When to Stop Reviewing in Technology-Assisted Reviews
Sampling from an Adaptive Distribution to Estimate Residual Relevant Documents
Li, D.; Kanoulas, E.
DOI
10.1145/3411755
Publication date
2020
Document Version
Final published version
Published in
ACM Transactions on Information Systems
License
Article 25fa Dutch Copyright Act (https://www.openaccess.nl/en/in-the-netherlands/you-share-we-take-care)
Link to publication

Citation for published version (APA):
When to Stop Reviewing in Technology-Assisted Reviews: Sampling from an Adaptive Distribution to Estimate Residual Relevant Documents

DAN LI and EVANGELOS KANOULAS, University of Amsterdam

Technology-Assisted Reviews (TAR) aim to expedite document reviewing (e.g., medical articles or legal documents) by iteratively incorporating machine learning algorithms and human feedback on document relevance. Continuous Active Learning (CAL) algorithms have demonstrated superior performance compared to other methods in efficiently identifying relevant documents. One of the key challenges for CAL algorithms is deciding when to stop displaying documents to reviewers. Existing work either lacks transparency—it provides an ad-hoc stopping point, without indicating how many relevant documents are still not found, or lacks efficiency by paying an extra cost to estimate the total number of relevant documents in the collection prior to the actual review.

In this article, we handle the problem of deciding the stopping point of TAR under the continuous active learning framework by jointly training a ranking model to rank documents, and by conducting a “greedy” sampling to estimate the total number of relevant documents in the collection. We prove the unbiasedness of the proposed estimators under a with-replacement sampling design, while experimental results demonstrate that the proposed approach, similar to CAL, effectively retrieves relevant documents; but it also provides a transparent, accurate, and effective stopping point.

CCS Concepts: • Information systems → Retrieval effectiveness; Retrieval efficiency;

Additional Key Words and Phrases: Total recall, active sampling, unbiased estimator

ACM Reference format:
https://doi.org/10.1145/3411755

1 INTRODUCTION

Given an information need, exhaustive search aims to retrieve all the relevant information, if possible. There is a real demand for exhaustive search in domains like electronic discovery, systematic review, investigation, research, or even the construction of datasets for information retrieval evaluation [13, 14, 16, 20–22, 24, 26, 32]. Electronic discovery involves searching electronic business
records such as correspondence, memos, emails, and balance sheets for documents that are relevant or responsive to a lawsuit or a government investigation. It is an important aspect of the civil litigation process in the United States [27]. Missing relevant business records may cause significant impact on a lawsuit or government investigation. Systematic review is a type of literature review that uses systematic methods to reliably bring together the findings from multiple studies that address a question and are often used to inform policy and practice, e.g., the development of medical guidelines in evidence-based medicine [28]. Test collections are fundamental in Information Retrieval (IR) evaluation. Assessing large-scale relevance labels is inevitable in order to build a high-quality test collection. Missing relevant labels in test collections causes bias when evaluating retrieval models [14]. The electronic business records, studies, or documents in these domains are usually large and their number is growing rapidly, making the task of identifying all relevant information both complex and time consuming. Technology-Assisted Reviews (TAR) tackles the problem by using classification or ranking algorithms to identify the relevant documents based on relevance feedback from expert reviewers, until a substantial number (or all) of the relevant documents have been identified.

The Continuous Active Learning (CAL) approach and its extensions have demonstrated high effectiveness when used in the TAR process [7, 8, 10, 12]. Given a document collection and a query, a ranker (or a classifier) is trained to identify documents to be shown to expert reviewers for relevance assessment. Then, the assessed documents are used as training data to re-train the ranker. As more and more documents are identified by the ranker and assessed by the reviewers, the training data grows and the performance of the ranker improves. The TAR process continues until “enough” relevant documents have been found. “Enough” is often specified by the additional cost (in terms of reading irrelevant documents) required to find more relevant documents, or by the importance of the missing relevant documents, e.g., toward resolving a legal dispute [7], or even as a percentage of relevant documents. In this work, we assume the latter, i.e., the experts specify a desired recall level to determine when should the reviewing process end.

In this case, the goal of the ranker is two-fold: first, to identify relevant documents as early as possible during the TAR process, and second, to accurately estimate the total number of relevant documents, $R$, in the collection so that we can stop the TAR process in a transparent manner, when we reach the required level of recall.

The two goals are, to some extent, conflicting in terms of the optimal strategy of stopping the TAR process. To demonstrate this, consider the following example. Suppose we have an unknown perfect ranking model that ranks all the relevant documents before the non-relevant documents. The most effective strategy in terms of achieving high recall is to select documents starting from the top of the list and moving toward the bottom. However, in order to know the total number of relevant documents in the collection, one needs to examine all the documents in the collection. On the other hand, a random sampling design can produce an unbiased estimator of the total number of relevant documents, $R$, by selecting a relatively low number of documents to review, leading, however, to low recall if the sampled documents are the only ones the reviewer reads.

There is little work that tackles the problem of automatically stopping the TAR process and they follow two directions. The first direction [8, 15, 25] proposes different methods to determine an ad-hoc stopping point. However, such approaches lack transparency because they provide no information on how many relevant documents are still not found. The second direction [9, 36] first samples documents to obtain an unbiased estimate of $R$, paying a significant assessment cost, and then employs a TAR method to rank and find the required number of relevant documents.

---

1A sampling design contains a sampling distribution, the manner to sample documents from the distribution (e.g., with replacement or without replacement), and some statistical estimators of desired values such as population total [35].

None of the directions that we just described can both produce effective rankings and support, transparently, the decision to stop reviewing documents.

In this work, we tackle the problem of determining the stopping point of document selection for constructing test collections that balance the cost of assessing document relevance and the gain of identifying relevant documents by proposing a novel framework for the TAR process. The framework consists of a ranking module, a sampling module, an assessment module, an estimation module, and a stopping module (see Figure 1 and Section 3.1). The ranking module provides a ranked list of documents based on their predicted relevance. The sampling module consists of a sampling distribution and a manner to sample documents from the distribution. The estimation module provides an estimator of $R$ as well as its variance based on the sampling distribution and the sampled documents. The stopping module determines whether to stop the TAR process. Our framework approaches TAR in an active learning manner following the CAL approach [8]. The major difference from the CAL approach is that we allow random sampling from all the documents in the collection instead of greedily assessing documents from top to bottom, in a with-replacement sampling design that can both collect many relevant documents (Section 3.2) and produce an unbiased estimator of $R$ with low variance (Section 3.3). Further, we devise different stopping strategies based on the estimated $R$ and its variance (Section 3.5).

To summarize, in this work, we make the following contributions:

— We propose a novel framework for the TAR process that allows us to conduct a “greedy” sampling over documents in order to collect as many relevant documents as possible, and produce a sequence of statistically unbiased estimators of $R$.
— We provide a proof of the unbiasedness of the proposed estimators of $R$ under our sampling design and also empirically verify the unbiasedness.
— We validate the effectiveness of the proposed framework and provide a detailed analysis of its components on various datasets including the Conference and Labs of the Evaluation Forum (CLEF) Technology-Assisted Reviews in Empirical Medicine datasets [20–22], the Text Retrieval Conference (TREC) Total Recall datasets [16], and the TREC Legal datasets [13].
— We reproduce a large number of baselines, and we release the code, along with the code of our work.

2 RELATED WORK

In this section, we first introduce the CAL approaches since our framework is developed based on these approaches. Then, we compare the merits and drawbacks of different methods that tackle the problem of stopping the TAR process. Finally, we explain the main differences between our method and existing work.

2.1 Continuous Active Learning Approaches

The TAR process aims to iteratively retrieve a substantial number (if not all) of the relevant documents in a collection—which makes it a total recall problem. Cormack and Grossman proposed a family of CAL approaches used in various total recall tasks, including technology-assisted reviews in electronic discovery and in empirical medicine, as well as the construction of test collections in information retrieval [7, 8]. The first AutoTAR method proposed by Cormack and Grossman [7] mainly consisted of an initial selection of training documents by a simple keyword search, and subsequent selections by active learning. The method significantly outperformed simple active learning with uncertainty sampling in terms of human reviewing cost. Cormack and Grossman [8] later proposed a different instantiation of AutoTAR, which enhances the first CAL method through a handful of adaptions including the Term Frequency–Inverse Document
Frequency (TF-IDF) features, a single relevant seed document, pseudo non-relevant documents, and exponentially increasing batch size of documents to be reviewed at each iteration. AutoTAR is considered the current state-of-the-art method for total recall tasks.

However, AutoTAR leaves the question of when to terminate the reviewing process open. In practice, there is no way to know the number of relevant documents in a collection before inspecting and labeling every document. It is thus impossible to know exactly what level of recall has been reached during the process. Hence, one is facing a dilemma between high recall and low cost in terms of reviewing documents. Stopping reviewing documents too early will result in missing many valuable relevant documents; stopping too late will cause unnecessary cost when there are no more relevant documents to be found.

So far, researchers have developed various approaches to solve the “stopping” problem. We introduce them in the following two sections.

### 2.2 Topic-Wise Approaches

In their AutoTAR experiment, Cormack and Grossman [8] observed that the gain curve (i.e., recall as a function of the number of documents reviewed) shows clearly diminishing returns at some point, and that if a substantial number of relevant documents have been found with high precision and precision drops later on, the vast majority of relevant documents have most likely been found. Inspired by this observation, Cormack and Grossman [9] proposed the Knee method, the Target method, and the Budget method. The Knee method defines a knee of the gain curve through a simple geometric algorithm [33]. The TAR process stops when the slope after the knee diminishes to less than a ratio (e.g., $\frac{1}{6}$) of the slope before the knee. In the Target method Cormack and Grossman [9] first review a randomly sampled set of documents, until a pre-defined number (e.g., 10) of them are judged relevant, which is the target set. The documents in the collection are ranked and retrieved within the AutoTAR framework without knowledge of the target set, until each document in the target set has been retrieved. In the Budget method Cormack and Grossman [9] combine the Knee and the Target method. The TAR process stops when all the documents in the target set have been retrieved and the slope after the knee diminishes to less than a ratio (e.g., $\frac{1}{6}$) of the slope before the knee.

It has been shown empirically that the Knee method is the most effective one [9]. However, the Knee method is a “blind” method, and it does not indicate how many relevant documents have not been found.

To solve the aforementioned problem, Cormack and Grossman [10] proposed the Scalability of Continuous Active Learning (SCAL) method, which is designed to achieve high recall for a large scale or infinite document collection. In the first step, SCAL randomly samples a large subset of documents from the document collection in order to have an accurate $\hat{R}$ – the estimator of $R$. Stratified sampling is applied to the subset to calculate $\hat{R}$: it first splits the subset into many small strata, and then randomly samples documents from each stratum to estimate the total number of relevant documents in each stratum; finally, $\hat{R}$ is calculated by summing up the total number of relevant documents over all the strata and multiplying it by a calibration factor. At each iteration, a ranking model is also trained by using the sampled labeled documents as the training data. In the second step, $\hat{R}$ is used to define a threshold with which SCAL can select a ranking model from the sequence of ranking models and produce a ranked list of documents for the reviewer to review.

