Decomposing Complex Queries for Tip-of-the-tongue Retrieval

Kevin Lin♠ Kyle Lo♡ Joseph E. Gonzalez ♠ Dan Klein♠
♠University of California Berkeley ♡ Allen Institute for AI
(k-lin,jegonzal,klein)@berkeley.edu kylel@allenai.org

Abstract

When re-finding items, users who forget or are uncertain about identifying details often rely on creative strategies for expressing their information needs—complex queries that describe content elements (e.g., book characters or events), information beyond the document text (e.g., descriptions of book covers), or personal context (e.g., when they read a book). Standard retrieval models that rely on lexical or semantic overlap between query and document text are challenged in such retrieval settings, known as tip of the tongue (TOT) retrieval. We introduce a simple but effective framework for handling such complex queries by decomposing the query with an LLM into individual clues, routing those as subqueries to specialized retrievers, and ensembling the results. Our approach takes advantage of off-the-shelf retrievers (e.g., CLIP for retrieving images of book covers) or incorporate retriever-specific logic (e.g., date constraints). We show that our framework incorporating query decomposition into retrievers can improve gold book recall up to 6% absolute gain for Recall@5 on WhatsThatBook, a new collection of 14,441 real-world query-book pairs from an online community for resolving TOT inquiries.

1 Introduction

Tip of the tongue (TOT) refers to the retrieval setting in which a user is unable to formulate a precise query that identifies a sought item, even if the user knows they’ve encountered this item before. For example, users searching for movies they watched or books they read long ago often resort to complex and creative queries that employ a diverse set of strategies to express information relevant to the sought item—high-level categories (e.g., topic, genre), content details from the movie or book (e.g., events, characters), references to personal context (e.g., when they last read the book), descriptions of extratextual elements (e.g., movie promotional posters, book covers), and more. In fact, in an annotation study of TOT queries for movies, Arguello et al. (2021) found over 30 types of informational facets that users may include when crafting queries. Figure 1 shows a TOT query and its corresponding gold book.

Figure 1: To resolve tip-of-the-tongue queries about books, our approach decomposes long, complex queries into subqueries routed to specific retrieval experts, each trained to handle a particular aspect of a book.

A key challenge in TOT retrieval is that queries are not just longer and more complex than those in popular retrieval datasets, but resolving them requires an enriched document collection since
query-document relevance can’t necessarily be established from document content alone (see Table 1). For example, in Figure 1, the query’s description of the book cover—The cover of this book was yellow with a silhouette of a cat—can be highly useful for identifying the book, but necessitates the book’s representation to contain that information.

In this work, we present a simple yet effective technique for improving TOT retrieval: First, we decompose queries into individual subqueries or clues that each capture a single aspect of the target document. Then, we route these subqueries to expert retrievers that can be trained individually. Finally, we combine their results with those from a base retriever that receives the original query. Experiments show improvement in gold book recall over description-only retrieval baselines on a set of 14,441 real-world query-book pairs collected from an online forum for resolving TOT inquiries, complete with cover images and metadata.

2 Decomposing Queries

In this section, we describe our a simple but effective method for tackling long, complex TOMT queries. Given a collection of documents \(d_1, \ldots, d_n\) and a textual query \(q\), the TOT retrieval task aims to identify the sought document \(d^*\). The input (raw) documents are semi-structured; each document \(d\) contains metadata fields \(d^{(1)}, \ldots, d^{(k)}\). In our the WhatsThatBook dataset, the documents are books that have the fields that correspond to plot, its publication year, an image of its book cover, title, genre, and author etc. Missing elements take on a default value (e.g., blank image, earliest publish date in overall book collection).

2.1 Method

First, the query decomposer takes a query \(q\) and outputs a set of subqueries \(q^{(1)}, \ldots, q^{(k)}\), for \(k\) metadata fields. To do this, we use in-context learning with a large language model (LLM) to generate a subquery \(q\) that is relevant to that field or optionally output the string "N/A" if the \(q\) does not contain any relevant information to the field; this is repeated for each field and can be run in independently in parallel for each field.

We experiment with two prompting strategies, extractive and predictive. Extractive prompting aims to generate subqueries that are purely extractions from the original query. The LLM is instructed to output "N/A" if there is not information relevant to the metadata field.

