Data Mining and Knowledge Discovery
https://doi.org/10.1007/s10618-023-00961-5

SAL$_\tau$: efficiently stopping TAR by improving priors estimates

Alessio Molinari[1,2] · **Andrea Esuli**[2]

Received: 10 November 2022 / Accepted: 12 July 2023

Abstract
In high recall retrieval tasks, human experts review a large pool of documents with the goal of satisfying an information need. Documents are prioritized for review through an active learning policy, and the process is usually referred to as Technology-Assisted Review (TAR). TAR tasks also aim to stop the review process once the target recall is achieved to minimize the annotation cost. In this paper, we introduce a new stopping rule called SAL$_\tau^R$ (SLD for Active Learning), a modified version of the Saerens–Latinne–Decaestecker algorithm (SLD) that has been adapted for use in active learning. Experiments show that our algorithm stops the review well ahead of the current state-of-the-art methods, while providing the same guarantees of achieving the target recall.

Keywords Active learning · Technology-assisted review · TAR · e-Discovery · Systematic review

1 Introduction

In high recall retrieval tasks, the goal is to find all (or almost all) the documents which are relevant to a given information need, from an unlabelled set of documents (often called the pool P). Examples of these tasks are e-discovery, systematic reviews in empirical medicine, and online content moderation.

Responsible editor: Sriraam Natarajan.

✉ Andrea Esuli
andrea.esuli@isti.cnr.it

Alessio Molinari
alessio.molinari@isti.cnr.it

[1] Department of Computer Science, University of Pisa, 56124 Pisa, Italy

[2] Istituto di Scienza e Tecnologie dell'Informazione "A. Faedo", Consiglio Nazionale delle Ricerche, 56124 Pisa, Italy

Published online: 28 August 2023

In these scenarios, the simplest strategy to guarantee a high recall target is to have in-domain human experts reviewing the whole pool of documents, that is, labelling each document as either relevant or non-relevant. When working with large collections, however, this operation becomes incredibly expensive, both in terms of time and costs: indeed, the number of data items that can be annotated are usually limited by either the availability of the reviewer, or the money one is willing to invest in the process (the *annotation budget*).

Given some time/cost budget, annotating a random sample of the data is a suboptimal approach: the reviewing process is usually aided by one of the many machine learning techniques which go under the name of "Active Learning" (AL, Dasgupta and Hsu 2008; Huang et al. 2014; Lewis and Gale 1994). AL methods adopt a human-in-the-loop model, in which the human expert labels items selected by an automatic classifier. The classifier is then updated, exploiting the additional knowledge coming from new labels, in an iterative process. In the high-recall scenarios mentioned earlier, the human-in-the-loop annotation workflow is usually referred to as Technology-Assisted Review (TAR, Cormack et al. 2010; Grossman and Cormack 2011; Kanoulas et al. 2019).

One of the most challenging issues in TAR applications is the so-called "when-to-stop" problem: that is, we need to choose when to stop the AL process, in order to jointly minimize the annotation effort and satisfy the information need, e.g., a target recall value. Recently, IR literature has proposed many stopping methods (Cormack and Grossman 2016; Li and Kanoulas 2020; Oard et al. 2018; Yang et al. 2021a): the when-to-stop issue is usually tackled by either changing the sampling policy of the AL algorithm (see Sect. 2.1), by crafting task-specific heuristics, and/or by estimating the currently achieved recall. In this paper, we focus on the latter approach, and propose a new technique based on the Saerens–Latinne–Decaestecker (SLD) algorithm (Saerens et al. 2002), adapting it to the AL workflow typically leveraged in TAR processes.

The paper is structured as follows: Sect. 2 describes the related work; we then analyze the shortcomings of the SLD algorithms when used "as-is" in AL scenarios (Sect. 3); we then propose a solution to this problem, our own method in Sect. 4. Experiments and results are discussed in Sects. 5 and 6. Section 7 concludes.

2 Background and related work

2.1 Active learning: relevance sampling

AL algorithms prioritize the annotation of certain data items over others. Two of the most well-known AL techniques, still used to this day, were presented in 1994 by Lewis and Gale (1994): Active Learning via Relevance Sampling (ALvRS) and Active Learning via Uncertainty Sampling (ALvUS).

Algorithm 2.1 shows the typical structure of an AL process with a stopping rule. Given a data pool of unlabelled documents P, the reviewer annotates a "seed" set of documents $S \subset P$ that defines the initial training set L. An iterative procedure is then started, in which L is used to train a classifier ϕ, which is then exploited by an

AL policy *pol* to select the next set of documents to be presented to the reviewer. In ALvRS, ϕ is used to rank the documents in $(P \setminus L)$ in decreasing order of their posterior probability of relevance $\Pr(\oplus|\mathbf{x})$. The reviewer is asked to annotate the b documents for which $\Pr(\oplus|\mathbf{x})$ is highest (with b the *batch size*), which, once annotated, are added to the training set L. The classifier is retrained on the new training set and the process repeats until a stopping condition is met, e.g., the annotation budget is exhausted or a target recall is reached. ALvRS is most-effective and has been mostly used when we are interested in finding all the items relevant to a given information need, as quickly as possible.

Algorithm 1: Schema of an AL process.

Input : Pool of documents P to be reviewed; Batch size $b = 100$; budget $t = |P|$; target recall R; AL policy *pol*;

$i \leftarrow 0$;
$S \leftarrow$ `random_initial_seed()` ;
$L \leftarrow S$;
$U \leftarrow P \setminus L$;
do
  $i \leftarrow i + 1$;
  $\phi_i \leftarrow$ `train_clf`(L);
  $B_i \leftarrow$ `select_via_pol`(pol, ϕ_i, U, b) ;
  $L \leftarrow L \cup B_i$;
  $U \leftarrow P \setminus L$;
while `should_not_stop`$(\phi_i, B_i, L, U, t, R)$;

The ALvUS policy is a variation of ALvRS, where documents are ranked in ascending order of $|\Pr(\oplus|\mathbf{x}) - 0.5|$, i.e., we top-rank the documents which the classifier is most uncertain about. ALvUS can be useful when we want to build a high-quality training set to later train a machine learning model on it, and has not been employed as often as ALvRS in TAR applications. While many other AL techniques have been proposed over the years (Dasgupta and Hsu 2008; Huang et al. 2014; Konyushkova et al. 2017), in this work we focus especially on ALvRS, and in particular on its variant called Continuous Active Learning (CAL), proposed by Cormack and Grossman (2015), which is specifically tailored to TAR applications.

Both ALvRS and ALvUS policies suffer from what is called *sampling bias* (Dasgupta and Hsu 2008; Krishnan et al. 2021), i.e. the fact that, due to the document selection policy, the sample of annotated items L is not representative of P, nor of the unlabelled set U (see Sect. 3 for a more thorough explanation and analysis). In order to investigate how and when a classifier is affected by this bias, Esuli et al. (2022) have introduced a policy called *Rand*(*pol*). The *Rand*(*pol*) policy is an oracle policy, i.e., it requires knowing the true labels of all documents in the pool. It is thus a synthetic policy designed to investigate the issue of sampling bias of a given AL policy *pol*.

