AudioBoost: Increasing Audiobook Retrievability in Spotify Search with Synthetic Query Generation

Enrico Palumbo[1], Gustavo Penha[2], Alva Liu[3], Marcus Eltscheminov[4], Jefferson Carvalho dos Santos[4], Alice Wang[5], Hugues Bouchard[7], Humberto Jesús Corona Pampin[2], Michelle Tran Luu[6]
Spotify
[1]Italy, [2]Netherlands, [3]Sweden, [4]Brazil, [5]USA, [6]UK, [7]Spain
{enricop,gustavop}@spotify.com

Abstract

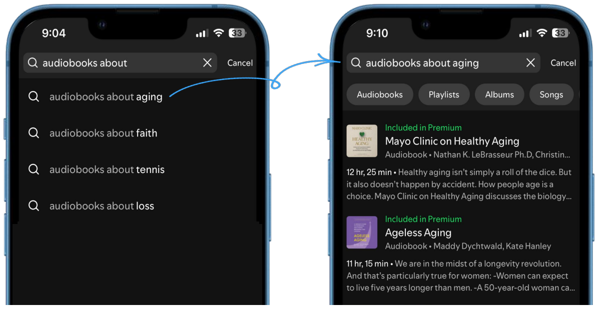
Spotify has recently introduced audiobooks as part of its catalog, complementing its music and podcast offering. Search is often the first entry point for users to access new items, and an important goal for Spotify is to support users in the exploration of the audiobook catalog. More specifically, we would like to enable users without a specific item in mind to broadly search by topic, genre, story tropes, decade, and discover audiobooks, authors and publishers they may like. To do this, we need to 1) inspire users to type more exploratory queries for audiobooks and 2) augment our retrieval systems to better deal with exploratory audiobook queries. This is challenging in a cold-start scenario, where we have a retrievabiliy bias due to the little amount of user interactions with audiobooks compared to previously available items such as music and podcast content. To address this, we propose AudioBoost, a system to boost audiobook retrievability in Spotify's Search via synthetic query generation. AudioBoost leverages Large Language Models (LLMs) to generate synthetic queries conditioned on audiobook metadata. The synthetic queries are indexed both in the Query AutoComplete (QAC) and in the Search Retrieval engine to improve query formulation and retrieval at the same time. We show through offline evaluation that synthetic queries increase retrievability and are of high quality. Moreover, results from an online A/B test show that AudioBoost leads to a +0.7% in audiobook impressions, +1.22% in audiobook clicks, and +1.82% in audiobook exploratory query completions.

Keywords

Synthetic Query Generation, Cold Start, Retrieval, Query Suggestion, LLMs

ACM Reference Format:
Enrico Palumbo[1], Gustavo Penha[2], Alva Liu[3], Marcus Eltscheminov[4], Jefferson Carvalho dos Santos[4], Alice Wang[5], Hugues Bouchard[7], Humberto Jesús Corona Pampin[2], Michelle Tran Luu[6]. 2025. AudioBoost: Increasing Audiobook Retrievability in Spotify Search with Synthetic Query Generation. In *Proceedings of RecSys (EARL Workshop@RecSys)*. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction



audiobooks about

audiobooks about aging
audiobooks about faith
audiobooks about tennis
audiobooks about loss

audiobooks about aging

Mayo Clinic on Healthy Aging
Audiobook • Nathan K. LeBrasseur Ph.D, Christin...
12 hr, 25 min • Healthy aging isn't simply a roll of the dice. But it also doesn't happen by accident. How people age is a choice. Mayo Clinic on Healthy Aging discusses the biology...

Ageless Aging
Audiobook • Maddy Dychtwald, Kate Hanley
11 hr, 15 min • We are in the midst of a longevity revolution. And that's particularly true for women: -Women can expect to live five years longer than men. -A 50-year-old woman ca...

Figure 1: Illustrative example - Synthetic queries generated by an LLM are used as query completions to support query formulation (left) and for retrieval (right) to improve the retrievability of cold-start entities, i.e. audiobooks, in Spotify search.