Wallace et al. [36] proposed several estimators of $R$ and let reviewers decide when to stop by showing them how close they are to $\hat{R}$. However, there is no guarantee that the estimators they proposed are statistically unbiased.

Di Nunzio [15] proposed a thresholding method by investigating the interaction between the probability of observing document $d$ given the current relevant documents – $p(d | R)$ and the same...
probability given the non-relevant documents – \( p(d \mid NR) \). First, a document \( d \) is represented as two coordinates \((p(d \mid NR), p(d \mid R))\), denoted by \((x, y)\). Then, the original problem of classifying (or ranking) documents can be transformed into an intuitive geometric problem of finding two decision lines in the form of \( y = ax + \beta \): (1) a line \( y = ax + \beta_{rel} \) that fits the existing assessed relevant documents, and (2) a parallel line \( y = ax + \beta_{least} \) passing through the least relevant documents. The TAR process stops when all the documents between the two lines are assessed.

Modeling the distribution of relevant and non-relevant documents and fitting them over scores in a reasonable way, which is called score distribution method [6, 23, 37], has been studied since the early days of information retrieval and is beneficial to a wide range of applications, as well as the “stopping” problem [1, 30]. Hollmann and Eickhoff [18] proposed a thresholding method based on score distributions. It first fits a Gaussian distribution to the scores of relevant articles from the relevance feedback of random sampled articles, and then estimates the total number of relevant documents at any rank position.

### 2.3 Cross-Topic Approaches

Losada et al. [25] proposed a stopping method for accurate evaluation of retrieval systems using partially labeled test collections, with the estimation of \( R \) being a byproduct. During the training phase, a document pool, which consists of multiple ranked lists of documents from different retrieval systems is formed; a power law function is used to model the number of new relevant documents at each pool depth; the number of relevant documents is also known; during the test phase, for a test query, a certain number of documents are assessed based on which the similarity between the pattern of relevance of the current test query and the pattern of relevance of each training query is calculated; finally, \( R \) is estimated by averaging the number of relevant documents of each training query weighted by their similarity.

Although the work of Losada et al. [25] studies the stopping problem, its goal is to achieve an accurate evaluation of retrieval systems, and not high recall, per se. Besides, it assumes the existence of different retrieval systems, which produce multiple ranked lists. Pooling these systems is critical since it is then that the number of new relevant documents at each pool depth follows a power law distribution, based on which the method is designed. The method could be adapted, reducing the pool of ranking systems to a single ranking, and the pool of documents to the top-\( k \) documents of that ranking. Despite the fact that such a method could be used toward stopping the reviewing process, such an adaptation goes beyond the intentions of Losada et al. [25].

### 2.4 Summary

Inspired by Cormack and Grossman [8–10], we propose to jointly estimate \( R \) and improve the ranker at each iteration by “greedily” sampling documents. Our method is able to decide when to stop the TAR process with transparency—by estimating how many relevant documents are still missing. Such a model has several merits. First, it is topic-wise independent, which means no extra training topics are needed. Second, it does not need an extra procedure to estimate \( R \); instead, it iteratively utilizes the sampling module to collect relevant documents as well as to estimate \( R \). Third, instead of estimating \( R \) once, it produces \( \hat{R} \) at every iteration, compensating for variance and reducing the risk of very wrong estimations. Fourth, it calculates \( \hat{R} \) as well as the variance estimator of \( \hat{R} \), enabling \( \hat{\text{var}}(\hat{R}) \) to also contribute to the stopping decision.

### 3 Method

In this section, we first introduce the overall framework; then we elaborate on the sub-modules. The notation used throughout this section is summarized in Table 1.
Table 1. Notation Used in Section 3

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q$</td>
<td>The input topic</td>
</tr>
<tr>
<td>$C_q$</td>
<td>Document collection for topic $q$</td>
</tr>
<tr>
<td>$N$</td>
<td>Total number of documents in $C_q$</td>
</tr>
<tr>
<td>$R$</td>
<td>Total number of relevant documents in $C_q$</td>
</tr>
<tr>
<td>$S$</td>
<td>Set of sampled documents</td>
</tr>
<tr>
<td>$n$</td>
<td>Total number of documents in $S$</td>
</tr>
<tr>
<td>$t$</td>
<td>Iteration number</td>
</tr>
<tr>
<td>$b_t$</td>
<td>Number of documents to be sampled at $t$-th iteration</td>
</tr>
<tr>
<td>$k$</td>
<td>Number of pseudo-relevant documents added per iteration</td>
</tr>
<tr>
<td>$L_t$</td>
<td>Labeled documents (training set) at $t$-th iteration</td>
</tr>
<tr>
<td>$U_t$</td>
<td>Unlabeled documents at $t$-th iteration</td>
</tr>
<tr>
<td>$d_i^t$</td>
<td>Document indexed by $i$ at $t$-th iteration</td>
</tr>
<tr>
<td>$r_i^t$</td>
<td>Rank of $d_i^t$</td>
</tr>
<tr>
<td>$y_i^t$</td>
<td>Relevance label of $d_i^t$</td>
</tr>
<tr>
<td>$p_i^t$</td>
<td>Selection probability of $d_i^t$</td>
</tr>
<tr>
<td>$\pi_i$</td>
<td>First-order inclusion probability of $d_i$</td>
</tr>
<tr>
<td>$\pi_{i,j}$</td>
<td>Second-order inclusion probability of $d_i$ and $d_j$</td>
</tr>
<tr>
<td>$\cdot$</td>
<td>Operation of removing duplicates</td>
</tr>
<tr>
<td>$\hat{}$</td>
<td>Operation of cumulating units in previous iterations</td>
</tr>
<tr>
<td>$\tilde{}$</td>
<td>Estimator of a variable or statistic</td>
</tr>
</tbody>
</table>

3.1 The Framework

We propose a novel framework for the TAR process and we call it autostop. The framework mainly consists of a ranking module, a sampling module, an assessment module, an estimation module, and a stopping module (see Figure 1).

Given a topic, a reviewer is interested in, denoted by $q$, a collection of documents for the topic, denoted by $C_q$, the reviewer can specify a target recall level, denoted by $recall_t$, which indicates
what proportion of relevant documents retrieved by the framework makes the reviewer satisfied. The framework iteratively trains a ranking model to produce a ranked list of documents from which it samples documents for the reviewer to read toward satisfying her or his information need; then, the framework outputs an estimator of the total number of relevant documents of the collection, and an estimator of the variance of the estimator. The process stops once the estimated recall exceeds the target recall.

We describe the framework in Algorithm 1. We start with the ranking and the sampling process. A ranking model will be trained from scratch and produce a ranked list for the documents in $C_q$. Let $L_t$ denote the (labeled) training set for the ranking model at iteration $t$, and $U_t$ denote the unlabeled set at iteration $t$. Initially, $L_0$ is empty and we fill it with a pseudo document $d_0$. We follow the AutoTAR method [8] to construct the pseudo document, i.e., we use the description of the topic. For the cases where there is no description for the topic, one can always use the query expanded by relevant feedback. Next, we also add non-relevant documents into the training set. At each iteration, $L_t$ is augmented with $k$ documents, which are sampled uniformly and without replacement from $U_t$ (in line 3). These documents are temporarily labeled non-relevant, same as the AutoTAR method [8]. In line 4, a ranking model is trained on $L_t$. In line 5, the ranking model predicts the probability of relevance over all the documents in $L_t$ and $U_t$ (which equals $C_q$). These documents are ranked in the order of decreasing relevance probability. In line 6, a sampling distribution $P_t$ is constructed based on the ranked list of documents. We propose to use the AP Prior distribution and provide the details in Section 3.2. In line 7, a number of $b_t$ documents are sampled independently and with replacement from $P_t$.

After the documents are sampled, we start the assessment process. In line 8, the reviewer assesses the relevance of the sampled documents. Note that the sampled $b_t$ documents may contain duplicates; therefore, the reviewer only needs to assess the unique documents. Moreover, the sampling design is with-replacement; therefore, it is possible to sample documents that have been assessed before. As a consequence, the reviewer only assesses at most $b_t$ documents. In line 9, we follow the same method with the AutoTAR method [8], i.e., we remove the temporary documents from $L_t$. Meanwhile, we add the $b_t$ assessed documents (which may contain duplicates) in $L_t$ and remove them from $U_t$.

Given the sampled documents, their relevance labels, and their sampling probabilities, we can estimate the total number of relevant documents—denoted by $\hat{R}_t$, as well as the variance of $\hat{R}_t$—denoted by $\hat{\text{var}}(\hat{R}_t)$. They are calculated in line 10. We propose to use two estimators for $R_t$, which are the Horvitz-Thompson estimator and the Hansen-Hurwitz estimator. We provide the details in Section 3.3.

Finally, the stopping module uses $\hat{R}_t$ and $\hat{\text{var}}(\hat{R}_t)$ to decide whether to stop the TAR process or not. We propose two stopping strategies, an optimistic and a conservative one, and we explain them in detail in Section 3.5.

### 3.2 Sampling Design

In this section, we elaborate on our sampling design with a focus on the sampling distributions. Note that in Algorithm 1, we need to sample documents from a distribution $P_t = \{p'_i\}$ (for notation simplicity, we omit $t$ and use $P = \{p_i\}$). Ideally, $p_i$ should be positively correlated with the relevance label at position $i$, which allows an estimator $\hat{R}$ with low variance (see the explanation in Section 3.3). However, the relevance labels are not known until documents are assessed by the reviewers. What we have instead is a ranking model that can predict relevance probability and output a list of ranked documents. Therefore, we construct $\mathcal{P}$ based on the output ranked list of documents.
ALGORITHM 1: Automatic thresholding algorithm

Input: Topic \( q \); document collection \( C_q \), target recall \( \text{recall}_t \).
Output: A list of labeled documents \( \{(d_t^i, y_t^i) \mid i = 1, 2, \ldots; t = 1, 2, \ldots\} \); a list of estimator \( \{\hat{R}_t \mid t = 1, 2, \ldots\} \) and \( \{\hat{\text{var}}(\hat{R}_t) \mid t = 1, 2, \ldots\} \).