Rand(*pol*) observes the prevalences of labels in the L set produced by *pol* and produces its own L^{Rand} set which exhibits the same prevalences, but using a random

document selection policy. The idea is that substituting the selection policy of *pol* with random sampling keeps the content of elements in L^{Rand} unbiased with respect to P, while using the same prevalences produced by *pol*. The comparison of results produced by *pol* and *Rand*(*pol*) is thus a way to understand whether and how much sampling bias is affecting the decisions of a *pol*-based classifier: being a controlled random sampling, the *Rand*(*pol*) policy should produce a "sampling bias free" dataset.

2.2 TAR tasks and workflows

TAR processes usually operate in a "needle in a haystack" scenario, i.e. the number of relevant items is just a tiny fraction of the whole collection of documents. Three of the main TAR real-world applications are: e-discovery, in the legal domain; the production of systematic reviews in empirical medicine, and online content moderation. In this work, we focus on the first two tasks.

2.2.1 TAR for e-discovery

E-discovery is an important aspect of the civil litigation in many (but not only) common law countries: in e-discovery a large pool of documents P need to be reviewed in order to find all items "responsive" (i.e., relevant) to the litigation matter. The documents labelled as responsive are "produced" by the defendant party, and disclosed to the other party in the civil litigation. However, the former party holds the right to keep some of these documents "hidden", putting them in a private log only available to the jury: this is only allowed if the "logged" documents are deemed to contain "privileged" information (e.g., intellectual property). Making different misclassification errors (i.e., producing a document which contains privileged information) can bring about different costs for the defendant party, based on the severity of the error committed. It is worth noticing, however, that there are only few works which really take e-discovery costs into consideration (Oard et al. 2018; Yang et al. 2021b).

In e-discovery, the review usually happens in two stages: (i) documents are first reviewed by responsiveness (i.e., relevancy) by a team of junior reviewers; (ii) the documents judged as responsive are then passed on to a second team of senior reviewers (with an hourly rate several times higher than the junior team's), which mainly re-review the documents by privilege.[1] As it can be inferred, annotating documents by privilege is usually a much more costly and delicate operation than annotating by responsiveness (Oard et al. 2018).

[1] Notice, however, that this procedure may be conducted in another order, with different teams of reviewers, and in many other configurations. See, for instance, Yang et al. (2021b).

2.2.2 TAR for systematic reviews

In empirical medicine, a systematic review discusses (ideally) all medical literature relevant to a given research question. The production of a systematic review is usually carried out by one or more physicians and can span even years (Michelson and Reuter 2019; Shemilt et al. 2016). A systematic review usually collects a large pool of documents P by issuing a boolean query on a (medical literature) search engine. Then, similarly to e-discovery, systematic reviews are conducted in two stages: (i) a first one, where doctors review document abstracts to determine their relevance and (ii) a second one, where documents which passed the first phase are reviewed in their entirety.

The production of systematic reviews has recently attracted the interest of the IR community (Callaghan and Müller-Hansen 2020; O'Mara-Eves et al. 2015; Lease et al. 2016; Wang et al. 2022), which has focused on several aspects of the process, from improving the query formulation issued to search engines, to finding the optimal stopping criterion, reducing the annotation costs (and the time spent on a systematic review).

2.2.3 One-phase TAR and two-phase TAR workflows

TAR workflows can usually be divided in "one-phase" and "two-phase" approaches (Yang et al. 2021a), where:

1. In one-phase TAR workflows, we assume that there is a single review process that is stopped when some condition is met. Relevance sampling is usually the preferred AL technique, since the aim is to annotate the highest number of relevant items in the shortest amount of time possible;
2. In two-phase TAR workflows, a review team annotates a sample of the data pool, on which a classifier is trained and later used to help a second review team complete the process. The two review teams may and usually have different per-document costs.

Note that the number of phases is not related to how many stages are required by the specific task: both approaches work with multi-stage review (see also Yang et al. (2021b) on when either of the two workflows might be preferred).

This paper focuses on one-phase TAR workflows, although the new method we propose might as well be used in two-phase workflows. This choice is in line with the recent literature, which has mostly focused on one-phase reviews (Cormack and Grossman 2016, 2020; Li and Kanoulas 2020; Yang et al. 2021a) [with the notable exception of Oard et al. (2018)].

2.3 Stopping methods for TAR

Finding a way to stop the AL process as soon as a target recall R is reached is a key aspect in lowering the cost of human review, the other being using an AL policy that efficiently selects relevant documents over non-relevant ones. The target recall is usually very high, in many cases equal or close to 100%, transforming the task in a "total recall" task; other high target recalls are often seen in TAR applications, such as 80% or 90% (Li and Kanoulas 2020; Yang et al. 2021a, b).

The scope and aim of this paper is on stopping algorithms which work inside an AL process without changing the sampling policy. We thus do not consider "interventional stopping rules", e.g. Li and Kanoulas (2020). As observed in (Yang et al. 2021a, §2), we argue that interventional methods are (i) less efficient than AL (i.e., they usually trade off annotation costs for a safer recall estimation) and (ii) less applicable in real case scenarios, where reviewers are often limited to using some AL methods (i.e., relevance sampling) provided by a specific software. We will now give an overview of the most relevant methods, which are then compared to our proposal in the experiments.

2.3.1 The knee and budget method

The Knee method was first proposed by Cormack and Grossman (2016). The method is based on a gain curve for a one-phase TAR workflow, i.e., a plot of how the number of positive documents increases as more documents are reviewed, during the AL process. The method, based on Satopaa et al. (2011), empirically finds "knees" in the plot, ideally stopping the process when the effort of continuing to review documents is not supported by the retrieval of a sufficient amount of positive documents.

The Budget method (Cormack and Grossman 2016) is a heuristic variant of the knee method, where the process is stopped no earlier than when at least 70% of the document collection has been reviewed. This follows the observation that, if we were to review by random sampling, we would expect to achieve a recall of 0.7 when reviewing 70% of the collection; by using an AL technique, we expect the recall to be much higher. After the 70% threshold has been reached, the Budget method stops the review if a knee test passes (detailed in Cormack and Grossman (2016); Yang et al. (2021a)), and if the number of relevant items found $Rel(L)$ is somewhat large, i.e.: $|L| \geq 10\ P/Rel(L)$. Both methods do not allow users to specify a target recall.

2.3.2 The Callaghan Müller–Hansen (CMH) method

Callaghan and Müller-Hansen (2020) propose a stopping heuristic based on estimating the probability of having reached the target recall and comparing it against a confidence level. The CMH method consists of a first part of the process that uses

an AL method and a confidence level which, once reached, stops the AL process and starts a second part that continues reviewing using random sampling and a higher confidence level. Following an approach similar to Yang et al.'s (2021a) for picking our baselines, we will not use the random sampling part of CMH in our experiments and only use its heuristic method as a stopping rule; notice, however, that the random sampling and the confidence level estimation which follows the heuristic stopping criterion is applicable to any of the other methods we explore in this paper (including our method presented in Sect. 4).