Search engines in online content platforms have to disambiguate user queries against a huge number of potential catalog items across several possible content types. For instance, if the user searches for "*motivation*" at Spotify, they could be looking for a specific song with the word *motivation* in the title, a motivational playlist that fits their current vibe, or a podcast/audiobook about finding good habits that lead to motivation. To find the most relevant items for a user, modern search engines rely on a variety of signals, such as the user's history and the item's popularity. However, the estimation of the "true" item popularity is challenging in a cold-start scenario where a new type of content has been recently introduced in the platform. In such a scenario, most listeners are not used to engaging with the new content type, and interactions are mostly concentrated on previously available items, hence negatively impacting the retrievability (i.e. the chance of being retrieved) [1] of new items. An important goal is then to raise awareness about the newly introduced items, helping users discover the breadth and depth of the catalog, while, at the same time, supporting authors and publishers in getting their work exposed and connecting with more audience.

Research on search mindsets [2] has highlighted that when users search for broader topics such as "*indie rock*" or "*psychology*" they are more prone to engage with system recommendations and suggestions, thus providing a chance to increase the retrievability of under-served content [11]. Previous work has shown that query suggestions can be used effectively to lead users to type more

EARL Workshop@RecSys, Sept 22–26, 2025, Prague, Czech Republic

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

exploratory, broad queries [3, 7]. In [8] the authors propose a particularly promising paradigm to increase the retrievability of a set of target items that leverages LLMs to perform query generation controlling for the intent (i.e. broad vs narrow). More specifically, in [8] the authors show that exploratory synthetic queries can be generated and used as 1) query suggestions influencing the query distribution toward more exploratory queries 2) to improve retrieval for exploratory queries via document augmentation. Crucially, both steps need to be done to maximize the effectiveness of the approach. While promising, so far this approach has been tested on general tail items rather than on new item types and has never been shown to work at scale in a production system.

In this work, we describe AudioBoost, a system that increases audiobook retrievability in Spotify Search via synthetic query generation. AudioBoost leverages LLMs to generate synthetic queries conditioned on audiobook metadata (e.g. title, author, description). The LLM is prompted using a taxonomy that is defined for the audiobook use case and leveraging a chain-of-thought prompting strategy (Sec. 2.1). Then, the synthetic queries are used as candidates for query completions so that the users can be inspired to type more audiobook exploratory queries (Sec. 2.2). Finally, we perform document augmentation adding the synthetic queries to the audiobook representations in the retrieval system (Sec. 2.3). We use AudioBoost to generate synthetic queries for all audiobooks in the catalog and perform a set of offline evaluations to check the validity of the proposed approach in a controlled setting (Sec. 3). More in detail, we perform a simulation showing that the synthetic queries increase audiobook retrievability as expected (Sec. 3.1), that the quality of the synthetic queries is high using an *LLM-as-a-judge approach* (Sec. 3.2). We finally test our method online in a large-scale A/B test, showing that it leads to +0.7% in audiobook impressions, +1.22% in audiobook clicks, and +1.82% in audiobook exploratory query completions.

In addition to being effective, AudioBoost is also appealing for practical reasons in an industry scenario. AudioBoost is highly scalable - the query generation and indexing steps are performed in offline pipelines, without affecting the latency and with a reasonable cost.

2 Approach

In this section, we describe how the AudioBoost pipeline works and its main components (Fig. 2).