1. \( t = 0 \), \( L_0 = \{\text{pseudo document } d_0\} \);
2. while not stop do
3. \( t += 1 \);
   // Sampling
4. Temporarily augment \( L_t \) by uniformly sampling \( k \) documents from \( U_t \), labeled non-relevant;
5. Train a ranking model on \( L_t \);
6. Rank all the documents in \( C_q \) with the ranker trained over \( L_t \);
7. Construct sampling distribution \( P_t \) over the ranked documents via Equation (1);
8. Sample \( b_t \) documents from the distribution \( P_t \);
9. Render relevance assessments for the sampled documents;
10. Remove the \( k \) temporary documents from \( L_t \);
11. Place the \( b_t \) assessed documents in \( L_t \), and remove them from \( U_t \);
12. \( b_{t+1} = b_t + \lceil b_t \cdot \frac{1}{10}\rceil \);
   // Estimation
13. Calculate \( \hat{R}_t \) and \( \hat{\text{var}}(\hat{R}_t) \) via Equation (5–9);
   // Stopping
14. if \( \hat{R}_t \) and \( \hat{\text{var}}(\hat{R}_t) \) satisfy stopping strategy then
15. \( \text{stop} = \text{True} \);
16. end
end

The probability over document rank is defined as the probability of sampling a document at a certain rank of the ranked list. The underlying assumption is that a ranking model satisfies the Probability Ranking Principle (PRP) \([31]\), which dictates that the probability of relevance monotonically decreases with the rank of the document. Any distribution that agrees with PRP can be used.

When choosing the sampling distribution, we are faced with a tradeoff between collecting as many relevant documents as possible and an accurate estimator of \( R \) with low variance. There are many ways to construct a distribution to sample from, for example, sampling from the output probability of relevance produced by the trained ranking model, or sampling from a power law distribution produced based on document ranks. We tried both and neither performed well enough. To meet the goal, we propose to use AP-prior distribution. It is a widely used sampling distribution in the task of information retrieval evaluation, and it has been empirically proved to be a good prior for the relevance of documents in a ranked list \([2–5, 24, 38, 39]\).

**AP-prior Distribution.** The AP-prior distribution is proposed in Refs \([4]\) and \([29]\), which defines the probability at each rank position on the basis of the contribution of this rank position in the calculation of average precision. The idea is that we first rewrite *average precision* as \( AP = \frac{1}{N} \sum_{1 \leq i < j \leq N} \frac{1}{2} y_i y_j \), where \( i, j \) denote the position in the ranked list, and \( y \) denotes the relevance label; then we can construct a distribution of random variable \( (y_i, y_j) \) over position pairs \( (i, j) \) that satisfies the expectation \( E(y_i, y_j) \) equals \( AP \); finally, we add up all the probabilities associated
with all pairs involving a given position $r$ and get $p (r) = \frac{1}{2N} \left( 1 + \frac{1}{r} + \frac{1}{r+1} + \cdots + \frac{1}{N} \right) \approx \frac{1}{2N} \log \frac{N}{r}$.

For more details, the readers are referred to Refs [4] and [29].

Thus, the AP-prior distribution is defined as follows:

$$p_i = \frac{1}{Z} \log \frac{N}{r_i}, \quad r_i \in [1, N],$$  \hspace{1cm} (1)

where $Z = \sum_{i=1}^{N} \log \frac{N}{r_i}$ is the normalization factor.

**Inclusion probability.** In order to have a simple expression for inclusion probability, we adopt *sampling with replacement* as our sampling method. At each iteration $t$ and for each draw, a document is sampled independently from one of the aforementioned distributions. Let selection probability denote the probability that a document is sampled for a draw, and inclusion probability the probability that a document is included in the sample set considering all the draws. The first-order and second-order inclusion probabilities under sampling with replacement are indispensable to calculate $\hat{R}$. Under sampling-with-replacement design, the first-order inclusion probability $\pi_i$ is given by:

$$\pi_i = 1 - \prod_{t=1}^{T} \left( 1 - p_i^t \right)^{n_t}. \hspace{1cm} (2)$$

The second-order inclusion probability $\pi_{i,j}$—the probability of any two different document $d_i$ and $d_j$ being included—is given by:

$$\pi_{i,j} = \pi_i + \pi_j - \left[ 1 - \prod_{t=1}^{T} \left( 1 - p_i^t - p_j^t \right)^{n_t} \right]. \hspace{1cm} (3)$$

### 3.3 Estimate $R$ and $\text{var}(\hat{R})$

As mentioned in Section 1, one of our goals is to estimate $R$—the total number of relevant documents in the document collection for a given topic. In this section, we first clarify the exact expression of $R$ with regard to the population, then introduce its estimators on the sample set. Also, for notation simplicity, sometimes we omit the subscript $t$.

Let $C_q = \sum_{i=1}^{N} d_i$ denote a population of documents, and let $y_i$ be an indicator variable of $d_i$, with $y_i = 1$ if the document $d_i$ is relevant, and $y_i = 0$ otherwise. The population total is defined as the summation of all $y_i$, i.e., the total number of relevant documents in the collection. We write $R$ as follows:

$$R = \sum_{i=1}^{n} y_i. \hspace{1cm} (4)$$

Suppose our framework produces a sample set at each iteration $t$, denoted by $S_t$. Let $\tilde{S}_t$ denote the cumulated sample set till iteration $t$, $\tilde{S}_t = \cup_{k=1}^{t} S_k$. Furthermore, we also let $S'_t$ denote the subset of $\tilde{S}_t$ and $\tilde{S}'_t$ only contains unique documents. Correspondingly, let $n_t$ be the number of documents in $S_t$, $\tilde{n}_t$ be the number of documents in $\tilde{S}_t$, and $\tilde{n}'_t$ be the number of documents in $\tilde{S}'_t$.

Note that when sampling documents, each document has different sampling probability compared to other ones; therefore, we can not apply a simple estimator for equal sampling probability (e.g., $\frac{N}{\tilde{n}_t} \sum_{i \in \tilde{S}_t} y_i$) to estimate $R$. Instead, we employ Horvitz-Thompson estimator and Hansen-Hurwitz estimator, both of which can estimate $R$ as well as $\text{var}(\hat{R})$.

Both of them are designed for sampling with unequal probabilities, Hansen-Hurwitz estimator is only restricted for with-replacement sampling, while Horvitz-Thompson estimator is for any design. On the other hand, the inclusion probability of Horvitz-Thompson estimator is not easy.
to calculate when the sampling design is complex. One can choose to use one of the estimators according to their sampling design. We provide the proof of unbiasedness of the two estimators under our sampling design in Section 3.4. We also provide the theoretical conditions of zero variance estimators in this section. For more details of the derivation, the reader can refer to Ref. [35].

3.3.1 Horvitz-Thompson Estimator. The Horvitz-Thompson estimator provides an unbiased estimator of population total under any sampling design, with or without replacement [19]. It is written as 
\[ \hat{\tau} = \sum_{i \in S'} \frac{y_i}{\pi_i}, \]
where \( S' \) is a subset of \( S \), \( S' \) only contains unique units, and \( \pi_i \) is the inclusion probability for unit \( i \).

In our case, the population total is \( R \). The Horvitz-Thompson estimator of \( R \) on \( \tilde{S}_t \) is written as:
\[ \hat{R}^{HT}_t = \sum_{i \in \tilde{S}_t} \frac{y_i}{\pi_i}. \]  
Furthermore, it is good to know whether the estimator is stable. An estimator with low variance is desirable. In order to calculate the variance, we need to conduct the same sampling procedure many times, which is not feasible in practice. Luckily, there is an unbiased estimator of the variance of the population total estimator, given by:
\[ \text{var}1(\hat{R}^{HT}_t) = \sum_{i \in \tilde{S}_t} \left( \frac{1}{\pi_i^2} - \frac{1}{\pi_i} \right) y_i^2 + 2 \sum_{i > j} \left( \frac{1}{\pi_i \pi_j} - \frac{1}{\pi_{ij}} \right) y_i y_j. \]  
(6)

As \( \text{var}1(\hat{R}^{HT}_t) \) may produce negative values, the true variance must be non-negative. Hence, we also adopt another estimator of variance, given by:
\[ \text{var}2(\hat{R}^{HT}_t) = \frac{N - \tilde{n}'_t}{N \tilde{n}'_t - 1} \left( \frac{1}{\tilde{n}'_t} \frac{1}{\pi_i} - \hat{R}^{HT}_t \right)^2. \]  
(7)

Note that when \( \pi_i = \frac{\tilde{n}'_t}{N} \), the variance of \( \hat{R}^{HT}_t \) equals 0 [19].

3.3.2 Hansen-Hurwitz Estimator. Hansen-Hurwitz estimator provides an unbiased estimator of population total under sampling with replacement, and the sampling distribution should be the same for each draw [35]. In our case, the sampling distribution changes at each iteration and converges to the ultimate distribution produced by the ranking model trained on the whole documents. This is the major difference with the setting of the vanilla Hansen-Hurwitz estimator. In Section 3.4.1, we will prove that under varying sampling distributions at different iterations, \( \hat{R}^{HH}_t \) is also unbiased.

The Hansen-Hurwitz estimator of \( R \) on \( \tilde{S}_t \) is expressed as follows:
\[ \hat{R}^{HH}_t = \frac{1}{\tilde{n}_t} \sum_{k=1}^{T} \sum_{i \in \tilde{S}_k} \frac{y_i}{\tilde{p}_i^k}. \]  
(8)

\(^2\)It is biased yet “conservative” (meaning it tends to be larger than the actual variance). The intuition is taking the Horvitz-Thompson estimator as the average of i.i.d. random variable.

\(^3\)Note that if taken in the view of importance sampling, Hansen-Hurwitz estimator equals to the estimation of population total obtained by importance sampling, denoted by \( q_t = \frac{\tilde{n}'_t}{\tilde{n}_t} \frac{y_i}{\pi_i} \). The sample variance of \( q_t \) is defined by \( s^2 = \frac{1}{\tilde{n}'_t - 1} \sum_{i=1}^{\tilde{n}'_t} (q_i - \hat{R}^{HT}_t)^2 \). The alternative variance estimator is \( \text{var}2(\hat{R}^{HT}_t) = \frac{N - \tilde{n}'_t}{N \tilde{n}'_t - 1} s^2 \) [35].
Similarly, there is also an estimator of the variance of the population total estimator, given by:

\[
\text{var} \left( \hat{R}_t^{HH} \right) = \frac{1}{n_t (n_t - 1)} \sum_{k=1}^{t} \sum_{i \in S_k} \left( \frac{y_i}{p_i^k} - \hat{R}_t^{HH} \right)^2.
\]  

(9)

Note that when \( \frac{y_i}{p_i^k} \) is a constant, the variance of the Hansen-Hurwitz estimator equals 0 [35, page 68].

For both Horvitz-Thompson and Hansen-Hurwitz, there is no guarantee that their estimators of variance are unbiased.

3.4 Unbiasedness

3.4.1 Hansen-Hurwitz Estimator.

**Lemma 3.1.** Given that the sampling design is with replacement, and at each draw \( l \), a unit is independently sampled from a different distribution \( \{p_i^l\} \) from the previous draws. Let \( S \) denote the sample set. The Hansen-Hurwitz estimator of population total is unbiased, i.e., \( \mathbb{E} \left( \sum_{y_i^l \in S} \frac{y_i^l}{p_i^l} \right) = \tau \).