CMH heuristic treats batches of previously screened documents as if they were random samples (an assumption somewhat similar to the one we make in Sect. 5.1); for subsets $A_i = \{d_{N_{seen}-1}, ..., d_{N_{seen}-i}\}$ of these documents they compute $p = \Pr(X \leq k)$, where $X \sim \text{Hypergeometric}(N, K_{tar}, n)$: n is the size of the subsample, N is the total number of documents and $K_{tar} = \lfloor \frac{\rho_{seen}}{R} - \rho_{AL} + 1 \rfloor$ represents the minimum number of relevant documents remaining at the start of sampling. This is done for all sets A_i with $i \in N_{seen} - 1...1$; p_{min} is the value where the null-hypothesis (i.e., recall being below target) is lowest. The review of documents proceeds with AL until $p_{min} < 1 - \alpha$; α is a confidence level, which is set to 95%.

2.3.3 The QuantCI method

The QuantCI method proposed by Yang et al. (2021a) leverages the classifier predictions (a logistic regression) to estimate the current recall, computes a confidence interval based on variance in the predictions, and stops the reviewing process when the lower bound of the confidence interval reaches the target recall.

More specifically, the estimated recall $\hat{R}$ is computed as:

$$\hat{R} = \frac{\widehat{Rel}(L)}{\widehat{Rel}(P)} = \frac{\sum_x^{|L|} \Pr(\oplus|x)}{\sum_x^{|P|} \Pr(\oplus|x)} \tag{1}$$

This estimate is based on modeling the relevance of a document i as the outcome of a Bernoulli random variable $D_i \sim Bernoulli(\Pr(\oplus|x))$. The 95% confidence interval (CI) is then computed as:

$$\pm 2 \sqrt{\frac{1}{\widehat{Rel}(P)^2} Var(D_L) + \frac{\widehat{Rel}(L)^2}{\widehat{Rel}(P)^4} \left(Var(D_L) + Var(D_U) \right)} \tag{2}$$

The authors tested their method with and without the confidence interval (i.e., using the recall estimate as is, not lowering it with the confidence interval) resulting in two stopping techniques, called QuantCI and Quant.

2.3.4 The QBCB method

The Quantile Binomial Confidence Bound (QBCB) stopping rule, proposed by Lewis et al. (2021), leverages on theory of quantile estimation to define a stopping rule that is a function of the target recall, a confidence value, and the size r of a human annotated sample of relevant documents P_r. Given these three values, QBCB returns the minimum number of relevant documents $j \leq r$ from P_r that have to be found while annotating P to have a statistical guarantee to reach the target recall on P within the given confidence value. The actual equation used by QBCB is:

$$\sum_{k=0}^{j-1} \binom{r}{k} t^k (1-t)^{r-k} \geq 1 - \alpha \tag{3}$$

where $1 - \alpha$ is the confidence level. The equation is tested for increasing values of j until it is satisfied. This methods thus requires an initial phase of human annotation of documents randomly sampled from P in order to define the set P_r. A larger size of P_r raises this initial annotation cost, yet it produces a more accurate estimation of j, with a lower expected annotation cost for the main annotation phase. Experiments in Lewis et al. (2021) on various dataset of different difficulty and prevalence have different optimal values for r, which minimize the average overall annotation cost, in the range of 30 to 60 samples (see Lewis et al. 2021, Fig. 3), with a more accurate targeting of recall for larger r values.

2.3.5 The IPP method

The Inhomogeneous Poisson Process Power (IPP) was recently proposed by Sneyd and Stevenson (2021). The authors actually propose several stopping rules based on counting processes, that is, stochastic models of the number of occurrences of an event over time (Sneyd and Stevenson 2021, §3). In order to be applied to TAR, the authors treat position in a search ranking as "time", and occurrences of relevant documents as "events".

Poisson processes assume that the number of occurrences follow a Poisson distribution: if the rate at which events occur varies, the process is said to be inhomogeneous. Furthermore, Poisson processes have a λ rate, a function representing the frequency with which events occur over the space (i.e., our ranking). More in details,

$$\Lambda(a, b) = \int_a^b \lambda(x) dx \tag{4}$$

If we indicate with $N(a, b)$ a random variable denoting the number of events occurring in the interval $(a, b]$, then:

$$P(N(a, b) = r) = \frac{[\Lambda(a, b)]^r}{r!} e^{-\Lambda(a,b)}, \tag{5}$$

where N has a Poisson distribution with expected value $\Lambda(a, b)$. The λ function is unknown, and the goal is to choose a suitable one for the problem at hand.

The IPP method uses a Poisson process with a power curve function as the rate function. Given a counting process such as IPP, the stopping criterion is then applied as in Algorithm 2 [which we report from (Sneyd and Stevenson 2021, Algorithm 1)].

Algorithm 2: Sneyd and Stevenson stopping algorithm [27].

Input : No. of documents in ranking n ;
target recall R ;
confidence level p ;
α initial sample size ;
β sample increment size
Output: stopping rank s

```
s ← α × n;
while s < n do
    Fit Counting Process (N(0,s)) to documents in range 1
     ... s;
    ρ ← CDF of N(0, s) > p;
    if ρR < rel(s) then
        break;
    end
    s ← s + β × n ;
end
return s
```

2.4 Using the SLD algorithm in AL processes

The Saerens–Latinne–Decaestecker (SLD) algorithm (Saerens et al. 2002) was proposed as a technique to adjust a priori and a posteriori probabilities (priors and posteriors here after) in prior probability shift (PPS) scenarios, i.e. when the prior probability Pr(y) diverges between the labelled L and the unlabelled U sets (Moreno-Torres et al. 2012). The algorithm works by iteratively and jointly updating the prior and posterior probabilities (see Algorithm 3).

Algorithm 3: The SLD algorithm [11].

