Towards Question-based High-recall Information Retrieval: Locating the Last Few Relevant Documents for Technology-assisted Reviews

JIE ZOU and EVANGELOS KANOULAS, University of Amsterdam, The Netherlands

While continuous active learning algorithms have proven effective in finding most of the relevant documents in a collection, the cost for locating the last few remains high for applications such as Technology-assisted Reviews (TAR). To locate these last few but significant documents efficiently, Zou et al. [2018] have proposed a novel interactive algorithm. The algorithm is based on constructing questions about the presence or absence of entities in the missing relevant documents. The hypothesis made is that entities play a central role in documents carrying key information and that the users are able to answer questions about the presence or absence of an entity in the missing relevance documents. Based on this, a Sequential Bayesian Search-based approach that selects the optimal sequence of questions to ask was devised. In this work, we extend Zou et al. [2018] by (a) investigating the noise tolerance of the proposed algorithm; (b) proposing an alternative objective function to optimize, which accounts for user "erroneous" answers; (c) proposing a method that sequentially decides the best point to stop asking questions to the user; and (d) conducting a small user study to validate some of the assumptions made by Zou et al. [2018]. Furthermore, all experiments are extended to demonstrate the effectiveness of the proposed algorithms not only in the phase of abstract appraisal (i.e., finding the abstracts of potentially relevant documents in a collection) but also finding the documents to be included in the review (i.e., finding the subset of those relevant abstracts for which the article remains relevant). The experimental results demonstrate that the proposed algorithms can greatly improve performance, requiring reviewing fewer irrelevant documents to find the last relevant ones compared to state-of-the-art methods, even in the case of noisy answers. Further, they show that our algorithm learns to stop asking questions at the right time. Last, we conduct a small user study involving an expert reviewer. The user study validates some of the assumptions made in this work regarding the user's willingness to answer the system questions and the extent of it, as well as the ability of the user to answer these questions.

CCS Concepts: • **Information systems** → **Information retrieval**; **Users and interactive retrieval**; *Environment-specific retrieval*;

Additional Key Words and Phrases: Technology-assisted reviews, SBSTAR, SBSTAR$_{\text{ext}}$, interactive search, asking questions

This article is an extended version of Zou et al. [2018].
This research was supported by the China Scholarship Council, the European Union, under Project No. H2020-EU.3.4. Societal Challenges—Smart, Green, and Integrated Transport (Grant No. 814961), the Google Faculty Research Awards program, and the Netherlands Organisation for Scientific Research (NWO), under Projects No. 016.Vidi.189.039 and No. 314-99-301.
Authors' addresses: J. Zou and E. Kanoulas, University of Amsterdam, Science Park 904, 1098 XH Amsterdam, The Netherlands; emails: {j.zou, e.kanoulas}@uva.nl.

1046-8188/2020/05-ART27 $15.00
https://doi.org/10.1145/3388640

ACM Reference format:
Jie Zou and Evangelos Kanoulas. 2020. Towards Question-based High-recall Information Retrieval: Locating the Last Few Relevant Documents for Technology-assisted Reviews. *ACM Trans. Inf. Syst.* 38, 3, Article 27 (May 2020), 35 pages.
https://doi.org/10.1145/3388640

1 INTRODUCTION

Technology-assisted Reviews (TAR) aims at locating all relevant documents in a collection while minimizing the manual effort required to review irrelevant documents. Successful applications of TAR include electronic discovery in legal proceedings [Cormack and Grossman 2014; Oard et al. 2018], systematic reviews in evidence-based medicine [O'Mara-Eves et al. 2015], and test collections construction in Information Retrieval (IR) evaluation [Cormack and Grossman 2018; Sanderson and Joho 2004]. A significant research question in TAR is how to minimize the human effort required to review irrelevant documents while finding (nearly) all relevant documents, given the cost of assessing a large document collection is high, especially when the assessors are domain experts or information specialists. To reduce this cost, TAR is typically performed as a three-phase process: (a) a Boolean query is carefully designed by an information specialist expressing what constitutes relevant information to search for in a document corpus,[1] (b) potentially relevant documents are identified within the result set of the Boolean query, by experts examining only a summary of these documents (typically the title and the abstract), and (c) the documents to be included in the review are located by experts reading the full text of the document that corresponds to the relevant abstracts. The most expensive phase in this process is the second one, the screening of titles and abstracts, since the Boolean query typically returns a rather large dataset, in the order of thousands.

Active learning techniques, which iteratively improve the prediction accuracy by interacting with the reviewers, are considered the state-of-the-art in TAR [O'Mara-Eves et al. 2015]. In particular, Cormack and Grossman [2014, 2017] have proposed a Continuous Active Learning (CAL) algorithm, called Baseline Model Implementation (BMI), which achieves the best performance in a number of high-recall tasks [Grossman et al. 2016; Kanoulas et al. 2017; Zou et al. 2018]. BMI repeatedly trains a logistic regression model to predict the relevance of documents. In every session, BMI returns the top-scored documents to users (i.e., expert reviewers) to review and label. Then these labeled documents are added to the training dataset to re-train the logistic regression model. To speed-up the process of re-training, instead of a single document, a batch of documents is returned to the user at every iteration and the batch size increases exponentially with the iterations [Cormack and Grossman 2014, 2017]. In this way, if E is the number of labeled documents at the end of the process, then the number of iterations, and hence re-training steps, is $O(logE)$. While CAL algorithms have shown to be effective in finding relevant documents in a collection [Cormack and Grossman 2014; Grossman et al. 2017], the percentage of relevant documents identified typically reaches a plateau at 80%–90% of all relevant documents in the collection. This often happens after reviewing and labeling 20%–40% of the collection [Kanoulas et al. 2017; Suominen et al. 2018]. This is illustrated in Figure 1, which presents recall as a function of the number of documents manually reviewed in the CLEF 2017 e-Health Lab [Kanoulas et al. 2017]. Finding the remaining 10% of relevant documents (typically 1 to 3 relevant documents) needs reviewing almost the entire collection.

[1] In some cases, during the first phase a handful of relevant documents is also available; we do not consider this case in this work.

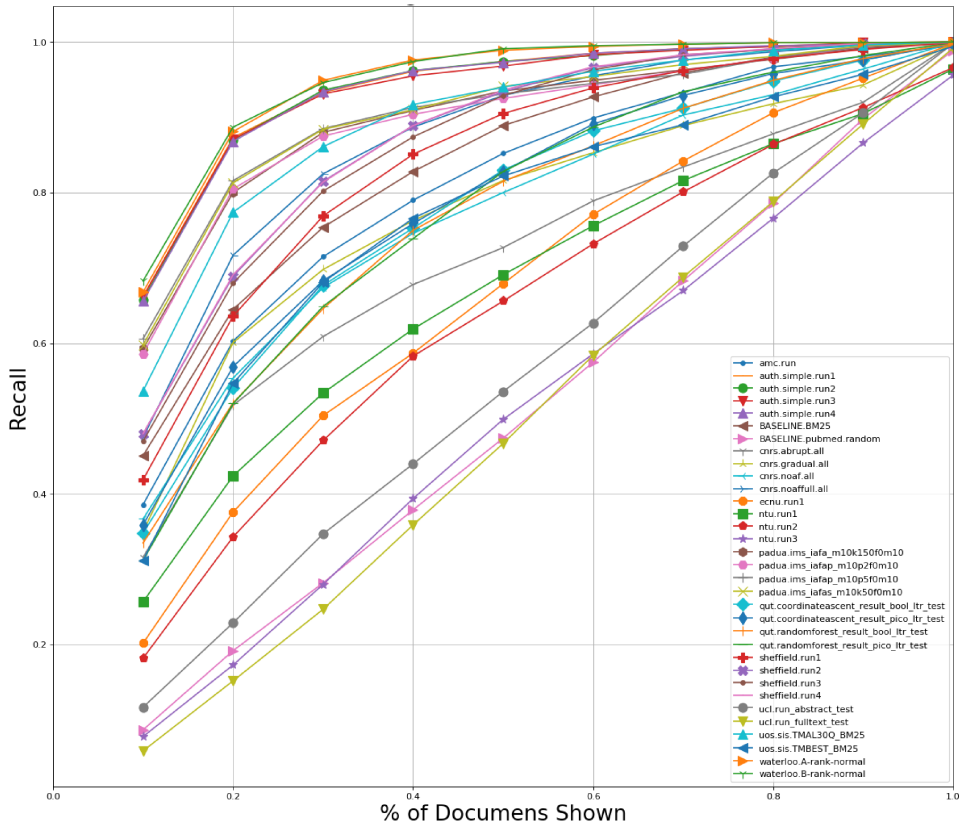


Fig. 1. Recall achieved by participating runs at CLEF 2017 e-Health task at different percentages of reviewed documents.

To overcome the above challenge, Zou et al. [2018] aim to retrieve these last few missing relevant documents by asking direct questions to reviewers about the information carried in the missing documents, instead of requesting relevance feedback on them. Zou et al. [2018] propose a Sequential Bayesian Search-based method [Wen et al. 2013] for TAR, called SBSTAR. SBSTAR applies CAL up to a certain number of documents reviewed, e.g., 20%–40% of the collection. Then, it switches to directly asking yes/no questions to reviewers focusing on questions about the expected presence of an entity in the missing relevant documents. In their work, TAGME is used to identify entities; therefore, an entity is defined as a sequence of informative terms (also called spots) in the input text [Ferragina and Scaiella 2010]. SBSTAR works by constructing a prior belief over document relevance on the basis of the ranking model learned by CAL. Based on the prior belief, it applies Generalized Binary Search (GBS) over entities to find the optimal entity, i.e., the one that dichotomizes the probability mass of the modeled document relevance, to ask to the reviewer. After the question is being answered by the reviewer, a posterior belief is obtained to be used for the selection of the next question or the final ranking of documents.

SBSTAR in Zou et al. [2018] made the assumption that the reviewers always provide the correct answer to the asked question, which is a strong assumption. Further, Zou et al. [2018] made some assumptions that the reviewers are willing to answer a number of questions and the effort

Table 1. A Running Example

Topic: Mini-Cog for the diagnosis of Alzheimer's disease dementia and other dementias within a community setting
Query: mini-Cog OR (MCE and (cognit* OR dement* OR screen* OR Alzheimer*))
Already found relevant documents by CAL:
ID: 22508578, ID: 17567931, ID: 16534774, ID: 15877567, ID: 14511167, ID: 11113982
Target document (missing relevant document):
ID: 20473827, Title: [Evaluation] of the [Functional Activities Questionnaire (FAQ)] in [cognitive screening] across four [American] [ethnic groups]. Abstract: The purpose of this [study] was to examine the performance of the [Functional Activities Questionnaire (FAQ)] in four [American] [ethnic groups] (N = 691), evaluate the influence of [demographic factors] and [depressive symptoms] on the [FAQ] and compare its performance with two [cognitive screening] [measures], the Mini-Cog and the [MMSE] ...
Asked questions from question pool:
Question: Are the documents about [study] (0.09) ? Answer: Yes/No/Not Sure
Question: Are the documents about [patient] (0.01)? Answer: Yes/No/Not Sure
Question: Are the documents about [cognitive screening] (0.15)? Answer: Yes/No/Not Sure
Question: Are the documents about [evaluation] (0.18)? Answer: Yes/No/Not Sure
Question: Are the documents about [dementia] (0.30)? Answer: Yes/No/Not Sure
Question: Are the documents about [medication] (0.29)? Answer: Yes/No/Not Sure
Question: Are the documents about [clinic] (0.29)? Answer: Yes/No/Not Sure
...

The topic describes the topic of the systematic review conducted. The Query reflects the Boolean query designed for this systematic review. The documents with IDs 22508578, 17567931, 16534774, 15877567, 14511167, 11113982 are relevant documents found by the CAL algorithm. In this example there is a single missing document, which is shown next. This is the document that our interactive algorithm will attempt to locate. Each candidate document, that is each document in the set of returned documents by the Boolean query, which have not been reviewed yet by the reviewer, is annotated by an entity recognition algorithm. The entities recognized in the missing document are shown in the example indicated by blue square brackets. The question pool is then constructed by using all of the entities in all the candidate documents. For example, the entity [study] appears in the missing relevant document, while [medication] does not, but instead appears in some irrelevant candidate documents. For each question round, we select the question from the question pool to ask to the user. We do that by selecting the question with the lowest score calculated by our objective function. In our example, in the first round, we selected the question "Are the documents about [study]?," which received a score of 0.09 and it was the lowest score in that round, while in the second round, we chose the question "Are the documents about [patient]?," which had a score of 0.01 and this was the lowest score in that round.

for answering a direct question about entities are at most as much as providing the relevance of a document, which remain unverified. Last, in SBSTAR of Zou et al. [2018] the number of questions to be asked is a predefined parameter. Setting this number depends on human experience, previous empirical results, or manual search (e.g., grid search) in a cross validation setup, which is costly, not flexible, and typically not optimal. In this work, we extend the work of Zou et al. [2018] in three ways: (a) we incorporate the chance that user may erroneously answer the system questions to our previously introduced SBSTAR model; (b) conduct a small user study to validate some of the assumptions in Zou et al. [2018]; and (c) we propose the $\text{SBSTAR}_{\text{ext}}$ algorithm, which determines when to stop asking questions on-line and automatically without the burden of predefining this parameter. The algorithm continues asking questions until a stopping criterion is met. The stopping criterion is set by our trained classifier based on dynamically extracted features. Once the prediction by the trained classifier is to "stop asking," the posterior belief is used to produce the final relevant documents list.

To sum up, this work extends Zou et al. [2018] in the following directions[2]:

- **E1** We propose three rudimentary models of reviewers' noisy answers when answering the generated questions. (See Section 4.1—Simulating reviewers.)
- **E2** We conduct an analysis on the basis of user simulations on the noise tolerance of the algorithm by Zou et al. [2018]. (See Section 4.4.)
- **E3** We propose a new objective function that accounts for the user's erroneous answers when selecting the next question to ask, and demonstrate its effectiveness. (See Sections 3.2 and 4.4.)
- **E4** We propose $\text{SBSTAR}_{\text{ext}}$ with a novel method to decide when to stop asking questions, and demonstrate its effectiveness. (See Sections 3.3 and 4.5.)
- **E5** All experiments are run both against abstract-level relevance and document-level relevance, with the latter being a harder problem, since relevant documents are only a fraction of the relevant abstracts. (See Section 4.)
- **E6** We conduct a small user study to validate the assumptions made regarding the users' willingness to answer a number of questions, their efforts, and their noisy answers. (See Section 4.6.)

The rest of this article is organized as follows. In Section 2, we summarize the related work. In Section 3, we introduce our approach and describe it in detail. Section 4 includes the experimental setup, the experimental results, and the corresponding analysis. The limitations and future work are presented in Section 5, while Section 6 concludes the article.