Predictive prompting aims to generate subqueries that are similar to the content of the metadata field. Note that the subqueries need not be extractions from the original query and can be inferred metadata or even hallucinations from the query decomposer. Using LLMs to generate subqueries affords us the ability to set the few-shot prompt generation targets to be predictions. This is important as the information in queries are rarely presented in a form amenable for matching with the corresponding document field. For example, books have publish dates, but queries will rarely mention these dates; instead, users may articulate personal context (e.g., “I read this book in high-school around 2002-2005”). Then to simplify the learning task for a date-focused retrieval expert, we might ask the LLM to predict a “latest possible publish date” (e.g., 2005).

See Table 2 for examples of extractive and predictive subqueries and Appendix A for examples of their prompts. In practice, we use GPT 3.5 (gpt-3.5-turbo) few-shot prompting with up to 8 in-context examples. Each field has its own prompt template and set of examples. Subqueries for different fields can be generated in parallel, as they are independent of each other.

2.2 Retrieval Experts

We have retriever models, or experts, that specialize to specific field types. Let \(R_1, \ldots, R_k\) represent these retrievers. Retrievers can be implemented as dense, sparse, or symbolic logic.

If a retriever requires training, we run the query decomposer over all query-document pairs \((q, d)\) in the training set. This produces effectively \(k\) training datasets, where each dataset is comprised of a subquery and document-field pair. For example, field \(j\) would have training dataset of examples \((q^{(j)}, d^{(j)})\).

At indexing time, each document’s field is indexed according to the specifications of its retriever expert. For example, if the retriever is implemented as an embedding model, then that specific field is converted into an embedding. On the other hand, if the retriever is a sparse model, then a sparse index would be built using just that specific field’s text.

At inference time, each retriever takes a subquery \(q^{(j)}\) and retrieves documents from its associated index of fields.

5522
DatasetQuery LengthLexical Overlap
MSMarco (Campos et al., 2016) 7.68 0.55
Natural Questions (Kwiatkowski et al., 2019) 10.35 0.52
BioASQ (Tsatsaronis et al., 2015) 14.82 0.58
TREC-COVID (Roberts et al., 2020) 15.94 0.41
SciFact (Wadden et al., 2022) 19.52 0.50
HotPotQA (Yang et al., 2018b) 22.78 0.45
TOMT (Bhargav et al., 2022) 136.50 0.25
WhatsThatBook 156.20 0.19

Table 1: Tip of the tongue (TOT) queries are significantly longer while also having less lexical overlap with the gold document, compared with queries in popular retrieval datasets. Query length is number of BPE (Sennrich et al., 2016) pieces, averaged across examples. Lexical overlap is fraction of whole words in query that occur in gold passage(s), averaged across examples.

2.3 Implementation details

In practice, for titles, authors, plot, and genre, we use Contriever (Izacard et al., 2021), a state-of-the-art dense retriever. For both models, we train for a total of 10,000 steps with a batch size of 16, learning rate of 1e-4. For titles, we finetune with 3,327 extracted subqueries. For our base retriever, we use the full training set of original book descriptions. For embedding search during inference, we use the faiss library and project all embeddings to 768 Johnson (2022).

For cover images, we use CLIP (Radford et al., 2021), a state-of-the-art image-text model that can be used for image retrieval by scoring matches between embedded images and their textual descriptions. Specifically, we finetune ViT-B/32 on 2,220 extracted subqueries using cross-entropy loss with batch size of 4, learning rate of 5e-5 and weight decay of 0.2 for 10 epochs with the Adam optimizer (Kingma and Ba, 2014). We select the model with the best top 1 retrieval accuracy on a validation set.

For publish dates using the predictive prompting, we use a symbolic function that heuristically scores 0 if a book was published after the subquery date (i.e. predicted latest publish date) and 1 otherwise. If necessary, we heuristically resolve the subquery to a year.

2.4 Combining retrieved results

In this work, we restrict to a simple strategy of using a weighted sum of all $k$ retrieval scores across the $(q^{(j)}, d^{(j)})$. That is, the final score is:

$$s(q, d) = \sum_{j=1}^{n} w_j R_j(q^{(j)}, d^{(j)})$$

All documents are scored in this manner, which induces a document ranking for a given query $q$. We tune the weights $w_j$ on the validation set and select the weights that have the best Recall@5.