Input : Class priors $\Pr_L(y_j)$ on L, for all $y_j \in \mathcal{Y}$;
Posterior probabilities $\Pr(y_j|\mathbf{x}_i)$, for all $y_j \in \mathcal{Y}$ and for all $\mathbf{x}_i \in U$;
Output: Estimates $\hat{\Pr}_U(y_j)$ on U, for all $y_j \in \mathcal{Y}$;
Updated posterior probabilities $\Pr(y_j|\mathbf{x}_i)$, for all $y_j \in \mathcal{Y}$ and for all $\mathbf{x}_i \in U$;

```
// Initialization
```
$s \leftarrow 0$;
for $y_j \in \mathcal{Y}$ **do**
  $\hat{\Pr}_U^{(s)}(y_j) \leftarrow \Pr_L(y_j)$; `// Initialize the prior estimates`
  for $\mathbf{x}_i \in U$ **do**
    $\Pr^{(s)}(y_j|\mathbf{x}_i) \leftarrow \Pr(y_j|\mathbf{x}_i)$; `// Initialize the posteriors`
  end
end
`// Main Iteration Cycle`
while *stopping condition* = *false* **do**
  $s \leftarrow s + 1$;
  for $y_j \in \mathcal{Y}$ **do**
    $\hat{\Pr}_U^{(s)}(y_j) \leftarrow \frac{1}{|U|} \sum_{\mathbf{x}_i \in U} \Pr^{(s-1)}(y_j|\mathbf{x}_i)$; `// Update the prior estimates`
    for $\mathbf{x}_i \in U$ **do**
      `// Update the posteriors`
      $$\Pr^{(s)}(y_j|\mathbf{x}_i) \leftarrow \frac{\frac{\hat{\Pr}_U^{(s)}(y_j)}{\hat{\Pr}_U^{(0)}(y_j)} \cdot \Pr^{(0)}(y_j|\mathbf{x}_i)}{\sum_{y_j \in \mathcal{Y}} \frac{\hat{\Pr}_U^{(s)}(y_j)}{\hat{\Pr}_U^{(0)}(y_j)} \cdot \Pr^{(0)}(y_j|\mathbf{x}_i)}$$
    end
  end
end
`// Generate output`
for $y_j \in \mathcal{Y}$ **do**
  $\hat{\Pr}_U(y_j) \leftarrow \hat{\Pr}_U^{(s)}(y_j)$; `// Return the prior estimates`
  for $\mathbf{x}_i \in U$ **do**
    $\Pr(y_j|\mathbf{x}_i) \leftarrow \Pr^{(s)}(y_j|\mathbf{x}_i)$; `// Return the adjusted posteriors`
  end
end

AL policies, such as relevance and uncertainty sampling, naturally tend to generate a high PPS: in particular, when $\Pr_P(\oplus)$ is fairly low to start with (as it usually is in TAR applications), the AL process generates a PPS such that $\Pr_L(\oplus) \gg \Pr_U(\oplus)$. Hence, using SLD to improve both our prevalence estimates and our posteriors might seem like a promising idea. However, recent works (Esuli et al. 2022; Molinari et al. 2023) have shown that in this context SLD has a disastrous effect on the posteriors. In the next sections, we analize this behaviour (Sect. 3) and propose a solution (Sect. 4) that enables using SLD in AL (and hence, in TAR).

3 An analysis of SLD shortcomings in active learning scenarios

Two recent studies (Esuli et al. 2022; Molinari et al. 2023) have shown how SLD can have disastrous behaviours in AL contexts, leading to an extremization of the posterior probability distribution, i.e. the fact that most (if not all) posterior probabilities $\Pr(\oplus|x)$ are dragged to either 0 or 1. Experiments from Esuli et al. (2022) show that when SLD is applied to *Rand*(*pol*) version of an AL policy, be it either ALvRS or ALvUS, the phenomenon of corruption of posteriors does not happen. The cause of the issue is thus to be found in the document selection policy, and in the fact that both ALvRS and ALvUS produce a *sampling bias* (Dasgupta and Hsu 2008; Krishnan et al. 2021), which leads to a dataset shift where $\Pr_L(y) \neq \Pr_U(y)$ and $\Pr_L(x|y) \neq \Pr_U(x|y)$.

Sampling bias emerges from AL algorithms due to both the selection policy *pol* and the initial seed S. S is usually very small (in many TAR applications, it may consist of a single positive instance). Considering the ALvRS policy, the classifier trained on S will have high confidence on documents similar to the few ones contained in S. As the AL process continues, the training set L will diverge from the underlying data distribution. A classifier trained on such a biased training set can hardly make sense of the whole document pool. Indeed, Dasgupta and Hsu (2008, §2) shows that the classifier can be overly confident in attributing the negative label to a cluster of data which actually contains several positive instances.

A visual representation of the sampling bias is shown in Fig. 1, from Molinari et al. (2023), which shows how the document selection is extremely focused on one small region of the positive instances.

When it comes to the classifier capability to estimate the prevalence of relevant documents in the unlabeled part of the pool U, this means that its estimates are going to be much lower than those of a classifier trained on a random and representative sample of the same population: we can see this in Table 1, where we compare the prevalence estimates of a calibrated SVM classifier trained on the ALvRS training set (at different sizes) and the same classifier trained on a controlled random sample of the pool, with same size and same positive prevalence (we call ALvRS SVM and *Rand*(RS) SVM, the two SVMs trained on the two different training sets). The estimate is the average of the posterior probabilities for the relevant class $\Pr(\oplus|x)$ for all $x \in U$.

The last two columns of the table report the true prevalence of the positive class in L and U, showing the strong PPS generated by AL techniques; clearly, the shift is stronger if $p_P(\oplus)$ is already fairly low to start with (which is usually the case in many TAR applications). In this scenario, the classifier usually overestimates $p_U(\oplus)$, given its bias on $p_L(\oplus)$ prevalence, which is what we see for *Rand*(RS) SVM. The values in the table seem to indicate that the ALvRS-based estimates are better than the *Rand*(RS) ones. We argue that this is actually due to the sampling bias: the classifier is very likely underestimating the prevalence of those positive clusters it does not know about; the end result is that the output prevalence estimate is much lower than the *Rand*(RS) and incidentally closer to the real prevalence of U.

In order to better visualize how sampling bias affects the AL trained classifier, we give a visual representation of this phenomenon on a synthetic dataset:

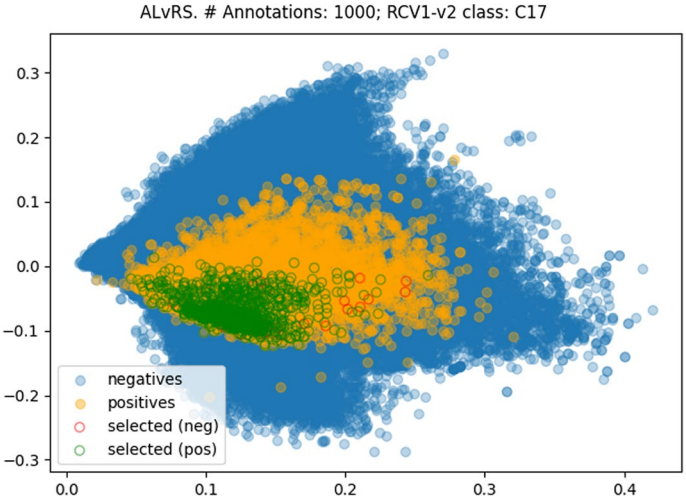

Fig. 1 LSA visualization of documents in the pool of an ALvRS experiment. Yellow/Blue indicates relevant/non-relevant documents. Green/Red circles indicates relevant/non-relevant documents selected via RS. A strong sampling bias selects documents in a restricted region. See Molinari et al. (2023, Fig. 1) (Color figure online)