2 RELATED WORK

2.1 Technology-assisted Reviews

A Technology-assisted Review (TAR) aims at locating as many relevant documents as possible in a collection, i.e., a high-recall task. The Text Retrieval Conference ("TREC") [Voorhees et al. 2005] has a history of studying the problem of high recall, starting with the TREC Legal track [Baron et al. 2006; Cormack et al. 2010; Hedin et al. 2009; Oard et al. 2008; Tomlinson et al. 2007], followed by the TREC Total-recall track [Grossman et al. 2016; Roegiest et al. 2015], which received a lot of attention in 2015 and 2016. Then, CLEF [Ferro and Peters 2019] focuses on the total-recall problem of TAR in empirical medicine [Kanoulas et al. 2017; Suominen et al. 2018]. BMI [Cormack and Grossman 2014], an AutoTAR CAL method, was provided to the participants of the Total-recall track as a baseline for comparison. No method evaluated in these tracks outperformed BMI, and thus BMI is recognized as the state-of-the-art [Grossman et al. 2016; Roegiest et al. 2015; Zhang et al. 2015]. It is a relevance-feedback method using supervised machine learning. Cormack and Grossman [2014] found CAL outperforms traditional supervised learning (i.e., "simple passive learning" (SPL)) and active learning (i.e., "simple active learning" (SAL)). Yu et al. [2016] also confirmed that CAL is effective for systematic reviews. Abualsaud et al. [2018] designed and implemented an efficient high-recall information retrieval system using CAL. CAL [Cormack and Grossman 2014] is an active learning method, which uses search as a first step to identify an initial set of potentially relevant and irrelevant documents for training a classifier, then presents a set of documents most likely to be relevant to the user, and solicits their feedback on their relevance. Then it is using these labeled documents as a training set to re-train the classifier. In detail, CAL first (1) creates a (set of) potentially relevant and irrelevant document(s); this can be done by different means; past work used the description of the TAR topic as a relevant document and a sample

[2]The source code is released in the following repository: https://github.com/JieZouIR/TAR.git.

of 100 documents from the collection as the set of irrelevant documents; then (2) trains a machine learning algorithm (e.g., a Support Vector Machine (SVM), or logistic regression) on this training set and uses it to predict the next most-likely relevant documents; typically a batch of documents is returned at every iteration with the batch size increasing exponentially with the iterations to speed-up the re-training process [Cormack and Grossman 2014, 2017]; (3) collects the relevance feedback for all of the presented documents, and amends the training set; and (4) repeats (2) and (3) until some stopping criterion is met, e.g., none of the presented documents is relevant. Despite its effectiveness, the method suffers from locating the last few relevant documents [Kanoulas et al. 2017; Zou et al. 2018].

Evaluation and comparative studies around TAR methods have also attracted the attention of researchers. Zhang et al. [2018c] used a simulation framework to evaluate sentence-level relevance feedback. Zhang et al. [2018a] conducted a controlled user study with 50 users to evaluate a retrieval system using the full document or selected paragraph as relevance feedback in CAL. McDonald et al. [2018] presented an evaluation of active learning strategies for sensitivity reviews. The evaluation of user-in-the-loop systems has also been studied using human subjects and simulated human responses [Cormack and Mojdeh 2009; Soboroff and Robertson 2003; SPARK-JONES 1975]. Grossman et al. [2017] performed a comparative study on automatic and semi-automatic document selection for the TREC 2016 Total-recall track.

The summary or abstract of documents has been shown to be an effective information source for accurate and efficient relevance judgments. Tombros and Sanderson [1998] found reviewers could locate more relevant documents by reviewing the extracted summary, while making fewer labeling errors. Further, Sanderson [1998] found that "reviewers can judge the relevance of documents from their summary almost as accurately as if they had access to the document's full text." They showed that an assessor took 61 seconds to assess each full document while spending 24 seconds to assess each summary on average. Zhang et al. [2018c] also suggested that a system that presents relevant sentences could reach high recall more efficiently compared to a system that presents the entire document. Systematic reviews also use article abstracts to filter out irrelevant articles efficiently.

A number of works that aim at deciding when to stop reviewing articles in TAR have been presented in the past. Wallace et al. [2010] and Yu et al. [2018] stop training their models when a pre-defined number of relevant studies is found. Di Nunzio [2018] proposed a variable threshold approach to stop labeling once the percentage of non-relevant documents over the total number of judged documents reaches a fixed threshold. The most recent approach proposed, called the "knee" method [Cormack and Grossman 2015b, 2016a, 2016b, 2017], uses a simple geometric criterion [Satopaa et al. 2011] to make a stopping decision. The "knee" method uses the fall-off in the slope of the gain curve (number of judged relevant documents vs. review effort) as a stopping criterion. However, the "knee" method is a heuristic method and does not indicate how many missing relevant documents are there. Cormack and Grossman [2016c] proposed the SCAL method, which first estimates the number of relevant documents R in a collection by randomly sampling and labeling a large subset of the documents. Di Nunzio [2018] proposed a heuristic thresholding method, the two-dimensional BM25, based on the interaction of the two probabilities used by the BM25 model: $P(d|R)$ and $P(d|NR)$—the probability of observing document d given the currently judged relevant documents R, and the currently judged non-relevant documents NR. Their method stops judging once the proportion of non-relevant documents over the total number of judged documents exceeds a fixed value. There are more studies that attempt to estimate the number of relevant documents, R, based on which one can decide when to stop presenting documents to the user of a search system [Arampatzis et al. 2009; Losada et al. 2019; Wallace et al. 2013]. Arampatzis et al. [2009] proposed methods to select the cut-off point where to stop reading a ranked List that optimizes a given evaluation metric. Losada et al. [2019] proposed a diversified group of stopping

methods and proposed a method to estimate the number of relevant documents R. They estimate R based on power-law distribution and the similarity between the pattern of the relevance of the test query and the pattern of the relevance of each training query. All the aforementioned methods need extra assessment budget to estimate the number of relevant documents R to decide when to stop the TAR process. Our SBSTAR$_{\text{ext}}$ method differs by (1) training a classifier based on dynamic features, and (2) automatically determining when to *stop asking questions* (3) without relying on the predefined threshold or evaluation metrics.

2.2 Interactive Search

Interactive Information Retrieval has always received significant attention in the research community [Buckley and Robertson 2008; Chai et al. 2007; Ruotsalo et al. 2018]. Compared with the traditional approaches, interactive information retrieval achieves high recall with a human-in-the-loop. It suggests putting the human-in-the-loop and learning a relevance model throughout an interactive search process, where users provide feedback on the relevance of presented documents, and the model adapts to this feedback [Cormack and Grossman 2015a; Grossman and Cormack 2010]. TREC firstly introduced the "pooling method" [SPARK-JONES 1975], which selects the top-ranked documents for assessment, and recognizes all other documents as non-relevant. Then Interactive Searching and Judging (ISJ) method has been shown to get comparable quality relevance to the pooling method with considerably less effort. ISJ utilizes the form of repeatedly formulating queries and examines the top results of a relevance-ranking search engine [Cormack et al. 1998]. Most of the methods take special treatment of the query [Lavrenko and Croft 2017; Robertson and Jones 1976; ROCCHIO 1971; Salton and Buckley 1990; Zhai and Lafferty 2001], typically expanding it with terms from labeled documents. However, query expansion has shown suboptimal performance [Cormack and Grossman 2014], in part, because handling the relationship between the original query and feedback documents is challenging [Lv and Zhai 2009]. Quantifying relevance on the basis of users' queries, or learning a model of relevance from past queries, cannot always capture the minute details of relevance [Cormack and Grossman 2014; Grossman and Cormack 2010; Zou et al. 2018]. Active learning [Krishnakumar 2007] and multi-armed bandits [Dudík et al. 2015; Hofmann et al. 2013; Radlinski et al. 2008] have been proposed to iteratively learn task-specific models; however, they both suffer from the multi-modality of relevance. Cormack and Mojdeh [2009] proposed a combination of ISJ and CAL. Cormack and Grossman [2015a] proposed a continuous learning-based algorithm, BMI, for TAR, which works by iteratively training an SVM classifier on user's relevance feedback over the predicted most relevant documents. The proposed algorithm is considered state-of-the-art. However, the human effort remains extremely high trying to find the last few relevant documents [Kanoulas et al. 2017; Zou et al. 2018]. Different from the aforementioned methods, which focus on receiving feedback at the level of documents, our interactive method asks explicit questions to the users in terms of entities contained in the documents of the collection.

Interactive retrieval methods have also been studied in community-based question answering (cQA). Successful applications in the field include expert finding [Zhao et al. 2014, 2016], question retrieval [Bae and Ko 2019; Chen et al. 2018], understanding and summarizing answers [Liu et al. 2008], question routing in providing answers for unanswered questions [Li and King 2010], and inference rules discovery from text [Lin and Pantel 2001]. Zhao et al. [2016] proposed a random-walk-based learning method with recurrent neural networks from a novel viewpoint of learning ranking metric embeddings to search the right experts for answering the questions. To solve the problem of lexical gaps between questions, Chen et al. [2018] presented a model to retrieve similar questions for cQA platforms to resolve users' queries by applying random walk with a recurrent neural network. They highlight a valuable investigation for considering

both question contents and the asker's social interactions. Bae and Ko [2019] instead presented a translation-based language model to solve the lexical gap problem for retrieving questions. To solve the cold-start problems, Wan et al. [2018] and Zhao et al. [2014] exploited knowledge from multiple sources to support question answering. A hybrid system to retrieval expertise to help to answer questions is also proposed [Kundu and Mandal 2019]. Liu et al. [2008] suggested that users can reuse the best answers from similar questions as search result snippets, and highlighted the effectiveness of applying automatic summarization techniques to summarize answers. Li and King [2010] introduced the concept of Question Routing to retrieve suitable questions to the right answerers to answer. They proposed a Question Routing framework by considering not only users' expertise but also the availabilities of users for providing answers. To assist with the mismatch between different expressions in questions and texts, Lin and Pantel [2001] proposed an unsupervised method to retrieve inference rules from question answering text. Different from the aforementioned works, which focus on retrieving related questions, answers, answerers, or inference rules for question answering, we focus on the TAR task by asking "yes" or "no" questions to reviewers to locate the missing relevant documents in this article.

Similar to our work, Wen et al. [2013] proposed a Sequential Bayesian Search algorithm for solving the problem of efficiently asking questions in an interactive search setup. They learn a policy that finds items in a collection using the minimum number of queries. Then Kveton and Berkovsky [2015, 2016] proposed a generalized linear search (GLS) method to combine generalized search and linear search in recommender systems. The SBSTAR algorithm that we introduced in previous work [Zou et al. 2018] differs from the aforementioned work by being applied to unstructured text for the ranking of documents using entities to construct the questions to ask. Further, it updates the model after each question rather than updating by each episode for locating the target item. Moreover, the SBSTAR algorithm is used to find the last few relevant documents in TAR by asking "yes" or "no" questions to reviewers. This work extends our previous work [Zou et al. 2018], by incorporating the user's erroneous answers to the SBSTAR model by introducing a new objective function. Further, it provides an analysis on the basis of user simulations. For the user simulations three rudimentary noisy answer models are introduced. Also, we propose an extension of the algorithm, $\text{SBSTAR}_{\text{ext}}$, to decide when to stop asking questions automatically. Last, we conduct a small user study to validate the assumptions made regarding the users' willingness to answer a number of questions, their efforts, and their noisy answers.

2.3 Question Pool Construction

Entities are recognized as the most vital source of information in text [Erosheva et al. 2004; Rosen-Zvi et al. 2004]. Based on this assumption, Zou et al. [2018] construct the pool of questions using entity annotation, thus focusing on generating questions regarding the presence or absence of an entity in relevant documents. That is, Zou et al. [2018] instantiate the question candidate set by identifying entities in the related documents by using TAGME [Ferragina and Scaiella 2010], an entity-linking algorithm [Cornolti et al. 2018; Liu et al. 2017, 2019], which is widely used in prior research [Ernst et al. 2017; Hasibi et al. 2015; Park et al. 2013; Xiong and Callan 2015; Xiong et al. 2017; Zou et al. 2018]. No filters on annotation score (a confidence score for a word or phrase being annotated as an entity) are used for TAGME's results, i.e., all annotations are being considered, which is also a widely used setting in previous work [Raviv et al. 2016; Xiong and Callan 2015; Zou et al. 2018]. One can also easily combine TagME entities with labeled topics [Zou et al. 2015, 2016, 2017], keywords extraction [Campos et al. 2018], or other information extraction [Maslennikov and Chua 2010; Riloff and Lehnert 1994] for the annotations. In this article, as in Zou et al. [2018], we take a rudimentary approach by using TAGME to annotate entities in documents, and represent documents by embedding them in the entity space. An example of an annotated document can be

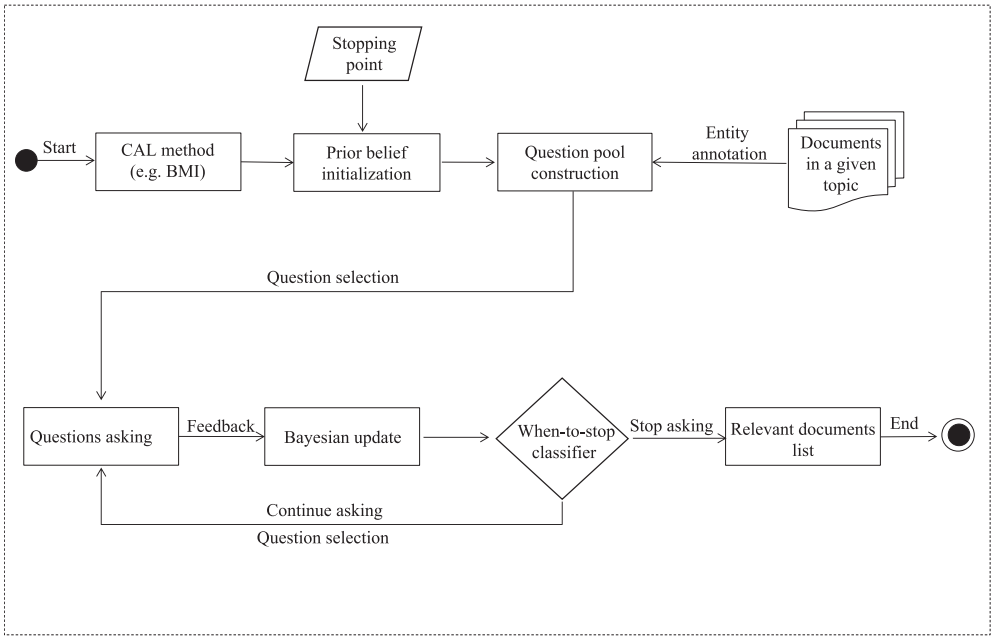


Fig. 2. Pipeline of our approach.

seen in Table 1. Each one of the entities annotated in this example document, as well as all other documents in the collection, is used to create a question pool. A subset of this pool can also be seen in Table 1. After that, the algorithm asks a sequence of questions in the form: "Are the documents about [entity]?" to find the reviewer's target documents. The question template here is ad hoc and one can also use other defined templates. The reviewers can respond with a "yes," a "no," or a "not sure," with "not sure" ensuring that the reviewers are not forced to give the wrong answer when they are not sure about it.

3 METHODOLOGY

In this section, we first describe in detail the SBSTAR algorithm introduced by Zou et al. [2018], and in particular the selection of questions asked to a reviewer in a sequential fashion (Section 3.1), given the constructed question pool by entity annotation (Section 2.3). Then, we present in detail the extensions to the SBSTAR algorithm and, in particular, (a) accounting for reviewers' noisy answers (Section 3.2) and (b) introducing an algorithm that decides when to stop asking questions to reviewers (Section 3.3).