**Proof.** First consider the one unit case. Assume there is only one unit in \( S \); thus, there are \( N \) possible sample sets for \( S \). Let \( y_S \) denote the sample value and \( p_S \) the selection probability; thus, Hansen-Hurwitz estimator can be rewritten as \( \tau_S = \frac{y_S}{p_S} \). Its expected value over all possible sample sets is:

\[
\mathbb{E}(\tau_S) = \sum_{S=1}^{N} \tau_S p_S = \sum_{S=1}^{N} y_S = \tau.
\]  

(10)

For the \( n \) unit case, take each unit \( i \) as a sample set with only one unit, which is denoted by \( S_i \). Thus, \( \mathbb{E} (\tau_{S_i}) = \tau \).

As each \( S_i \) is independent from any other \( S_j \), we have

\[
\mathbb{E}(\tau_S) = \mathbb{E} \left( \frac{1}{n} \sum_{i=1}^{n} \tau_{S_i} \right)
= \frac{1}{n} \left( \sum_{i=1}^{n} \mathbb{E}(\tau_{S_i}) \right)
= \frac{n \tau}{n} = \tau.
\]  

(11)

Equation (11) to Equation (12) is because \( \{\tau_{S_i}\} \) is a set of independent variables; thus, we can swap expectation and summation. Equation (12) to Equation (13) is because Equation (10) holds. \( \square \)

3.4.2 Horvitz-Thompson Estimator. The unbiasedness of Horvitz-Thompson estimator is proved [35, page 76]. We rephrase the proof below for reader’s convenience.

**Lemma 3.2.** Given that the sampling design is with replacement, and at each iteration \( t \), a set of units are independently sampled from a different distribution \( \{p_i^t\} \) from the previous iterations. Let \( S' \) denote the sample set where all the duplicated units are removed. The Horvitz-Thompson estimator of population total is unbiased, i.e., \( \mathbb{E} \left( \sum_{y_i \in S'} \frac{y_i}{p_i} \right) = \tau \).

**Proof.** Let random variable \( z_i \) denote whether the \( i \)-th unit is included in the sample set: \( z_i = 1 \) means that the \( i \)-th unit is in the sample set and \( z_i = 0 \) means that the \( i \)-th unit is not in the sample set. For any unit \( i \), \( z_i \) is a Bernoulli random variable; thus, \( \mathbb{E} z_i = \rho \left( z_i = 1 \right) = \pi_i \)
The Horvitz-Thompson estimator can be rewritten as follows, where each \( y_i \) is regarded as a constant:

\[
\sum_{y_i \in S'} y_i = \sum_{i=1}^{N} y_i z_i \pi_i.
\]  

(14)

The expected value is:

\[
E\left(\sum_{y_i \in S'} y_i \pi_i \right) = E\left(\sum_{i=1}^{N} y_i z_i \pi_i \right) = \sum_{i=1}^{N} y_i E z_i \pi_i = \sum_{i=1}^{N} y_i = r.
\]  

(15)

Equations (15) to (16) are because \( \{y_i z_i \pi_i\} \) is a set of independent variables; thus, we can swap expectation and summation. Equation (16) to (17) is because \( E z_i = P(z_i = 1) = \pi_i \).

3.5 Stopping Strategy

We propose two stopping strategies based on \( \hat{R} \) and its estimated variance, \( \hat{\text{var}}(\hat{R}) \). They represent different levels of conservativeness in terms of deciding when to stop the TAR process.

For notational simplicity, we omit the subscript \( t \). Let \( r \) denote the number of unique relevant documents that have already been read by the reviewer. Let \( \hat{R} \) denote the estimated total number of relevant documents, \( \hat{\text{var}}(\hat{R}) \) the estimated variance, and \( \text{recall}_t \) the target recall specified by the reviewer.

**Optimistic.** This strategy examines whether \( \frac{r}{\text{recall}_t} \geq \hat{R} \), and if so, stops the TAR process. The intuition is straightforward: if we have collected more relevant documents than the target number we estimated, we should feel confident to stop.

**Conservative.** This strategy examines whether \( \frac{r}{\text{recall}_t} \geq \hat{R} + \sqrt{\hat{\text{var}}(\hat{R})} \), and if so, stops the TAR process. This inequality is more conservative than the first one; that is, it continues providing document to the reviewer until it accumulates a higher confidence that the target recall is reached.

4 EXPERIMENTAL SETUP

4.1 Research Questions

The goal of the TAR process is to identify as many relevant documents as possible with minimal cost of returned irrelevant documents, and to stop exactly at the target recall specified by the reviewer. Based on this, our research questions are as follows.

**RQ1** Can the framework autostop proposed in Section 3 achieve target recall with minimal assessment cost and stop the document selection process on time?

**RQ2** Are the estimated \( \hat{R} \) using the Horvitz-Thompson estimator (Section 3.3.1) and the Hansen-Hurwitz estimator (Section 3.3.2) unbiased with low variance?

**RQ3** How does the proposed framework trade off high recall against accurate \( \hat{R} \)?

**RQ4** How does the estimation module (Section 3.3) and the stopping module (Section 3.5) impact the performance of the model?
4.2 Dataset

We test our framework on a wide range of datasets including the CLEF Technology-Assisted Reviews in Empirical Medicine datasets (EMED for short), the TREC Total Recall datasets (TR for short), and the TREC Legal datasets (LEGAL for short).

**EMED dataset.** The TAR in EMED Task at CLEF aims at evaluating search algorithms that seek to identify all studies relevant for conducting a systematic review in empirical medicine.

Three datasets have been released—namely, EMED 2017 [20], EMED 2018 [21, 34], and EMED 2019 [22]. EMED 2017 and EMED 2018 only include Diagnostic Test Accuracy reviews, while EMED 2019 includes three new review types, which are Intervention, Prognosis, and Qualitative systematic reviews. We use all the Diagnostic Test Accuracy reviews in the three datasets for our experiments. There are 42, 30, and 31 reviews (topics), respectively. We select 12 topics in EMED 2017 as the training topics for hyper-parameter fine-tuning of baselines and our method. The 12 topics are selected in the way that their prevalence values ranging from small to large value. We use the 30 topics in EMED 2017, 30 topics in EMED 2018, and 31 topics in EMED 2019 as the test topics.

For each topic, a topic description, a subset of the PubMed abstracts that are related to the topic and need to be ranked, along with the relevance judgments of the studies in this set are provided. Each topic description provides a topic title, which is the title of the corresponding systematic review article, like “Galactomannan detection for invasive aspergillosis in immunocompromised patients.” The titles usually contain terms in biomedical literature, not like general queries users submit to search engines in their everyday life. The document collection—PubMed Baseline Repository—comprises biomedical literature from MEDLINE, life science journals, and online books.

**TR dataset.** The TREC Total Recall track [16] aims to evaluate methods designed to achieve very high recall—as close as 100%—with a human assessor in the loop.

We use the athome1 and athome4 dataset provided in the TREC 2016 Total Recall track. Athome1 contains 10 topics and athome4 contains 34 topics. We use the 10 topics in athome1 as the training topics and the 34 topics in athome4 as the test topics.

Each topic supplies a short topic title, typically one to three words, like “Olympics” or “bottled water”. The topics are similar to queries people submit to search engines in their everyday life. The document collection consists of the Jeb Bush’s emails. It consists of 290,099 emails from Jeb Bush’s eight-year tenure as Governor of Florida.

**LEGAL dataset.** The TREC Legal track [13] focuses on evaluation of search technology for discovery of electronically stored information in litigation and regulatory settings. We use the dataset provided in the interactive task in the TREC 2010 Legal track. It includes topic 301, 302, 303, and 304. As we did not find any other topics that have the same document collection and relevancy labels, we use 301 and 302 as the training topics and 303 and 304 as the test topics.

Each topic consists of a mock complaint and document requests, where the mock complaint sets forth the legal and factual basis for the hypothetical lawsuit that motivates the discovery requests, and the document requests specify the categories of documents that must be located and produced. The document collection is a processed variant of the Enron email dataset, containing about 685,592 email messages captured by the Federal Energy Review Commission (FERC) from Enron, in the course of its investigation of Enron’s collapse. Similarly with the TR dataset, we use the same strategy to generate the subset of documents for each topic.

Table 2 lists the statistics of the datasets used in our experiments. For more details of topic splits readers can refer to Table 8. We calculated prevalence, which is defined as the percentage of

---

5The TR dataset is publicly available conditioned on a usage agreement. https://plg.uwaterloo.ca/~gvcormac/total-recall/.
6The dataset is available at https://trec-legal.umiacs.umd.edu.
relevant documents in the given documents for a topic. Overall the prevalence is low for all the datasets, ranging from 0.10% to 2.15%, indicating it is not an easy task to retrieve all the relevant documents. Figure 2 further shows the boxplot of topic prevalence of the five test datasets. The three EMED datasets have larger prevalence than TR, followed by LEGAL.

### 4.3 Evaluation Metrics

The effectiveness of the proposed framework can be interpreted in three aspects: (1) to maximize the recall, (2) to minimize the assessment cost, and (3) to stop the document selection process on time in the sense that the difference between the achieved recall and the target recall is minimized.

The recall metric can be used for the evaluation of (1). It is formally defined as:

\[
recall = \frac{r}{R},
\]

where \( r \) is the number of relevant documents identified, and \( R \) is the total number of relevant documents.

The cost metric can be used for the evaluation of (2). It is formally defined as:

\[
cost = \frac{n}{N},
\]

where \( n \) is the number of documents shown to the reviewer for assessment, and \( N \) is the total number of documents for the topic.
The relative error between the achieved recall and the target recall can be used for the evaluation of (3). It is formally defined as:

$$RE = \frac{|recall_c - recall_t|}{recall_t},$$  \hspace{1cm} (20)

where \(recall_t\) denotes the target recall specified by the reviewer.

Following the convention in IR evaluation, it is usually desired to have one metric that considers all the three aspects. However, there does not exist such a metric. As an alternative, we use an existing metric \(loss_{er}\) for the combined evaluation of (1) and (2). \(loss_{er}\) is introduced for the first time in Ref. [9] and latter used as a metric for the total recall task in the CLEF technology assisted reviews in empirical medicine overview tracks [20, 21, 22]. It is defined as:

$$loss_{er} = (100\% - recall_c)^2 + \left(\frac{100}{N}\right)^2 \left(\frac{n}{R + 100}\right)^2,$$  \hspace{1cm} (21)

where \(recall_c\) is the achieved recall when the TAR process is stopped. The intuition behind \(loss_{er}\) is straightforward: \(100\% - recall_c\) expresses a loss due to the inability to find all relevant documents or to achieve a recall of 100%, and \(\frac{n}{R}\) expresses a loss in terms of the effort spent in assessing both relevant and non-relevant documents. Note that the 100% in the formula means that a target recall of 100% is wanted. One may propose to use \(recall_t - recall_c\) to replace \(100\% - recall_c\) in order to evaluate system performance given different target recall levels, but using \(recall_t - recall_c\) penalizes systems that achieve recall higher than the target recall. Also note that the rationale of using \(\frac{100}{N}\) and \(\frac{n}{R + 100}\) can be understood in this way: \(aR + b\) (in this case, \(a = 1, b = 100\)) represents a reasonable amount of effort to achieve high recall, which has been empirically shown in the TREC total recall tracks [16, 32]; \(n\) represents the actual effort spent in reviewing documents; \(\frac{n}{R + 100}\) actually says how many percentages of the reasonable effort is spent in reviewing documents. \(\frac{100}{N}\) is a fixed weight that balances the two types of loss. Readers are also recommended to read the original explanation in Ref. [9].