Table 1 Prevalence estimates of an SVM classifier trained on a ALvRS and *Rand*(RS) training set respectively, compared to true prevalences of the L and U sets

Size of L	$\hat{p_U}(\oplus)$			
	ALvRS	*Rand* (RS)	$p_L(\oplus)$	$p_U(\oplus)$
2000	0.048	0.141	0.500	0.058
4000	0.065	0.158	0.709	0.040
8000	0.063	0.121	0.686	0.013
16,000	0.008	0.064	0.405	0.003
23,149	0.004	0.048	0.284	0.002

1. We generate an artificial dataset consisting of 10,000 data items with four clusters (blue, yellow, purple and pink in Fig. 2). Blue and yellow clusters are the positive clusters (i.e., every item in these clusters has a positive $\oplus$ label); purple and pink clusters are the negative clusters (i.e., every item in these clusters has a negative $\ominus$ label). Notice that negative clusters are much more populated than positive ones (i.e., the overall positive prevalence is low);
2. We start the active learning process with two positive items coming from one of the positive clusters and 10 negative items, randomly sampled from the negative clusters. We then annotate 500 documents with the ALvRS policy and generate an analogous training set with the *Rand*(RS) policy. The two training sets are shown with "X" markers in Fig. 2a, b, for ALvRS and *Rand*(RS) respectively;
3. We show the estimated (and the true) proportion of positive items remaining in each cluster, for a Calibrated SVM trained on the ALvRS and *Rand*(RS) training sets respectively. Furthermore, we show in the title the true $\mathrm{P}_U(\oplus)$ and estimated prevalence ($\mathrm{P}_U^{\mathrm{ALvRS}}(\oplus)$ and $\mathrm{P}_U^{Rand}(\oplus)$) on the test set.

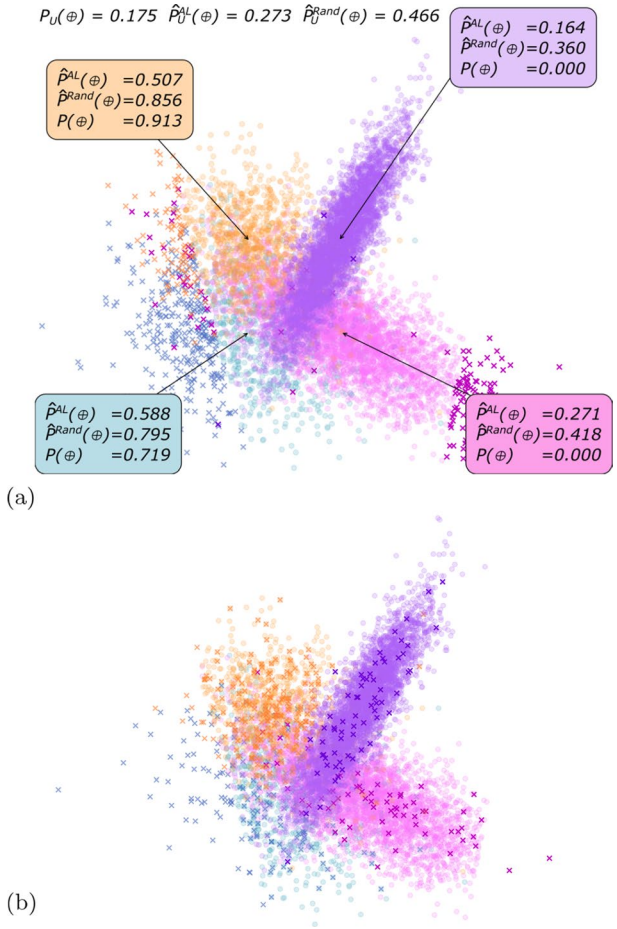

Fig. 2 ALvRS and *Rand*(RS) applied to synthetic data. **a** ALvRS training set (marked with X) and prevalence estimation of an SVMtrained on this training set. We report a *Rand*(RS) trained classifier estimationas well for completeness. The blue and yellow clusters are the "positive" clusters,whereas the pink and purple ones are the "negative" ones. **b** The Rand(RS) policy training set (marked with X), used for the Rand classifierin (**a**).

In Fig. 2a we see how the AL process annotates a specific subregion of positive items. This in turn "misleads" the classifier to output a much lower prevalence than the *Rand*(RS) classifier for the clusters it has never seen during training; notice that, being the overall positive prevalence quite low, the prevalence estimates of the AL classifier seem better than the *Rand*'s.

3.1 How is sampling bias related to SLD failures in active learning contexts?

The reason why sampling bias is a key element in our analysis lies beneath the main assumptions made by SLD on the posterior probability distribution of the classifier: SLD reasonably assumes to be in the scenario represented by the *Rand*(RS) policy rather than the AL one. More precisely, it assumes that there is no dataset shift on the conditional probability $\Pr(x|y)$ between the labelled set L and U, i.e., that $\Pr_L(x|y) = \Pr_U(x|y)$. If this assumption holds, the trained classifier will have a bias on L which will "uniformly" translate (i.e., in our previous example, the bias is consistent for all regions of the graph) to the posterior probabilities $\Pr_U(\oplus|x)$ on the unlabelled set. In a PPS scenario, this means that the classifier estimate of the prevalence (i.e., the average of the classifier posteriors) will be closer to L, and that they can be "adjusted" consistently across the whole distribution. Nonetheless, when using an active learning policy such as ALvRS or ALvUS, we not only generate prior probability shift, but we also affect the distribution of the conditional probability $\Pr(x|y)$, such that $\Pr_L(x|y) \neq \Pr_U(x|y)$.

Let us now focus on one of the key updates in SLD: the priors ratio (Line 13 of Algorithm 3), which is later multiplied by the posteriors. This is defined as:

$$\frac{\hat{\Pr}_U^{(s)}(\oplus)}{\Pr_L(\oplus)} \tag{6}$$

This linear relation is represented by the red line of Fig. 3. When $\hat{\Pr}_U(\oplus) = \Pr_L(\oplus)$, the ratio is 1, and posteriors do not change. However, as $\hat{\Pr}_U(\oplus)$ drifts further away from $\Pr_L(\oplus)$ (recall that this latter quantity is constant for all iterations in SLD), the ratio becomes progressively smaller, resulting in a multiplication of $\Pr_U(\oplus|x)$ by a number very close to 0.[2] We argue this is one of the main culprits of degenerated outputs from SLD.