The pipeline of our approach is shown in Figure 2, and a running example is provided in Table 1. In particular, while offline, our framework analyzes all the documents in the collection and generates a pool of clarifying questions to be asked to users. During search time, a user submits a query for a certain topic and a search algorithm responds with a ranked list of documents. In our setup, we employ interactive search, using a certain CAL algorithm, the BMI. BMI responds to the user's query with a short ranked list of documents, of a predefined size, and the user is requested to provide relevance feedback over each one of the returned documents. Once feedback is provided, the algorithm is trained over it and produces another ranked list of documents to be shown to the user, and so on. At a certain point, the BMI algorithm decides to stop returning documents, assuming

Table 2. Notations

Notation	Explanation
$\mathcal{D}, \mathcal{E}$	document set, entity set
$\mathcal{D}^*$	target document set, $\mathcal{D}^* \subseteq \mathcal{D}$
d, e_l	document $d \in \mathcal{D}$, selected entity in the l-th question $e_l \in \mathcal{E}$
$e_l(\mathcal{D}^*)$	the reviewer reply indicating whether or not the target documents contain the entity e_l
$\pi_l^*(d)$	probability distribution of reviewer preferences over the document d during the l-th question
$\mathbb{P}$	prior belief over the reviewer preferences π^*
α	the Dirichlet parameter of $\mathbb{P}$
N_q	the number of questions to be asked
U_l	the candidate document space during the l-th question
$Z_l(d)$	indicator function for model updating
$h(e)$	error rate of reviewer answers for the entity e
β	the tradeoff parameter for $h(e)$
$D_{training}, D_{testing}$	training set, testing set for the when-to-stop classifier
Max_q	the max number of questions to be asked for the when-to-stop classifier

that there is no other relevant document in the collection, or that the effort to find the last few relevant documents is too high. After this stopping point, our algorithm is run to ask clarifying questions. The BMI algorithm is based on a probabilistic classifier (logistic regression), hence at its stopping point a probability over relevance can be calculated for every document in the collection. This probability is used to construct a prior belief of the user's interest over all documents in the collection. Based on this prior belief, our model selects a clarifying question to ask to the user by picking the entity that best splits the probability mass of the user's interest over the document in the collection into two halves. The algorithm receives the user's answer and on the basis of this answer, the prior belief is updated to a posterior belief. This posterior belief is used now by our model to select the next query to ask to the user. In the meantime, a classifier is employed to decide whether indeed a next clarifying question should be asked to the user or the algorithm should produce the final ranked list of the remaining of the documents in the collection.

3.1 Sequential Bayesian Search for TAR

The SBSTAR algorithm introduced by Zou et al. [2018] is described in Algorithm 1. The notation used throughout the article is summarized in Table 2. The input to the algorithm is the document collection, $\mathcal{D}$, the set of annotated entities in the documents, $\mathcal{E}$, a prior belief, $\mathbb{P}_0$, which we model as a Dirichlet distribution parameterized by α, and the number of questions to be asked, N_q. The document set $\mathcal{D}$ is built by running a Boolean query against a biomedical article collection, e.g., PubMed, which constitutes the current approach taken by experts when working on systematic review [Kanoulas et al. 2017]. This document collection is further reduced by removing the documents that are already discovered by BMI and labeled by experts.

The algorithm assumes that there is a target document set $\mathcal{D}^* \subseteq \mathcal{D}$ the reviewer is interested in, which is the last few relevant documents missed by BMI (e.g., the single missing relevant document with ID #20473827 in our running example). We also assume there is a probability distribution modeling the preference of the reviewer over the documents, π^*, over $\mathcal{D}$, and the target documents are drawn i.i.d. from this distribution. Further, we assume that we have a prior belief $\mathbb{P}$ over the reviewer preferences π^*, which is a probability density function over all the possible realizations of π^*. The system updates its belief when the system observes a reviewer's answer of

ALGORITHM 1: SBSTAR [Zou et al. 2018]

input: A document set, $\mathcal{D}$, the set of annotated entities in the documents, $\mathcal{E}$, a prior belief over document relevance, $\mathbb{P}_0$, and a number of questions to be asked, N_q

foreach *topic* **do**
  $l \leftarrow 1$
  while $l \leq N_q$ **do**
    Compute the reviewer preference:
      $\pi_l^*(d) = \mathbb{E}_{\pi \sim \mathbb{P}_l}[\pi(d)]\ \forall d \in \mathcal{D}$
    Use GBS to find the optimal target entity:
      $e_l = \arg\min_e |\sum_{d \in \mathcal{D}} (2\mathbb{1}\{e(d) = 1\} - 1)\pi^*(d)|$
    Ask the question about e_l and observe the reply $e_l(\mathcal{D}^*)$
    Remove e_l from entity pool
    $l \leftarrow l + 1$
    Update the system's belief $\mathbb{P}_l$ using Bayes' rule:
      $\mathbb{P}_{l+1}(\pi) \propto \pi(d)\mathbb{P}_l(\pi)\ \forall \pi$
  end
end

a question, which is sampled i.i.d. from π^*. At each interaction round l, the reviewer preference $\pi_l^*(d)$ is calculated based on the system's prior belief $\mathbb{P}_l$ over π^*,

$$\pi_l^*(d) = \mathbb{E}_{\pi \sim \mathbb{P}_l}[\pi(d)] \quad \forall d \in \mathcal{D}. \tag{1}$$

Then, the algorithm uses GBS to find the entity, e_l, that best dichotomizes the probability mass of the predicted document relevance, we ask whether the entity e_l is present in the missing target documents that the user wants to find, observe the reply $e_l(\mathcal{D}^*)$, and remove e_l from the entity pool.

In this work, we consider two settings, regarding the user answers to the system questions. In the first setting, similar to Cormack and Grossman [2014], Cormack and Lynam [2005], Drucker et al. [2001], and Roegiest et al. [2015], we use the assumption that the human is infallible and will answer the questions correctly, i.e., he/she will respond with $e(d) = 1$ if the document d contains the entity, while $e(d) = 0$ if the document d does not. The entity in the question is from the entity pool, which is extracted from the corpus. The selection of entities is sequential, that is, we choose an entity to ask a question on taking into consideration all the previous entities chosen and the corresponding answers. This GBS strategy considers a generalized form of binary search based on the reviewer's preference on document relevance $\pi_l^*(d)$. It chooses the entity that is the most discriminative at each step, which is the one that can split the expected accumulated reviewer's preference $\pi_l^*(d)$ closest to two halves. In particular, it selects the entity with a minimal question selection score by the following objective function:

$$\left| \sum_{d \in \mathcal{D}} (2\mathbb{1}\{e(d) = 1\} - 1)\pi^*(d) \right|, \tag{2}$$

where $\{e(d) = 1\}$ is either 0 or 1, and thus the term of $(2\mathbb{1}\{e(d) = 1\} - 1)$ is either -1 or 1. In our running example, the first question the system asks is about "[study]," since its question selection score is 0.09 and it is the smallest among all question scores calculated in the first iteration. The reviewer preferences $\pi_l^*(d)$ will be updated by each question and answer, and so will the GBS-based question selection.

After that, the system's belief $\mathbb{P}_l$ is updated using Bayes' rule. The reviewer preference π^* is a multinomial distribution over documents $\mathcal{D}$; hence, we model the prior, $\mathbb{P}_0$, by the conjugate prior of the multinomial distribution, i.e., the Dirichlet distribution, with parameter α. In principle, the prior belief $\mathbb{P}_0$ based on α can be set by using any retrieval algorithm. In this work, the prior belief $\mathbb{P}_0$ is initialized as $Dir(\alpha)$, with the initial α computed by using the probability of a document being relevant provided by the CAL trained logistic regression; i.e., $\alpha(d) = Pr(d = rel), \forall d \in \mathcal{D}$. Further, we define the indicator vector $Z_l(d) = \mathbb{1}\{e_l(d) = e_l(\mathcal{D}^*)\}$, where $\mathcal{D}^*$ represents the target documents, and $e_l(\mathcal{D}^*)$ is 1 if e_l is present in all the documents in $\mathcal{D}^*$. Intuitively, $Z_l(d)$ is 1 if the entity e_l is both in the target documents and in d, or if it is neither in the target documents nor in d. From Bayes' rule, the posterior belief at the beginning of question l is

$$\mathbb{P}_l = Dir\left(\alpha + \sum_{j=0}^{l-1} Z_j\right). \tag{3}$$

From the properties of the Dirichlet distribution, then we have

$$\pi_l^*(d) = \mathbb{E}_{\pi\sim\mathbb{P}_l}[\pi(d)] = \frac{\alpha(d) + \sum_{j=0}^{l-1} Z_j(d)}{\sum_{d'\in\mathcal{D}}(\alpha(d') + \sum_{j=0}^{l-1} Z_j(d'))}, \tag{4}$$

where $\alpha(d)$ is the ith entry of α, which corresponds to document d. And thus the reviewer preference π_l^* can be updated by counting and re-normalization. After the last question is being asked, the system generates the relevance ranking list based on the reviewer preference $\pi_{N_q}^*$ over the documents in the collection that have not been presented by TAR.

3.2 Accounting for Noisy Answers

In Algorithm 1, Zou et al. [2018] make the assumption, that reviewers, when presented with an entity, know with 100% confidence whether the entity appears in the target documents. In this work, to relax this assumption, we propose a noise-tolerant version of the algorithm (**E3**). That is, we allow the user to make mistakes and provide the algorithm with wrong answers. In this work, we assume that user mistakes are related (in different ways) with the entity the question is asked upon, and we model this by $h(e)$, which model the probability that the user will give the wrong answer to a question about entity e. We integrate $h(e)$ into the new objective function, at line 7 of Algorithm 1:

$$e_l = \arg\min_e \left| \sum_{d\in\mathcal{D}} (2\mathbb{1}\{e(d) = 1\} - 1)\pi^*(d) \right| + 2\beta * h(e), \tag{5}$$

where β trades $h(e)$ with the probability mass. The $h(e)$ is defined in the range from 0 to 0.5, with the highest error rate of 0.5 means that the expert reviewer gives random answers. The first term that is on the left side of the plus sign, the GBS strategy term, ranges from 0 to 1. To ensure that both terms are in the same range, we multiply $h(e)$ by 2, which enhances clarity in case of manual inspection, even though it is not necessary. The precise definition of $h(e)$ will be provided in Section 4. After observing the noisy answer, we update the posterior system belief.

3.3 When to Stop Asking Questions

In Zou et al. [2018] it was observed that a number of questions are effective to identify the last few relevant documents. However, different queries and different stopping points of the CAL algorithm may require a different number of questions to be asked. In this section, we describe our question stopping method. Different from defining the number of questions in advance, we propose an SBSTAR extension algorithm SBSTAR$_{ext}$, which explores an automatic method of determining when to stop. In particular, SBSTAR$_{ext}$ trains a classifier based on a number of extracted features to

ALGORITHM 2: SBSTAR$_{\text{ext}}$

input: A document set, $\mathcal{D}$, the set of annotated entities in the documents, $\mathcal{E}$, a prior belief over document relevance, $\mathbb{P}_0$, and the max number of questions to be asked, Max_q

```
Training a classifier using training set:
  Classifier(D_training)
foreach topic ∈ D_testing do
    l ← 1
    Stop ← False
    while l ≤ Max_q and Stop = False do
        Compute the reviewer preference:
          π*_l(d) = E_{π∼P_l}[π(d)] ∀d ∈ D
        Use GBS to find the optimal target entity:
          e_l = arg min_e | Σ_{d∈U_l} (2𝟙{e(d) = 1} − 1)π*(d)|
        Ask the question about e_l and observe the reply e_l(D*)
        Remove e_l from entity pool
        Reduce the set of documents of candidate version space U_l:
          U_{l+1} = U_l ∩ i ∈ D : e_l(i) = e_l(D*)
        l ← l + 1
        Update the system's belief P_l using Bayes' rule:
          P_{l+1}(π) ∝ π(d)P_l(π)∀π
        Compute the features of Table 3
        Predict the label of trained classifier Pred_classifier using the computed features:
        Stop = Pred_classifier
    end
end
```

dynamically decide whether to stop or continue to ask questions, at every interactive round. After thinking which factors could affect the decision of when to stop asking questions, we define seven dynamic features including the CAL "Stopping point," the "# of yes/no questions" asked so far, whether the last question received an answer that "Is_yes/no," the "# of candidate(s)" documents left, the "difference in # of candidates," the "difference in (user) preference," and the "difference of top 1" ranked document.[3] The features we defined are shown in Table 3. The intuition behind using these features is the following: different stopping points for BMI may affect the number of missing documents and thus affect the number of questions to locate the last few relevant documents. "# of yes/no questions" and "Is_yes/no" also affect when to stop asking questions, since having asked many questions already may indicate that it is time to stop, while receiving a "not sure" answer as opposed to "yes" or "no" indicates that no further information was obtained in this last round. As for "# of candidate(s)" documents left and "difference in # of candidates," a smaller number of documents left or smaller difference in the number of candidates before and after asking a question indicates a reduction in the space of possible documents and hence hint higher chances to stop asking questions. "difference in (user) preference" and the "difference of top 1" measure

[3]An external experiment is conducted. We first define as many factors as possible that may affect when to stop asking questions, and then we filter out the factors that have very small or no influence for when to stop asking questions by performance experiments; at the end seven features are left.

Table 3. The Defined Features for Deciding When to Stop Asking Questions

Feature	Description
Stopping point	The percentage of documents reviewed through CAL
# of yes/no questions	The number of questions asked that got yes or no feedback from reviewer
Is_yes/no	Whether the last question asked got yes/no answer from reviewer or not
# of candidates	The number of documents in the current user space, split by GBS
difference in # of candidates	The reduction of the number of documents in the current user space compare with last user space
difference in preference	The change of user preference π^* in term of last user preference π^*
difference of top 1	Whether the highest ranked document change or not

the change in user preference and in the top document, respectively, and thus provide a signal of whether eliciting further user preferences may make a difference or not in the ranking and thus whether it is time to stop asking questions. This is by no means an exclusive set of useful features and more could be engineered; however, our experiments indicate that these features are effective enough. In this work, we use the SVM, random forests, and feedforward neural network classifiers, and we show the performance comparison results of them in Section 4.5.

The SBSTAR$_{\text{ext}}$ algorithm is presented in Algorithm 2. Let $\{(x_1, y_1), \ldots, (x_n, y_n)\}$ be a training data set, in which x_i denotes a question instance (each question is represented by a seven-feature vector) and $y_i \in \{\text{“}True\text{”}, \text{“}False\text{”}\}$ denotes a classification label. For each query (i.e., topic) in the training space $D_{training}$, which contains all of the training documents for this query, we sequentially ask 100 questions (on the basis of the SBSTAR algorithm), and label each question as “True” or “False.” “True” means stop asking while “False” means continue asking. To decide whether to label a question as “True” or “False,” we look at the ranking of documents produced by the SBSTAR algorithm after each question is being asked. There is a point (question) after which the position of the target relevant documents does not change anymore. All questions up to that point are labeled as “False,” while all the following up to 100 questions are marked as “True.” That means we have 100 question instances and assign 100 labels for each topic-stopping point pair.

We first train a when-to-stop classifier using the training set. During testing time, for each document in $D_{testing}$, we first compute the user preference using the belief $\mathbb{P}_l$, and find the optimal entity e_l that best splits the probability mass of the predicted document relevance. We ask whether the entity e_l is present in the missing target documents that the user wants to find, observe the reply $e_l(\mathcal{D}^*)$, and remove e_l from the entity pool. Then, we reduce the candidate document space, $U_l \subseteq \mathcal{D}$, and update the system's belief $\mathbb{P}_{l+1}$ using Bayes' rule. After that, we compute the features of Table 3 and predict the output of the when-to-stop classifier by using the pre-trained classifier model. If the output of the trained classifier model using the dynamically computed features is “False,” then the SBSTAR$_{\text{ext}}$ system continues by selecting the next question to ask and updates the posterior. If the output of the trained classifier model is “True,” then we stop asking questions and generate the final recommended relevant documents ranked list.

4 EXPERIMENTS AND ANALYSIS

In this work, we attempt to answer the following research questions:

RQ1 How does the stopping point of CAL, as well as the number of questions asked by SBSTAR affect the performance of the algorithm?

SBSTAR is affected by two parameters: (a) the stopping point of the CAL algorithm; if CAL stops too early it may be harder to locate all the remaining documents; and (b) the number of questions asked by SBSTAR. We will explore the effect on the model performance of varying stopping points and the number of asked questions.

RQ2 How effective is SBSTAR in finding the remaining relevant documents compared to the baselines?