2.1 Query Generation

To generate synthetic queries for audiobooks we first mapped out a taxonomy containing different types of audiobook descriptors. To define such taxonomy, we looked into how users search for audiobooks using both internal search queries and also requests issued at Reddit on the */r/booksuggestions/*[1] forum. For example, users might start a thread asking other users for "*books with a teen protagonist who overcomes many challenges*", revealing that descriptions of the characters might serve as good queries. This manual approach has led to the following taxonomy:

(1) Genres, e.g. "*juvenile fiction*". Genre-specific descriptors are those where users are seeking book recommendations within a particular literary genre such as horror, romance, detective, fantasy, etc.
(2) Themes or topics, e.g. "*global politics*". Descriptors based on specific subjects themes or topics such as societal issues, self-improvement, education, etc.
(3) Characters Descriptions, e.g. "*heroic protagonist*". Descriptors about the personality, development, relationships, or unique characteristics of the protagonists or other significant characters within the story, e.g. heroic protagonist, book with great villains, etc.
(4) Moods, e.g. "*adventurous*". Descriptors related to evoking a specific emotional response or mood, for example, books that make you cry, light books with elements of humor, dark or grim books, etc.
(5) Settings, e.g. "*China's Cultural Revolution.*". Descriptors that focus on the reader's desire to immerse themselves in a specific location, period, or mood through the book they read, for example, Halloween reads, Christmas books, books set in Venice, books with cozy spooky magical vibes, etc.
(6) Personal situations, e.g. "*dealing with loss.*". Descriptors tailored to specific circumstances or challenges people are facing in their personal lives, e.g. parenting and family, dealing with loss or grief, books on relationships, personal life transition, etc.
(7) Story tropes, e.g. "*enemies to lovers.*". Descriptors that explore specific narrative themes or plot devices commonly found in literature. These tropes provide a framework for the story and often resonate with readers due to their familiarity and emotional impact, e.g. coming-of-age, forbidden love, found family, characters don't initially like each other, starting from a rough spot, etc.
(8) Target audiences, e.g. "*children's literature.*". Descriptors based on the target audience of the book, e.g. family time, bedtime stories for adults.
(9) Objective-based, e.g. "*to learn Japanese.*". descriptors related to a specific activity or objective, e.g. meditation, sleep stories, etc.
(10) Named entities, e.g. "*Britney Spears.*". Non-fictional entities related to the book.

Based on this taxonomy we use a prompt approach to generate queries in a chain-of-though manner. Before asking the LLM to generate queries, we first ask it to generate descriptors under the 10 categories of this taxonomy and then to use such descriptors to generate two types of synthetic queries:

(11) Queries, e.g. "*realistic fiction audiobooks*".
(12) Compound queries, e.g. "*Stephen King supernatural fiction audiobooks*".

Queries use broader descriptors and combinations from the previous types with the "*audiobook*" suffix, whereas *Compound queries* leverage a combination of narrow attributes (author names) and broader descriptors. Besides instructions on how to generate descriptors for each of the 12 types described above, we also use two in-context learning examples in the prompt.

The metadata used for each audiobook as input to the model is the audiobook title, audiobook author(s), audiobook description,