In other words, let us assume that $\Pr_L(x|y) = \Pr_U(x|y)$, and that L is then a representative sample of U. A classifier trained and biased on L will tend to shift any prevalence estimate toward $\Pr_L(\oplus)$. When the classifier estimated $\hat{\Pr}_U(\oplus)$ is lower (or higher) than $\Pr_L(\oplus)$, SLD deems the true $\Pr_U(\oplus)$ to be even lower (or higher). Indeed, this works very well when the previous assumption holds, e.g., for *Rand*(pol). As a matter of fact, SLD has been a state-of-the-art technique for prior and posterior probabilities adjustments in PPS for 20 years. However, ALvRS and similar techniques generate a $\Pr_L(x|y) \neq \Pr_U(x|y)$ type of shift (as well as PPS), and, as a result, we cannot apply SLD update with confidence: we should rather find a way to apply a milder correction, when possible, or no correction at all when we have no way of estimating how "far" we are from the SLD assumption.

[2] This is the case for low prevalence scenarios, typical of TAR. The opposite case is the one with very high prevalence, in which the posteriors for the relevant class are all pushed to 1.

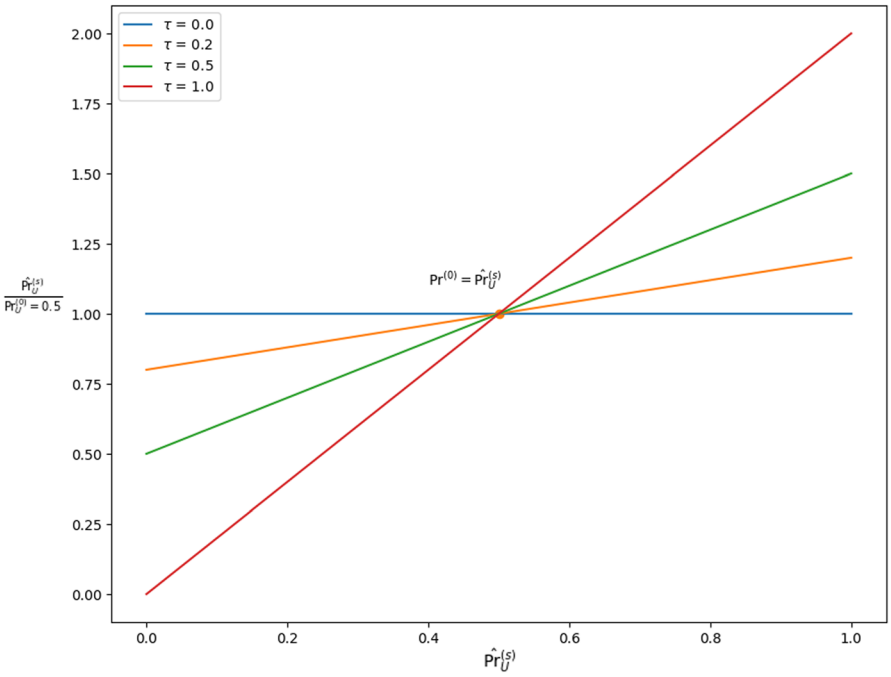

Fig. 3 The τ-based correction to the priors ratio of SLD that we propose. The value of this ratio (i.e., the y axis) is multiplied by the posteriors during SLD iterations. Notice that when $\tau = 1$ we get the SLD original ratio, whereas when $\tau = 0$ we multiply the posteriors by 1, i.e. we do not change the classifier posteriors

4 Adapting the SLD algorithm to active learning

As discussed in previous section, in AL scenarios, the priors ratio defined in the SLD algorithm should be milder in order to avoid extreme behaviours. A simple but possibly effective action is to directly add a correction factor τ to the priors ratio equation, so that:

- when $\tau = 1$ we get the original SLD algorithm;
- when $\tau = 0$ the ratio always equals 1, i.e., we do not apply any correction;
- all other intermediate values adjust the ratio, making it milder with respect to SLD original ratio.

We thus define a correction to the priors ratio of SLD:

$$\delta = -\left[\tau \cdot \left(-\frac{\hat{\mathrm{Pr}}_U^{(s)}(y)}{\hat{\mathrm{Pr}}_U^{(0)}(y)} + 1\right) - 1\right] \tag{7}$$

the new ratio, which we call δ, will then be multiplied by the posteriors at every SLD iteration. We show how τ affects the slope of the ratio in Fig. 3. We call our

method " **S**LD for **A**ctive **L**earning " or SAL_τ for short. The complete algorithm is reported in Algorithm 4.

Algorithm 4: The SAL_τ algorithm. Changes with respect to SLD are highlighted.

Input : Class priors $\Pr_L(y_j)$ on L, for all $y_j \in \mathcal{Y}$;
Posterior probabilities $\Pr(y_j|\mathbf{x}_i)$, for all $y_j \in \mathcal{Y}$ and for all $\mathbf{x}_i \in U$;
Correction factor τ;
Output: Estimates $\hat{\Pr}_U(y_j)$ on U, for all $y_j \in \mathcal{Y}$;
Updated posterior probabilities $\Pr(y_j|\mathbf{x}_i)$, for all $y_j \in \mathcal{Y}$ and for all $\mathbf{x}_i \in U$;

`// Initialization`
$s \leftarrow 0$;
for $y_j \in \mathcal{Y}$ **do**
  $\hat{\Pr}_U^{(s)}(y_j) \leftarrow \Pr_L(y_j)$; `// Initialize the prior estimates`
  for $\mathbf{x}_i \in U$ **do**
    $\Pr^{(s)}(y_j|\mathbf{x}_i) \leftarrow \Pr(y_j|\mathbf{x}_i)$; `// Initialize the posteriors`
  end
end
`// Main Iteration Cycle`
while *stopping condition = false* **do**
  $s \leftarrow s + 1$;
  for $y_j \in \mathcal{Y}$ **do**
    $\hat{\Pr}_U^{(s)}(y_j) \leftarrow \frac{1}{|U|} \sum_{\mathbf{x}_i \in U} \Pr^{(s-1)}(y_j|\mathbf{x}_i)$; `// Update the prior estimates`
    $\delta \leftarrow -\left[\tau \cdot \left(-\frac{\hat{\Pr}_U^{(s)}(y)}{\hat{Pr}_U^{(0)}(y)} + 1\right) - 1\right]$;
    for $\mathbf{x}_i \in U$ **do**
      `// Update the posteriors`
      $\Pr^{(s)}(y_j|\mathbf{x}_i) \leftarrow \frac{\delta \cdot \Pr^{(0)}(y_j|\mathbf{x}_i)}{\sum_{y_j \in \mathcal{Y}} \delta \cdot \Pr^{(0)}(y_j|\mathbf{x}_i)}$;
    end
  end
end
`// Generate output`
for $y_j \in \mathcal{Y}$ **do**
  $\hat{\Pr}_U(y_j) \leftarrow \hat{\Pr}_U^{(s)}(y_j)$; `// Return the prior estimates`
  for $\mathbf{x}_i \in U$ **do**
    $\Pr(y_j|\mathbf{x}_i) \leftarrow \Pr^{(s)}(y_j|\mathbf{x}_i)$; `// Return the adjusted posteriors`
  end
end

In the next section we detail how we set the value of τ.