3 Datasets

WhatsThatBook We introduce the WhatsThatBook dataset consisting of query-book pairs collected from a public online forum on GoodReads for resolving TOT inquiries about books. On this forum, users post inquiries describing their sought book and community members reply with links to books on GoodReads as proposed answers. If the searcher accepts a book as the correct answer, the post is manually tagged as SOLVED and a link to the found book is pinned to the thread. For these solved threads, we take the original inquiry as our query $q$ and the linked book as gold $d^*$. At the end,WhatsThatBook contains 14,441 query-book pairs. Each query corresponds to a unique book. Finally, these books are associated with pages on GoodReads, which we used to obtain publication year metadata and images of book covers. We further collect 172,422 negative books (books that do not correspond to a gold query in our dataset) to make the setting more realistic, for a total of 186,863 books. To collect negative books, crawl books authored by crawling books by the authors of the positive books.

---

2https://huggingface.co/facebook/contriever
3https://huggingface.co/sentence-transformers/clip-ViT-B-32

5This is a simplification of community interactions. Threads also may include dialogue between original poster and members but this is beyond the scope of our work.
<table>
<thead>
<tr>
<th>Query</th>
</tr>
</thead>
<tbody>
<tr>
<td>I think I saw this in a used store once and I remember saying to my new husband “my daughter use to read that book to her little brother” and it’s funny because on the outside cover is a little girl reading a book to her little brother. It’s called....my book, or my story, or something simple like that. It would be about 15 or more years old. The girl was blond and the boy brunet....I think!!!! Inside was the cutest little sentences and my kids use to do what each page said...thank you!!</td>
</tr>
</tbody>
</table>

**Extracted Title:** It’s called....my book, or my story, or something simple like that.

**Extracted Date:** 15 years old (2006 or earlier)

**Extracted Cover:** The outside cover is a little girl reading a book to her little brother.

**Actual Cover:**

---

<table>
<thead>
<tr>
<th>Query</th>
</tr>
</thead>
<tbody>
<tr>
<td>I read a book in 2008 or 2009 that was part of a series for young adults. It was fantasy, and about several families of witches and warlocks. The main character was named Holly.</td>
</tr>
</tbody>
</table>

**Extracted Genre:** for young adults, fantasy

**Predicted Plot:** Holly is a young witch who comes from a long line of magical families...

**Predicted Cover:** Young woman with long, curly hair holding a wand and surrounded by swirling colors and symbols

**Actual Cover:**

---

<table>
<thead>
<tr>
<th>Query</th>
</tr>
</thead>
<tbody>
<tr>
<td>Written in the 1990s, by [probably] one of those romance novelists (not Nora Roberts though). The cover was just a snowed-in cabin on a mountain side at night. In a nut shell – An escaped prisoner takes a woman captive and steals her car with her in it!!...Any ideas what this might be called? I haven’t forgotten it after all these years.</td>
</tr>
</tbody>
</table>

**Extracted Author:** by [probably] one of those romance novelists (not Nora Roberts though)

**Predicted Cover:** Dark, mysterious cabin obscured by falling snow with the silhouette...

**Extracted Cover:** The cover was just a snowed-in cabin on a mountain side at night.

**Actual Cover:**

---

<table>
<thead>
<tr>
<th>Query</th>
</tr>
</thead>
<tbody>
<tr>
<td>A rootless foster child, Julie Mathison had blossomed under the love showered upon her by her adoptive family. Now a lovely and vivacious young woman...</td>
</tr>
</tbody>
</table>

**Actual Author:** Judith McNaught

**Actual Cover:** (see right)

Table 2: Query-document pairs, their generated sub-queries or clues, and corresponding gold document fields. Clues can be extracted directly from the query or predicted as a best-guess attempt to match the actual document field. Clues can be about the book’s title, author, date, cover, genre, or general plot elements.
Table 3: Number of queries organized by results from running query decomposition with extractive prompting. Rows correspond to number of clues Extracted (between one and six). For instance, top row is queries with only a single extracted clue and bottom row is queries with all clue types found. Num Posts counts number of queries in each bucket. Cell counts correspond to number of times a given clue type was extracted in a post, and percentages normalize against Num Posts. For example, 61.9% of queries with two extracted clues have one “Date” subquery.

Table 4: Results on test set of WhatsThatBook. (Top) Baselines operate directly over queries and descriptions of the books. (Middle) We use the top performing model, Contriever, on the concatenated representations of subqueries (E-extractive, P-predictive, Q-rewritten queries) or the hypothetical document (H). (Bottom) Subqueries routed to individual retriever experts.

4 Experiments

4.1 Baseline models

Models that use original queries. We evaluate our approach against several popular retrieval models that have been used as baselines for a range of other retrieval datasets (see Table 1). For text-only models—BM25 (Robertson and Walker, 1997; Robertson and Zaragoza, 2009), Dense Passage...
Table 5: Results on test set of Reddit-TOMT (Books).