[1] https://www.reddit.com/r/booksuggestions/

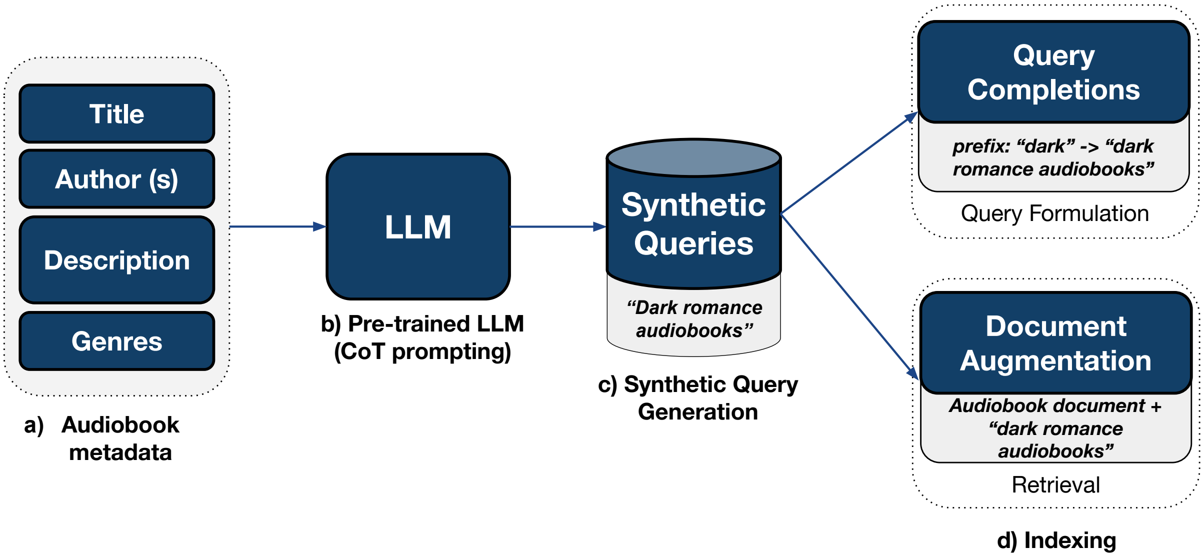

Title
Author (s)
Description
Genres
a) Audiobook metadata
LLM
b) Pre-trained LLM (CoT prompting)
Synthetic Queries
"Dark romance audiobooks"
c) Synthetic Query Generation
Query Completions
prefix: "dark" -> "dark romance audiobooks"
Query Formulation
Document Augmentation
Audiobook document + "dark romance audiobooks"
Retrieval
d) Indexing

Figure 2: AudioBoost pipeline a) We collect metadata about a given audiobook b) we use a pre-trained LLM with chain-of-thought prompting to generate audiobook descriptors and synthetic queries c) we store the generated synthetic queries in a table d) we index synthetic queries as query completion and concatenate them to the document metadata in the retrieval system before indexing (document augmentation).

and BISAC genres[2]. For each audiobook in the catalog, we use an LLM to generate the 12 types of descriptors.

2.2 Query Completions

To inspire users to type more exploratory queries for audiobooks and support them at the query formulation stage, we index the synthetic queries in the Query AutoComplete system. The Query AutoComplete (QAC) system presents query completions as the user types a query, matching the current prefix. For instance, if a user is typing a prefix "*audiob*" we want to be able to suggest different exploration ideas such as "*audiobooks for children*", "*audiobooks about love*", "*audiobooks in french*". The QAC system has multiple sources of query completions, including item titles, complete queries from the search logs, and synthetic queries. For a given query prefix p, each source j produces a set of K candidate query completions q_{jk} with $k = 1, .., K$. The set of K candidate query completions per source is determined by a predefined prefix matching score $v = v(p, q_{jk})$ and a global score $s = s(q_{jk})$ that models the inherent likelihood of the query completion q_{jk}. Then, all candidate query completions are combined and a re-ranker model is trained to select the top-N completions that are most relevant for the user prefix based on a number of features such as prefix matching, popularity, and user preferences [3].

We create a new source of query completions with synthetic queries for audiobooks. To define the global score s we cannot rely on historical data for the popularity of the query, as most synthetic queries are not well represented in the search logs. Hence, we rely on the popularity of the audiobooks associated with the queries instead, compounded with a score that measures the broadness of the query, since we aim to prioritize broad queries that favor the catalog exploration. More specifically, given a query q such as "*ancient history audiobooks*" and a list of audiobooks $A = \{a_i\}$ from which q was generated, we define:

$$s = median\_popularity(q) * broadness(q) \qquad (1)$$

where $broadness(q) = log(|A| + 1)$. Broadness is higher for broad synthetic queries that are associated with many distinct audiobooks such as "*ancient history audiobooks*" and lower for highly specific queries such as "*jrr tolkien first audiobooks*".

2.3 Document Augmentation

To improve the number of queries that retrieve audiobooks for such synthetic broad queries we rely here on document expansion for sparse retrieval systems. Similar to *doc2query* [5, 6], besides indexing the existing metadata of each audiobook, we also index the synthetic descriptors generated for them (types 1–12 from Sec 2.1):

- document = "title - author - description - genres"
- augmented_document = "title - author - description - genres - descriptors - synthetic queries"

The sparse retrieval system is based on BM25 [10], a keyword-matching system that models documents as a bag of words and assigns weights based on word frequencies. In this context, doing document augmentation with the synthetic queries has two potential effects: (1) modifying the weight of words that already exist in the item metadata by repeating them and (2) adding new words that were not covered by the audiobook metadata.