4.1 Estimating SAL$_\tau$ τ across active learning iterations

We have seen that SLD works on the output of a *Rand*(RS)-based classifier (Esuli et al. 2022). We can build our estimate of τ for SAL$_\tau$ by measuring how much the ALvRS classifier posteriors are more or less distributed like those of a *Rand*(RS) classifier. The more ALvRS posteriors diverge from *Rand*(RS) ones, the milder SLD correction should be, up to the point where we do not use SLD at all.

Let us consider the posteriors for the relevant class, i.e., $\Pr(\oplus|x)$: we collect a vector $\mathcal{A}$ of posteriors $\Pr(\oplus|x)$ for all documents in U for the classifier trained on the ALvRS training set, and an analogous one $\mathcal{R}$ for the classifier trained on the *Rand*(RS) training set. We define τ as the cosine similarity between these two vectors:

$$\tau = \text{cosine similarity}(\mathcal{A}, \mathcal{R}) = \frac{\mathcal{A} \cdot \mathcal{R}}{||\mathcal{A}||\ ||\mathcal{R}||} \tag{8}$$

Since $\Pr(\oplus|x) \geq 0$ by definition of probability, the cosine similarity is naturally bounded between 0 and 1, a required property of our τ parameter. In other words, we apply the SLD update when AL posteriors are similar to posteriors for which we know the assumption made in SLD holds (i.e., $\Pr_L(x|y) = \Pr_U(x|y)$, see Sect. 3); we apply an accordingly milder correction the further the AL posteriors are from this assumption.

Equation (8) would be a good solution, as well as an impossible one, since the *Rand*(pol) policy requires knowledge of the labels (e.g., relevancy) for the entire data pool.

We thus resort to an heuristic formulation based on the evolution of the classifier during the iterations of the AL process.[3] Given the batch of documents B_i reviewed at the i-th iteration, we define $\mathcal{A}_{\phi_i}$ and $\mathcal{A}_{\phi_{i-1}}$ as:

$$\mathcal{A}_{\phi_i} = \Pr^{(i)}(\oplus, x) = \phi_i(x)|x \in B_i \tag{9}$$

$$\mathcal{A}_{\phi_{i-1}} = \Pr^{(i-1)}(\oplus, x) = \phi_{i-1}(x)|x \in B_i \tag{10}$$

i.e., the posteriors on B_i returned by the classifiers ϕ_i and ϕ_{i-1}. The τ parameter is then defined as the cosine similarity between $\mathcal{A}_{\phi_i}$ and $\mathcal{A}_{\phi i-1}$. The assumption we make is that:

- At early iterations, there will not likely be substantial differences between the two vectors (thus making the cosine similarity useless);
- However, as the active learning (AL) process progresses, this could assist us in obtaining an estimation of the sampling bias that impacts the classifier.

[3] This idea of leveraging previously annotated batches is more or less similar to what Callaghan and Müller-Hansen (2020) proposed in their method (see also Sect. 2.3.2).

Let us consider the ALvRS policy, in a scenario similar to the one depicted in Fig. 2a, that is, overall prevalence is low and we start with a small seed set: in the first iterations we are likely going to find many positive items; that is, when we compare ϕ_i and ϕ_{i-1} predictions on B_i they are likely to be similar, since the "clusters" of data available to ϕ_i are probably the same that were available to ϕ_{i-1}. However, as we review all the positive items in a cluster, the process is forced to explore items in the neighbour clusters. This is when the cosine similarity should be effective: by comparing the posterior distribution of ϕ_i to that of ϕ_{i-1} for the same B_i documents, we should be able to assess the impact of the new documents on the classifier. Indeed, if ϕ_i accessed previously unseen clusters, the posteriors on B_i might radically change and, in turn, roughly give us an estimate of how much sampling bias was affecting ϕ_{i-1} predictions.

Finally, let us consider one last issue: we said that in the first AL iterations $\mathcal{A}_{\phi_i}$ will likely be similar to $\mathcal{A}_{\phi_{i-1}}$, and their distance cannot be used as an indication of sampling bias. How can we establish when to use our method and when to fallback to the classifier posteriors? In lack of better solutions (which we defer to future works), we introduce a hyperparameter α:

- At every iteration i, we apply SAL_τ and obtain a new set of posterior probabilities $\Pr^{\text{SAL}_\tau}(\oplus|x)$ and a prevalence estimation $\Pr^{\text{SAL}_\tau}(\oplus)$ (computed as $\frac{\sum_x^X \Pr^{\text{SAL}_\tau}(\oplus|x)}{|X|}$);
- We measure the Normalized Absolute Error (NAE) between the estimated prevalence and the true prevalence of batch B_i, where $\text{NAE}(\Pr_{B_i}(y), \hat{\Pr}_{B_i}^{\text{SAL}_\tau}(y))$ is defined as:

$$\text{NAE} = \frac{\sum_{j=1}^{Y} |\Pr_{B_i}(y) - \hat{\Pr}_{B_i}^{\text{SAL}_\tau}(y_j)|}{2\left(1 - \min_{y_j \in Y} \Pr_{B_i}^{\text{SAL}_\tau}(y_j)\right)} \tag{11}$$

- If $\text{NAE} > \alpha$ we do not use our SAL_τ method.

The simple intuition behind this heuristic is that when the NAE between the true prevalence and the prevalence estimation of SAL_τ is too high, then the estimates of SAL_τ are likely going to be poor on the rest of the pool as well.

To recap, we give below an overview of how SAL_τ integrates into the active learning process (see Algorithm 5):

1. At each iteration i we employ an active learning policy, annotating a batch B_i of b documents, which are added to the training set L;
2. We train a classifier ϕ_i on L;
3. At each iteration $i > 1$, we compute the cosine similarity between the two vectors of scores $\mathcal{A}_{\phi_i} = \langle \Pr^{\phi_i}(\oplus|x) \; \forall \; x \in B_i \rangle$ and $\mathcal{A}_{\phi_{i-1}} = \langle \Pr^{\phi_{i-1}}(\oplus|x) \; \forall \; x \in B_i \rangle$. The cosine similarity will be used as the τ parameter in SAL_τ;

(a) We obtain a new set of posterior probabilities $\Pr^{\text{SAL}_\tau}(\oplus|x)$ and a new prevalence estimate $\Pr^{\text{SAL}_\tau}(\oplus)$ on U, using SAL$_\tau$ on the posteriors coming from ϕ_{i-1};

4. We compute NAE between SAL$_\tau$ prevalence estimate for B_i and the true prevalence of B_i. If this is lower than a threshold α, we consider SAL$_\tau$-based probabilities to be the correct ones, otherwise we fall back to ϕ_i-based probabilities. In a TAR process this means that we can use SAL$_\tau$-based probabilities to estimate the recall and decide when to stop reviewing documents.