We compare the SBSTAR model with three different baselines (see Section 4.1 in detail). This research question is used to confirm the effectiveness of the SBSTAR model, and investigate the extent to which the SBSTAR model outperforms state-of-the-art methods. Both **RQ1** and **RQ2** were investigated by Zou et al. [2018]. In this work, we extend all the experiments including both abstract-level and document-level relevance. Finding relevant documents is a harder problem compared to finding relevant abstracts, since the former are only a fraction of the latter.

RQ3 What is the influence of noisy answers over the SBSTAR performance?

We investigate the effect of reviewers' noisy answers on the performance of our algorithm. We consider three noise settings: one with a fixed error rate for all of the questions (entities), one for which the error rate is a function of the term frequency of the entity, and one is defined on the basis of target documents. This research question explores the robustness of our SBSTAR model to noise.

RQ4 Is our proposed method for deciding when to stop effective?

Previous work needed to define the number of asked questions as an input parameter to the algorithm. We investigate how dynamically deciding when to stop asking questions performs.

4.1 Experimental Setup

Dataset. The dataset used in the experiments is the collection released by the Technological-assisted Reviews in Empirical Medicine Task of The CLEF 2017 e-Health Lab[4] [Goeuriot et al. 2017; Kanoulas et al. 2017], which is also well adopted by other works [Cormack and Grossman 2017; Di Nunzio 2018; Lee and Sun 2018; Scells et al. 2018]. The collection contains 50 topics, and 266,967 abstracts of MEDLINE articles identified by PMID, and the relevance judgments for each of these articles against the 50 topics, both at an abstract and at a document level. Each topic file is in a text format and contains four sections: topic ID, topic title, the query, and a list of PMIDs of documents. The query corresponds to the Boolean query used to obtain the PMIDs relevant to the given topic, which need to be re-ranked. Each document file linked by PMID is in the XML format and contains the titles, abstracts, and metadata for an article. First, we use the java SAX parser to extract the title and abstract of XML files. After removing documents without abstract, we ended up with 221,654 documents. The remaining preprocessing steps include tokenization, elimination of stop-words, stemming and case unification. For each topic the relevant documents were also provided. In systematic reviews there are typically two levels of relevance judgments. The first is at abstract level: the expert submits a Boolean query and examines the titles and abstracts of the returned set, judging whether these returned abstracts summarize potentially relevant articles. That is, the expert provides a relevance label for each article ID returned by the query based on the abstract. The second is at full text level, where the full document that corresponds to the previously identified relevant abstracts is read and the relevant ones are identified. That is, the expert refines the relevance label for those article IDs that were assigned a positive label before at abstract level.

[4] https://sites.google.com/site/clefehealth2017/task-2.

Table 4. The Number of Missing Documents in Different Stopping Points on Abstract Level (Top) and Document Level (Bottom)

Stopping point	0%	10%	15%	20%	25%	30%	35%	40%
# of missing docs	4,661	1,225	758	485	311	203	134	81
Stopping point	45%	50%	55%	60%	65%	70%	75%	80%
# of missing docs	52	35	24	12	4	3	3	3

Stopping point	0%	10%	15%	20%	25%	30%
# of missing docs	1,093	123	56	23	10	5
Stopping point	35%	40%	45%	50%	55%	60%
# of missing docs	5	3	3	2	2	0

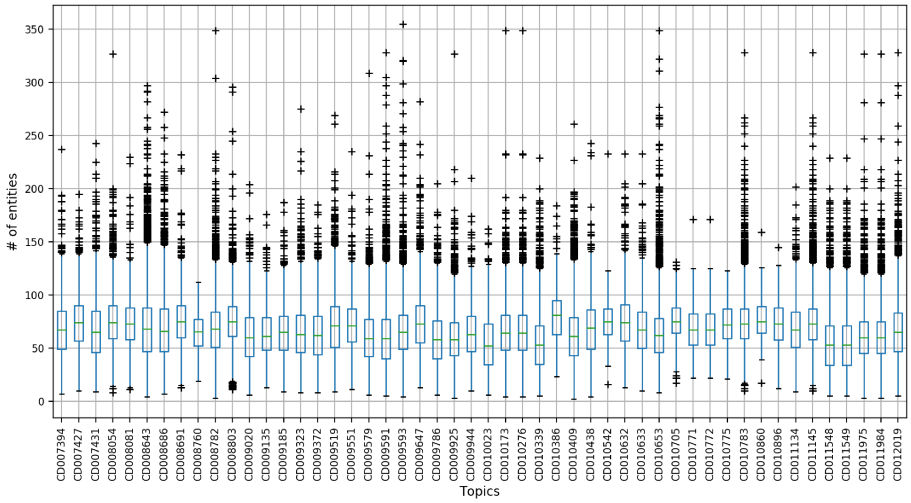


Fig. 3. The five-number summary of entities for each topic.

For the first three research questions **RQ1**–**RQ3**, we test our model over all of the 50 topics. For **RQ4**, same with the Technological-assisted Reviews in Empirical Medicine Task[5] [Kanoulas et al. 2017] in CLEF 2017[6] and other works [Cormack and Grossman 2017; Di Nunzio 2018; Lee and Sun 2018; Scells et al. 2018], we use 20 topics as the training set and test our model on the remaining 30 topics. The number of missing documents in different stopping points on abstract level (top) and document level (bottom) is shown in Table 4. The five-number summary of entities for each topic is shown in Figure 3. From Figure 3, we can see different topics have a similar five-number summary trend. The medians of the number of entities in different topics are between 50 and 80.

Evaluation measures. Same with Zou et al. [2018], we use two evaluation metrics that were the official metrics in CLEF 2017 e-Health Evaluation Lab [Kanoulas et al. 2017]: Mean Average Precision (MAP) and last_rel, which is the position of the last relevant document in the ranking, which approximates the user effort made, in terms of documents that need to be reviewed, to find

[5]https://sites.google.com/site/clefehealth2017/task-2.
[6]http://clef2017.clef-initiative.eu/index.php.

all relevant documents in the collection:

$$MAP = \frac{\sum_{q=1}^{Q} AP(q)}{Q}, \tag{6}$$

where Q is the number of topics in the testing dataset; in our experiments, Q is 50. $AP(q)$ is the average precision of the topic q:

$$AP = \frac{\sum_{l=1}^{n} P(l) * rel(l)}{\#\text{ of relevant documents}}, \tag{7}$$

where n is is the number of documents in the ranking list, l is the rank in the sequence of retrieved documents, $P(l)$ is the precision until l, i.e., the number of relevant documents out of the top-ranked documents, and $rel(l)$ is the ground truth capturing whether the document is relevant to this topic.

Simulating reviewers. The experiments depend on the ability of the reviewers to answer the questions asked to them by our model. In this work, we simulate users following past work in interactive algorithms [Zhang et al. 2018b; Zou and Kanoulas 2019]. We also conduct a small user study described in Section 4.6. We simulate users under two different settings: (1) we assume that the user will respond to the questions knowing precisely whether an entity appears or not in the missing relevant documents. Here the user model assumption is that the reviewer has an initial target document set in mind, which is deterministic but unknown. If an entity is contained in all missing relevant documents, then the reviewer will respond with a "yes" answer, if an entity is absent with a "no" answer, and for anything in between, with a "not sure" answer. This setting is the same with Zhang et al. [2018b], which assumes that the user fully knows the value of the question on an aspect; (2) we allow the users give to the wrong answer to our system with a given probability.

In this latter case, we consider three noisy answers settings, regarding the error rate for each entity $h(e)$:

(a) In the first setting all entities have the same chance to invoke a wrong answer and hence $h(e)$ is equal across entities; in this case, we experiment with different error rates that range from 0.1 to 0.5 with a step of 0.1. An error rate $h(e)$ of 0.5 means that the user has a 50% probability to give the wrong answer (i.e., randomly give the answer).

(b) In the second setting, we assume that users are more confident in their answers about an entity e if e is frequently occurring in the documents related to a given topic (query), and we define $h(e)$ as a function of average term frequency (TF) of an entity e across all documents, which lies in the range of (0, 0.5]:

$$h(e) = \frac{1}{2(1 + TF_{avg}(e))}, \tag{8}$$

where $TF_{avg}(e)$ represents the average term frequency of an entity e across all documents related to a given topic. In our experiments this subset of documents related to a topic is provided to us by the way the collection has been constructed, but one could think of other heuristics to define topic-related documents (such as running a ranking function, e.g., BM25 and considering the top-1,000 documents).

(c) In the third setting the noisy answers are modeled by multi-target property. We assume that the reviewer is more confident about the entity that tends to appear concurrently in all of the missing relevant documents. In this case $h(e)$ is also in the range (0, 0.5] and is

defined as

$$h(e) = \frac{\min(N_{\{e(\mathcal{D}^*)=1\}}, N_{\{e(\mathcal{D}^*)=0\}})}{N_{\{e(\mathcal{D}^*)=1\}} + N_{\{e(\mathcal{D}^*)=0\}}}, \tag{9}$$

where $N_{\{e(\mathcal{D}^*)=1\}}$ represents the number of target documents containing entity e for a given topic, while $N_{\{e(\mathcal{D}^*)=0\}}$ represents the number of target documents that do not contain entity e for a given topic. In all three settings, and during simulations, once it is decided that a wrong answer will be provided to the system, the simulator will randomly choose a wrong answer out of two wrong answers available.

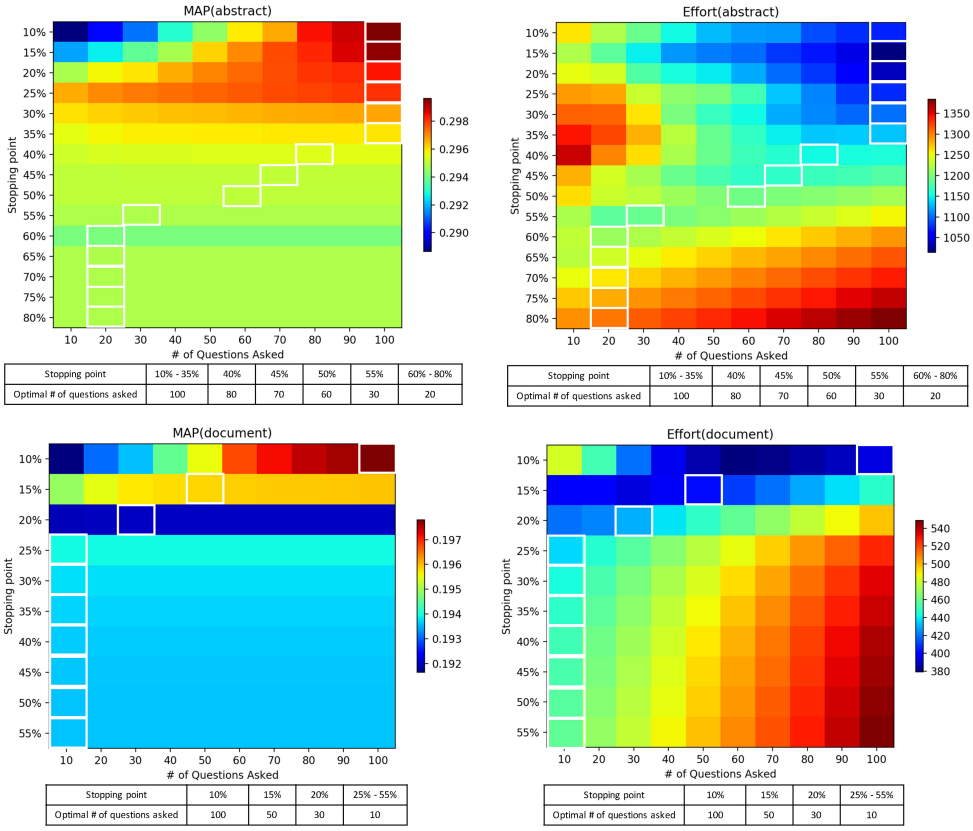
Note that the selection of these three noise settings is ad hoc; error can be defined as any other function of any other characteristic of entities, reviewers or topics. One should conduct a large user study to identify how and why users give erroneous answers to such system questions, but we leave this as future work.

Baselines. Same with Zou et al. [2018], we compare our method to three baselines, (1) **BMI** [Cormack and Grossman 2017], which is the state-of-the-art CAL algorithm applied without any stopping criterion until the entire collection is reviewed, (2) **BMI + LR**, which applies BMI until a number of documents are reviewed (stopping point) and then ranks the remaining of the collection on the basis of the trained logistic regression model, and (3) **BMI + Random**, which applies BMI until a number of documents are reviewed (stopping point) and then randomly chooses entities to ask about.

4.2 RQ1 the Effect of the CAL Stopping Point and the Number of Questions

The SBSTAR algorithm is parameterized by (a) the point BMI stops providing documents to reviewers for relevance feedback and (b) the number of questions on entities asked consequently to the reviewer. To better understand their impacts on the performance of the model, we report the MAP and total effort required to reach 100% recall on abstract level and document level. We report the total effort on abstract level as the same with Zou et al. [2018], but we also report an additional total effort result on document level and two additional MAP results on abstract level and document level in this work. Figure 4 shows the heatmap of the MAP (left) and the total effort (right) required to reach 100% recall on abstract level (top) and the added document level (bottom). The x-axis is the number of asked questions ranging from 10 to 100, and the y-axis is the stopping point as a percentage of the collection shown to the reviewer by BMI. We measure the effort on basis of two indicators: (a) the total number of documents required to be reviewed to reach 100% recall (i.e., the last_rel measure); this consists of both the documents ranked by BMI before the stopping point and the documents ranked by SBSTAR after the stopping point; and (b) the number of questions asked by SBSTAR. The effort is calculated as the sum of the two numbers, by making the simplifying assumption that answering a question takes the same effort as judging the relevance of a document. From this figure, we can see the visualization of the evolution of the MAP and effort over the number of asked questions, and we can also see the trends of MAP and effort across different stop rates.