[2] https://www.bisg.org/BISAC-Subject-Codes-main

3 Offline Results

In this section, we describe the results of our offline experimentation where we tested the increase in retrievability due to the synthetic query generation in different conditions, the quality of synthetic queries as evaluated with an *LLM-as-a-judge* approach.

3.1 Retrievability Simulation

Dataset. For this experiment, we used a sample of data containing search successes from user logs. Specifically, we took a sample of query and entity pairs from a single day for three entity types at the platform: audiobooks, playlists, and podcast shows, where the user country is *US*, the query is a reformulation (not the initial query issued by the user in our instant search system) and with more than 5 characters. The resulting distribution of distinct queries to entity types is as follows: 12.77% for audiobooks, 43.47% for playlists, and 43.76% for podcasts.

Evaluation methodology. We employ a BM25 model (we use Pyterrier [4] implementation with default hyperparameters), indexing each entity with their textual metadata (title, description, and genres when available). We have four different configurations in this experiment. Configuration 1 is the baseline, where synthetic queries are not used. Configurations 2 and 3 are partial solutions that do only one part of the proposed solution at a time (document expansion with synthetic queries in configuration 2 and synthetic query suggestion in configuration 3). Configuration 4 uses the synthetic queries in both query suggestion and document expansion.

Evaluation metric. The retrievability of an entity e, as defined by [1]: $r(\mathbf{e}) = \sum_{\mathbf{q}\in\mathbf{Q}} o_q \cdot f\left(k_{eq}, c\right)$, where $\mathbf{Q}$ is the set of queries, o_q is the weight of each query—here we use 1 for all queries—and $f\left(k_{eq}, c\right)$ is 1 if the entity e is ranked above c by the search system (in our experiments we set $c = 100$) and 0 otherwise. For configurations 1 and 2, the set of queries is a sample of 20k queries from the logs. Adding a sample of the synthetic queries (also 20k queries) for the query set $\mathbf{Q}$ used to calculate the retrievability (configurations 3 and 4) in this simulation assumes that such synthetic query suggestions made by the system would be clicked by at least one user. The **retrievability share** of an entity type is the sum of the retrievability for entities of that type. The retrievability percentage share of each entity type is calculated by dividing the sum of the retrievability of the entities of that type by the sum of retrievability for all entities.

Results. Table 1 shows the results of this simulation, for each configuration considering that the entire set of synthetic queries suggested would receive clicks. In configuration 2, some queries from the logs that would retrieve playlists or shows now return more audiobooks. For example, the following queries from the logs "*audiobooks*", "*growth*" and "*christian*" retrieve more audiobooks when compared to configuration 1, due to synthetic queries that were added to the representation of audiobooks (i.e. document expansion) such as "*Self-Help Audiobooks for Spiritual Growth*" and "*Christian Audiobooks*".

In configuration 3, we change the query distribution to have more audiobook-focused queries in the system (remember that the original distribution of the dataset is skewed towards queries for playlists and podcast shows), and thus we can increase the number of audiobooks retrieved, as some of the synthetic queries will match with the title, description, and genre metadata already available.

Table 1: Results for the retrievability simulation when all suggested synthetic queries are clicked.

Retrievability percentage share		
↓Query set \ Retrieval →	BM25	BM25 + doc augmentation
Queries from logs	**Configuration 1** *audiobook*: 23.36 % *playlist*: 17.84 % *podcast*: 58.78 %	**Configuration 2** *audiobook*: 26.20 % *playlist*: 16.67 % *podcast*: 57.11 %
Queries from logs + query completions	**Configuration 3** *audiobook*: 34.49 % *playlist*: 9.72 % *podcast*: 55.78 %	**Configuration 4** *audiobook*: **56.97** % *playlist*: 8.13 % *podcast*: 34.89 %

The most effective approach though is a combination of both helping users to issue more broad audiobook queries and also modifying the retrieval system so that such queries lead to audiobooks (configuration 4).

