'It's on the tip of my tongue': A new Dataset for Known-Item Retrieval

Samarth Bhargav
s.bhargav@uva.nl
IRLab, University of Amsterdam
The Netherlands

Georgios Sidiropoulos
g.sidiropoulos@uva.nl
IRLab, University of Amsterdam
The Netherlands

Evangelos Kanoulas
e.kanoulas@uva.nl
IRLab, University of Amsterdam
The Netherlands

ABSTRACT

The tip of the tongue known-item retrieval (TOT-KIR) task involves the 'one-off' retrieval of an item for which a user cannot recall a precise identifier. The emergence of several online communities where users pose known-item queries to other users indicates the inability of existing search systems to answer such queries. Research in this domain is hampered by the lack of large, open or realistic datasets. Prior datasets relied on either annotation by crowd workers, which can be expensive and time-consuming, or generating synthetic queries, which can be unrealistic. Additionally, small datasets make the application of modern (neural) retrieval methods unviable, since they require a large number of data-points. In this paper, we collect the largest dataset yet with 15K query-item pairs in two domains, namely, Movies and Books, from an online community using heuristics, rendering expensive annotation unnecessary while ensuring that queries are realistic. We show that our data collection method is accurate by conducting a data study. We further demonstrate that methods like BM25 fall short of answering such queries, corroborating prior research. The size of the dataset makes neural methods feasible, which we show outperforms lexical baselines, indicating that neural/dense retrieval is superior for the TOT-KIR task.

CCS CONCEPTS

• **Information systems → Document filtering**; *Web searching and information discovery*; *Document representation*; *Retrieval models and ranking*; *Test collections.*

KEYWORDS

Known Item Retrieval; Tip of the tongue known item retrieval;

ACM Reference Format:
Samarth Bhargav, Georgios Sidiropoulos, and Evangelos Kanoulas. 2022. 'It's on the tip of my tongue': A new Dataset for Known-Item Retrieval. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining (WSDM '22), February 21–25, 2022, Tempe, AZ, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3488560.3498421

1 INTRODUCTION

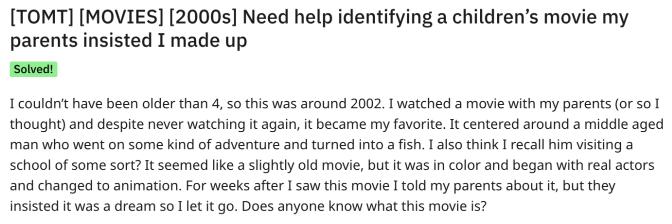
The tip of the tongue known-item retrieval (TOT-KIR) task involves a user searching for an item which they had previously encountered, for which they are unable to recall the precise identifier [2]; for instance, they are unable to recall the name of a movie viewed several years ago, characterized by the phrase 'it lies on the tip of my tongue'. More precisely, the task involves the retrieval of a *single* item from a potentially large collection of items, given a descriptive query with possibly imprecise information about the item, along with other information such as the circumstances of encountering the item, the sentiment/emotion associated with it, the medium it was encountered in, etc [2, 7–9]. While this is similar to Known-Item Retrieval (KIR) or item re-finding, we note that TOT-KIR is distinct from KIR: in TOT-KIR the identifier is unknown, the item is searched after a much longer period of time after encountering it, the requests contain imprecise/incorrect information i.e false memories [7–9] and the queries are verbose [2].

This task is particularly hard for existing search systems, which is evidenced by the emergence and subsequent popularity of several online communities[1] that allow users to pose TOT-KIR queries. The challenging nature of this task is further evidenced by references to previous search attempts in these posts [2], highlighting the need for datasets and methods to tackle this problem.

This difficulty may stem from the nature of the query itself. For instance, most queries are *verbose* compared to typical searches, and may contain *imprecise or incorrect descriptions* from a user's (typically) faint recollection of the item. Indeed, the incidence of false memories has been previously studied for KIR [2, 7–9]. Furthermore, the query may contain terms that *might not be useful* or even *harmful* for retrieval performance, such as previous search attempts, the context or medium it was encountered in, or *hedging words* like 'probably', 'maybe', etc [2]. In some cases, users mention other items to inform other users that the known-item are *not* among them. In addition, the text may contain *indirect references* ('actor that looks like Tom Cruise') which are not resolvable using methods like BM25, requiring more advanced methods [2]. Research in TOT-KIR is impeded by the lack of large scale or open collections [8, 9].

We make two contributions in this paper. First, we describe a process to extract known-item queries and their corresponding 'gold' items using a heuristic, from the Tip of my Tongue sub-reddit (https://www.reddit.com/r/tipofmytongue/), a sub-community in the social media website *Reddit*. We collect this dataset with a focus on *precision* i.e gather only queries for which we can get the 'gold' known-item with a high confidence, instead of gathering as

[1]Other sub-reddits like https://www.reddit.com/r/whatsthatbook/, or online communities such as https://irememberthismovie.com/ [2]


[TOMT] [MOVIES] [2000s] Need help identifying a children's movie my parents insisted I made up

Solved!

I couldn't have been older than 4, so this was around 2002. I watched a movie with my parents (or so I thought) and despite never watching it again, it became my favorite. It centered around a middle aged man who went on some kind of adventure and turned into a fish. I also think I recall him visiting a school of some sort? It seemed like a slightly old movie, but it was in color and began with real actors and changed to animation. For weeks after I saw this movie I told my parents about it, but they insisted it was a dream so I let it go. Does anyone know what this movie is?

Figure 1: A popular post in the subreddit, containing a title (top, bold text), a flair (in green) and a description (bottom). A subset of accompanying comments are shown in Figure 2. The full post can be viewed here.

many data points as possible. We collect 15K data points from two domains, namely *Books* and *Movies* , and show that the heuristic we propose is accurate using data quality checks. This dataset, which we term *Reddit-TOMT* , is the largest collection of known-item queries yet. Alternatives to the process outlined in this paper are (a) annotation by crowd workers [2] or experts [7] or (b) simulated queries [3, 6, 13]. However, the former is expensive and/or time-consuming, while the latter might not account for characteristic properties such as false memories[7–9]. Our heuristic allows for a cost-effective, quick, extensible method to build collections for TOT-KIR.