We report the best performing baseline (Contriever) using the original queries and the best performing model using our approach that uses the predicted subqueries (P).

<table>
<thead>
<tr>
<th>Model</th>
<th>Top 5</th>
<th>Top 10</th>
<th>Top 20</th>
<th>Top 100</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contriever</td>
<td>42.8</td>
<td>51.1</td>
<td>58.6</td>
<td>77.7</td>
</tr>
<tr>
<td>Ours (P)</td>
<td>44.2</td>
<td>53.6</td>
<td>61.3</td>
<td>79.9</td>
</tr>
</tbody>
</table>

Retrieval (DPR) (Karpukhin et al., 2020), Contriever (Izacard et al., 2021), and ColBERT (Khattab and Zaharia, 2020)—the document representation is simply the concatenation of all available document fields into a single text field. For our image-only baseline—CLIP (Radford et al., 2021)—the document representation is only the embedded book cover. All baselines receive the same input (full) query, and are finetuned with the same hyperparameters (§2.2) except using the full training set (as opposed to just examples with a successful subquery extraction).

Models that use generated queries. To evaluate the effect of generating subqueries independently without training specialized retrievers, we also train the top performing text-only model from the set of models with queries enriched with LLMs. We experiment with using the concatenation of all subqueries generated by the query decomposer as the query representation for both the exhaustive and predicted subqueries. To isolate the effect of generating separate subqueries, we also use LLM to generate a single rewritten query that is used as input to Contriever, by simply prompting LLMs to generate a single new query that better retrieves the desired item (prompt in Appendix A). In addition, we generate hypothetical documents and use these as the query, similar to Gao and Callan (2022) though we differ in that we train on these generations as queries while Gao and Callan (2022) restricted their use to the zero-shot setting.

4.2 Results

Table 4 shows the test results on WhatsThatBook. We use Recall@K metric as our primary metric since each query has exactly one correct item.

**Contriever is the best-performing baseline model.** In this setting with low lexical overlap, we see that dense retrievers like DPR and Contriever outperform sparse retrievers like BM25. Without extracting clues about the book cover, using CLIP on its own is not effective, likely due to its limited context window.

**Query decomposition improves performance.** When using the rewritten queries without training individual expert retrievers, we observe that concatenating the predicted clues improve the model. However, both generating rewritten queries and entire hypothetical documents perform worse than just using the original document as input. Our approach to decompose queries and route clues to specialized retrievers improves performance (ranging from +6 to +7 Recall@K across all cutoffs) over the next best baseline retriever. Table 6 shows the results for each individual expert retriever on WhatsThatBook and the books subset of TOMT.

**Performance degrades predictably with corpus size.** To see how well the performance scales as more documents are added, we also report the performance of the base performing baseline and our model as we add more documents to the corpus. We evaluate the performance for different corpus sizes by varying the number of negative books added to the corpus. Figure 2 shows that for both our decomposition and the baseline, the performance drops significantly as more negative books are added. There is a sharp decline of performance when negative books are first introduced (between corpus size 1 and 2). Since the negative books are collected from the authors of the positive books, they may be more difficult to distinguish than a random book. As more documents are added, there is a marginal decrease in performance smoothly. Furthermore, our decomposition method performance better than the baseline for different corpus sizes and has slightly less performance degradation as more documents are added to the corpus.

**Metadata fields exhibit long-tailed behavior.** The query decomposer generates a plot subquery in at least 90% of the queries for both WhatsThatBook and TOMT. Dates occur in a large proportion of the queries, but are not specific enough to be effective identifying the book. While subqueries for the author appear very infrequently, when they do appear, they are much more effective than more generic subqueries such as dates or genres. Images
Table 6: Results of individual retrieval experts on the test set of *WhatsThatBook* and *Reddit-TOMT (Books)*. E and P indicate extractive and predictive prompts. Extractive prompts scores are calculated only over the "N/A" queries.