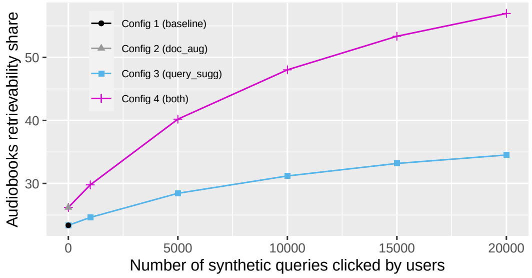


Figure 3: Offline simulation showing the impact on retrievability share of audiobooks when increasing the number of clicks towards suggested synthetic query completions.

We can see in Figure 3 that the increase in retrievability share for audiobooks increases as long as more users click on the suggested queries when using the proposed approach (configuration 4).

3.2 LLM evaluation

We evaluate the quality of the synthetic queries using an *LLM-as-a-judge* approach [9, 12], where a powerful LLM is used as an evaluator of another LLM's output. This approach has been shown to correlate with human evaluation both in public benchmarks [9, 12] and in internal assessments on similar tasks.

We instruct the evaluator LLM_{eval} using a few-shot prompting approach to judge synthetic queries on several dimensions:

- *quality*: queries need to be complete, well-formatted, without misspellings
- *relevancy*: queries need to be relevant to the audiobook metadata that was provided as input
- *diversity*: queries need to be not redundant (at the audiobook level).

- *broadness*: queries need to refer to general topics, and genres, rather than to specific audiobooks

All metrics are boolean and evaluated at the query level, except for *diversity* which is a single value for the group of queries associated with an audiobook. We obtain high scores on all dimensions: $quality = 99.7\%$, $relevancy = 97.2\%$, $diversity = 81.5\%$, $broadness = 84.2\%$.

4 Online Results

We run a large-scale A/B test for 3 weeks comparing the default QAC and retrieval system to a treatment that uses AudioBoost, namely where we 1) add the synthetic queries as an additional source of query completions 2) use the synthetic queries to perform document augmentation in a sparse retrieval system (Fig. 2).

We measure several metrics online to account for audiobook retrievability and exploratory searches for audiobooks at the SERP (Search Engine Results Page) level:

- *impressions*: number of audiobook impressions per SERP
- *clicks*: number of clicks on audiobooks per SERP
- *coverage*: overall number of query completions shown per SERP
- *exploration*: number of clicks on exploratory[3] query completions leading to audiobook interactions per SERP

We observe that AudioBoost leads to +0.7% in impressions, +1.22% in clicks, +0.03% in coverage, and +1.82% in exploration. At the same time, guardrail metrics that check for the overall engagement with QAC and overall search effectiveness are neutral. All reported results are statistically significant with a t-test with a p-value of 1%.

5 Conclusions

In this work, we have introduced AudioBoost, a system to increase audiobook retrievability in Spotify search via synthetic query generation. Building up on previous research work on query generation, we have shown that synthetic query generation is a viable strategy to support exploratory search in a cold-start scenario where new item types have been introduced in the catalog in a production setting. We have used LLMs to generate synthetic queries for audiobooks and we have indexed them at the query formulation stage, inspiring users to type more exploratory queries for audiobooks, and at the retrieval stage, fulfilling such queries with relevant search results. Offline results have confirmed the validity of the proposed solution, in terms of retrievability, coverage increases and query quality. Online results show its effectiveness at scale in a real production system with millions of users.

AudioBoost increases impressions and clicks on audiobooks, inspiring users to explore the catalog and helping authors and publishers gain visibility and engage with new listeners. More in general, this work highlights that synthetic query generation is a powerful strategy to increase the visibility of a set of target items in a search system. AudioBoost is also appealing from the productionalization point of view, as the query generation and indexing steps can be performed offline in a batch pipeline, with reasonable cost and without affecting the latency of the QAC and retrieval systems. Given these results, we have rolled out AudioBoost in production.