Second, we benchmark this dataset with both sparse/lexical baselines and a neural/dense method [12, 25], with the goal of investigating if TOT-KIR requests can be addressed by methods robust to low lexical overlap. We show that dense methods outperform lexical methods by a large margin on *Reddit-TOMT* . We have released our code, data and benchmarks, which can be accessed at https://github.com/samarthbhargav/tomt-data.

2 RELATED WORK

In the following section, we revisit early work in Known-Item Retrieval (KIR) from Library Information Sciences, followed by recent work. We end the section with other datasets and methods for Known Document/Item search.

Known-Item Retrieval, also termed Known-Item Search, has a long history in Library and Information Sciences. Buckland [4] critique the distinction between 'known item searches' and 'subject searches'. Lee et al. [16] further discuss the complexity of defining the task in this context. They also note challenges with the task: The type of relationship with the item, whether it is direct (first-hand experience) or not (second-hand); the existence of the item, and if the belief that the item exists is justified; and finally, issues with the accuracy of the request. In this regard we assume (a) that item requested exists, and that the user has encountered this item before; (b) the information contained in the query may or may not be accurate, and may contain extraneous information irrelevant to the task.

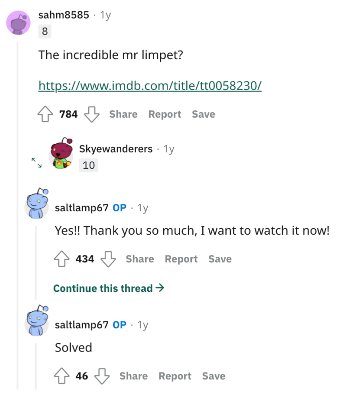
KIR or (item) 're-finding' is defined as *repeated* searching for a document that was previously found or accessed [21]. These queries have been found to constitute up to 40% of all queries, with about a third being identical[22]. Re-finding/KIR in general is distinct from TOT requests. KIR queries are much shorter[22], and comparatively less time passes before a repeat query[5].

Data in this domain is typically not public, motivating approaches for simulating known-item queries. Azzopardi et al. [3] generate synthetic data by first selecting a document and then generating a query corresponding to it. Query terms are sampled from a probability distribution over document terms, prioritizing discriminative terms, with noise added to mimic false memories. This general approach has been adopted for specific domains such as personal information management [6] and emails [13]. Kim and Croft [14] study type-specific metadata for known item search on desktops using a test collection collected using a human-computer game on a pseudo-desktop. Participants were shown items and tasked to generate queries which rank the correct item higher; however this study does not model false memories, a key characteristic of TOT-KIR. Other datasets for re-finding or KIR are: Homepage finding (TREC 10), Named-page finding (TREC 11), and the Known-Email search (TREC 2005, Enterprise). Hauff et al. [8], Hauff and Houben [9] note that false memories and other inaccuracies need to be modeled for realistic queries, and corpora need to be public, contain queries generated organically i.e without automatically generated queries from a predetermined known-item. *Reddit-TOMT* not only contains real queries, but can be periodically updated or easily extended to other domains.

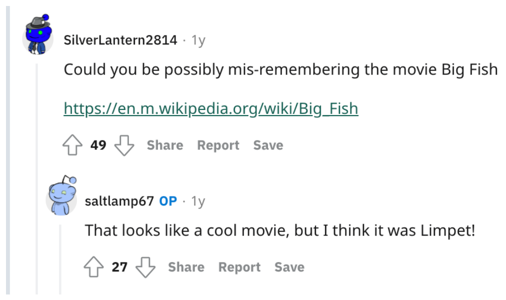
Hagen et al. [7] curate a corpus of 2,755 known-item questions, each mapped to a web page in the ClueWeb09 collection, from the *Yahoo!* answers community. 240 of these questions were annotated with false memories and corresponding correct information. These collections focus on KIR queries, we now describe work that focus on 'Tip of the Tongue' (TOT) requests. The phenomena of false memories, the lengthy duration between encountering and querying for an item, as well as increased verbosity are characteristic of TOT queries, compared to the KIR task. In addition, TOT queries involve imprecision and unreliable information, is typically 'one-off', and emphasizes identification over navigation [2]. Arguello et al. [2] study the problem using the irememberthismovie.com website, scraping 1000 movies and having crowd workers label the correct answer from replies, in addition to sentence level annotation of the queries. We corroborate their observation that BM25 methods maybe insufficient for this task, providing additional evidence that advanced methods/ representations might be needed to solve this task. In contrast to Arguello et al. [2], we use heuristics instead of crowd workers, consider multiple domains, and gather a larger pool of candidate items, making the task more realistic. Jørgensen and Bogers [11] conduct a similar study on 250 posts from the 'Tip of my Joystick' community, a sub-community similar to the one explored in this paper. *Reddit-TOMT* is by far the largest dataset available for this task, while requiring no crowd-sourcing or hand-labeling.

3 DATA COLLECTION METHODOLOGY

The data collection process outlined in this section produces around 15K query/known-item pairs, using data spanning 2017-2020, collected from the /r/TipOfMyTongue *Reddit* community. The first step, which is to collect, filter and categorize user requests, is outlined in Section 3.1. Next, the process to extract gold known-items,


sahm8585 · 1y
8
The incredible mr limpet?
https://www.imdb.com/title/tt0058230/
784 Share Report Save
Skyewanderers · 1y
10
saltlamp67 OP · 1y
Yes!! Thank you so much, I want to watch it now!
434 Share Report Save
Continue this thread →
saltlamp67 OP · 1y
Solved
46 Share Report Save

(a) **The comment tree containing the gold answer, to which the original poster replies with 'Solved'**


SilverLantern2814 · 1y
Could you be possibly mis-remembering the movie Big Fish
https://en.m.wikipedia.org/wiki/Big_Fish
49 Share Report Save
saltlamp67 OP · 1y
That looks like a cool movie, but I think it was Limpet!
27 Share Report Save

(b) **Another comment tree, containing an answer rejected by the user**

Figure 2: Two comment trees, one containing the gold answer, and one containing a suggestion rejected by the user - which consider as a 'negative'

along with other items, is described in Section 3.2. Since this process is automatic, we perform a quality check to verify the efficacy of our heuristic, which is described in Section 3.3. The following paragraph introduces the community and describes the structure of a typical request, called a submission.

