuansan Wang. 2018. A Web-scale system for scientific knowledge exploration. In *Proceedings of ACL 2018, System Demonstrations*. 87–92.

[45] Amanpreet Singh, Mike D'Arcy, Arman Cohan, Doug Downey, and Sergey Feldman. 2023. SciRepEval: A Multi-Format Benchmark for Scientific Document Representations. In *EMNLP*. 5548–5566.

[46] Fangbo Tao, Honglei Zhuang, Chi Wang Yu, Qi Wang, Taylor Cassidy, Lance M Kaplan, Clare R Voss, and Jiawei Han. 2016. Multi-Dimensional, Phrase-Based Summarization in Text Cubes. *IEEE Data Eng. Bull.* 39, 3 (2016), 74–84.

[47] 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 *NeurIPS Datasets and Benchmarks Track*.

[48] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971* (2023).

[49] Jianyou Wang, Kaicheng Wang, Xiaoyue Wang, Prudhviraj Naidu, Leon Bergen, and Ramamohan Paturi. 2023. Scientific Document Retrieval using Multi-level Aspect-based Queries. In *NeurIPS Datasets and Benchmarks Track*.

[50] Kexin Wang, Nandan Thakur, Nils Reimers, and Iryna Gurevych. 2022. GPL: Generative Pseudo Labeling for Unsupervised Domain Adaptation of Dense Retrieval. In *NAACL*. 2345–2360.

[51] Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models

are zero-shot learners. *arXiv preprint arXiv:2109.01652* (2021).
[52] HongChien Yu, Chenyan Xiong, and Jamie Callan. 2021. Improving query representations for dense retrieval with pseudo relevance feedback. In *CIKM*. 3592–3596.
[53] Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. 2021. Optimizing dense retrieval model training with hard negatives. In *SIGIR*. 1503–1512.
[54] Hang Zhang, Yeyun Gong, Yelong Shen, Jiancheng Lv, Nan Duan, and Weizhu Chen. 2022. Adversarial Retriever-Ranker model for Dense Retrieval. In *ICLR*.
[55] Yu Zhang, Hao Cheng, Zhihong Shen, Xiaodong Liu, Ye-Yi Wang, and Jianfeng Gao. 2023. Pre-training Multi-task Contrastive Learning Models for Scientific Literature Understanding. In *Findings of EMNLP*. 12259–12275.
[56] Zhi Zheng, Kai Hui, Ben He, Xianpei Han, Le Sun, and Andrew Yates. 2020. BERT-QE: Contextualized Query Expansion for Document Re-ranking. In *Findings of EMNLP*. 4718–4728.
[57] Wangchunshu Zhou, Yuchen Eleanor Jiang, Ethan Wilcox, Ryan Cotterell, and Mrinmaya Sachan. 2023. Controlled text generation with natural language instructions. In *ICML*. 42602–42613.