It is important to notice that, while this work focuses on the retrieval stage, many of these techniques could be applied similarly to the re-ranking stage of a typical search engine system, and in the future, we plan to investigate this direction further.

We also aim to further explore the intersection of synthetic query generation with other approaches that have shown to work well to address the cold-start problem, such as explore-exploit strategies and/or content-based recommendations.

[3] The classification of exploratory queries is associated to the broadness of the query, i.e. number of distinct items they are associated with, and other factors.

References

[1] Leif Azzopardi and Vishwa Vinay. 2008. Retrievability: An evaluation measure for higher order information access tasks. In *Proceedings of the 17th ACM conference on Information and knowledge management*. 561–570.

[2] Ang Li, Jennifer Thom, Praveen Chandar, Christine Hosey, Brian St. Thomas, and Jean Garcia-Gathright. 2019. Search Mindsets: Understanding Focused and Non-Focused Information Seeking in Music Search. In *The World Wide Web Conference* (San Francisco, CA, USA) *(WWW '19)*. Association for Computing Machinery, New York, NY, USA, 2971–2977. https://doi.org/10.1145/3308558.3313627

[3] Henrik Lindstrom, Humberto Jesus Corona Pampin, Enrico Palumbo, and Alva Liu. 2024. Encouraging Exploration in Spotify Search through Query Recommendations. In *Proceedings of the 18th ACM Conference on Recommender Systems* (Bari, Italy) *(RecSys '24)*. Association for Computing Machinery, New York, NY, USA, 775–777. https://doi.org/10.1145/3640457.3688035

[4] Craig Macdonald, Nicola Tonellotto, Sean MacAvaney, and Iadh Ounis. 2021. PyTerrier: Declarative experimentation in Python from BM25 to dense retrieval. In *Proceedings of the 30th acm international conference on information & knowledge management*. 4526–4533.

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

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

[7] Enrico Palumbo, Andreas Damianou, Alice Wang, Alva Liu, Ghazal Fazelnia, Francesco Fabbri, Rui Ferreira, Fabrizio Silvestri, Hugues Bouchard, Claudia Hauff, Mounia Lalmas, Ben Carterette, Praveen Chandar, and David Nyhan. 2023. Graph Learning for Exploratory Query Suggestions in an Instant Search System. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management* (Birmingham, United Kingdom) *(CIKM '23)*. Association for Computing Machinery, New York, NY, USA, 4780–4786. https://doi.org/10.1145/3583780.3615481

[8] Gustavo Penha, Enrico Palumbo, Maryam Aziz, Alice Wang, and Hugues Bouchard. 2023. Improving Content Retrievability in Search with Controllable Query Generation. In *Proceedings of the ACM Web Conference 2023* (Austin, TX, USA) *(WWW '23)*. Association for Computing Machinery, New York, NY, USA, 3182–3192. https://doi.org/10.1145/3543507.3583261

[9] Hossein A Rahmani, Emine Yilmaz, Nick Craswell, Bhaskar Mitra, Paul Thomas, Charles LA Clarke, Mohammad Aliannejadi, Clemencia Siro, and Guglielmo Faggioli. 2024. LLMJudge: LLMs for Relevance Judgments. *arXiv preprint arXiv:2408.08896* (2024).

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

[11] Federico Tomasi, Rishabh Mehrotra, Aasish Pappu, Judith Bütepage, Brian Brost, Hugo Galvão, and Mounia Lalmas. 2020. Query Understanding for Surfacing Under-served Music Content. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management* (Virtual Event, Ireland) *(CIKM '20)*. Association for Computing Machinery, New York, NY, USA, 2765–2772. https://doi.org/10.1145/3340531.3412741

[12] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems* 36 (2023), 46595–46623.