In addition to the aforementioned metrics, it is interesting to see how often the framework achieves a target recall over different topics because the proposed framework relies on a random sampling of documents. We use reliability [9] for the evaluation. reliability is defined as the percentage that an estimator achieves the target recall for all topics [9]. It is expressed as:

$$reliability = \frac{|\{q \mid recall_c(q) \leq recall_t(q), q \in Q\}|}{|Q|},$$  \hspace{1cm} (22)

where \(Q\) is a set of topics, and \(recall_c(q)\) and \(recall_t(q)\) are the achieved recall and the target recall for topic \(q\).

For RQ1, we use all the aforementioned metrics. For RQ2, we compare the difference between \(R\) and \(\hat{R}\) through a scatter plot. For RQ3, we interpret the ability of identifying relevant documents as the gain, and the assessment effort spent in reviewing documents as the loss. We use a curve of recall as a function of cost for the evaluation. RQ4 is an effectiveness analysis of the submodules of the proposed framework; therefore, we use the same metrics as in RQ1.

All the metrics are calculated using open-sourced script tar_eval.\(^7\)

### 4.4 Baselines

We compare our approach with several representative baselines, including two greedy methods designed for high recall and early stopping, namely Knee [9] and Target [9], one hybrid method of greedy method and sampling method designed for high recall and early stopping, namely SCAL

\[^7\text{https://github.com/CLEF-TAR/tar.}\]
and two variants of Score Distribution [18], which are designed for high recall and accurate estimator of $R$. The Knee method is the state-of-the-art method for the total recall task. In what follows, we elaborate on the baseline methods and their precise implementation.

### 4.4.1 AutoTAR.

AutoTAR [8] aims to improve recall by repeatedly selecting documents for users to review. AutoTAR is considered the current state-of-the-art method for total recall problems. As it lays the foundation of several methods, we provide the algorithm in Algorithm 2.

We are faced with several design choices in implementing the method. For example, in line 6 of Algorithm 2, when training the ranking model, we are faced with choices such as the corpus and the minimum term frequency used for the TF-IDF vector of documents, and the regularization strength weight used for the Logistic Regression.

We ran our implementation of the AutoTAR method against the 30 test topics in the 2017 CLEF technology-assisted reviews in empirical medicine track, as this allows us to compare the results of our implementation with those reported in Ref. [11]. We use grid search to sweep over all possible configurations. The corpus varies between the document texts of the whole 30 topics or only of the target topic. The minimum term frequency varies among 2, 3, or 5. The regularization weight varies among 0.001, 1.0, and 10,000. We find that the best configuration is to use document texts of only the target topic as the corpus, set the minimum term frequency to 2, set the regularization weight to 1.0, which is slightly different from that reported in Ref. [11]. We report metrics that are also reported in Ref. [11]: they are $ap$—average precision, $norm \_area$—area under the cumulative recall curve normalized by the optimal area, and $last \_rel$—minimum number of documents returned to retrieve all relevant documents. With the best configuration we achieve an $ap$ score of 0.191, an $norm \_area$ score of 0.947, and a $last \_rel$ score of 493, and the metrics are 0.189, 0.948, and 461 reported in Ref. [11]. We use the same configuration for all the latter methods.

**ALGORITHM 2: AutoTAR**

**Input:** Topic $q$; document collection $C_q$, target recall $recall_t$.

**Output:** A ranked list of documents $D_q = \{d_1, d_2, \ldots, d_{|C_q|}\}$.

**Parameter:** None.

1. $t = 0$, $L_0 = \{\text{pseudo document } d_0\}$, $D_q = []$;

while $C_q$ is not exhausted do

2. $t = t + 1$;

3. Temporarily augment $L_t$ by uniformly sampling $k$ documents from $U_t$, labeled non-relevant;

4. Train a ranking model on $L_t$;

5. Rank all the documents in $C_q$ with the ranker trained over $L_t$;

6. Select the top $b_t$ documents from the ranked list and append them in $D_q$;

7. Render relevance assessments for the selected documents;

8. Remove the $k$ temporary documents from $L_t$;

9. Place the $b_t$ assessed documents in $L_t$, and remove them from $U_t$;

10. Update batch size: $b_{t+1} = b_t + \lfloor \frac{b_t}{10} \rfloor$;

end

### 4.4.2 Knee.

On the basis of the AutoTAR method, Cormack and Grossman [9] propose the Knee method. It defines a knee of the gain curve through a simple geometric algorithm [33], and stops the TAR process when the slope after the knee diminishes to less than a ratio of the slope before the knee. Let $G = \{(x_i, z_i) \mid i = 1, \ldots, t\}$ denote the data points observed on the gain curve until the $t$-th iteration, $x_i$ denote the cost or percentage of documents reviewed at the $i$-th iteration, and $z_i$ denote the recall at the $i$-th iteration. Let $i^*$ denote the iteration where the knee is
detected. Let $\rho$ denote the slope ratio, defined as $\rho = \frac{\left| \{d_j | r_{d_j} \leq i^* \}, y_{d_j} = 1 \right|}{\left| \{d_j | t > r_{d_j} > i^* \} \right|} \frac{t - i^*}{i^*}$, where $r_{d_j}$ is the rank of document $d_j$, and $y_{d_j}$ is the relevance label of $d_j$. If $\rho$ is larger than a bound, the TAR process is stopped. We implemented the method described in Ref. [9] and summarized it in Algorithm 3.

We run the Knee method on the 30 test topics in the 2017 CLEF technology-assisted reviews in the empirical medicine track, as it is possible to compare our performance with the performance of AutoTAR reported in Ref. [11]. As suggested in Ref. [9], a dynamic bound, $\text{bound} = 156 - \min (R_t, 150)$, has been empirically proven to be effective, where $R_t$ is the number of relevant documents found so far at the $t$-th iteration. Similar to AutoTAR, we also swept over all possible configurations of the ranking model. The best configuration is the same with that of AutoTAR. With the best configuration, we achieve a loss er score of 0.610 and a recall score of 0.999; the metrics are 0.657 and 1.000, reported in Ref. [11].

**ALGORITHM 3: Knee**

**Input:** Topic $q$; document collection $C_q$, target recall $\text{recall}_t$.

**Output:** A ranked list of documents $D_q = [d_1, d_2, \ldots]$. 

**Parameter:** slope ratio bound: $\text{bound}$, number of documents should be reviewed when knee detection takes effect: $\beta$.

1. $D_q = [], G = []$; 
2. $t = 0$, $L_0 = \{\text{pseudo document } d_0\}$; 
3. while $C_q$ is not exhausted & not stop do 
   4. $t += 1$; 
   5. Temporarily augment $L_t$ by uniformly sampling $k$ documents from $U_t$, labeled non-relevant; 
   6. Train a ranking model on $L_t$; 
   7. Rank all the documents in $C_q$ with the ranker trained over $L_t$; 
   8. Select the top $b_t$ documents from the ranked list and append them in $D_q$; 
   9. Render relevance assessments for the selected documents; 
   10. Remove the $k$ temporary documents from $L_t$; 
   11. Place the $b_t$ assessed documents in $L_t$, and remove them from $U_t$; 
   12. Append $(x_t, z_t)$ to $G$; 
   13. Update batch size $b_{t+1} = b_t + \lceil \frac{b_t}{10} \rceil$; 
   14. if $|L_t| \geq \beta$ then 
      15. if knee is detected & $\rho \geq \text{bound}$ then 
         16. stop = True; 
      end 
   end 
end 

4.4.3 Target. Similar to the Knee method, the Target method [9] is designed for high recall tasks. It consists of two steps. In the first step, it randomly samples documents until a pre-defined number of documents is judged relevant. In the second step, the documents in the collection are ranked and retrieved with the AutoTAR method without knowledge of the target set, until each relevant document in the target set has been retrieved. Note that Target is designed to achieve high recall, but not 100% recall. It has been strictly proven that the size of the target set guarantees a recall of 70% with 95% probability under certain assumptions [9].

We implemented the method described in Cormack and Grossman [9] and summarized it in Algorithm 4.
We ran the Target method on the combination of athome1, athome2, and athome3 datasets, which consists of 30 topics and 290,099 Jeb Bush emails, 465,147 hacker forum documents, and 902,434 local news documents. The datasets are released by the TREC 2015 Total Recall track and results are reported in Ref. [9]. There is one hyper-parameter to be tuned: relevant document number in the target set \( n_{rel} \). The recommended value reported in Ref. [9] is 10. Based on this, we also conducted a hyper-parameter sweeping: we alternate \( n_{rel} \) between 5, 10, and 15. The best hyper-parameter configuration (\( n_{rel} = 5 \)) leads to a recall of 0.93 and number of reviewed documents of 49,151, while the reported metrics in Ref. [9] are 0.91 and 44,079.