The /r/TipOfMyTongue community is a sub-community where users post TOT requests (referred to as a *submission* or *post*) which are then answered by other users in the community, sometimes in a collaborative manner. The user who makes the submission, called the 'Original Poster', provides a *title* and a *description*. An example post is shown in Figure 1, which has a *flair* assigned to it by the moderators. Suggestions by other users form a *comment forest*, where the roots of the trees correspond to top-level replies. Two such trees are shown in Figure 2. We use all of these fields to filter and then extract query and known-item pairs.

3.1 Filtering

We first filtered submissions based on the flair text. By default, a request is marked '*Open*', and if the search is successful, this is changed to '*Solved*'. Some posts (prior to 2019) did not conform to this, are otherwise empty (no flair), or the post is deleted (if a user deletes a submission, the content is removed, but the submission can remain). These were labeled '*Other*', '*Unknown*' and '*Removed*' respectively. We gathered submissions made in the 2017-2020 period, using the Pushshift API[2] (to extract submission ids/URLs), and the Python Reddit API Wrapper [3] to gather a total of 843,493 submissions. After skipping posts marked 'Over 18', a total of 793,226 submissions were obtained, of which 291,741 were marked *Solved* and 143,974 *Open*. We used only *Solved* threads in the following steps. A breakdown of submissions categorized by its status is shown in Figure 3a. The data suggest that only about half of the requests were solved in 2020, if *Removed* posts were not considered. This may be because of incomplete or inaccurate information, which may be exacerbated by false memories, a feature common in such requests [2, 7]. The titles were then used to categorize posts.

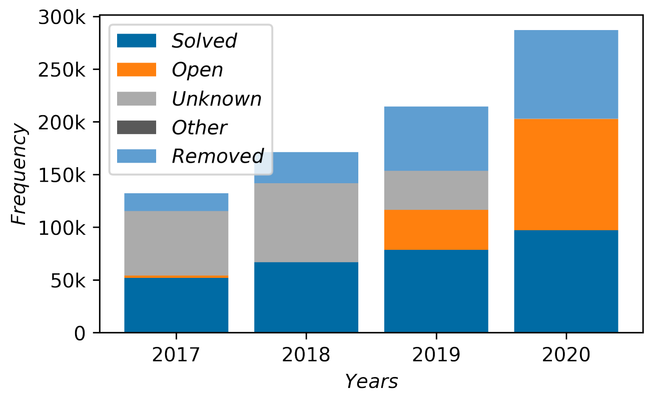
Several titles conformed to a pattern, despite being free-text: they begin with '*[TOMT]*', followed by a description of the category in square braces i.e '[TOMT][*Category*]'. For instance, '[TOMT] [Movie] [80/90s]' indicates the user is looking for a Movie that might have been made in 1980s or 1990s. We extracted the text in the second set of braces ([*Category*]), and grouped them into two categories corresponding to *Books* and *Movies* . This grouping was done by one author and independently verified by another author to ensure quality. 69 such categories including Television series, were grouped into a '*Movies* ' category, while 22 were grouped into '*Books* ', with 68,724 and 15,613 submissions respectively. Requests for movies were most frequent in the community, while *Books* form a much smaller fraction, as shown in Figure 3b. Apart from the flair, title and description, each submission has a comment forest. We noted that most posts, especially in *Books* , had only a single top-level reply with minimal back-and-forth between users. Crucially, the rules of the community state that users are asked to reply to the accepted answer with 'Solved'. The next section describes a heuristic to use this to extract 'gold' answers.

3.2 Extracting Gold Answers

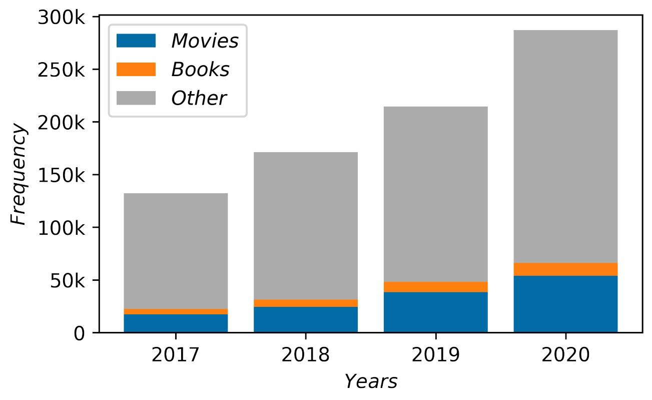
Since a typical 'Solved' post has a canonical comment by the original poster, we design a heuristic to extract gold answers from solved posts, outlined in Algorithm 1. Our approach can be summarized as: (1) traverse the comment trees to find the reply which indicates that the author has accepted the answer ('Solved!') (2) construct a path from this node to the root of the tree (3) extract URLS from this path, and link this to corresponding item page (4) extract query-item pairs by picking requests with a single candidate . The URLs are examined only if they are from (1) Wikipedia (2) IMDb [4] (for *Movies*) (3) GoodReads [5] (for *Books*) with each such URL considered

[2]https://pushshift.io/api-parameters/
[3]https://praw.readthedocs.io/en/latest/

[4]A community website for digital media: https://www.imdb.com/
[5]A community website for Books https://www.goodreads.com/



(a) Submission Status



(b) Category

Figure 3: Submission statistics for the subreddit for 2017-2020, with the number of posts steadily increasing. Note that several posts remain unsolved in 2020. The categorization used to produce (b) is described in Section 3.1

as a candidate for the gold answer[6]. As this heuristic may extract multiple candidates, only submissions with a single candidate were considered, which is justified since our goal is to gather *accurate* query-item pairs in an *automated* manner. We leave the utilization of submissions with multiple candidates to future work.