<table>
<thead>
<tr>
<th>Subquery</th>
<th>Top 5</th>
<th>Top 10</th>
<th>Top 20</th>
<th>Top 100</th>
<th>% not N/A</th>
<th>Top 5</th>
<th>Top 10</th>
<th>Top 20</th>
<th>Top 100</th>
<th>% not N/A</th>
</tr>
</thead>
<tbody>
<tr>
<td>plot (E)</td>
<td>29.0</td>
<td>36.4</td>
<td>44.9</td>
<td>64.2</td>
<td>90.4</td>
<td>45.5</td>
<td>56.2</td>
<td>61.0</td>
<td>76.0</td>
<td>94.3</td>
</tr>
<tr>
<td>plot (P)</td>
<td>27.9</td>
<td>34.7</td>
<td>42.4</td>
<td>61.7</td>
<td>100</td>
<td>43.2</td>
<td>50.5</td>
<td>58.8</td>
<td>77.1</td>
<td>100</td>
</tr>
<tr>
<td>dates (E)</td>
<td>1.3</td>
<td>2.1</td>
<td>3.4</td>
<td>6.0</td>
<td>54.8</td>
<td>0.0</td>
<td>1.0</td>
<td>2.2</td>
<td>2.8</td>
<td>59.8</td>
</tr>
<tr>
<td>dates (P)</td>
<td>0</td>
<td>0</td>
<td>1.1</td>
<td>2.3</td>
<td>100</td>
<td>0.0</td>
<td>1.8</td>
<td>2.3</td>
<td>4.4</td>
<td>100</td>
</tr>
<tr>
<td>cover (E)</td>
<td>5.3</td>
<td>6.7</td>
<td>7.7</td>
<td>14.1</td>
<td>21</td>
<td>8.6</td>
<td>11.6</td>
<td>15.5</td>
<td>17.7</td>
<td>35.8</td>
</tr>
<tr>
<td>cover (P)</td>
<td>3.1</td>
<td>4.4</td>
<td>5.6</td>
<td>7.2</td>
<td>100</td>
<td>5.8</td>
<td>8.4</td>
<td>10.0</td>
<td>13.4</td>
<td>100</td>
</tr>
<tr>
<td>title (E)</td>
<td>11.7</td>
<td>12.9</td>
<td>14.6</td>
<td>19.8</td>
<td>29.0</td>
<td>13.4</td>
<td>17.5</td>
<td>22.2</td>
<td>37.4</td>
<td>43.9</td>
</tr>
<tr>
<td>title (P)</td>
<td>4.3</td>
<td>5.7</td>
<td>8.2</td>
<td>13.4</td>
<td>100</td>
<td>14.3</td>
<td>17.2</td>
<td>21.0</td>
<td>35.5</td>
<td>100</td>
</tr>
<tr>
<td>genre (E)</td>
<td>2.7</td>
<td>3.4</td>
<td>5.9</td>
<td>17.4</td>
<td>28.2</td>
<td>1.3</td>
<td>4.4</td>
<td>5.2</td>
<td>28.8</td>
<td>26.8</td>
</tr>
<tr>
<td>genre (P)</td>
<td>3.8</td>
<td>6.8</td>
<td>10.9</td>
<td>24.4</td>
<td>100</td>
<td>6.7</td>
<td>8.6</td>
<td>12.3</td>
<td>25.5</td>
<td>100</td>
</tr>
<tr>
<td>author (E)</td>
<td>6.0</td>
<td>8.0</td>
<td>8.0</td>
<td>9.1</td>
<td>6.9</td>
<td>7.1</td>
<td>8.2</td>
<td>9.6</td>
<td>9.6</td>
<td>9.9</td>
</tr>
<tr>
<td>author (P)</td>
<td>0</td>
<td>1</td>
<td>1.1</td>
<td>3.2</td>
<td>100</td>
<td>1.9</td>
<td>3.0</td>
<td>4.7</td>
<td>14.5</td>
<td>100</td>
</tr>
</tbody>
</table>

Figure 2: Recall at k of the top performing baseline (Contriever) and our decomposition baseline for vary amounts of negatives documents added to the corpus.

are much more effective when using decomposition compared to the image only CLIP baseline, as the image retriever model is able to access the visual part of the text.

**Predictive prompting performs better than extractive prompting.** Overall, we find that predicting subqueries is more effective than extracting. Subqueries are generated for every query, thus, there is more data for the specialized retriever experts to train on, compared to extractive clues where some retrievers only have a small fraction of the dataset for finetuning. Moreover, the predictive clues allows the LLM to make inferences based on information that is not explicitly present in the query. For example, the LLM is able to make inferences about the genre effectively based on the full query even when the user does not explicitly discuss the genre. Another benefit of the extractive clue is that the subqueries are more grounded in the original query.