**Algorithm 4: Target**

**Input:** Target topic \( q \), document collection \( C_q \), target recall \( recall_t \).

**Output:** A ranked list of documents \( D = [d_1, d_2, \ldots] \).

**Parameter:** Relevant document number in the target set: \( n_{rel} \).

1. \( D = [] \);
2. \( t = 0 \), \( L_0 = \{\text{pseudo document } d_0\} \);
3. // Sample set, target set
   Sample documents uniformly from \( C_q \) until \( n_{rel} \) relevant documents have been found or the collection is exhausted. Let \( S_{sample} \) denote the sampled set, \( R_{sample} \) denote the target set only containing the \( n_{rel} \) relevant documents;
4. if \( C_q \) is exhausted then
5.   \( D = \text{list}(C_q) \);
6.   \( \text{stop} = \text{True} \);
7. end
8. // Re-find relevant documents in the target set
9. Let \( S_{interact} \) denote documents re-found, \( R_{interact} \) denote the relevant ones.
10. \( S_{interact} = R_{interact} = \emptyset \);
11. while \( C_q \) is not exhausted & not stop do
12.   \( t + = 1 \);
13.   Temporarily augment \( L_t \) by uniformly sampling \( k \) documents from \( U_t \), labeled non-relevant;
14.   Train a ranking model on \( L_t \), let \( f_t \) denote the corresponding ranking function;
15.   Rank all the documents in \( C_q \) with \( f_t \);
16.   Select the top \( b_t \) documents from the ranked list and add them in \( S_{interact} \);
17.   Place the \( b_t \) assessed documents in \( L_t \), and remove them from \( U_t \);
18.   Remove the \( k \) temporary documents from \( L_t \);
19.   \( b_{t+1} = b_t + \left\lfloor \frac{b_t}{10} \right\rfloor \);
20. // Stop the TAR process
21.   if \( R_{interact} \subseteq R_{sample} \) then
22.     \( \text{stop} = \text{True} \);
23. end
24. // Final ranked list
25. Apply \( f_T \) on \( S_{sample} \cup S_{interact} \) to produce a ranked list \( D \), where \( T \) denotes the last iteration;

**4.4.4 SCAL.** SCAL [10] is designed to achieve high recall for large scale or infinite document collection. To this end, it uses a large fixed-sized sample of the collection to generate the ranker, estimate the prevalence, and determine the cutoff necessary for a particular target recall.
When to Stop Reviewing in Technology-Assisted Reviews

Algorithm 5: SCAL

**Input:** Topic \( q \); document collection \( C_q \), target recall \( \text{recall}_t \).

**Output:** A ranked list of documents \( D = [d_1, d_2, \ldots] \).

**Parameter:** Size of the document sample set: \( N_U \), sub-sample size of a batch: \( b \), calibration factor: \( \eta \).

\[
D = [];
\]

\[
t = 0, \quad L_0 = \{\text{pseudo document } d_0\}, \quad U_0 = U, \quad b_0 = 1;
\]

// Large sample set

\[
\text{// Construct a series of rankers, training sets, and estimators}
\]

\[
\text{while } U \text{ is not exhausted \& not stop do}
\]

\[
t = t+1;
\]

\[
\text{Temporarily augment } L_t \text{ by uniformly sampling } k \text{ documents from } U_t, \text{ labeled non-relevant;}
\]

\[
\text{Train a ranking model on } L_t, \text{ let } f_t \text{ denote the corresponding ranking function;}
\]

\[
\text{Rank all the documents in } U_t \text{ with } f_t;
\]

\[
\text{Select the top } b_t \text{ documents from the ranked list;}
\]

\[
\text{Calculate the sub-sample size at this iteration: } \hat{b}_t = b \text{ is } \hat{R} = 1 \text{ or } b_t < b, \text{ otherwise } \hat{b}_t = b;
\]

\[
\text{Render relevance assessments for the } \hat{b}_t \text{ documents;}
\]

\[
\text{Remove the } k \text{ temporary documents from } L_t. \text{ Place the } \hat{b}_t \text{ documents in } L_t, \text{ and remove the } b_t \text{ documents from } U_t \text{ (note } \hat{b}_t \text{ and } b_t \text{ are different);}
\]

\[
\hat{R}_{t+1} = \hat{R}_t + \frac{\hat{r}_t}{\hat{b}_t}, \text{ where } \hat{r}_t \text{ is the number of relevant documents in the sub-sample;}
\]

\[
b_{t+1} = b_t + \left\lceil \frac{b_t}{10} \right\rceil;
\]

\[
\text{end}
\]

// Cutoff the ranked list

\[
\hat{R} = \eta \cdot \hat{R}_T, \text{ where } T \text{ denotes the last iteration;}
\]

\[
t^* = \arg\min_t (\hat{R}_t \geq \text{recall}_t \cdot \hat{R});
\]

\[
\text{threshold} = \min (\{f_t(d) \mid d \in U_0 \setminus U_t, \text{ d is relevant}\});
\]

\[
\text{Produce a ranked list } D = \{d \mid f_T(d) \geq \text{threshold}, d \in C_q\};
\]

---

We implemented the method described in Ref. [10] and summarize it in Algorithm 5. SCAL solves three major problems. First, in order to deal with the large scale or infinite document collection, a large document set is uniformly sampled (line 3). Second, in order to save the effort of reviewing documents, only a finite number instead of the exponentially increasing batch size of documents are sampled and reviewed. The reviewed documents comprise a stratified sample of the entire collection, which is used for training the ranker and calculating \( \hat{R} \) (line 10). Third, it defines a cutoff to stop the TAR process based on the list of rankers and estimators (lines 18–20).

Note that line 18 is different from the corresponding line in the original work [10], which is \( \text{threshold} = \max \left(\{f_t(d) \mid d \in U_0 \setminus U_t\}\right) \). During our effort to reproduce the published results, we found that taking the maximum score over documents in \( U_0 \setminus U_t \) will cutoff the final ranked list at a very high rank. When using min, we can achieve similar recall to that reported in Ref. [10], but of much larger cost. We further found that using only relevant document scores to determine the threshold can achieve similar recall and cost as in Ref. [10].

We ran SCAL on the athome1 dataset, which consists of 10 topics and 290,099 Jeb Bush emails, released with the TREC 2015 Total Recall track, and compare our results to that reported in Ref. [10]. There are three hyper-parameters: the size of the large document sample set \( N_U \) (%), the
sub-sample size of a batch $b$, and the calibration factor $\eta$. The recommended configuration reported in Ref. [10] is $N_U = 1.0$ (the whole 290,099 emails), $b = 50$, and $\eta = 1.05$. Based on this, we conducted a grid search with $N_U \in [0.6, 0.8, 1.0]$, $b \in [30, 50, 70, 90, 110]$, $\eta \in [1.0, 1.05]$. We also compare our line 18 and the responding line in Ref. [10]: determining the threshold with either max or min, filtering either documents in the bucket, or documents in the sampled set, or only relevant documents in the sampled set.

We find that indeed our line 18 leads to similar results as those reported in Ref. [10]. The best hyper-parameter configuration ($N_U = 1.0$, $b = 110$, $\eta = 1.05$) leads to recall of 0.91 and the number of reviewed documents of 13,198, where the metrics are 0.90 and 9504 in Ref. [10].

### 4.4.5 Score Distribution

The score distribution methods proposed by Ref. [18] provide ways to estimate the total number of relevant documents at any rank position, making it possible to stop the TAR process at an appropriate cutoff in order to achieve a target recall.

We implement the two methods described in Ref. [18]—score distribution using training topics ($SD$-$training$) and score distribution not using training topics ($SD$-$sampling$). The main difference is the way of collecting scores of relevant documents. $SD$-$training$ needs extra training topics and the corresponding relevance labels of the documents in the collection, while $SD$-$sampling$ needs to sample documents and render relevance labels from users. The two versions that we implemented are summarized in Algorithms 6 and 7. One notable design choice we are faced with is the ranking model. We use BM25 as the ranking model as it is impossible to use exactly the same ranking model with AutoTAR. One reason is that the features used for the logistic regression ranking model of AutoTAR are document-wise TF-IDF vectors, and training data is interactively obtained from user relevance feedbacks; however, the two score distribution methods are not interactive methods. Thus, extra training topics are needed, and the features must be topic-document-wise if we train a cross-topic ranking model. Therefore, it is not possible to use exactly the same ranking model with AutoTAR. Another reason is that what matters for the score distribution methods is the distribution of scores instead of the absolute value of each score [1].

We ran $SD$-$training$ and $SD$-$sampling$ on the 30 test topics in the TAR in EMED Task at CLEF 2017, as this makes it possible to compare our implementation with the performance reported in Ref. [18]. For $SD$-$training$, we achieved a recall of 0.994 and precision of 0.042, while the metrics reported are 0.989 and 0.057. For $SD$-$sampling$, we fine-tune the hyper-parameter sample size $p_{sample}$ (%) with $p_{sample} \in [0.1, 0.2, 0.3, 0.4, 0.5]$. The best configuration ($p_{sample} = 0.2$) leads to a recall of 0.978 and precision of 0.062, with Hollmann and Eickhoff [18] reporting 0.948 and 0.079.

### 4.5 Implementation

Similar to AutoTAR and related methods, our proposed framework is topic-wise independent, which means a brand new ranking model is trained at the beginning of the TAR process for each topic.

For fair comparison in terms of precisely stopping the TAR process at a target recall, we use the same document representation and ranking model as AutoTAR. Specifically, we use TF-IDF document representation and logistic regression for ranking: we consider all documents associated to a given topic as the corpus, and construct a TF-IDF vector for each document as its representation. We use scikit-learn (a Python Machine Learning toolkit)\(^8\) for the implementation. In particular, we use TfidfVectorizer with its default configuration to preprocess the input text and generate the TF-IDF vector, while we use LogisticRegression with the default configuration (which is

\(^8\)https://scikit-learn.org/.
ALGORITHM 6: Score distribution using training topics (SD-training)

**Input**: Target topic \( q \), the corresponding document collection \( C_q \), target recall \( \text{recall}_t \), training topics, the corresponding document collections, and the full relevance judgement sets.

**Output**: A ranked list of documents \( D = \{d_1, d_2, \ldots \} \).

**Parameter**: None.

// Training topics
1. Produce ranking scores for the relevant documents of all the training topics by applying BM25 ranking function;
2. Fit a Gaussian distribution \( \mathcal{N}(\mu, \sigma^2) \) over the scores of these relevant documents;

// Target topic
3. Produce ranking scores for all documents of the target topic by applying BM25 ranking function;
4. Find the point where the cumulative density probability of \( \mathcal{N}(\mu, \sigma^2) \) equals \( 1 - \text{recall}_t \);
5. Extract from \( C_q \) all the documents of which the ranking scores are larger than the point;

// Final ranked list
6. Sort the extracted documents by the descending order of ranking scores, denoted as \( D \);

ALGORITHM 7: Score distribution not using training topics (SD-sampling)

**Input**: Target topic \( q \), the corresponding document collection \( C_q \), target recall \( \text{recall}_t \).

**Output**: A ranked list of documents \( D = \{d_1, d_2, \ldots \} \).

**Parameter**: Sample size \( p_{\text{sample}} \) (%).

// Sample documents
1. Produce a ranked list of all the documents and a ranked list of normalized score by applying BM25 ranking function on \( C_q \);
2. Sample uniformly \( |C_q| \cdot p_{\text{sample}} \) documents, denoted by \( D_1 \);

// Fit Gaussian distribution
3. Fit a Gaussian distribution \( \mathcal{N}(\mu, \sigma^2) \) over the scores of the sampled relevant documents;
4. Find the point where the cumulative density probability of \( \mathcal{N}(\mu, \sigma^2) \) equals \( 1 - \text{recall}_t \);
5. Extract from \( C_q \) the documents of which the ranking scores are larger than the point, denoted by \( D_2 \);

// Final ranked list
6. Output the final ranked list \( D \) by merging \( D_1 \) and \( D_2 \) by the descending order of ranking scores;

also proven to be the best configuration when reproducing the AutoTAR baseline) for the logistic regression models.

We also implemented and reproduced all the baselines: AutoTAR, Knee, Target, SCAL, SD-training, and SD-sampling given that their source code is not available online. The code used in this work will be released as open source code online.9

5 EXPERIMENTS
In this section, we conduct experiments on all the datasets to answer the aforementioned research questions.