Each candidate consists of the following: (1) an identifier uniquely identifying the known-item (2) a title associated with the item (3) a description of the item (4) additional meta data (if applicable) . The following two paragraphs detail the data extracted for each subset, followed by a discussion of other domains and the extraction of other items to make the task more realistic.

3.2.1 Movies . The IMDb ID was used as the identifier for *Movies* , since this is referred to most frequently in the comments. If, instead, a Wikipedia URL was found, the IMDb ID (property P345) was obtained using the corresponding WikiData ID associated with the URL. Similarly, if the URL was an IMDb ID, we linked this movie back to a Wikidata entry, while ensuring that the IMDb ids from both sources matched. The title and description of the movie were extracted from Wikipedia (using Wikiplots, similar to [2]), if available, otherwise from IMDb .

3.2.2 Books . We used the GoodReads 'Work ID' as the identifier for *Books* , instead of ISBN-10/13 since books can have multiple editions. If a Wikipedia URL was available instead, we extracted the ISBN-10/13 using the corresponding Wikidata entity, and linked it back to GoodReads (and vice versa). We utilized the Book Graph dataset [23, 24] (extracted in late 2017) for matching ISBNs to work IDs and for extracting titles and descriptions. Additional information such as reviews could also be used towards improving retrieval performance. While we focus on *Books* and *Movies* , it is straightforward to extract similar data for other domains.

3.2.3 Other domains. The subreddit has several other categories not considered in this work. For instance, there were 50,163 solved submissions corresponding with the (free-text) categories 'song' and 'music'. Creating a dataset for other domains is straightforward and involves (1) a method to map a URL to an identifier (and optionally link this to other sources like Wikipedia) (2) methods to extract data for a particular item, or to link it to existing data (i.e `get_data` in Algorithm 1) .

3.2.4 Other candidates and negatives. In addition to the query-item pairs described above, we gathered other items that were not selected as 'gold' for any query. These were sourced from candidates which were not selected as gold answers, including those that were discarded in the filtering step (i.e submissions with $>$ 1 candidates found in `get_answers`). These items increase the number of candidates in the retrieval pool, making the task more realistic. In addition to these, we extracted *negatives* sourced from submissions where a query-item pair was extracted. An item is considered a *negative* for a given query, if it was proposed as an answer but ended up being rejected by the user. These items are termed negatives since other users confused them with the 'gold' item. These items were not extracted from the comment tree containing the accepted answer. As there were too few negatives for *Books* , we did not include them in the data or experiments. The resulting dataset statistics are reported in Table 2. Note that some queries have the same item, which is why the documents are fewer than the queries. In *Reddit-TOMT* , the most frequently requested item is '*Mindhunters*' (21 times!) in *Movies* , '*The Transall Saga*' (8 times) in *Books* .

3.3 Data Quality Checks

Three aspects of the data were examined in this study: (1) Heuristic Accuracy: whether the answer picked by the heuristic matched the answer picked by the author of the post (2) Data Leakage: whether the answer itself was in the text [7] (3) Malformed Query: Whether

[6]These websites were selected after a preliminary analysis showed that these were the most frequent

[7]*Reddit* allows a user to edit the original submission. We observed that some descriptions were edited by the original poster to include the answer after the request was solved, which prompted this aspect of the study. It is an unwritten rule that edits are appended to the end of the submission.

Algorithm 1: Heuristic for finding the 'gold' known item from a comment forest, given the author of the request

```
function get_solved_node comment_forest, original_poster
    for comment_tree in comment_forest do
        reply_stack = [comment_tree] ;
        while reply_stack is not empty do
            reply = reply_stack.pop();
            if reply.author == original_poster then
                if reply contains 'Solved' then
                    return comment_tree, reply;
                end
            end
        end
    end
end
function get_answers submission
    comment_tree, solved_node =
      solvedPath(submission.comment_forest,
      submission.author);
    urls = extract_urls_to_path(comment_tree,
      solved_node);
    candidates = [] ;
    for url in urls do
        if url domain is wikipedia or imdb (or goodreads)
          then
            gold_answer = get_data(url);
            candidates.append(gold_answer);
        end
    end
    return candidates;
end
```

the query was usable e.g if it contained only a URL, it was considered unusable. This happens if the query pointed to an image, or a screenshot of a movie . Two of the authors first independently labeled the data using Label Studio, and then resolved disagreements, on 3 subsets: `QAMovies`, 100 randomly selected data points from the *Movies* ; `QAMovies>1`, 100 randomly selected data points from *Movies* , where the depth of the reply containing the answer is *greater than one*; `QABooks`, 100 randomly selected data points from the *Books* subset. As there were too few data points where the answer positions were greater than one in the *Books* subset, we didn't conduct a study similar to `QAMovies>1` for *Books* . The results are reported in Table 1 and discussed below.

3.3.1 Heuristic Accuracy. The accuracy for `QAMovies>1` (96%) is slightly lower than that for `QAMovies` (98%), which indicates that the heuristic works slightly worse for answers located deeper in a comment tree. This is likely to happen only in cases where the original poster rejects an answer containing a URL but accepts a plain-text answer (no URL) in a deeper reply, which is not picked up by the heuristic. However, we noted there are relatively few posts where the answer is located deeper in the comment tree. We note that the overall accuracy of 97.6% might be under-estimated,

Table 1: Results of the data study outlined in Section 3.3. Overall accuracy is 97.6%, highlighting the efficacy of the heuristic outlined in Algorithm 1.

QA Set	Question	Count
	Correct/Incorrect Answer	98 / 2
`QAMovies`	Well-formed / Malformed Query	98 / 2
	Answer in submission / Not in submission	94 / 6
	Correct/Incorrect Answer	96 / 4
`QAMovies>1`	Well-formed / Malformed Query	86 / 14
	Answer in submission / Not in submission	99 / 1
	Correct/Incorrect Answer	99 / 1
`QABooks`	Well-formed / Malformed Query	97 / 3
	Answer in submission / Not in submission	100 / 0

since most of the answers were found at the root of the comment tree (99.9% of *Books* and 98.9% of *Movies*).