**Trade-off exists between generating subquery and individual retriever expert performance.** Between the prompting strategies, we find that there is often a trade-off between how selective the query decomposer is with generating a subquery for a metadata field, and the effectiveness when generated. For most of the metadata retrieval queries, the extractive prompting approach is slightly more effective than predictive prompting on the examples that it does not predict "N/A" on.

5 Error Analysis

We sample 50 predictions where the top performing model where the model fails to get the book correct in the top 5 and categorize the types of errors made by the model in Table 7. The most common kind of error is failure to model the relationship between the document and query, which happens in instances where there may be events that dense models fail to capture indicating that there is still headroom on the task. Moreover, documents are sometimes brief or written in a style that is not similar to other descriptions. Lastly, because users are recalling memories, some of these can be false memories that lead to errors in retrieval.
I think this was a teen book. Don’t remember Author or character names. All I remember is that the girl loses her memory, it is not "Memoirs of a teenage amnesiac". It was fiction book and it wasn’t comic or manga. I think the girl was involved in a car accident where someone hit her while she was walking? Apparently she was quite wild and broke instruments and was quite hated before she had the accident or something like that for things she done "to people, but she can’t remember any of it. ...

Title: Kat Got Your Tongue
Plot: After a terrible car accident, Kat wakes up with no idea who she is, and no memory of anything before the crash. She doesn’t even recognize her mum, much less her friends from school—Poppy, Jade, and the mysterious Tina. Only after she finds her old diary—written in a voice no longer her own—does Kat begin to discover the terrible secrets of her previous life. This incredib

Title: The Daughters of Ardmore Hall
Plot: An unbalanced woman seeks to destroy her daughters.

Title: Moon Tiger
Plot: The elderly Claudia Hampton, a best-selling author of popular history; lies alone in a London hospital bed. Memories of her life still glow in her fading consciousness, but she imagines writing a history...

Table 7: Error types from the top performing model. Book representation are simplified here without all metadata for space constraints.

6 Related Work

Dense methods for document retrieval. Document retrieval has a long history of study in fields like machine learning, information retrieval, natural language processing, library and information sciences, and others. Recent years have seen the rise in adoption of dense, neural network-based methods, such as DPR (Karpukhin et al., 2020) and Contriever (Izacard et al., 2022), which have been shown can outperform sparse methods like BM25 (Robertson and Walker, 1997; Robertson and Zaragoza, 2009) in retrieval settings in which query-document relevance cannot solely be determined by lexical overlap. Researchers have studied these models using large datasets of query-document pairs in web, Wikipedia, and scientific literature-scale retrieval settings (Campos et al., 2016; Sciavolino et al., 2021; Roberts et al., 2020). Many retrieval datasets have adopted particular task formats such as question answering (Kwiatkowski et al., 2019; Yang et al., 2018b; Tsatsaronis et al., 2015) or claim verification (Thorne et al., 2018; Wadden et al., 2022). We direct readers to Zhao et al. (2022) for a comprehensive, up-to-date survey of methods, tasks, and datasets.

Known-item and TOT retrieval. Tip of the tongue (TOT) is a form of known-item retrieval (Buckland, 1979; Lee et al., 2006), a long-studied area in the library and information sciences. Yet, lack of large-scale public datasets has made development of retrieval methods for this task difficult. Prior work on known-item retrieval focused on constructing synthetic datasets (Azzopardi et al., 2007; Kim and Croft, 2009; Elsweiler et al., 2011). For example, Hagen et al. (2015) released a dataset of 2,755 query-item pairs from Yahoo! answers and injected query inaccuracies via hired annotators to simulate the phenomenon of false memories Hauff and Houben (2011); Hauff et al. (2012), a common property of TOT settings.

The emergence of large, online communities for resolving TOT queries has enabled the curation of realistic datasets. Arguello et al. (2021) categorized the types of information referred to in TOT queries...
from the website I Remember This Movie.\footnote{https://irememberthismovie.com/} Most recently, Bhargav et al. (2022) collected queries from the Tip Of The Tongue community on Reddit\footnote{https://www.reddit.com/r/tipofmytongue/} and evaluated BM25 and DPR baselines. Our work expands on their work in a key way: We introduce a new method for retrieval inspired by long, complex TOT queries. In order to test our method on a large dataset of TOT queries, we collected a new dataset of resolved TOT queries such that we also had access to metadata and book cover images, which were not part of Bhargav et al. (2022)’s dataset.