We observe that the MAP is always increasing or stay stable with the increasing number of questions although at different stop rates, for both abstract level and document level. It is obvious that we get more accurate results when we asked the reviewer more questions. And the overall trends of MAP over different stop rates are increasing and then drop down in the early stage (less than 80 questions for abstract level and less than 60 questions for document level, respectively). It means that the stop rate should be set as a suitable value, should not too high or too low. The MAP always keeps decreasing trend in the late stage (greater than or equal to 80 questions for abstract level and greater than or equal to 60 questions for document level, respectively). This might be



Stopping point	10% - 35%	40%	45%	50%	55%	60% - 80%
Optimal # of questions asked	100	80	70	60	30	20

Stopping point	10% - 35%	40%	45%	50%	55%	60% - 80%
Optimal # of questions asked	100	80	70	60	30	20

Stopping point	10%	15%	20%	25% - 55%
Optimal # of questions asked	100	50	30	10

Stopping point	10%	15%	20%	25% - 55%
Optimal # of questions asked	100	50	30	10

Fig. 4. Heatmap of the MAP and the total effort required to achieve total recall on abstract level (top) and document level (bottom). The total effort is naively defined as the sum of the rank of the last relevant document and the number of asked questions. For MAP, the more red the heat map, the better the model performance. For effort, the more blue the heat map, the better the performance. The optimal number of questions for the corresponding stopping point is designated with the white boundary box and the tables below.

because that SBSTAR can always get a good relevance ranking list when the number of asked questions is high and the ranking list before the stop rate (i.e., is equal to the ranking list of BMI) lowers the mean of average precision. The highest MAP is achieved when the stop rate is 10% and the number of questions is 100. We also observe that the overall trend of effort is growing with the number of questions asked when the stopping point is greater than 55%, while the effort is decreasing when the stopping point is less than 55% for abstract level. As for document level, the overall trend of effort increases with the number of questions asked when the stopping point is greater than 10%, while the effort decreases when the stopping point is equal to 10%. This might be because the missing relevant documents are very few when the stopping point is in a high value, in which case asking many questions only leads to higher effort. Furthermore, SBSTAR(N_q = opt.) can reduce the effort effectively when the stopping point is less than or equal to 50% for abstract level, while the stopping point is equal to 10% for document level, respectively. The effort fluctuates over different stopping points and the effort is relatively lower when the stopping point is between 15%

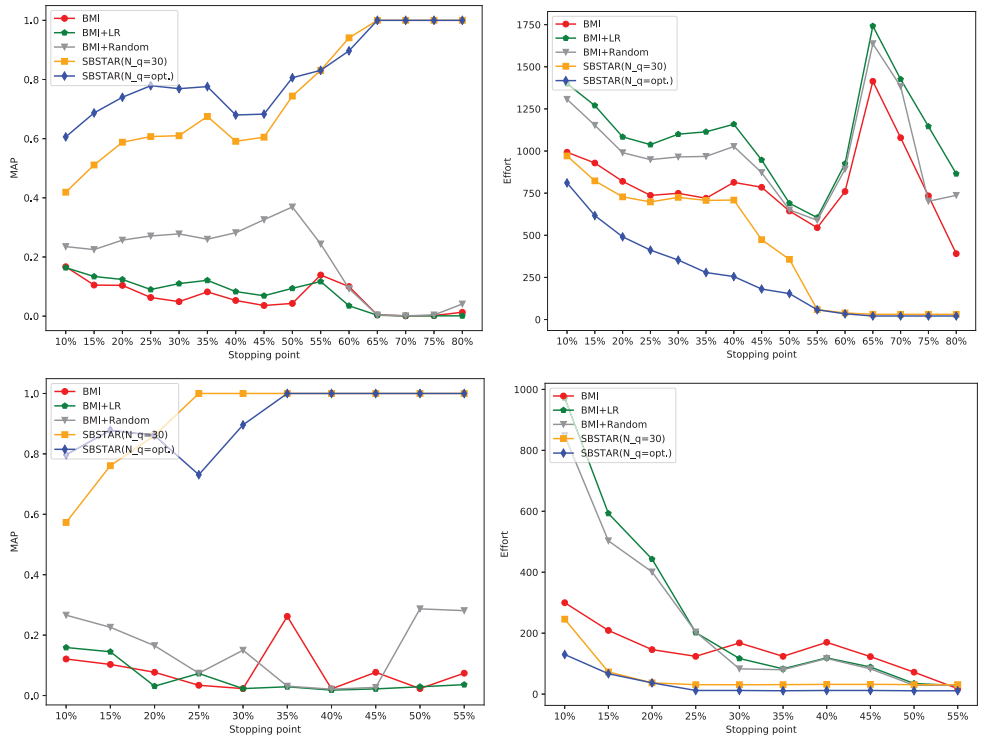


Fig. 5. Comparison of performance measured by MAP and total effort with different stopping points (the percentage of documents to be reviewed through BMI) on abstract level (top) and document level (bottom). N_q = opt. stands for N_q = optimal. The near-optimal number of questions were asked by SBSTAR ($N_q = opt.$) and BMI+Random for each stopping point is indicated in Figure 4 of **RQ1**. SBSTAR performs better than baselines.

and 20%, and between 45% and 55% for abstract level, while when the stopping point is between 10% and 15% for document level, respectively. The lowest effort is achieved when the stopping point is 15% and the number of asked questions is 100 for abstract level while the stopping point is between 10% and the number of asked questions is 60 for document level, respectively.

4.3 RQ2 the Performance of the SBSTAR Method

We compare the performance of SBSTAR with the state-of-the-art baselines in this section to explore the effectiveness of the SBSTAR model. In this research question, different from **RQ1**, MAP and total effort are computed only on the basis of the documents ranked after the stopping point, since we want to isolate the model effectiveness. The three compared baselines are shown in Section 4.1. Figure 5 shows the comparison results measured by MAP and effort on abstract level and additional document level. The values with the best performance are shown in boldface. As indicated in Figure 5, the SBSTAR algorithm achieves the highest results when compared to the three baselines. The SBSTAR algorithm outperforms BMI, BMI + LR, and BMI + Random on both MAP and effort. When considering the abstract level relevance labels, the results show the SBSTAR model can improve BMI by 0.439 to 0.999 points on MAP, it can improve BMI + LR by 0.442 to 0.999, and improve BMI + Random by 0.357 to 0.999. Note again that MAP is measured against

the missing documents, that we consider the ranking to start after the BMI stopping point. The results show the SBSTAR model can relatively reduce the effort by 18.3% to 98.5%, 42.1% to 98.8%, and 38% to 98.7% compared with BMI, BMI + LR, and BMI + Random, respectively. When considering the document level labels, the results show the SBSTAR model can improve BMI by 0.676 to 0.978 MAP points, it can improve BMI + LR by 0.638 to 0.982, and it can improve BMI + Random by 0.531 to 0.979. Again MAP is measured against the missing documents, that we consider the ranking to start after the BMI stopping point. The results show the SBSTAR model can relatively reduce the effort by 38.9% to 92.9%, 60.7% to 94.1%, and 62.1% to 94.1% compared with BMI, BMI + LR, and BMI + Random, respectively. The SBSTAR algorithm exceeds BMI + Random baseline, which indicates, as expected that our question selection strategy is better than choosing questions randomly. The SBSTAR algorithm also greatly improves over the BMI and BMI + LR baselines, and even the performance of BMI + Random is superior to the BMI and BMI + LR. This clearly suggests that a theoretically optimal sequence of entity-centered questions can be rather effective. It is expected that BMI will outperform BMI + LR because of repeatedly training the LR classifier [Cormack and Grossman 2017]. There exist some results that BMI + LR outperforms BMI, especially when the stopping point is less than and equal to 50%. This might be because that repeatedly enriching the training data and training LR classifier does not push all of the ranking positions of the missing relevant documents up, but instead by pushing part of the missing relevant documents up and part of them down, since there are multiple missing relevant documents. When the stopping point is larger than 50%, BMI outperforms BMI + LR obviously, since there are very few missing relevant documents. Additionally, we add the results for SBSTAR when the number of asked questions is equal to 30, i.e., SBSTAR(N_q=30), to show the performance of our SBSTAR model when asking a fixed smaller number of questions instead of the optimal number of questions. The results show that SBSTAR can still perform better than the state-of-the-art baseline BMI when asking 30 questions. Table 9 provides a few examples of a sequence of questions session.

4.4 RQ3 the Effect of Noisy Answers

Given that the user may not always know the right answer to a system's question, we also explore the noise-tolerance of the SBSTAR algorithm toward answering **RQ3**. We develop a noise-tolerant version of the SBSTAR algorithm as shown in Section 3.2 and investigate what the influence of noisy answers is. We simulate the noisy answer of the user under three settings. In the first setting, we fix the probability of error and consider it as a parameter that ranges from 10% to 50% with a step 10%. The results when $h(e)$ is a fixed number are shown in Figure 6. It is obvious and expected that overall MAP decreases with the increase of the probability of error while the effort (in terms of the number of documents that need to be reviewed for the reviewer to reach the last relevant document) increases with the increase of the probability of error. When the probability of error is equal to 50%, which means the user answer the question with "yes," "no," or "not sure" at random, we observe very low performance. However, when the probability of error is equal to 10%, the values of the two metrics still achieve relatively good performance. Further, note that, in comparison to the state-of-the-art baseline, the BMI model, for a 10% wrong answer rate, the MAP of our method is still higher.

In the second setting, we define $h(e)$ as a function of the average term frequency of the entity e as shown in Equation (8). In this case, we perform the experiment changing β in the range of 0 to 1 with a step of 0.1. The results of MAP when $h(e)$ is modeled by term frequency are shown in Table 5 and the results of effort are shown in Table 6. Optimal results in different stopping points are shown in bold. As we can see from the table, the performance as measured by the two metrics when using the optimal β is higher than or equal to that when $\beta = 0$, which suggests that the objective function of our noise-tolerant version of SBSTAR algorithm is effective.

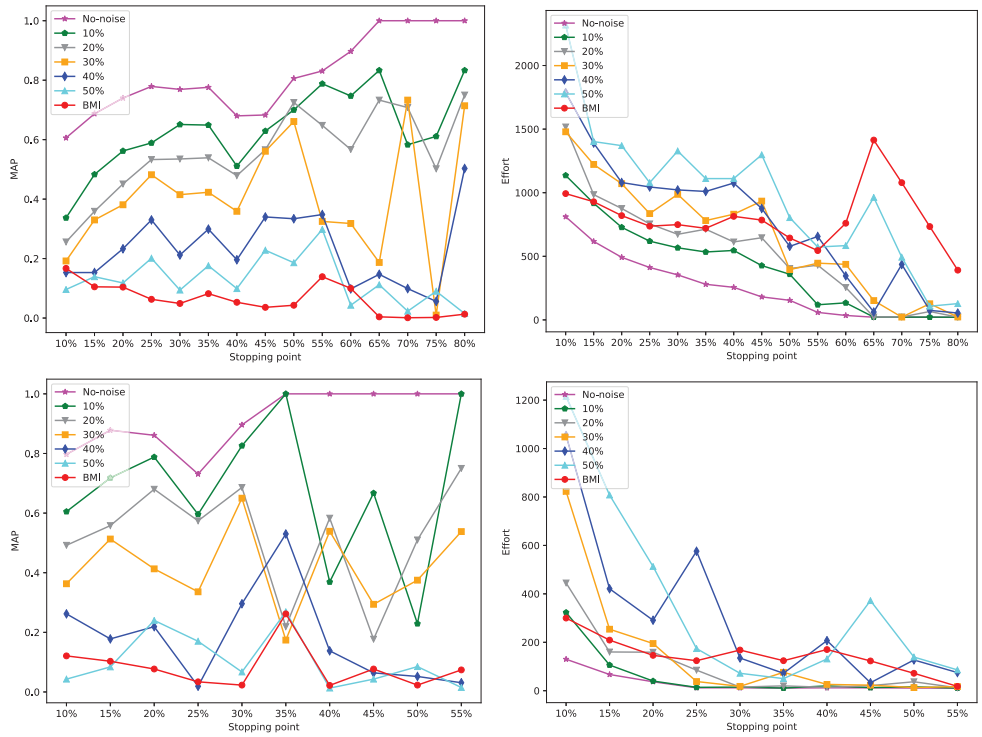


Fig. 6. The results of noisy answers for MAP and the total effort required to achieve total recall on abstract level (top) and document level (bottom) when $h(e)$ is a fixed, ranging from10% to 50% with a step 10%. The near-optimal number of questions were asked for each stopping point is indicated in Figure 4 of **RQ1**. Our model still achieves relatively good performance for a 10% wrong answer rate.

In the third setting $h(e)$ is defined on the basis of the target documents as shown in Equation (9). The results of this experiment are shown in Figure 7. From Figure 7, despite with noise, our method still achieves comparable effort and greatly outperforms the baselines in terms of MAP.

4.5 RQ4 the Effectiveness of Stopping to Ask Question

To answer **RQ4**, we explore the effectiveness of our stopping algorithm $\text{SBSTAR}_{\text{ext}}$. We first perform the experiment using SVM classifier with different max number of questions Max_q to be asked, from 100 to 500 with a step of 100. The results of the stopping algorithm on abstract level (top) and document level (bottom) are shown in Figure 8. As we can see from Figure 8, different Max_q generates different results. Obviously, the MAP is increasing with an increase of Max_q for abstract level; this might be because the increase of Max_q will increase the number of questions asked in some topics. As for document level, the MAP is increasing and finally stays stable with the increase of Max_q. This might be because the increase of Max_q will increase the number of questions asked in some topics in the beginning, and finally stops increasing, since the Max_q is too high so that the number of questions in all of topics is less than the Max_q. The total effort required to achieve 100% recall fluctuates with different Max_q when the stopping point is low and then stays stable when the stopping point is set as a high value. When Max_q = 300 for abstract level, most of MAP and effort outperform original SBSTAR. The average MAP in

Table 5. The Results of Noisy Answers for MAP on Abstract Level (Top) and Document Level (Bottom) When $h(e)$ Is Modeled by Term Frequency

Stopping point	No noise	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
10%	0.606	0.147	0.147	0.175	0.143	0.160	0.124	0.108	0.149	0.105	**0.189**	0.168
15%	0.687	0.179	0.201	0.246	0.235	0.194	0.219	**0.254**	0.224	0.162	0.213	0.217
20%	0.740	0.251	0.257	0.300	0.247	0.222	0.292	0.301	0.241	0.261	0.243	**0.355**
25%	0.779	0.342	0.321	0.325	0.316	**0.403**	0.308	0.391	0.265	0.326	0.329	0.313
30%	0.769	0.289	0.295	0.349	0.272	0.262	0.327	0.328	0.340	0.250	0.345	**0.358**
35%	0.776	0.321	0.300	0.360	0.299	0.317	0.362	**0.401**	0.328	0.326	0.293	0.367
40%	0.680	0.219	0.289	0.359	0.281	0.301	**0.381**	0.298	0.347	0.305	0.282	0.338
45%	0.683	0.446	0.320	0.384	0.362	0.308	0.464	**0.538**	0.437	0.366	0.434	0.507
50%	0.806	0.331	0.379	0.418	0.438	0.465	0.376	**0.566**	0.428	0.478	0.506	0.522
55%	0.831	0.415	0.503	0.500	0.418	0.317	0.436	0.530	0.436	0.299	0.407	**0.532**
60%	0.897	0.180	0.335	0.266	0.580	**0.639**	0.323	0.106	0.375	0.290	0.291	0.243
65%	1.000	0.252	0.026	0.068	0.393	**0.614**	0.342	0.611	0.018	0.117	0.193	0.042
70%	1.000	0.019	0.098	0.513	0.418	**0.667**	0.013	0.181	0.061	0.338	0.142	0.339
75%	1.000	0.041	0.337	0.034	0.212	0.215	**0.375**	0.340	0.371	0.201	0.338	0.169
80%	1.000	0.051	0.195	0.159	0.243	0.118	0.278	0.095	0.335	0.362	**0.459**	0.155

Stopping point	No noise	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
10%	0.797	0.150	0.274	0.280	0.262	0.275	0.262	0.266	0.237	**0.335**	0.313	0.332
15%	0.878	0.338	0.376	0.311	0.360	0.420	0.336	**0.425**	0.422	0.385	0.416	0.344
20%	0.861	0.419	0.425	**0.560**	0.373	0.379	0.426	0.503	0.399	0.452	0.512	0.367
25%	0.731	0.236	0.223	0.220	**0.475**	0.124	0.375	0.267	0.256	0.336	0.186	0.223
30%	0.896	0.374	**0.637**	0.119	0.275	0.436	0.361	0.181	0.540	0.264	0.317	0.355
35%	1.000	0.515	0.160	0.406	**0.688**	0.513	0.458	0.271	0.183	0.198	0.386	0.360
40%	1.000	0.273	0.066	0.169	0.341	0.569	0.261	0.140	0.569	**0.667**	0.071	0.187
45%	1.000	0.080	0.500	0.113	0.268	0.533	0.029	0.528	0.438	0.369	0.293	**0.613**
50%	1.000	**1.000**	0.505	0.556	1.000	0.507	0.417	0.545	0.750	0.625	0.417	0.545
55%	1.000	**1.000**	0.031	0.512	0.750	0.750	0.313	1.000	0.750	0.200	0.750	0.258

β is ranging from 0 to 1 with a step 0.1. Optimal results in different stopping points are shown in bold. The near-optimal number of questions for each stopping point is indicated in Figure 4 of **RQ1**. Results with optimal β are better than or equal to that when $\beta = 0$, which suggests that the objective function of our noise-tolerant version of SBSTAR algorithm is effective.

different stopping points is improved by 3.5% and the average effort is improved by 36.4%. Note that, the original SBSTAR uses the optimal number of asked questions by grid search in **RQ1**. As for document level, we can see that our method for deciding when to stop asking questions automatically still achieves a very high MAP while greatly reduces the effort. When $Max_q = 300$, the average MAP in different stopping points is reduced by 20.6% but the average effort is improved by 36.1%.