5.1 Stopping Effectiveness

This experiment is designed to answer RQ1. In this experiment, methods to be compared are provided with a target recall. A method is considered better than another whether (1) it achieves higher recall, (2) it spends less assessment cost, and (3) the achieved recall when it decides to stop the TAR process is as close as possible to the target recall. As explained in Section 4.3, we use a portfolio of metrics including recall, cost, RE, loss$e_r$, and reliability for evaluation.

5.1.1 Experimental Setting. As our framework provides the flexibility to target the TAR process for recall, we vary the target recall value between 0.8, 0.9, and 1.0 to see whether the proposed framework is capable of stopping the TAR process on time. We focus on \{0.8, 0.9, 1.0\} because high recall is usually desired in the TAR process.

The proposed framework provides multiple choices for its sampling module, estimation module, and stopping module. The estimator varies between the Horvitz-Thompson estimator and the Hansen-Hurwitz estimator, and the stopping strategy varies between optimistic and conservative. In total, there are five variations of the proposed framework: HT-opt, HT-con1 (variance is calculated based on Equation (6)), HT-con2 (variance is calculated based on Equation (7)), HH-opt, and HH-con (variance is calculated based on Equation (9)).

By running the five variations on the corresponding training topics of the EMED, TR, and LEGAL datasets, we find the combination of APPrior and HT-con1 performs the best. In the rest of the work, we use this configuration as our method. The impact on the stopping effectiveness with regards to the sampling methods (APPrior, PowerLaw, MixtureUniform), estimators (HT and HH), and stopping strategies (optimistic and conservative) can be found in Section 5.4.

This experiment is conducted on all five test datasets including EMED 2017, EMED 2018, EMED 2019, TR, and LEGAL. Note that for a topic on TR and LEGAL, there are no associated documents filtered with some initial ranking techniques like on the EMED datasets; therefore, we use the whole document collection as the associated documents for each topic. On the other hand, our method typically needs a large amount of memory (or a long time depending on the implementation) when calculating the first order and the second order inclusion probabilities. For example, it needs about 20GB memory to deal with topics associated with about 15,000 documents. As a consequence, it is impossible to run our method directly on our computational resources. As an alternative, for the TR and LEGAL datasets, we split the whole collection into buckets of 1,000 documents, run our method over these buckets, and concatenate all the sampled documents for the final reviewing. This adaption is only for our method. All the baselines are run on the TR and LEGAL dataset as it is.

We compare our method with the Knee, Target, SCAL, SD-training, and SD-sampling methods. The Knee and Target methods are designed to achieve 100% recall, and the rest are designed to stop the TAR process at a specified target recall. We present the results of the Knee method and Target methods only when the target recall is 1.0. We fine-tune all the methods (if needed) on the training topics when the target recall is 0.8, 0.9, and 1.0, respectively. The Knee method has two hyper-parameters: slope ratio bound, bound; and number of reviewed documents when knee detection takes effect, $\beta$. The recommended value reported in Ref. [9] is $\text{bound} = 156 - \min (R_t, 150)$ and $\beta = 1,000$. Based on this, we conducted a grid search for $\text{bound} \in [3, 6, 10, 15, 156 - \min (R_t, 150)]$, and $\beta \in [100, 1000]$. The Target method has one hyper-parameter: relevant document number in the target set $n_{rel}$. The recommended value reported in Ref. [9] is $n_{rel} = 10$. Based on this, we conducted a grid search by alternating $n_{rel} = [5, 10, 15]$. SCAL has three hyper-parameters: the size of the large document sample set $N_U$, the sub-sample size of a batch $b$, and the calibration factor $\eta$. The recommended configuration reported in Ref. [10] is $N_U = 1.0$, $b = 30$, and $\eta = 1.05$. Based on this, we conducted a grid search for $N_U \in [0.6, 0.8, 1.0]$, $b \in [30, 50, 70, 90, 110]$, and
Table 3. Fine-Tuned Parameters

<table>
<thead>
<tr>
<th>target recall</th>
<th>Knee bound</th>
<th>β</th>
<th>Target bound</th>
<th>SCAL n_{rel}</th>
<th>SCAL N_U</th>
<th>SCAL b</th>
<th>SCAL η</th>
<th>SD-training</th>
<th>SD-sampling p_{sample}</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.8</td>
<td>156 − min(R_t, 150)</td>
<td>1,000</td>
<td>15</td>
<td>0.8</td>
<td>30</td>
<td>1.05</td>
<td>—</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>0.9</td>
<td>156 − min(R_t, 150)</td>
<td>1,000</td>
<td>15</td>
<td>0.9</td>
<td>30</td>
<td>1.05</td>
<td>—</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>156 − min(R_t, 150)</td>
<td>1,000</td>
<td>15</td>
<td>1.0</td>
<td>30</td>
<td>1.05</td>
<td>—</td>
<td>0.1</td>
</tr>
</tbody>
</table>

η ∈ [1.0, 1.05]. SD-training has no hyper-parameter. SD-sampling has one hyper-parameter: sample size p_{sample} ∈ [0.1, 0.2, 0.3, 0.4, 0.5]. The best configuration of these baselines for different target recall is shown in Table 3.

5.1.2 Results.

Effectiveness in terms of high recall and low cost. Achieving 100% recall is desired in electronic discovery, systematic review, investigation, research, and the construction of datasets for information retrieval evaluation. It is interesting to see whether the stopping methods can achieve 100% recall with low cost.

In Table 4, the target recall is set to 100%, and we focus on the recall, cost and loss value. On the EMED 2017 dataset, all the methods can achieve very high recall (97.8% to 99.9%); among them, Knee only needs around 30% cost, while Target, SCAL, SD-sampling and our method needs double cost (61.4% to 76.2%), and SD-training needs to review almost the full document collection. When comparing the loss value, which considers both recall and cost, Knee achieves the best performance, and our method achieves the second best performance. Similar results can be observed on the EMEM 2018, EMED 2019, and TR datasets. Note that the performance of all methods on LEGAL is very different from those on the remaining four datasets. The cost on TR and LEGAL are always larger than that on EMED. This is because our method applied on TR and LEGAL is a slightly different from the original algorithm, as mentioned at the beginning of this section. Specific reason is that we train a new ranking model from scratch for each split of the document collection, which leads to reviewing more non-relevant documents. This issue can be addressed by training a global ranking model for all the splits; we leave this for the future work.

In order to better understand the relation of cost and recall, we further visualize the cost and recall columns of Table 4. Figure 3 shows the scatter plots of different methods, and the cost and recall values are averaged over all topics. When the target recall is set to 100% (subplots (a) to (e) in Figure 3), the Knee method, the SD-training method, and our method are on the Pareto frontier, indicating they achieve a good balance between cost and recall.

Overall, when the goal is achieving high recall and low cost, the Knee method performs the best. It is a greedy method designed for total recall tasks; there is no sampling mechanism and the goal is to collect many relevant documents as fast as possible. Our method achieves the second best performance. It is a sampling method that trades off the assessment cost for the estimation of R.
Fig. 3. Visualization of stopping effectiveness on five datasets. $x$-axis is cost, $y$-axis is $\text{recall}_c$. Each point in the plots represents one method. The desirable situation is to stop the TAR process exactly when $\text{recall}_c = \text{recall}_t$. 
Effectiveness in terms of on-time stopping. Another goal of the proposed framework is to stop the TAR process on time. It is also interesting to see whether the recall achieved when the TAR process is stopped is close to the target recall and how often the achieved recall is equal to or higher than the target recall. We vary target recall value between 0.8, 0.9, and 1.0 in Tables 4, 5, and 6, and focus on the RE and reliability values.

First, let us examine each table and compare the performance of different methods. Our method achieves either the best or comparable RE and reliability values among all baselines, indicating that it can stop the TAR process on time. The Knee method performs slightly worse than our method because it stops the TAR process early; thus, it is more likely to achieve a recall less than 100%. The Target method performs worse than the Knee method because it needs much more assessment cost while still achieving slightly lower recall than the Knee method; the observation is similar to the conclusion in Ref. [9]. Note that given a target recall, although it is possible to fine-tune the hyper-parameters for the Knee method, or adapt the method by collecting a subset of the target set for the Target method, these tricks do not represent their original intentions. Therefore, we only report the results when \( \text{recall}_t = 1.0 \) for Knee and Target. The SCAL method, which also stops the TAR process based on an estimator of \( R \), can achieve the target recall accurately, but the cost is higher than Knee as expected. The SD-training method estimates the cutoff of stopping reviewing documents by fitting a Gaussian distribution of the ranking scores of relevant documents of training data. It is not an interactive method. Instead, the documents are only ranked one time. SD-training does not stop the TAR process on time, and it also needs more cost. SD-sampling is similar to SD-training; the difference is that the documents that provide ranking scores are sampled from the documents of the current topic, instead of training topics. SD-sampling stops the TAR process better than SD-training.
Now, let us examine how the $RE$ and $reliability$ values change across the three tables. It can be found that the $RE$ and $reliability$ values of all the methods increase when the target recall decreases. $SCAL$ stops the TAR process on time when target recall is 1.0 and 0.9, but not on time when target recall is 0.8. $SD$-sampling stops the TAR process more accurately than $SD$-training, indicating sampling relevant documents within one topic is effective for on-time stopping. The aforementioned observations are also supported in the scatter plots in Figure 3 (values averaged over topics) and Figure 4 (topic-wise values). The desirable situation is to stop the TAR process exactly when $recall_c = recall_t$.
Overall, when the goal is to stop the TAR process on time, our method performs better than Knee. Our method also performs similarly well with SCAL and SD-sampling and performs better than SD-training.

5.2 Estimating $R$

This experiment is designed to answer RQ2. We examine whether the estimator $\hat{R}$ is unbiased with low variance. Further, as the estimator $\hat{R}$ is expected to be more accurate with more sampled documents, we also examine how the estimator $\hat{R}$ changes as the target recall changes.

5.2.1 Experimental Setting. The experimental setting is the same as in the previous experiment. We only report results for the EMED 2017 dataset since the results for the other datasets are similar. Because Knee, Target, SD-training, and SD-sampling do not provide an estimation of $R$, we only compare our method with SCAL.

5.2.2 Results. For each topic, we record $R$ and $\hat{R}$ when the TAR process stops and present them in scatter plots shown in Figure 5. We also report the MSE metric to see how far the estimated value is from the true value.

We can see that most points of our method lie on the diagonal line when $\text{recall}_t = 1.0$, indicating that our method accurately estimates $R$; whereas, when $\text{recall}_t = 0.9$ and $\text{recall}_t = 0.8$, $R$ is
under-estimated. But if we also take $\hat{\vartheta}(R)$ into consideration, i.e., the bar of one standard deviation $\hat{R} \pm \hat{\vartheta}(R)$, it is more likely that the bar covers $R$ when $\text{recall}_l = 0.9$ and $\text{recall}_t = 0.8$. The major reason of under-estimation for low target recall are due to the low prevalence of the topics in the collection. At the early iterations, the estimator fluctuates drastically. With more documents, especially more non-relevant documents for low-prevalence topics being sampled, the inclusion probabilities are generally approaching 1.0. Hence, finding a new relevant document will only slightly increase $\hat{R}$, and this makes the estimator slowly approaching the true value. It is interesting to study how the accuracy of the estimator changes with different prevalences. We leave this in the future work.