3.3.2 Data Leakage. The data seem to have very few answers in the query itself. To prevent any data leakage (i.e the name of the answer is available in the test), we picked only posts with edited descriptions (see footnote in previous page), and examined these to check if they had tokens from the title of the known-item in the last sentence. If such a token occurred, the last sentence is removed and the description is updated. We note that there were only 5 such descriptions (0.22%) in *Books* , and 66 (0.49%) in *Movies* .

3.3.3 Malformed Query. We removed queries which had either an empty description, or have a short description with only a URL. This resulted in 628 (4.45%) and 61 (2.56%) queries being removed from *Movies* and *Books* respectively.

Despite the efficacy of our approach, it is important to note its limitations, which we outline here. First, users only post requests to these communities *only if* their TOT queries fail using existing search engines, which creates a sampling bias in the types of posts. However, this is an unavoidable bias inherent in approaches that source data from communities like Reddit [11], Yahoo Answers [7] or other community websites [2]. Second, since we filter out *unsolved* queries, it contains only queries which can be resolved i.e *Reddit-TOMT* cannot reveal why TOT queries fail, which we believe can be important for a deeper understanding of the task. In addition, the filtering process considerably reduces the number of queries e.g using submissions with links may lead to several queries (without URLs) being missed out. This can happen if the user uses plain text without linking to these websites, for instance. While we attempted to capture the latter using entity linking, the large number of false positives (even after filtering out entities without IMDb IDs for *Movies*) made this approach unviable without annotation, which is contrary to our goal. In addition, since we didn't consider replies made by users in subsequent interactions with other users, some queries themselves might be 'unsolvable' without taking these *clarifications* into account. However, as these submissions are in the minority, this issue is minimized. Finally, we note that the sub-reddit contains only English posts.

Subset	No. queries	No. positive documents	No. other candidates	No. negatives	Total documents
Books	2319	1910	710	-	2620
Movies	13469	8845	4797	1221	14863
Total	15788	10755	5507	1221	17483

Table 2: Overall statistics of *Reddit-TOMT* .

4 EXPERIMENTS

This section describes the methods we used as benchmarks. Two approaches, Lexical/sparse (Section 4.1.1), and Neural/Dense retrieval (Section 4.1.2) is described in Section 4.1, followed by the experimental setup in Section 4.2.

4.1 Benchmarks

We tested the performance of a state-of-the-art dense retriever as well as traditional lexical-based ones on *Reddit-TOMT* . We used BM25 [20] and PL2 [1], two lexical retrieval methods, and Dense Passage Retrieval [12], a dense retrieval method.

4.1.1 Sparse Retrieval Benchmarks. BM25 [20] is a standard baseline in IR tasks, and is considered difficult to outperform. PL2, a divergence from randomness (DFR) model, is the 2-Poisson model with Laplace after-effect, with normalization [1], which may be suited for early-precision tasks [10, 19]. For both baselines, we used the py-Terrier[18] framework's implementations. The data was processed using Terrier's default processor, which removes stop words and stems the tokens using the Porter stemmer.

4.1.2 Dense Retrieval Benchmarks. In contrast to traditional methods which treat each document as a bag of tokens, dense retrieval conducts retrieval in a *dense* representation space, by encoding each document/query into a latent space. By projecting documents into a latent space, models can account for 'noise', for example, synonymy, allowing for retrieval based on semantics rather than lexical overlap. We used Dense Retrieval (DR) [12, 25] as a benchmark for this purpose. DR is a passage retrieval model where dense representations are learned from pairs of questions and passages (or answers) by a dual-encoder model, without any additional pretraining. Given a question q, alongside a relevant passage p^+ and a set of irrelevant passages $\{p_1^-, p_2^-, \ldots, p_m^-\}$, the model learns to rank relevant passages higher by optimizing the negative log likelihood (NLL) of the positive passage. In our problem, a relevant passage is the description of the 'gold' known-item, and negative passages are other items. The following section describes the experimental setup.

4.2 Experimental Setup

We first split the data into a train (80%), validation (10%) and test set (10%), by randomly sampling query-item pairs, both for *Books* and *Movies* separately. We index the descriptions of each candidate from the candidate pool, which consists of both positive and negative candidates.

For BM25, we tuned the c, k_1 and k_3 parameters. We performed grid search using the following values: $c \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1\}$, $k_1 \in \{0.3, 0.6, 0.9, 1.2, 1.4, 1.6, 2\}$, and $k_3 \in \{0.5, 2, 4, 6, 8, 10, 12, 14, 20\}$. For these methods, which do not involve training a model, we tuned the hyper-parameters on the train-validation sets, and report the performance on the test set.

We trained DR on the training set for 25 epochs on 2 × 12GB GPUs. Instead of using a bi-encoder architecture with separately parameterized query and passage encoders, we share the parameters for both. In preliminary experiments, we found that this performed better, perhaps due to the descriptive nature of the queries. We used a RoBERTa-based[17] encoder, an Adam [15] optimizer with a learning rate of $2e-5$, and linear scheduling with a warm-up ratio of 0.1. We trained using in-batch BM25 negatives, with a batch size of 4 and one additional negative per question. Finally, the maximum passage and query length was limited to 512; truncating anything beyond this length. Hard negatives for DR were obtained using BM25 (default parameters). In addition, since we extracted negatives from the data (as opposed to BM25), we experimented with using these in place of BM25 negatives in the training process. In this setting, which we term DR_{HN}, if a negative from *Reddit-TOMT* wasn't available, BM25 negatives were used instead. We note that only negatives were available only for 1444 queries in the training set.

We evaluate the benchmarks on the following metrics, after truncating the ranking list to 1000:

- *Recall@1 (R@1)* and *Recall@10 (R@10)* (referred to as Success@1/10 in [2]): Which measures the ability of a method to return the correct result in the top 1 or 10 list respectively. This follows Arguello et al. [2].
- *Mean Reciprocal Rank (MRR)*: MRR is the average of the reciprocal ranks of results. Higher numbers indicate that known-items were positioned higher in the ranking.