We also compare the performance of different classification models, including SVM, random forests (RF), and feedforward neural network (NN). Here, we use the optimal Max_q of previous experiment, 300, in Figure 8. The MAP and total effort on abstract level (top) and document level (bottom) are shown in Table 7. The average number of asked questions is also shown in parentheses

Table 6. The Results of Noisy Answers for the Total Effort Required to Reach 100% Recall on Abstract Level (Top) and Document Level (Bottom) When $h(e)$ Is Modeled by Term Frequency

Stopping point	No noise	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
10%	811	1854	**1665**	1724	1856	1840	1874	2048	1761	1983	1813	1729
15%	617	1585	1423	1484	1349	**1234**	1454	1299	1384	1483	1290	1552
20%	492	1374	**1040**	1155	1090	1119	1124	1194	1122	1080	1248	1124
25%	412	1006	998	1028	**946**	990	1008	1001	1047	1116	1042	1134
30%	354	1053	974	1128	986	**937**	1173	1114	1267	1087	1058	1074
35%	280	**844**	908	974	959	981	971	1056	985	889	916	975
40%	256	1203	1233	1263	1134	1023	1102	1207	1014	**977**	1122	1067
45%	181	1172	1039	966	**814**	863	966	1136	1132	1026	1155	846
50%	154	942	530	704	837	**436**	745	581	651	602	503	869
55%	59	576	427	303	452	354	239	378	300	292	439	**157**
60%	35	365	278	496	**162**	297	529	183	275	366	607	316
65%	21	690	276	105	45	58	76	**23**	120	410	89	94
70%	21	212	54	**30**	134	159	118	302	58	508	31	402
75%	21	51	140	204	57	159	**45**	85	51	51	522	240
80%	21	121	69	34	34	34	212	32	239	144	**29**	32

Stopping point	No noise	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
10%	130	1078	1056	1013	969	943	1175	1069	986	**889**	963	997
15%	67	325	301	267	284	709	430	478	**267**	402	654	532
20%	37	143	170	**118**	397	168	290	128	286	178	205	252
25%	12	113	42	321	**18**	90	31	35	42	62	69	31
30%	12	35	24	34	41	**19**	50	36	38	32	35	21
35%	11	41	22	16	**15**	83	21	17	20	22	18	24
40%	12	29	44	128	26	17	53	38	18	**13**	36	38
45%	12	103	**13**	23	17	19	150	27	23	18	255	15
50%	11	**11**	66	15	**11**	49	13	16	12	13	13	16
55%	11	**11**	49	31	12	12	15	**11**	12	15	12	43

β is ranging from 0 to 1 with a step 0.1. Optimal results in different stopping points are shown in bold. The near-optimal number of questions for each stopping point is indicated in Figure 4 of **RQ1**. Results with optimal β are better than or equal to that when $\beta = 0$, which suggests that the objective function of our noise-tolerant version of SBSTAR algorithm is effective.

following the total effort. For the comparison, the WEKA API and its default model parameters[7] are used.[8] As we observe in Table 7, mostly the SVM and the random forest achieve higher MAP and lower effort than neural network. That is, the SVM and the random forest classifiers perform better. When the stopping point is set to the high value (here: the stopping point is greater than or equal to 65% on abstract level and greater than or equal to 40% on document level), there are some peaks and the performance fluctuates strongly on MAP. This might be because all of the relevant documents are successfully found for most of topics and there are very few relevant documents

[7] https://www.cs.waikato.ac.nz/ml/weka/index.html.

[8] We use WEKA API and default model parameters in WEKA to perform the classifier models, i.e., for SVM, C = 1.0, and kernel = RBF; for random forests, the max number of trees = 100; for neural network, learning rate = 0.3, momentum = 0.2, and number of hidden neurons = 4.

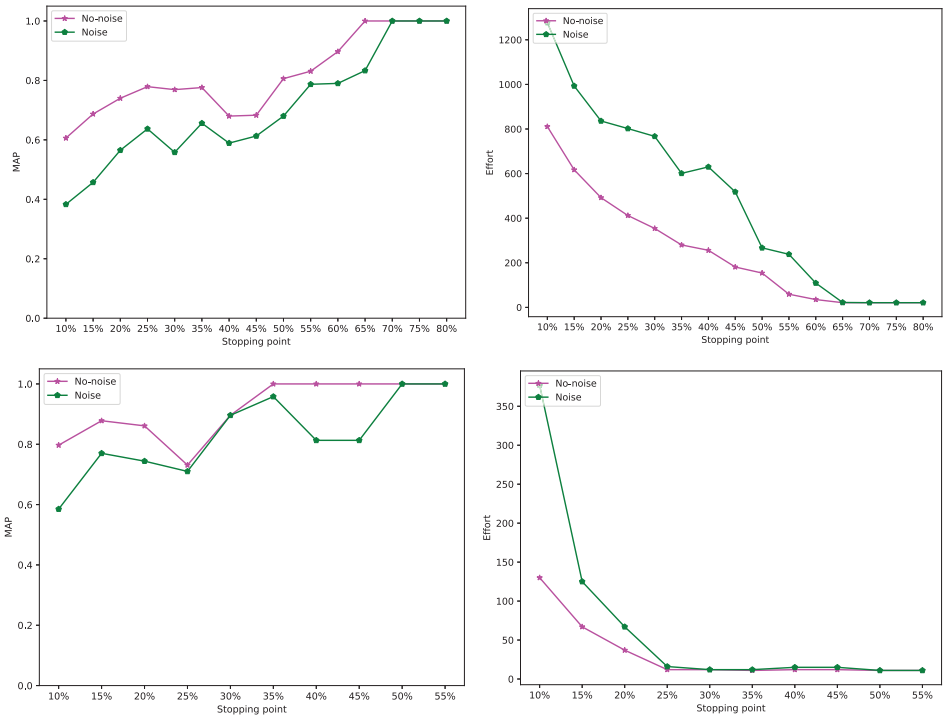


Fig. 7. The results of noisy answers for MAP and the total effort required to reach total recall on abstract level (top) and document level (bottom) when $h(e)$ is defined by using multi-target property. The near-optimal number of questions for each stopping point is indicated in Figure 4 of **RQ1**.

remaining (here: less than five relevant documents in total). We further conduct an analysis to understand the relative importance of different features in predicting when to stop. Similar to Genuer et al. [2010], we use the trained random forests to inspect the relative feature importance. The results of feature importance computed by random forests on abstract level (top) and document level (bottom) are shown in Table 8. From Table 8, we can see the relative feature importance on abstract level and document level, respectively. It is observed that the most important features are "# of candidates" and "# of yes/no questions" and the least important features are "Is_yes/no" and "different of top 1."

4.6 Online User Study

In addition to the simulated users, we also develop an online system and involve an information specialist,[9] whose job is to conduct searches in TAR with the assistance of medical experts, to conduct a small online user study to confirm some of the assumptions made in this work and evaluate how well our recommender system works "*in situ*." The ideal users would be medical experts, who are conducting a TAR to write a review paper within their well-understood topic and have already found most of the relevant documents using a BMI-based platform, and now

[9]The information specialist is an internationally renowned expert in the domain of evidence synthesis with more than 10 years work expertise, a member of the Council of the international Cochrane Collaboration, and a member of the International Collaboration of the automation of systematic reviews (ICASR).

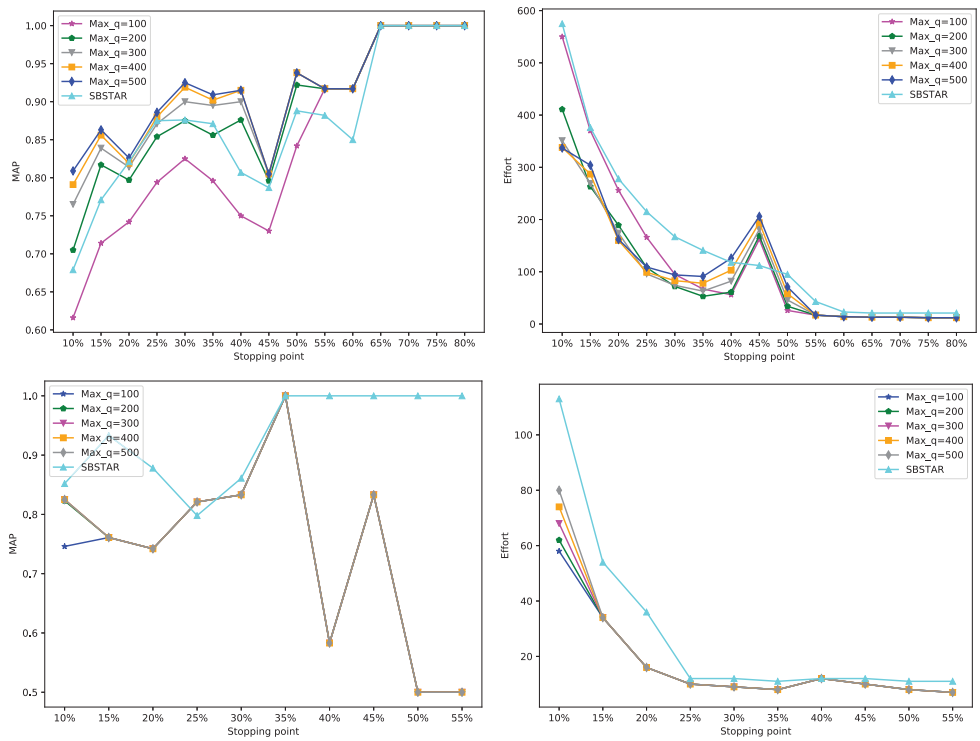


Fig. 8. The results of when to stop on abstract level (top) and document level (bottom) with different max number of asked questions. The near-optimal number of questions for each stopping point for SBSTAR are set as optimal number of questions, which is indicated in Figure 4 of **RQ1**.

they converse with our system embedded in the platform to find their last few missing relevant documents. However, finding such an expert user base who are currently conducting a TAR is not feasible. Thus, in our study, we asked the information specialist to choose all the studies he feels comfortable with out of the 50 topics in our collection. Then, we assume that the BMI has already run, and we provide the specialist with all the found relevant documents. The specialist is guided to review these relevant documents very carefully to familiarize himself with the topic. The missing relevant documents here are unknown to the specialist. After the specialist indicates that he is familiar with the topic, the conversation with the system can start. A question is selected by our algorithm to be asked to the specialist. The specialist is required to provide an answer for each question according to his expert knowledge and the topic information he read during the previous step, and then our system updates its relevance belief based on the provided answer. The specialist can stop answering questions any time during his interaction with the system. When stopping the interaction with the system, he is asked a number of exit questions about his experiences with the system. Note that this is by no means an *in situ* optimal user study.

During our user study the information specialist decided to work on seven topics, i.e., seven systematic reviews. We collected 193 rounds of questions made from our system toward the information specialist on these seven topics. First, we want to understand how many questions the user is willing to answer, and how well does the user answers them. From the collected data, we observe that the specialist answered an average number of 28 questions per topic using our system.

Table 7. The Results of When to Stop on Abstract Level (Top) and Document Level (Bottom) with Different Classifiers

	MAP			Effort		
Stopping point	SVM	random forests	neural network	SVM	random forests	neural network
10%	0.765	0.651	0.491	351 (159)	526 (122)	724 (60)
15%	0.839	0.697	0.617	269 (125)	481 (97)	601 (52)
20%	0.814	0.758	0.550	174 (95)	489 (81)	513 (39)
25%	0.871	0.812	0.696	96 (69)	347 (39)	371 (24)
30%	0.900	0.746	0.643	74 (66)	221 (48)	214 (29)
35%	0.895	0.823	0.695	63 (58)	94 (34)	46 (27)
40%	0.900	0.807	0.679	82 (77)	77 (71)	54 (34)
45%	0.805	0.832	0.756	181 (45)	46 (39)	46 (29)
50%	0.938	0.857	0.753	46 (44)	26 (21)	27 (16)
55%	0.917	0.988	0.858	17 (16)	17 (16)	16 (11)
60%	0.917	0.917	0.550	14 (12)	14 (12)	15 (9)
65%	1.000	1.000	0.100	13 (12)	13 (12)	17 (7)
70%	1.000	1.000	0.250	13 (12)	13 (12)	13 (9)
75%	1.000	1.000	1.000	12 (11)	11 (10)	11 (10)
80%	1.000	0.500	0.200	12 (11)	11 (9)	12 (7)

	MAP			Effort		
Stopping point	SVM	random forests	neural network	SVM	random forests	neural network
10%	0.825	0.771	0.768	68 (62)	83 (63)	85 (77)
15%	0.761	0.608	0.851	34 (30)	36 (31)	31 (28)
20%	0.742	0.750	0.693	16 (13)	16 (13)	16 (13)
25%	0.821	0.929	0.895	10 (8)	10 (9)	12 (10)
30%	0.833	0.833	0.770	9 (7)	9 (7)	12 (9)
35%	1.000	0.833	1.000	8 (7)	9 (7)	11 (10)
40%	0.583	0.583	1.000	12 (9)	12 (9)	13 (11)
45%	0.833	0.833	0.225	10 (7)	11 (8)	13 (5)
50%	0.500	0.500	0.500	8 (6)	7 (5)	8 (6)
55%	0.500	0.500	0.100	7 (5)	6 (4)	12 (2)

The average number of asked questions is shown in parentheses following each total effort.

Further, in the exit questionnaire, the specialist declares that he is willing to answer 30 questions. In the exit questionnaire, he thinks the conversational system is helpful and he is willing to use it in the future. Last, the specialist provided the correct answers to the system's question 76% of the time, he was not sure about the answer 4% of the time, and he gave the wrong answer (i.e., his answer disagreed with the description of the missing relevant documents) 20% of the time. As we can see from Figure 6, most of the results of our model are higher than the state-of-the-art baseline BMI when there are 20% of noisy answers.

In this article, we calculate the total effort as the sum of the rank of the last relevant document and the number of asked questions. Therefore, we made the assumption that answering a direct question about entities requests at most as much effort as providing the relevance of a document. We attempt to validate this assumption through the user study. We recorded the time the specialist spent on reviewing each relevant document (title and abstract) before answering questions and

Table 8. Relative Feature Importance on Abstract Level (Top) and Document Level (Bottom) by Random Forests

Feature	Importance score
# of candidates	0.543490
# of yes/no questions	0.195774
Stopping point	0.112272
different of candidates	0.074370
different of preference	0.035941
Is_yes/no	0.027093
different of top 1	0.011060

Feature	Importance score
# of candidates	0.487968
# of yes/no questions	0.246702
Stopping point	0.126864
different of candidates	0.075180
different of preference	0.036955
different of top 1	0.013730
Is_yes/no	0.012601

the time the specialist spent on answering each question. From the collected data, we observe that the specialist took an average time of 55.8 seconds to screen each relevant document while spending only 7.8 seconds to answer a question, on average. Additionally, in the exit questionnaire, the specialist indicates that the system's questions were easy to answer. This observation is in agreement with the conclusions made in previous works, which conclude that when judging the relevance of documents it is more effective and efficient to judge a provided document summary [Sanderson 1998] or even better to judge relevant sentences [Zhang et al. 2018c].