Query Understanding and Decomposition. Our work on understanding complex information-seeking queries by decomposition is related to a line of work breaking down language tasks into modular subtasks (Andreas et al., 2016). More recently, LLMs have been used for decomposing complex tasks such as multi-hop questions into a sequence of simpler subtasks (Khot et al., 2022) or smaller language steps handled by simpler models (Jhamtani et al., 2023).

Related to decomposition of long, complex queries for retrieval is literature on document similarity (Mysore et al., 2022) or query-by-document (QBD) (Yang et al., 2018a). In these works, a common approach is decomposing documents into sub-passages (e.g. sentences) and performing retrieval on those textual units. The key differentiator between these works and ours is that document similarity or QBD are inherently symmetric retrieval operations, whereas our setting requires designing approaches to handle asymmetry in available information (and thus choice of modeling approach or representation) between queries and documents. In this vein, one can also draw parallels to Lewis et al. (2021), which demonstrates that retrieving over model-generated question-answering pairs instead of their originating documents can improve retrieval, likely due to improved query-document form alignment. In a way, this is similar to our use of LLMs to generate clues that better align with extratextual document fields, though our work is focused on query-side decomposition rather than document-side enrichment. More recently, Wang et al. (2023) propose using LLMs for decomposing different facets of complex queries for scientific documents.

7 Conclusion

We study tip of the tongue retrieval, a real-world information-seeking setting in which users issue long, complex queries for re-finding items despite being unable to articulate identifying details about those items. We introduce WhatsThatBook, a large challenging dataset of real-world TOT queries for books. We also introduce a simple but effective approach to handling these complex queries that decomposes them into subqueries that are routed to expert retrievers for specialized scoring. Our simple framework allows for modular composition of different retrievers and leveraging of pretrained models for specific modalities such as CLIP for document images. We experiment with different subquery generation strategies and find that generating predictions of document fields is more effective. We observe improvements up to 6% absolute gain over state-of-the-art dense retrievers for Recall@5 when incorporating query decomposition into existing retrievers.

Limitations

Given our proposed method is designed to handle TOT queries, there is an implicit assumption that the document collection covers the sought item with high probability. That is, a system tailored for TOT retrieval is unlikely to perform well if its index is missing too many books. While our dataset is large-scale, one limitation is that even a corpus of 187k documents does not cover the sought document sufficiently. Consider, for instance, the total number of books available on GoodReads (over 3.5B as of time of writing). Another limitation of our method is that the overhead for tackling another domain with this technique is non-trivial as the prompts and few-shot examples may not be directly transferable. We believe a promising avenue for future work is reducing the effort needed to bootstrap the design and training of retrieval experts and incorporating them into a query decomposer.

8 Acknowledgements

We thank the members of the Berkeley NLP group and the anonymous reviewers for their insightful feedback. This work was supported by a grant from DARPA SemaFor, and gifts from sponsors of Sky Computing and BAIR. The content does not necessarily reflect the position of the sponsors, and no official endorsement should be inferred.
References


Wayne Xin Zhao, Jing Liu, Ruiyang Ren, and Ji rong Wen. 2022. Dense text retrieval based on pretrained language models: A survey. ArXiv, abs/2211.14876.
A Prompts

Extractive Prompt

You are a utility built to take in forum queries from users looking for books and output the aspect that is about the cover. If there is not enough information, then output N/A. Do not guess and only output text that was in the query. Here are some examples:

Question: Hi there, I read this book in highschool around 2002-2005. From what I remember, the main character is nicknamed "Mouse" and she rides a big chestnut horse in jumper shows. I think this book may have been Australian. The cover just showed a chestnut horse and rider in mid jump. I think the title was one word--it may have been the name of the horse. I cannot remember the name or the author of this book. I have googled everything I can think of but I cannot to find this book and its driving me crazy! I'd be grateful for any help on this! Thank you! Cover Clue: The cover just showed a chestnut horse and rider in mid jump.

Question: So I read this book somewhere between 2008-2010 when I was in elementary school or middle school.