R@K measures whether a user manages to find the item in the initial ranks, while MRR measures *effort*, since the user might have to peruse items further down in the list. We report both the mean and the standard deviation across all queries for each metric. The following section reports and discusses the results of these benchmarks.

5 BENCHMARK RESULTS

The overall results are reported in Table 3. From the results, we can conclude the following: (1) Lexical methods like BM25 may be inadequate for known-item search, corroborating prior findings [2] on a larger dataset dataset (2) DR outperforms all lexical methods, indicating dense methods may be better suited for TOT-KIR . We go over these results in the following paragraphs.

We first note that BM25 outperforms PL2 in all settings. We further note that the performance on *R@10* using BM25, 0.3433, is higher compared to the performance noted in Arguello et al. [2] - 0.1327 another TOT-KIR dataset.

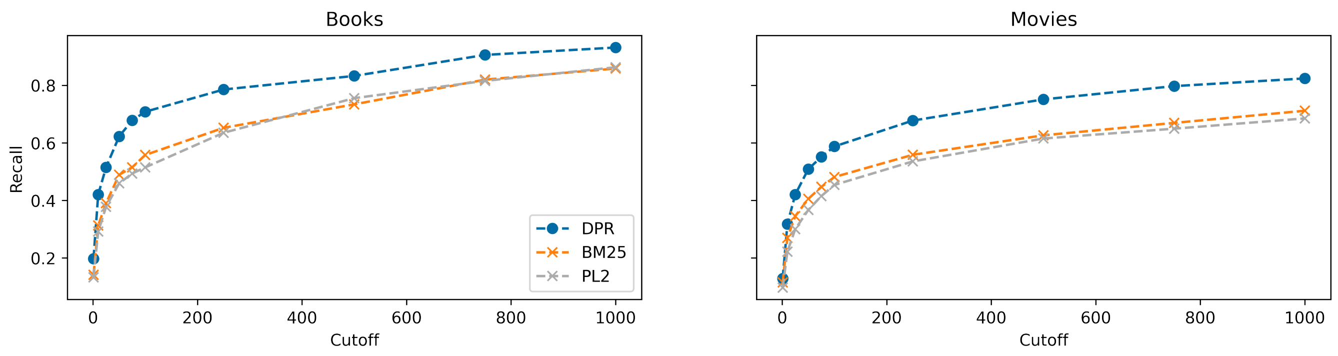


Figure 4: Average recall/success at different cut-offs.

Dataset	Method	R@1	R@10	MRR
Books	BM25	0.1416 (0.3487)	0.3133 (0.4638)	0.1971 (0.3444)
	PL2	0.1330 (0.3396)	0.2918 (0.4546)	0.1873 (0.3363)
	DR	**0.1974 (0.3981)**	**0.4206 (0.4937)**	**0.2783 (0.3835)**
Movies	BM25	0.1152 (0.3192)	0.2697 (0.4438)	0.1679 (0.3226)
	PL2	0.0973 (0.2964)	0.2221 (0.4157)	0.1423 (0.3009)
	DR	**0.1285 (0.3347)**	**0.3180 (0.4657)**	**0.1938 (0.3343)**

Table 3: Results from benchmarks: BM25 outperforms PL2, while DR outperforms BM25 by a large margin, indicating that dense methods are more suitable for TOT-KIR

To test if negatives harm performance, we conducted a controlled experiment, using *only* queries which have negatives. In the first run, for each query in the test set with a negative, we created an index consisting of only 'positive' documents. In the second run, in addition to positive documents, we index (only) negatives corresponding to the current query. Both use the default parameters for BM25. Comparing the two rankings, we found that negatives were placed higher than gold known-items for 46.5% queries, with the mean ranks being 411 for negatives and 1881 for the gold documents. This indicates lexical methods fail to retrieve the correct known items, instead ranking negatives higher for some queries. While we can't make a broader conclusion due to the limited number of queries with negatives, it might be that users are driven to these communities precisely because the wrong movies are retrieved by existing search engines. This is further evidenced by some users describing previous search attempts or asserting that the correct known item *is not* among certain items [2].

BM25 retrieves the correct known item as the first result for 14.16% of queries in *Books* . For *Movies* this number is lower, at 11.52%. The *R@10* numbers are about in the 0.26-0.31 range, suggesting that about a third of the queries can be resolved by a user perusing the top-10 results. Despite this, if we consider the typical scenario where a user views only the first few results, BM25 falls short: the majority (i.e approximately 70%) of queries seemingly cannot be answered using BM25/PL2. We note however, that the capability of existing search engines to resolve TOT-KIR queries is unknown, so a broader conclusion about its efficacy on queries cannot be made (see limitations in Section 3.3).

Initial experiments showed that DR (using a pretrained RoBERTa [17]) without training on *Reddit-TOMT* , achieves very poor performance, with a *R@1* of 0.0067(0.0815), *R@10* of 0.0223(0.1476) and a *MRR* of 0.0131(0.0903). This further motivates *Reddit-TOMT* , since deep models typically require a large training set to perform effectively.

Unsurprisingly, DR outperforms BM25 by a large margin for *Books* and a smaller but still large margin for *Movies* . For *Books* , DR retrieves the correct known-item in the top-10 for almost half of the queries, and for *Movies* this is reduced to a third. On average, the correct known-item is also placed higher in the result list, as evidenced by higher MRR scores.

DR$_{HN}$, which uses negatives from the dataset instead of the default BM25 negatives, performs slightly worse than DR, but is still better than lexical methods. DR$_{HN}$ has a lower *R@1* of 0.1248(0.3305) and a lower *MRR* of 0.1888(0.3295), but has a marginally better recall of 0.3210(0.4668) for *Movies* . While the performance differences are insignificant, we hypothesize that this might be due to multiple factors: First, only a fraction of the queries in the training set have negatives and access to additional negatives might change the results; second, since negatives are extracted from user comments, they might be noisy, making distinguishing between negatives and the correct item trivial, leading to little or no learning; finally other factors introduced in training deep models (like choice of model, etc) might also be a factor.