5 LIMITATIONS AND DISCUSSION

In this work, we pivot around the presence or absence of entities in the target documents to generate questions to ask to the user. Recognizing entities in biomedical texts is a research direction of its own, with significant recent work on neural methods, which further progresses the state-of-the-art [Hakala and Pyysalo 2019; Wang et al. 2019]. In this work, we use TAGME, which is widely used in prior research, in semantic mapping [Hasibi et al. 2015; Xiong and Callan 2015; Xiong et al. 2017], and in biomedical information labeling [Ernst et al. 2017; Park et al. 2013; Zou et al. 2018]. However, this automatic entity annotation may provide some irrelevant annotations or may miss some entities in the data. Our method could certainly benefit from better entity recognition and salience detection methods, constructing more reliable and salient question pools. Similarly, there may be a richer set of possible questions to be asked, questions that may or may not be answered with a "yes" or a "no." For instance, questions could be constructed by using labeled topics [Zou et al. 2015, 2017], keywords extraction [Campos et al. 2018], or other information extraction techniques [Maslennikov and Chua 2010; Riloff and Lehnert 1994]. Richer type of questions could also be constructed by identifying properties of the documents (entities in a knowledge base triplet representation) and their relation to the document. For example, the following entities could be identified in the document description: "author," "year," "publisher," "subject category," patient population, intervention, comparison, and outcome [Wallace et al. 2016]. Questions then could be constructed from the derived triplets [Reddy et al. 2017].

Table 9. Two Examples of a Sequence of Questions Asked by Our Model

Topic: Optical coherence tomography (OCT) for detection of macular oedema in patients with diabetic retinopathy		
Target documents: (missing relevant documents): ID: 15051203, Title: Comparison of the clinical diagnosis of diabetic macular edema with diagnosis by optical coherence tomography. ID: 9479300, Title: Topography of diabetic macular edema with optical coherence tomography.		
Question	**Answer**	**Rank of Last Relevant**
Are the documents about ...		188
diabetic macular edema (DME)	Yes	69
treatment	No	48
retina	Not Sure	48
optical coherence tomography (OCT)	Yes	27
measurements	Yes	17
evaluation	No	2

Topic: Human papillomavirus testing versus repeat cytology for triage of minor cytological cervical lesions		
Target documents (missing relevant documents): ID: 19116707, Title: Prevalence of human papillomavirus types 6, 11, 16 and 18 in young Austrian women - baseline data of a phase III vaccine trial. ID: 19331088, Title: Cervical cytology screening and management of abnormal cytology in adolescents.		
Question	**Answer**	**Rank of Last Relevant**
Are the documents about ...		988
Human Papillomavirus (HPV)	Yes	430
women	Not Sure	430
cervical cancer	Yes	224
infection	Yes	129
cancer	Yes	44
development	No	19
treatment	Not Sure	19
disease	Yes	6
clinic	No	5
cervical	Yes	2

Further, our work simulates user answers, noisy or not. Under the no-noise setting, we assume that when presented with an entity reviewers know whether the entity is present in all missing documents with 100% confidence. Under the noisy setting, we propose some noise model to explore the performance of our algorithm and relax the aforementioned 100% confidence assumption. We defined three different noise settings in this article. However, these three settings are ad hoc and the noise can be also defined as any function of any characteristic of entities, reviewers, or topics. A large user study could be particularly helpful in understanding how and why users give erroneous answers to such system questions. Our simulation setup is just a first step toward considering noisy answers, something that is utterly missing from past work on the topic. Our small user

study indicates that there is validity in the assumptions we have made, but yet again there is a need for an *in situ* larger study to confirm that.

It is also likely that an entity may be semantically related to the desired document, while not lexically present. In this work, we do not explore any semantic correlation modeling, but we leave it as future work.

Our when-to-stop algorithm is based on the extracted dynamic features. In this work, we extract seven related features to decide when to stop asking effectively. However, systematic feature engineering investigation may discover more strong features related to automatically stop asking and feature selection algorithm may yield improvements, which we leave as future work.

We apply GBS over entities to construct our objective function to find the best question to ask. Any other strategy learning techniques could also be used. We leave identifying more systematic objective functions as future work.

In this article, we focus on the task of Technology-assisted Reviews often performed in empirical medicine of legal e-discovery by expert reviewers. It is also possible to deploy our framework to other high-recall applications for naive users, but for that to be more effective, we might need to change our TAR-specific prior belief initialization model BMI to the corresponding task-specific prior belief initialization models, and most likely accounting for other characteristics of the entities in comparison with the background level of the user.

Last, in this work, we interact with the expert reviewers by asking questions to locate the last few relevant documents. Another possible way of locating these relevant documents is to keep reformulating the query. Query reformulation has shown its effectiveness to locate the targets for initial query mismatching and limited coverage [Dang and Croft 2010]. However, users need to find the association between queries and incorporate the new information gained from the previous search by themselves to reformulate the next query. Furthermore, query reformulation may generate some duplicate results and reviewing them will cost extra efforts. Our work automatically selects questions to ask and incorporates the answers to refine the search results, which can be a complement of keeping query reformulation. One can also combine our method with query expansion or reformulation techniques to a guided query expansion or reformulation.

6 CONCLUSION

In this work, we aim at achieving high recall in TAR. We describe our previously introduced interactive method, SBSTAR, which directly queries reviewers whether an entity is present or absent in missing relevant documents [Zou et al. 2018]. The framework applies a CAL algorithm on the relevance feedback from reviewers until a stopping point, which is a certain percentage of documents that have been reviewed, and then switches to locate the last few missing relevant documents. In this work, we present three rudimentary models to model reviewers' noisy answers, as well as the noise-tolerance algorithm in this work. We further conduct an analysis under different noisy answer simulation settings. We also propose the extension of SBSTAR, $\text{SBSTAR}_{\text{ext}}$, which performs a novel when-to-stop method by training a classifier to determine when to stop asking questions automatically, to avoid pre-setting the parameter of the number of questions when question asking. Additionally, we explore the performance of SBSTAR and $\text{SBSTAR}_{\text{ext}}$ at different levels, including the abstract level and document level. Experiments on the CLEF 2017 e-Health Lab dataset demonstrate SBSTAR can efficiently locate the missing relevant documents while asking for minimal reviewers' effort. When accounting for noisy answers, the noise-tolerance algorithm is also effective in the case that reviewers are not able to provide 100% correct answer. The experiment on the performance of $\text{SBSTAR}_{\text{ext}}$ demonstrates that our stopping to ask question model is effective. Last, we conduct a small user study that validates the availability of our assumptions about users' willingness and ability to answer a number of questions, as well as their efforts.

We introduce a question-based approach for the TAR task and validate the effectiveness of the question-based approach. However, this is just the first step toward an intelligent question-based system for assisting TAR and there are still rooms for improvement. A potential research direction is to incorporate the latest conversational/dialogue system techniques and question answering techniques to improve TAR, and to aid it toward a perfect intelligent system. Furthermore, we ask questions about informative terms to locate the last few relevant documents after deploying the CAL algorithm, which queries on documents. Instead of asking questions after deploying the CAL algorithm, one can also explore the switch mechanism for asking questions about informative terms and querying on documents in each iteration.

ACKNOWLEDGMENTS

All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors. We cordially thank Rene Spijker, a senior researcher at the University Medical Center Utrecht and Amsterdam University Medical Center, for his contribution to the user study of this work.

REFERENCES

Mustafa Abualsaud, Nimesh Ghelani, Haotian Zhang, Mark D. Smucker, Gordon V. Cormack, and Maura R. Grossman. 2018. A system for efficient high-recall retrieval. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'18)*. Association for Computing Machinery, New York, NY, 1317–1320. DOI: https://doi.org/10.1145/3209978.3210176

Avi Arampatzis, Jaap Kamps, and Stephen Robertson. 2009. Where to stop reading a ranked list? Threshold optimization using truncated score distributions. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'09)*. Association for Computing Machinery, New York, NY, 524–531. DOI: https://doi.org/10.1145/1571941.1572031

Kyoungman Bae and Youngjoong Ko. 2019. Improving question retrieval in community question answering service using dependency relations and question classification. *J. Assoc. Info. Sci. Technol.* 70, 11 (2019), 1194–1209. DOI: https://doi.org/10.1002/asi.24196 arXiv:https://asistdl.onlinelibrary.wiley.com/doi/pdf/10.1002/asi.24196

Jason R. Baron, David D. Lewis, and Douglas W. Oard. 2006. TREC 2006 legal track overview. In *Proceedings of the Text Retrieval Conference (TREC'06)*. Citeseer.

Chris Buckley and Stephen Robertson. 2008. *Relevance Feedback Track Overview: TREC 2008.* Technical Report. Microsoft Corporation, Redmond, WA.

Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Mário Jorge, Célia Nunes, and Adam Jatowt. 2018. A text feature-based automatic keyword extraction method for single documents. In *Advances in Information Retrieval*. Springer International Publishing, Cham, 684–691.

Joyce Y. Chai, Chen Zhang, and Rong Jin. 2007. An empirical investigation of user term feedback in text-based targeted image search. *ACM Trans. Info. Syst.* 25, 1 (2007), 3.

Zheqian Chen, Chi Zhang, Zhou Zhao, Chengwei Yao, and Deng Cai. 2018. Question retrieval for community-based question answering via heterogeneous social influential network. *Neurocomputing* 285 (2018), 117–124. DOI: https://doi.org/10.1016/j.neucom.2018.01.034

Gordon V. Cormack and Maura R. Grossman. 2014. Evaluation of machine-learning protocols for technology-assisted review in electronic discovery. In *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'14)*. Association for Computing Machinery, New York, NY, 153–162. DOI: https://doi.org/10.1145/2600428.2609601

Gordon V. Cormack and Maura R. Grossman. 2015a. Autonomy and reliability of continuous active learning for technology-assisted review. *arXiv preprint arXiv:1504.06868*.

Gordon V. Cormack and Maura R. Grossman. 2015b. Waterloo (Cormack) participation in the TREC 2015 total recall track. In *Proceedings of the Text Retrieval Conference (TREC'15)*.

Gordon V. Cormack and Maura R. Grossman. 2016a. "When to stop" Waterloo (Cormack) participation in the TREC 2016 total recall track. In *Proceedings of the Text Retrieval Conference (TREC'16)*.

Gordon V. Cormack and Maura R. Grossman. 2016b. Engineering quality and reliability in technology-assisted review. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'16)*. Association for Computing Machinery, New York, NY, 75–84. DOI: https://doi.org/10.1145/2911451.2911510

Gordon V. Cormack and Maura R. Grossman. 2016c. Scalability of continuous active learning for reliable high-recall text classification. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management (CIKM'16)*. Association for Computing Machinery, New York, NY, 1039–1048. DOI: https://doi.org/10.1145/2983323.2983776

Gordon V. Cormack and Maura R. Grossman. 2017. Technology-assisted review in empirical medicine: Waterloo participation in CLEF eHealth 2017. In *Proceedings of the Conference and Labs of the Evaluation Forum (CLEF'17)*. Retrieved from http://ceur-ws.org/Vol-1866/paper_51.pdf.

Gordon V. Cormack and Maura R. Grossman. 2018. Beyond pooling. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'18)*. Association for Computing Machinery, New York, NY, 1169–1172. DOI: https://doi.org/10.1145/3209978.3210119

Gordon V. Cormack, Maura R. Grossman, Bruce Hedin, and Douglas W. Oard. 2010. Overview of the TREC 2010 legal track. In *Proceedings of the 19th Text Retrieval Conference (TREC'10)*.

Gordon V. Cormack and Thomas R. Lynam. 2005. TREC 2005 spam track overview. In *Proceedings of the Text Retrieval Conference (TREC'05)*. 500–274.

Gordon V. Cormack and Mona Mojdeh. 2009. Machine learning for information retrieval: TREC 2009 web, relevance feedback and legal tracks. In *Proceedings of the Text Retrieval Conference (TREC'09)*.

Gordon V. Cormack, Christopher R. Palmer, and Charles L. A. Clarke. 1998. Efficient construction of large test collections. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'98)*. Association for Computing Machinery, New York, NY, 282–289. DOI: https://doi.org/10.1145/290941.291009

Marco Cornolti, Paolo Ferragina, Massimiliano Ciaramita, Stefan Rüd, and Hinrich Schütze. 2018. SMAPH: A Piggyback approach for entity-linking in web queries. *ACM Trans. Info. Syst.* 37, 1 (2018), 13.

Van Dang and Bruce W. Croft. 2010. Query reformulation using anchor text. In *Proceedings of the 3rd ACM International Conference on Web Search and Data Mining (WSDM'10)*. Association for Computing Machinery, New York, NY, 41–50. DOI: https://doi.org/10.1145/1718487.1718493

Giorgio Maria Di Nunzio. 2018. A study of an automatic stopping strategy for technologically assisted medical reviews. In *Advances in Information Retrieval*. Springer International Publishing, Cham, 672–677.

Harris Drucker, Behzad Shahrary, and David Gibbon. 2001. Relevance feedback using support vector machines. In *Proceedings of the 18th International Conference on Machine Learning (ICML'01)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, 122–129.

Miroslav Dudík, Katja Hofmann, Robert E. Schapire, Aleksandrs Slivkins, and Masrour Zoghi. 2015. Contextual dueling bandits. *arXiv preprint arXiv:1502.06362*.

Patrick Ernst, Arunav Mishra, Avishek Anand, and Vinay Setty. 2017. BioNex: A system for biomedical news event exploration. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'17)*. Association for Computing Machinery, New York, NY, 1277–1280. DOI: https://doi.org/10.1145/3077136.3084150

Elena Erosheva, Stephen Fienberg, and John Lafferty. 2004. Mixed-membership models of scientific publications. *Proc. Natl. Acad. Sci. U.S.A.* 101, suppl. 1 (2004), 5220–5227.

Paolo Ferragina and Ugo Scaiella. 2010. TAGME: On-the-fly annotation of short text fragments (by Wikipedia entities). In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management (CIKM'10)*. ACM, New York, NY, 1625–1628. DOI: https://doi.org/10.1145/1871437.1871689

Nicola Ferro and Carol Peters. 2019. *Information Retrieval Evaluation in a Changing World: Lessons Learned from 20 Years of CLEF*. Vol. 41. Springer.

Robin Genuer, Jean-Michel Poggi, and Christine Tuleau-Malot. 2010. Variable selection using random forests. *Pattern Recogn. Lett.* 31, 14 (2010), 2225–2236.

Lorraine Goeuriot, Liadh Kelly, Hanna Suominen, Aurélie Névéol, Aude Robert, Evangelos Kanoulas, Rene Spijker, João Palotti, and Guido Zuccon. 2017. CLEF 2017 eHealth evaluation lab overview. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction*. Springer International Publishing, Cham, 291–303.

Maura R. Grossman and Gordon V. Cormack. 2010. Technology-assisted review in e-discovery can be more effective and more efficient than exhaustive manual review. *Rich. JL Tech.* 17 (2010), 1.

Maura R. Grossman, Gordon V. Cormack, and Adam Roegiest. 2016. TREC 2016 total recall track overview. In *Proceedings of the 25th Text Retrieval Conference (TREC'16)*. Retrieved from http://trec.nist.gov/pubs/trec25/papers/Overview-TR.pdf.

Maura R. Grossman, Gordon V. Cormack, and Adam Roegiest. 2017. Automatic and semi-automatic document selection for technology-assisted review. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'17)*. ACM, New York, NY, 905–908. DOI: https://doi.org/10.1145/3077136.3080675

Kai Hakala and Sampo Pyysalo. 2019. Biomedical named entity recognition with multilingual BERT. In *Proceedings of The 5th Workshop on BioNLP Open Shared Tasks*. Association for Computational Linguistics, Hong Kong, China, 56–61. DOI: https://doi.org/10.18653/v1/D19-5709

Faegheh Hasibi, Krisztian Balog, and Svein Erik Bratsberg. 2015. Entity linking in queries: Tasks and evaluation. In *Proceedings of the International Conference on the Theory of Information Retrieval (ICTIR'15)*. Association for Computing Machinery, New York, NY, 171–180. DOI: https://doi.org/10.1145/2808194.2809473

Bruce Hedin, Stephen Tomlinson, Jason R. Baron, and Douglas W. Oard. 2009. *Overview of the TREC 2009 Legal Track*. Technical Report. National Archives and Records Administration, College Park, MD.