Question: It was about a girl who lived by the sea, and started out with her having dinner at this boarding house she lived in. And something happened (she received a fruit or something else that was banned in her town) and she had to hide it. Then, the landlady of her house got mad when she found out. The MC went down to the beach and saw this giant walrus thing (I can't remember clearly but I think there was also a guy who rode the walrus but maybe not) and the MC got on the walrus and they rode away to this magical land. I don't remember much else, except the story had something to do with fruits for some reason, and the characters mother was likely from this magical land. Also, in the end, the main character brings Color and happiness back to her seaside village and I think there's something else to do with strawberries. Please help! I've been searching for this book for years! Thanks so much <3 Cover Clue: N/A

Question: I remember reading this around 2008/2009. It was about a girl being prettier than her mother, and when the mother gets jealous she sends her daughter to a type of boarding school for people who are well-known (Like nobles and princesses. She may be royalty). The cover was purple and captivating. (which was why I picked the book to read "") with a girl's face on it. However, I'm not sure if the cover really had a face on it, or if it was completely purple. (view spoiler) I believe that at the end, a minion of the girl's mother helped the girl get rid of her mother. Maybe it was because the mother tried to kill her daughter for being beautiful, especially since she was getting older. (hide spoiler) Please help me find it. It was the first book that got me to love reading at that time. :D Cover Clue: The cover was purple and captivating, (which was why I picked the book to read "") with a girl's face on it. However, I'm not sure if the cover really had a face on it, or if it was completely purple.

Question: I probably read this book in the late 90s, early 2000s, and cannot find the title anywhere. The main character is an outcast in her high school who lives in a trailer park near a cemetery. From what I recall, she was picking flowers and trying to clear her mother's grave. She found one with a horse and rider in mid jump. (I can't remember clearly but I think there was also a face on it. However, I'm not sure if the cover really had a face on it, or if it was completely purple.) Please help me find it. It was the first book that got me to love reading at that time. :D Cover Clue: The cover was purple and captivating. (which was why I picked the book to read "") with a girl's face on it. However, I'm not sure if the cover really had a face on it, or if it was completely purple.

Question: I'm looking for a book there was two girls. the older one liked dumpster diving and had a duck shaped jar of memories that she could go into. the younger one was blonde and liked buying shirts from a thrift store and writing on them, and she couldn't remember anyone's faces. the younger one tried to mail her mom a coconut and the mailman was the only face she could remember. there was also a woman in a hospital who only said red shoes and the cover was green or blue. was read a couple years ago so probably 2017-2019ish? Cover Clue: the cover was green or blue.

Question: This book was blue (if that helps). And I think it had a mailbox on the front of it. There were three girls and they hated their rival school. So to get the two schools to be friends they made everyone send letters to another person in the school. One girl got a guy who wanted to impress a girl at his school so him and the girl he was sending letters went on "Practice Dates" And they ended up liking each other :) The other girl likes playing "Games" with her letter guy :) ; so does the third girl. (I believe that at the end, a minion of the girl's mother helped the girl get rid of her mother. Maybe it was because the mother tried to kill her daughter for being beautiful, especially since she was getting older. (hide spoiler) Please help me find it. It was the first book that got me to love reading at that time. :D Cover Clue: This book was blue (if that helps). And I think it had a mailbox on the front of it.

Question: So I do not remember but the title, author, or cover, but I remember a bit of the plot. I believe it starts out with the main characters set up on a date. They don't know each other's names, but their date consists of a sexy get to know you /photo shoot where they're not allowed to have sex but they do anyway. She gets pregnant but can't find him after. Flash forward a bit, she works for her father's large New York office and they make a deal to with with her baby daddy's office. He finds out she is pregnant and so the story goes. I hope this is enough to go on and someone recognizes it. Thank you! Cover Clue: N/A

Now here is the example, remember not to add in additional information that's not in the question.

Question: I read this book back around 2008 (I think) can't remember author but read two of her works 1) is about this girl who falls in love with a guy who is actually a dragon, he takes her back to his homeland and I think he is injured, there are other dragons there and his relatives are also dragons author gives very vivid imagery I believe there were types of dragons like fire and ice ones... 2) the second book (warning: spoilers ahead) is about this girl who has a sister named rose (not too sure about name) who is already engaged to some local village boy, but she falls in love with this cold man who is like "ivy" and they escape together for a while but she returns and the girl also loves him but he eludes her grasp, I remember the end she talks about the imagery on the wall of roses and ivy intertwining together... Any help much appreciated!!! :D Cover Clue:

Predictive Prompts

You are a utility for guessing titles of books. Given the book described below, what is a possible title for the book? Only return one predicted title without any extra text. Even if you're unsure, try to come up with something.

Description: ()

5532
Query Rewrite Prompt

"You are a utility for helping users find books. Given the user's book description below, generate a query that a user can copy-paste into a book database to find the book. The query should focus on the important aspects of the book that will help the database locate it. These can be keywords about the book's title, author, genre, year, or distinguishing character or plot elements.\n\nUser's Query: {}
\nDatabase Query: "