To further investigate the effort required to find items in a list of results, we compute and plot recall at multiple cut-offs in Figure 4. It is evident that DR outperforms BM25 and PL2, highlighting the inability of lexical methods to retrieve the correct result. We conclude that *Dense retrieval is more effective at the known-item retrieval task.* The following paragraph describes examples where either BM25/DR excel or fail.

Qualitative Analysis. Table 4 contains examples from the dataset. We note that BM25 excels at retrieving items with very discriminative words like 'geyser' (1) or 'atticus' (2). However, since false memories are common in KIR [7–9, 11], discriminative terms, e.g names, may lead to failure. Furthermore, the presence of hedging terms or otherwise imprecise descriptions aren't handled directly

#	Query	Correct Known-Item	BM25 Rank	DR Rank	Top Result BM25/DR
1	Old black and white -alien?- movie, where in the end the main alien die by a hot water geyser. Can't remember much else, I watched this movie like 30 years ago or so, asked about it a lot and still nothing.	Lobster Man from Mars*	2	-	Journey to the Center of the Earth* / The Best Years of our Lives *
2	dystopian fiction book, possibly YA, about a teen with lion feet ....There's a man named Atticus I think, who had crab hands ...	Above*	1	-	Above* / The Deserter*
3	Teens (and one kid) are sent to a military style boot camp. The guy in charge is African American and has a gap tooth. Only scene I vividly remember is the youngest kid crawling in mud while it's raining and the movie ended with the guy in charge smiling at the kids	Major Payne*	-	1	Camp Nowhere* / Major Payne*
3	Book where kids travel to an alternate reality with people "taken out of time"? I vaguely remember a book from my childhood ...	Jake Ransom and the Skull King's Shadow*	-	1	Amnesia* / Jake Ransom and the Skull King's Shadow*
4	Looking for a recent book, dark cover with a pentagram, eerie font for the title and such, but I don't remember the title or the author. Found it on the Barnes and Noble website ..	The Merciless IV*	-	9	The Mysteries of Harris Burdick* / I Hope They Serve Beer in Hell*
5	Childrens book about playing cards I barely remember anything ... (teacher read it to us in 1st grade, 1996) ... days of the week or something like "Tuesday" seems significant, either part of the title ...	One Monday Morning*	-	-	Huggly Goes To School* / You Are Special*

Table 4: Qualitative examples from the data, highlighting certain cases where BM25/DR succeed and fail. '-' indicates failure to retrieve in the top-1000. Clicking on '*' beside an item leads to its page.

by BM25. DR, however, appears to handle this better i.e (3), or when there is a low lexical overlap. Surprisingly DR ranks the gold item high for (4), which describes only the cover of the book. The last query (5) contains incorrect information ('Tuesday' instead of 'Monday') and both methods fail to retrieve the correct item. We present concluding remarks and future research directions in the next section.

6 CONCLUSION

In this paper, we outlined a novel, automated process to gather a dataset for known-item retrieval in two domains, *Books* and *Movies* , using heuristics for finding 'gold' known-items, along with negatives that other users may confuse the correct item with. Through data quality checks, we showed that this heuristic is accurate, rendering automated collection of TOT-KIR datasets feasible. In contrast to prior work, this process does not require human labeling, allowing for large scale datasets in diverse domains. Furthermore, this process renders synthetic known-item queries unnecessary. Using this algorithm, we collected 15,788 query/known-item pairs, the largest dataset to date for the TOT-KIR task. This large scale dataset allows for training neural models. We have open sourced the code to enable other researchers to collect additional data, perhaps in other domains.

Finally, we evaluated multiple benchmarks on this dataset, using both methods that rely on lexical overlap and methods that create dense representations for retrieval. We showed that lexical methods cannot retrieve the correct known item for a majority of the data, highlighting the difficulty of the task. Furthermore, we showed that a dense retrieval benchmark outperforms the lexical baselines by a large margin. We see many directions of potential future work. The first thread of work involves the data collection process, while the second is methodological.

While we focused on two domains, gathering data for other domains is straightforward, since only part of the algorithm (e.g the *get_data* method) needs to be adapted for the new domain. Apart from this, the data can be periodically expanded by considering recent submissions. In addition, while this work focused on only *solved* submissions, further research is required to understand why TOT requests fail. For instance, discriminative information might be missing in such queries, and identifying this could be useful e.g in the formulation of clarifying questions. The number of data points can be increased by using entity linking instead of using URLs, or by finding the correct known item from the pool of extracted candidates. Finally, data for items can be augmented from additional sources. For instance, we found some users who referred to reviews in their answers. Review information can be extracted and indexed as well, augmenting the information available for items - especially since TOT requests can contain plot information otherwise not contained in a synopsis ('spoilers'). It has been hypothesized that TOT queries can benefit from multi-hop reasoning to answer indirect queries [2]. Such methods can use entity information, which can be gathered from Wikidata using the Wikidata IDs we extract (for *Movies*). In addition, since we also collect all replies made by users, they can potentially be used to either build conversational agents or use clarifications in these replies to improve retrieval performance.

ACKNOWLEDGMENTS

The authors would like to thank Wilker Aziz, Antonis Krasakis, Svitlana Vakulenko, Vera Provatorova and Mohammad Aliannejadi for their suggestions. The authors also thank the reviewers for their valuable feedback. This research was supported by the NWO Innovational Research Incentives Scheme Vidi (016.Vidi.189.039), the NWO Smart Culture - Big Data / Digital Humanities (314-99-301), and the H2020-EU.3.4. - SOCIETAL CHALLENGES - Smart, Green And Integrated Transport (814961). All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

REFERENCES

[1] Gianni Amati and Cornelis Joost Van Rijsbergen. 2002. Probabilistic models of information retrieval based on measuring the divergence from randomness. *ACM Transactions on Information Systems (TOIS)* 20, 4 (2002), 357–389.