The SCAL baseline method also under-estimates $R$ for $\text{recall}_l = 0.9$ and $\text{recall}_t = 0.8$ but over-estimates $R$ for $\text{recall}_l = 1.0$.

Overall, the estimator $\hat{R}$ of our method is accurate, especially for tasks of high target recall, and summing up $\hat{\vartheta}(R)$ and $\hat{R}$ partially solves the under-estimation problem for tasks of low target recall.

### 5.3 Tradeoff Recall Against Estimating $R$

#### 5.3.1 Experimental Setting

This experiment is designed to answer RQ3. The motivation for this research question is that we introduce extra assessment costs when we use random sampling
instead of a greedy method. Hence, it is beneficial to know whether we sacrifice the effectiveness of the ranker too much to collect relevant documents for the ability to accurately estimate the number of missing relevant documents through sampling. In the previous experiment, we study the quality of the estimator $\hat{R}$ when the TAR process stops. This is a “static” experiment in the sense that it only considers $\hat{R}$ at the final iteration. In this experiment, we will show how the actual recall and the relative error between $R$ and $\hat{R}$ changes along the iteration. For each topic, we let the TAR process run until it almost exhausts all the documents, reaching at a cost of 95%.

5.3.2 Results. Figure 6 shows the tradeoff between high recall and accurate estimation of $R$. The AutoTAR method (the grey line) serves as an upper bound of recall. An interesting point is when AutoTAR achieves a recall of 100% with a cost of 40%; our method achieves a recall of 85% and also provides an estimation of $R$ with a relative error of 10%.

The mechanism of AutoTAR is to repeatedly select documents from the top of the ranking towards the bottom until all the documents are selected; it does not address the stopping problem. Built on top of the AutoTAR, the Knee method allows for automatic stopping by using a geometric algorithm to detect the knee of the recall curve. It does not need extra assessment cost to determine the stopping point; however, it does not provide any insight on how many relevant documents one would miss if one stops reviewing. It turns out that knowing this information does need to trade off a certain amount of recall. The overall finding is that we need to trade off around 15% of the relevant documents against estimating $R$ with a relative error of 10%.

Knowing an estimation of $R$ is necessary in many cases, for example, estimating the number of missing documents to study the reliability of the results of systematic reviews [17], and estimating
(a) EMED 2017, recall₇ = 0.9  
(b) EMED 2018, recall₇ = 0.9  
(c) EMED 2019, recall₇ = 0.9  
(d) TR, recall₇ = 0.9  
(e) LEGAL, recall₇ = 0.9  
(f) EMED 2017, recall₇ = 0.8  
(g) EMED 2018, recall₇ = 0.8  
(h) EMED 2019, recall₇ = 0.8  
(i) TR, recall₇ = 0.8  
(j) LEGAL, recall₇ = 0.8

Fig. 4. (Con’t) Topic-wise visualization of stopping effectiveness on five datasets. Color and marker together distinguish different methods. Each point represents one topic.
Fig. 5. $R$ vs. $\hat{R}$ on EMED 2017 (topic-wise). In each subplot, marker and color distinguish estimators; each point corresponds to a topic. The bar on each point indicates a one standard deviation ($\sqrt{\text{var}(\hat{R})}$) away from mean ($\hat{R}$). An unbiased estimator with zero variance should lead to points that lie on the $x = y$ line.

Fig. 6. Tradeoff recall against estimating $\hat{R}$. The dataset is EMED 2017. The grey line (AutoTAR) serves as an upper bound of the recall. The blue dotted line and the blue triangle line indicates how much recall our method trades off against accurately estimating $\hat{R}$.

<table>
<thead>
<tr>
<th>Recall</th>
<th>RE ↓</th>
<th>Loss ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>con1</td>
<td>con2</td>
</tr>
<tr>
<td>recall = 1.0</td>
<td>HH 0.003 — 0.023</td>
<td>0.159</td>
</tr>
<tr>
<td>recall = 0.9</td>
<td>HH 0.059 — 0.054</td>
<td>0.130</td>
</tr>
<tr>
<td>recall = 0.8</td>
<td>HH 0.086 — 0.037</td>
<td>0.124</td>
</tr>
</tbody>
</table>

*: ↑ means that higher values are better, and ↓ means that lower values are better.

the volume of social media posts to track brands’ popularity and product sales [40]. The framework can be applied in these application fields.

5.4 Model Component Analysis

5.4.1 Experimental Setting. This experiment is designed to answer RQ4. We examine how the Horvitz-Thompson (HT) and Hansen-Hurwitz (HH) estimators, and the optimistic (opt) and conservative (con) stopping strategies impact the performance of the proposed framework. The combination of the estimators and the stopping strategies give rise to the following options: HT-opt, HT-con1 (where the variance is calculated based on Equation (6)), HT-con2 (where the variance is calculated based on Equation (7)), HH-opt, and HH-con1 (where the variance is calculated based on Equation (9)). We conduct this experiment on the training topics of EMED 2017. Similar with the previous setting, we alternate the target recall level between 0.8, 0.9, and 1.0.

5.4.2 Results. We show the performance of each configuration with respect to loss and RE in Table 7.

**Estimator.** The Horvitz-Thompson estimator performs slightly better than the Hansen-Hurwitz estimator in terms of both loss and RE. In practice, if the goal is to achieve high effectiveness, the Horvitz-Thompson estimator is recommended. Note that the calculation of the Horvitz-Thompson estimator is of relative low-efficiency, and the way of calculating inclusion probability is not trivial if the sampling is not random sampling with replacement. In other words, if efficiency is a concern or the sampling is without replacement, the Hansen-Hurwitz estimator is recommended.

**Stopping strategy.** The conservative strategy performs slightly better than the optimistic strategy. In practice, if not missing relevant documents is very important and the cost is not the main concern, the conservative strategy is recommended. We find that in our experimental results, the estimator $\hat{R}$ tends to be lower than the true value $R$ (as shown in Figure 5); therefore, adding one standard deviation (remember that both the mean and variance are provided in the estimation module) to the estimated mean value does help to stop the TAR process more effectively. In all the experiments in this work, we have only tried to add one standard deviation; adding two or three standard deviations can be tried in order to mitigate the underestimation of $R$ in future work.

6 CONCLUSION

In this work, we have studied how to determine the stopping point of document selection in order to construct test collections that balance the cost of assessing document relevance and the gain of identifying relevant documents.
We propose a novel continuous active learning framework by jointly training a ranking model to rank documents, and conducting “greedy” sampling to estimate the total number of relevant documents in the collection. Within the framework, we propose to use the AP-Prior as the sampling distribution, the Horvitz-Thompson or Hansen-Hurwitz estimator to estimate the total number of relevant documents, and the optimistic or conservative strategy to determine whether to stop the TAR process or not. We prove the unbiasedness of the proposed estimators under a without-replacement sampling design. To examine the effectiveness of the proposed framework, we compared it against the Knee, Target, SCAL, SD-training, and SD-sampling methods and provided detailed analysis on various datasets including the CLEF Technology-Assisted Reviews in Empirical Medicine datasets [20–22], the TREC Total Recall datasets [16], and the TREC Legal datasets [13]. The experimental results demonstrate that the proposed method performs secondary to Knee but better than other baselines in terms of high recall and low cost; on the other hand, the proposed framework performs better than Knee and other baselines in terms of stopping the TAR process on time.

To sum up, the proposed framework combines the advantages of the continuous active learning approach with the advantages of sampling methods. It can effectively retrieve relevant documents similar to CAL but also provide a transparent, accurate, and effective stopping point. Specifically, it is recommended to use the AP-Prior distribution, and the Horvitz-Thompson estimator together with the conservative stopping strategy. If efficiency is also considered, the Hansen-Hurwitz estimator is recommend instead of the Horvitz-Thompson estimator.

There are limitations and several directions that we have not touched and can be followed in the future work. First, the sampling and estimation in the proposed framework is in a “sequential” manner, i.e., at each iteration, a certain number of documents are sampled and an estimator of $R$ is calculated. When the sequence of estimators are repeatedly compared with the target recall until the desired result is observed—the estimated recall exceeds the target recall, the process yields a biased estimate of recall, even though the estimator at each iteration is unbiased. This issue is inherent for sequential testing [37]. Second, the proposed framework is designed for small-scale document collections. It requires a relatively small list of documents instead of a collection containing millions of documents so that the calculation of mean and variance of $R$ is feasible. Currently, we adapted our framework on the large document collection of TR and LEGAL by randomly splitting the documents, running our algorithm, and then concatenating the sampled documents for the final reviewing. However, this is not the best solution. More work can be done such as training the ranking model globally to avoid sampling too many non-relevant documents. Third, like all other sampling methods, the proposed framework inherently has the high variance problem for low-prevalence topics. In practice, where prevalence is low, one must resort to solutions such as snowball sampling, capture-recapture, and other techniques that are common in biostatistics and medicine. It is worth studying how to integrate these solutions. Fourth, the performance of the ranking model can be further improved in order to produce a good ranked list of documents. Currently, only the document text instead of the topic text is employed for feature representation; the simple TF-IDF is used for document representation, the logistic regression model is used as the ranking model, and the ranking model is trained in a topic-wise manner, i.e., a new ranking model is trained from scratch for each topic. In future work, a stronger ranking model can be trained by addressing these issues. Finally, the current framework only considers the number of missing relevant documents to stop the TAR process. More factors should be taken into account such as the importance of the missing documents. For example, risk of bias and quality, two concepts from systematic review domain indicating how reliable the results of the studies included in a systematic review are, can be leveraged to determine stopping the TAR process [17].
### APPENDICES

Table 8. Topic Splits of the Datasets

<table>
<thead>
<tr>
<th>Topics</th>
<th>EMED</th>
<th>TR</th>
<th>LEGAL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>train (EMED 2017)</td>
<td>test (EMED 2017)</td>
<td>test (EMED 2018)</td>
</tr>
<tr>
<td>CD008686</td>
<td>CD008081</td>
<td>CD008081</td>
<td>CD008081</td>
</tr>
<tr>
<td>CD008953</td>
<td>CD009551</td>
<td>CD009551</td>
<td>CD009551</td>
</tr>
<tr>
<td>CD011548</td>
<td>CD008691</td>
<td>CD010236</td>
<td>CD010236</td>
</tr>
<tr>
<td>CD011859</td>
<td>CD008760</td>
<td>CD008760</td>
<td>CD008760</td>
</tr>
<tr>
<td>CD00925</td>
<td>CD009925</td>
<td>CD009925</td>
<td>CD009925</td>
</tr>
</tbody>
</table>

ACKNOWLEDGMENTS

All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

REFERENCES


Received October 2019; revised July 2020; accepted July 2020