Katja Hofmann, Shimon Whiteson, and Maarten de Rijke. 2013. Balancing exploration and exploitation in listwise and pairwise online learning to rank for information retrieval. *Info. Retriev.* 16, 1 (2013), 63–90.

Evangelos Kanoulas, Dan Li, Leif Azzopardi, and René Spijker. 2017. Technologically assisted reviews in empirical medicine overview. In *Proceedings of the Conference and Labs of the Evaluation Forum (CLEF'17)*. Retrieved from http://ceur-ws.org/Vol-1866/invited_paper_12.pdf.

Anita Krishnakumar. 2007. *Active Learning Literature Survey*. Technical Report. University of California, Santa Cruz.

Dipankar Kundu and Deba Prasad Mandal. 2019. Formulation of a hybrid expertise retrieval system in community question answering services. *Appl. Intell.* 49, 2 (Feb. 2019), 463–477. DOI: https://doi.org/10.1007/s10489-018-1286-z

Branislav Kveton and Shlomo Berkovsky. 2015. Minimal interaction search in recommender systems. In *Proceedings of the 20th International Conference on Intelligent User Interfaces (IUI'15)*. Association for Computing Machinery, New York, NY, 236–246. DOI: https://doi.org/10.1145/2678025.2701367

Branislav Kveton and Shlomo Berkovsky. 2016. Minimal interaction content discovery in recommender systems. *ACM Trans. Interact. Intell. Syst.* 6, 2 (2016), 15.

Victor Lavrenko and W. Bruce Croft. 2017. Relevance-based language models. *SIGIR Forum* 51, 2, 260–267. DOI: https://doi.org/10.1145/3130348.3130376

Grace E. Lee and Aixin Sun. 2018. Seed-driven document ranking for systematic reviews in evidence-based medicine. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'18)*. Association for Computing Machinery, New York, NY, 455–464. DOI: https://doi.org/10.1145/3209978.3209994

Baichuan Li and Irwin King. 2010. Routing questions to appropriate answerers in community question answering services. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management (CIKM'10)*. Association for Computing Machinery, New York, NY, 1585–1588. DOI: https://doi.org/10.1145/1871437.1871678

Dekang Lin and Patrick Pantel. 2001. Discovery of inference rules for question-answering. *Nat. Lang. Eng.* 7, 4 (2001), 343–360.

Ming Liu, Lei Chen, Bingquan Liu, Guidong Zheng, and Xiaoming Zhang. 2017. DBpedia-based entity linking via greedy search and adjusted Monte Carlo random walk. *ACM Trans. Info. Syst.* 36, 2 (2017), 16.

Ming Liu, Gu Gong, Bing Qin, and Ting Liu. 2019. A multi-view–based collective entity-linking method. *ACM Trans. Info. Syst.* 37, 2 (2019), 23.

Yuanjie Liu, Shasha Li, Yunbo Cao, Chin-Yew Lin, Dingyi Han, and Yong Yu. 2008. Understanding and summarizing answers in community-based question answering services. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING'08)*. Association for Computational Linguistics, 497–504.

David E. Losada, Javier Parapar, and Alvaro Barreiro. 2019. When to stop making relevance judgments? A study of stopping methods for building information retrieval test collections. *J. Assoc. Info. Sci. Technol.* 70, 1 (2019), 49–60.

Yuanhua Lv and ChengXiang Zhai. 2009. Adaptive relevance feedback in information retrieval. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM'09)*. Association for Computing Machinery, New York, NY, 255–264. DOI: https://doi.org/10.1145/1645953.1645988

Mstislav Maslennikov and Tat-Seng Chua. 2010. Combining relations for information extraction from free text. *ACM Trans. Info. Syst.* 28, 3 (2010), 14.

Graham McDonald, Craig Macdonald, and Iadh Ounis. 2018. Active learning strategies for technology-assisted sensitivity review. In *Advances in Information Retrieval*. Springer International Publishing, Cham, 439–453.

Douglas W. Oard, Bruce Hedin, Stephen Tomlinson, and Jason R. Baron. 2008. *Overview of the TREC 2008 Legal Track*. Technical Report. University of Maryland College of Information Studies, College Park, MD.

Douglas W. Oard, Fabrizio Sebastiani, and Jyothi K. Vinjumur. 2018. Jointly minimizing the expected costs of review for responsiveness and privilege in E-discovery. *ACM Trans. Info. Syst.* 37, 1 (2018), 11.

Alison O'Mara-Eves, James Thomas, John McNaught, Makoto Miwa, and Sophia Ananiadou. 2015. Using text mining for study identification in systematic reviews: A systematic review of current approaches. *System. Rev.* 4, 1 (Jan. 2015), 5. DOI: https://doi.org/10.1186/2046-4053-4-5

Meeyoung Park, Hariprasad Sampathkumar, Bo Luo, and Xue-wen Chen. 2013. Content-based assessment of the credibility of online healthcare information. In *Proceedings of the IEEE International Conference on Big Data*. IEEE, 51–58.

Filip Radlinski, Robert Kleinberg, and Thorsten Joachims. 2008. Learning diverse rankings with multi-armed bandits. In *Proceedings of the 25th International Conference on Machine Learning (ICML'08)*. Association for Computing Machinery, New York, NY, 784–791. DOI: https://doi.org/10.1145/1390156.1390255

Hadas Raviv, Oren Kurland, and David Carmel. 2016. Document retrieval using entity-based language models. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'16)*. Association for Computing Machinery, New York, NY, 65–74. DOI: https://doi.org/10.1145/2911451.2911508

Sathish Reddy, Dinesh Raghu, Mitesh M. Khapra, and Sachindra Joshi. 2017. Generating natural language question-answer pairs from a knowledge graph using a RNN-based question generation model. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 376–385. Retrieved from https://www.aclweb.org/anthology/E17-1036.

Ellen Riloff and Wendy Lehnert. 1994. Information extraction as a basis for high-precision text classification. *ACM Trans. Info. Syst.* 12, 3 (1994), 296–333.

Stephen E. Robertson and K. Spärck Jones. 1976. Relevance weighting of search terms. *J. Assoc. Info. Sci. Technol.* 27, 3 (1976), 129–146.

J. Rocchio. 1971. Relevance feedback in information retrieval. *Smart Retriev. Syst.-Exper. Autom. Doc. Process.* (1971), 313–323. Retrieved from https://ci.nii.ac.jp/naid/10000074359/en/.

Adam Roegiest, Gordon V. Cormack, Maura R. Grossman, and Charles Clarke. 2015. TREC 2015 total recall track overview. *Proc. TREC-2015* (2015).

Michal Rosen-Zvi, Thomas Griffiths, Mark Steyvers, and Padhraic Smyth. 2004. The author-topic model for authors and documents. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence (UAI'04)*. AUAI Press, Arlington, VA, 487–494.

Tuukka Ruotsalo, Jaakko Peltonen, Manuel J. A. Eugster, Dorota Głowacka, Patrik Floréen, Petri Myllymäki, Giulio Jacucci, and Samuel Kaski. 2018. Interactive intent modeling for exploratory search. *ACM Trans. Info. Syst.* 36, 4 (2018), 44.

Gerard Salton and Chris Buckley. 1990. Improving retrieval performance by relevance feedback. *J. Amer. Soc. Info. Sci.* 41, 4 (1990), 288–297.

Mark Sanderson. 1998. Accurate user directed summarization from existing tools. In *Proceedings of the 7th International Conference on Information and Knowledge Management (CIKM'98)*. Association for Computing Machinery, New York, NY, 45–51. DOI: https://doi.org/10.1145/288627.288640

Mark Sanderson and Hideo Joho. 2004. Forming test collections with no system pooling. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'04)*. ACM, New York, NY, 33–40. DOI: https://doi.org/10.1145/1008992.1009001

V. Satopaa, J. Albrecht, D. Irwin, and B. Raghavan. 2011. Finding a "Kneedle" in a Haystack: Detecting knee points in system behavior. In *Proceedings of the 31st International Conference on Distributed Computing Systems Workshops*. 166–171. DOI: https://doi.org/10.1109/ICDCSW.2011.20

Harrisen Scells, Leif Azzopardi, Guido Zuccon, and Bevan Koopman. 2018. Query variation performance prediction for systematic reviews. In *Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR'18)*. Association for Computing Machinery, New York, NY, 1089–1092. DOI: https://doi.org/10.1145/3209978.3210078

Ian Soboroff and Stephen Robertson. 2003. Building a filtering test collection for TREC 2002. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval (SIGIR'03)*. Association for Computing Machinery, New York, NY, 243–250. DOI: https://doi.org/10.1145/860435.860481

K. Spärck Jones. 1975. Report on the need for and provision of an "ideal" information retrieval test collection. *Computer Laboratory*. Retrieved from https://ci.nii.ac.jp/naid/10000151848/en/.

Hanna Suominen, Liadh Kelly, Lorraine Goeuriot, Aurélie Névéol, Lionel Ramadier, Aude Robert, Evangelos Kanoulas, Rene Spijker, Leif Azzopardi, Dan Li, Jimmy, João Palotti, and Guido Zuccon. 2018. Overview of the CLEF eHealth evaluation lab 2018. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction*. Springer International Publishing, Cham, 286–301.

Anastasios Tombros and Mark Sanderson. 1998. Advantages of query biased summaries in information retrieval. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'98)*. Association for Computing Machinery, New York, NY, 2–10. DOI: https://doi.org/10.1145/290941.290947

Stephen Tomlinson, Douglas W. Oard, Jason R. Baron, and Paul Thompson. 2007. Overview of the TREC 2007 legal track. In *Proceedings of the Text Retrieval Conference (TREC'07)*. Citeseer.

Ellen M. Voorhees, Donna K. Harman et al. 2005. *TREC: Experiment and Evaluation in Information Retrieval*. Vol. 63. MIT Press, Cambridge.

Byron C. Wallace, Issa J. Dahabreh, Kelly H. Moran, Carla E. Brodley, and Thomas A. Trikalinos. 2013. Active literature discovery for scoping evidence reviews: How many needles are there. In *Proceedings of the KDD Workshop on Data Mining for Healthcare (KDD-DMH'13)*.

Byron C. Wallace, Joël Kuiper, Aakash Sharma, Mingxi Zhu, and Iain J. Marshall. 2016. Extracting PICO sentences from clinical trial reports using supervised distant supervision. *J. Mach. Learn. Res.* 17, 1 (2016), 4572–4596.

Byron C. Wallace, Thomas A. Trikalinos, Joseph Lau, Carla Brodley, and Christopher H. Schmid. 2010. Semi-automated screening of biomedical citations for systematic reviews. *BMC Bioinform.* 11, 1 (2010), 55.

Yao Wan, Guandong Xu, Liang Chen, Zhou Zhao, and Jian Wu. 2018. Exploiting cross-source knowledge for warming up community question answering services. *Neurocomputing* 320 (2018), 25–34. DOI: https://doi.org/10.1016/j.neucom.2018.08.012

Xuan Wang, Yu Zhang, Xiang Ren, Yuhao Zhang, Marinka Zitnik, Jingbo Shang, Curtis Langlotz, and Jiawei Han. 2019. Cross-type biomedical named entity recognition with deep multi-task learning. *Bioinformatics* 35, 10 (2019), 1745–1752.

Zheng Wen, Branislav Kveton, Brian Eriksson, and Sandilya Bhamidipati. 2013. Sequential Bayesian search. In *Proceedings of the 30th International Conference on International Conference on Machine Learning (ICML'13)*. JMLR.org, II–226–II–234. Retrieved from http://dl.acm.org/citation.cfm?id=3042817.3042919.

Chenyan Xiong and Jamie Callan. 2015. EsdRank: Connecting query and documents through external semi-structured data. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management (CIKM'15)*. Association for Computing Machinery, New York, NY, 951–960. DOI: https://doi.org/10.1145/2806416.2806456

Chenyan Xiong, Jamie Callan, and Tie-Yan Liu. 2017. Word-entity duet representations for document ranking. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'17)*. Association for Computing Machinery, New York, NY, 763–772. DOI: https://doi.org/10.1145/3077136.3080768

Zhe Yu, Nicholas A. Kraft, and Tim Menzies. 2016. How to read less: Better machine-assisted reading methods for systematic literature reviews. *arXiv preprint arXiv:1612.03224* (2016).

Zhe Yu, Nicholas A. Kraft, and Tim Menzies. 2018. Finding better active learners for faster literature reviews. *Empir. Softw. Eng.* 23, 6 (2018), 3161–3186.

Chengxiang Zhai and John Lafferty. 2001. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings of the 10th International Conference on Information and Knowledge Management (CIKM'01)*. Association for Computing Machinery, New York, NY, 403–410. DOI: https://doi.org/10.1145/502585.502654

Haotian Zhang, Mustafa Abualsaud, Nimesh Ghelani, Mark D. Smucker, Gordon V. Cormack, and Maura R. Grossman. 2018a. Effective user interaction for high-recall retrieval: Less is more. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM'18)*. Association for Computing Machinery, New York, NY, 187–196. DOI: https://doi.org/10.1145/3269206.3271796

Haotian Zhang, Gordon V. Cormack, Maura R. Grossman, and Mark D. Smucker. 2018c. Evaluating sentence-level relevance feedback for high-recall information retrieval. *arXiv preprint arXiv:1803.08988*.

Haotian Zhang, Wu Lin, Yipeng Wang, Charles L. A. Clarke, and Mark D. Smucker. 2015. WaterlooClarke: TREC 2015 total recall track. In *Proceedings of the Text Retrieval Conference (TREC'15)*.

Yongfeng Zhang, Xu Chen, Qingyao Ai, Liu Yang, and W. Bruce Croft. 2018b. Toward conversational search and recommendation: System ask, user respond. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM'18)*. ACM, New York, NY, 177–186.

Zhou Zhao, Qifan Yang, Deng Cai, Xiaofei He, and Yueting Zhuang. 2016. Expert finding for community-based question answering via ranking metric network learning. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI'16)*. AAAI Press, 3000–3006.

Zhou Zhao, Lijun Zhang, Xiaofei He, and Wilfred Ng. 2014. Expert finding for question answering via graph regularized matrix completion. *IEEE Trans. Knowl. Data Eng.* 27, 4 (2014), 993–1004.

Jie Zou and Evangelos Kanoulas. 2019. Learning to ask: Question-based sequential Bayesian product search. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM'19)*. Association for Computing Machinery, New York, NY, 369–378. DOI: https://doi.org/10.1145/3357384.3357967

Jie Zou, Dan Li, and Evangelos Kanoulas. 2018. Technology-assisted reviews: Finding the last few relevant documents by asking yes/no questions to reviewers. In *Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR'18)*. Association for Computing Machinery, New York, NY, 949–952. DOI: https://doi.org/10.1145/3209978.3210102

Jie Zou, Ling Xu, Weikang Guo, Meng Yan, Dan Yang, and Xiaohong Zhang. 2015. Which non-functional requirements do developers focus on? An empirical study on stack overflow using topic analysis. In *Proceedings of the IEEE/ACM 12th Working Conference on Mining Software Repositories*. IEEE, 446–449.

Jie Zou, Ling Xu, Mengning Yang, Xiaohong Zhang, and Dan Yang. 2017. Toward comprehending the non-functional requirements through Developers' eyes: An exploration of Stack Overflow using topic analysis. *Info. Softw. Technol.* 84 (2017), 19–32.

Jie Zou, Ling Xu, Mengning Yang, Xiaohong Zhang, Jun Zeng, and Sachio Hirokawa. 2016. Automated duplicate bug report detection using multi-factor analysis. *IEICE Trans. Info. Syst.* 99, 7 (2016), 1762–1775.

Received August 2019; revised February 2020; accepted March 2020