[2] Jaime Arguello, Adam Ferguson, Emery Fine, Bhaskar Mitra, Hamed Zamani, and Fernando Diaz. 2021. Tip of the Tongue Known-Item Retrieval: A Case Study in Movie Identification. In *Proceedings of the 2021 Conference on Human Information Interaction and Retrieval* (Canberra ACT, Australia) *(CHIIR '21)*. Association for Computing Machinery, New York, NY, USA, 5–14. https://doi.org/10.1145/3406522.3446021

[3] Leif Azzopardi, Maarten de Rijke, and Krisztian Balog. 2007. Building Simulated Queries for Known-Item Topics: An Analysis Using Six European Languages. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Amsterdam, The Netherlands) *(SIGIR '07)*. Association for Computing Machinery, New York, NY, USA, 455–462. https://doi.org/10.1145/1277741.1277820

[4] Michael K Buckland. 1979. On types of search and the allocation of library resources. *Journal of the American Society for Information Science* 30, 3 (1979), 143–147.

[5] David Elsweiler, Morgan Harvey, and Martin Hacker. 2011. Understanding Re-Finding Behavior in Naturalistic Email Interaction Logs. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Beijing, China) *(SIGIR '11)*. Association for Computing Machinery, New York, NY, USA, 35–44. https://doi.org/10.1145/2009916.2009925

[6] David Elsweiler, David E. Losada, José C. Toucedo, and Ronald T. Fernandez. 2011. Seeding Simulated Queries with User-Study Data for Personal Search Evaluation. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Beijing, China) *(SIGIR '11)*. Association for Computing Machinery, New York, NY, USA, 25–34. https://doi.org/10.1145/2009916.2009924

[7] Matthias Hagen, Daniel Wägner, and Benno Stein. 2015. A Corpus of Realistic Known-Item Topics with Associated Web Pages in the ClueWeb09. In *Advances in Information Retrieval*, Allan Hanbury, Gabriella Kazai, Andreas Rauber, and Norbert Fuhr (Eds.). Springer International Publishing, Cham, 513–525.

[8] Claudia Hauff, Matthias Hagen, Anna Beyer, and Benno Stein. 2012. Towards Realistic Known-Item Topics for the ClueWeb. In *Proceedings of the 4th Information Interaction in Context Symposium* (Nijmegen, The Netherlands) *(IIIX '12)*. Association for Computing Machinery, New York, NY, USA, 274–277. https://doi.org/10.1145/2362724.2362773

[9] Claudia Hauff and Geert-Jan Houben. 2011. Cognitive Processes in Query Generation. In *Advances in Information Retrieval Theory*, Giambattista Amati and Fabio Crestani (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 176–187.

[10] Ben He and Iadh Ounis. 2005. Term Frequency Normalisation Tuning for BM25 and DFR Models. In *Advances in Information Retrieval*, David E. Losada and Juan M. Fernández-Luna (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 200–214.

[11] Ida Kathrine Hammeleff Jørgensen and Toine Bogers. 2020. *"Kinda like The Sims... But with Ghosts?": A Qualitative Analysis of Video Game Re-Finding Requests on Reddit.* Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3402942.3402971

[12] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Online, 6769–6781. https://doi.org/10.18653/v1/2020.emnlp-main.550

[13] Jinyoung Kim and W. Bruce Croft. 2009. Retrieval Experiments Using Pseudo-Desktop Collections. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management* (Hong Kong, China) *(CIKM '09)*. Association for Computing Machinery, New York, NY, USA, 1297–1306. https://doi.org/10.1145/1645953.1646117

[14] Jinyoung Kim and W. Bruce Croft. 2010. Ranking Using Multiple Document Types in Desktop Search. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Geneva, Switzerland) *(SIGIR '10)*. Association for Computing Machinery, New York, NY, USA, 50–57. https://doi.org/10.1145/1835449.1835461

[15] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[16] Jin Ha Lee, Allen Renear, and Linda C Smith. 2006. Known-item search: Variations on a concept. *Proceedings of the american society for information science and technology* 43, 1 (2006), 1–17.

[17] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[18] Craig Macdonald and Nicola Tonellotto. 2020. Declarative Experimentation in Information Retrieval using PyTerrier. *CoRR* abs/2007.14271 (2020). arXiv:2007.14271 https://arxiv.org/abs/2007.14271

[19] Vassilis Plachouras, Ben He, and Iadh Ounis. 2004. University of Glasgow at TREC 2004: Experiments in Web, Robust, and Terabyte Tracks with Terrier.. In *TREC*.

[20] Stephen Robertson and Hugo Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Found. Trends Inf. Retr.* 3, 4 (April 2009), 333–389. https://doi.org/10.1561/1500000019

[21] Sargol Sadeghi, Roi Blanco, Peter Mika, Mark Sanderson, Falk Scholer, and David Vallet. 2014. Identifying Re-Finding Difficulty from User Query Logs. In *Proceedings of the 2014 Australasian Document Computing Symposium* (Melbourne, VIC, Australia) *(ADCS '14)*. Association for Computing Machinery, New York, NY, USA, 105–108. https://doi.org/10.1145/2682862.2682867

[22] Jaime Teevan, Eytan Adar, Rosie Jones, and Michael A. S. Potts. 2007. Information Re-Retrieval: Repeat Queries in Yahoo's Logs. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Amsterdam, The Netherlands) *(SIGIR '07)*. Association for Computing Machinery, New York, NY, USA, 151–158. https://doi.org/10.1145/1277741.1277770

[23] Mengting Wan and Julian McAuley. 2018. Item Recommendation on Monotonic Behavior Chains. In *Proceedings of the 12th ACM Conference on Recommender Systems* (Vancouver, British Columbia, Canada) *(RecSys '18)*. Association for Computing Machinery, New York, NY, USA, 86–94. https://doi.org/10.1145/3240323.3240369

[24] Mengting Wan, Rishabh Misra, Ndapa Nakashole, and Julian McAuley. 2019. Fine-Grained Spoiler Detection from Large-Scale Review Corpora. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, 2605–2610. https://doi.org/10.18653/v1/P19-1248

[25] Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Min Zhang, and Shaoping Ma. 2020. RepBERT: Contextualized Text Embeddings for First-Stage Retrieval. *ArXiv* abs/2006.15498 (2020).