Algorithm 5: SAL$_\tau$ integration within an active learning process

Input: Pool of documents P to be reviewed; Active learning policy a; Initial seed set S; Batch size b; Budget t; threshold value α;

$i \leftarrow 0$;
$L \leftarrow S$;
$\phi_i \leftarrow$ `train_clf`(L);
$B_i \leftarrow$ `select_via_policy`(ϕ_i, a, P, b);
$L \leftarrow L \cup B_i$;
while $|L| < t$ **do**
  $i \leftarrow i + 1$;
  $U \leftarrow P \setminus L$;
  $\phi_i \leftarrow$ `train_clf`(L);
  $A_i \leftarrow \langle \Pr^{\phi_i}(\oplus|x)\ \forall x \in B_{i-1} \rangle$;
  $A_{i-1} \leftarrow \langle \Pr^{\phi_{i-1}}(\oplus|x)\ \forall x \in B_{i-1} \rangle$;
  $\tau \leftarrow$ `cosine_similarity`(A_i, A_{i-1});
  $\Pr^{\text{SAL}_\tau}(\oplus|x) \leftarrow$ `SAL`$_\tau$`(`$\Pr^{\phi_{i-1}}(\oplus|x)\ \forall x \in U, \hat{\Pr}_L^{\phi_{i-1}}(\oplus), \tau$`)`;
  if `NAE`$(\Pr_{B_{i-1}}(\oplus), \hat{\Pr}_{B_{i-1}}^{\text{SAL}_\tau}(\oplus)) < \alpha$ **then**
    `// The new posteriors can be used to, e.g., estimate recall`
    $\Pr(\oplus|x) \leftarrow \Pr^{\text{SAL}_\tau}(\oplus|x)\ \forall x \in U \cup \Pr^{\phi_i}(\oplus|x)\ \forall x \in L$;
    $\hat{R} \leftarrow \frac{\sum_x^L \Pr(\oplus|x)}{\sum_x^P \Pr(\oplus|x)}$;
  end
  $B_i \leftarrow$ `select_via_policy`(ϕ_i, a, P, b);
  $L \leftarrow L \cup B_i$;
end

4.2 Mitigating SAL$_\tau$ recall overestimation: SAL$_\tau^m$

As it will be clear in the results section (Sect. 6), SAL$_\tau$ achieves significant improvements with respect to the compared methods. However, SAL$_\tau$ also tends, in some

cases and especially for higher recall targets, to overestimate the recall, stopping the TAR process too early. Leaving the study of more complex approaches to future work, we propose a simple way to mitigate the issue of recall overestimation by raising by a margin value the target recall given in input to the method. We call this variant SAL_τ^m (SAL_τ with margin m), which trades off an increment in annotation costs for a safer TAR process that reduces the early stops. SAL_τ^m is actually SAL_τ with the only difference being the use of a target recall R^m determined as a function of the target recall R and the margin m:

$$R^m = R + (1 - R)m \tag{12}$$

The margin m ranges from 0 to 1. When $m = 0$, $\text{SAL}_\tau^m = \text{SAL}_\tau$; when $m = 1$, $R^m = 1$. A low value means fully trusting SAL_τ, a high value means accepting to label more documents to avoid early stops. In order to avoid adding a free parameter to the method we decided to set $m = R$, following the intuition that it is crucial to guarantee a given target recall R, the closer R is to 1. Equation (12) thus becomes:

$$R^m = R + (1 - R)R = 2R - R^2 \tag{13}$$

We call this configuration SAL_τ^R and comment upon its results in Sect. 6. We defer to future works the exploration of more informed methods to set m, which could possibly enable a more convenient trade-off between annotation costs and proper recall targeting.

5 Experiments

5.1 Using SAL_τ to stop a TAR process

The SAL_τ algorithm can be tested in any AL scenario where we need to improve priors and posteriors. In this paper, we focus on testing SAL_τ capabilities for TAR: our goal is to stop the review process as soon as a target recall R is reached, lowering the review cost.

We test the SAL_τ algorithm with three configurations:

1. The SAL_τ formulation of Algorithm 5;
2. SAL_τ^R, i.e., with margin, as described in Sect. 4.2;
3. SAL_τCI, a variant of SAL_τ that uses the confidence interval heuristic from the QuantCI technique (Yang et al. 2021a).

We compare against the following well-known methods (see Sect. 2):

1. The Knee method by Cormack and Grossman (2015);
2. The Budget method by Cormack and Grossman (2015)
3. The CHM method by Callaghan and Müller-Hansen (2020);
4. The QuantCI method by Yang et al. (2021a);

5. The QBCB method by Lewis et al. (2021);
6. The IPP by Sneyd and Stevenson (2021).

For all methods that require a confidence interval, we set it to 95%. The positive sample size r of the QBCB method (see Sect. 2.3.4) is instead set at 50, which in the results reported in the original paper appears to be a good trade-off between a low overall cost and a low variance in such cost across different samples.

5.2 The active learning workflow

We run the same active learning workflow for all tested methods. In most TAR applications (Yang et al. 2021a; Cormack and Grossman 2015), we usually have only a single positive document to start the active learning with, called the initial seed: for our experiments, we decide to seed the active learning process with an additional negative document randomly sampled from the document pool P; that is, our initial seed set S consists of a positive and a negative document, randomly sampled from P.

As mentioned earlier (Sect. 2), the active learning policy we pick is CAL (Cormack and Grossman 2015) (a variation of ALvRS) since this is the most common policy used in TAR tasks. As the batch size b, we follow (Yang et al. 2021a) and set it to 100. The classifier we use in all our experiments is a standard Logistic Regression, as this is also the classifier of choice in most (if not all) TAR applications. We test the target recalls values $\{0.8, 0.9, 0.95\}$.

5.3 Datasets

We run our experiments[4] on two well-known datasets: the RCV1-v2 and the CLEF Technology-Assisted Reviews in Empirical Medicine (EMED) datasets (specifically, the dataset made available for the CLEF 2019 edition). Both datasets have been already used to test TAR frameworks and algorithms, e.g., the MINECORE framework (Oard et al. 2018), the QuantCI stopping technique (Yang et al. 2021a) and Li and Kanoulas sampling methodology (Li and Kanoulas 2020). We also use a sample of the Jeb Bush Email collection to set SAL$_\tau$ α hyperparameter.

All text is converted into vector representation first converting it to lowercase, removing English stopwords and any term occurring in more than 90% of the documents in P; vectors are weighted using TF-IDF.

5.3.1 RCV1-v2

RCV1-v2 (Lewis et al. 2004) is a publicly available collection of 804,414 news stories from the late nineties, published on the Reuters website. RCV1-v2 is a multi-label multi-class collection, i.e., every document can be assigned to one or more

[4] The code is available at https://github.com/levnikmyskin/salt.

classes from a set $\mathcal{C}$ of 103 classes. Since for our experiments we need binary classification datasets (i.e., a document can either be relevant or not), for each class $c \in \mathcal{C}$ we consider each document d as either belonging to c or not, thus obtaining 103 binary datasets. In the experiments, we use a random sample of 10,000 documents, for efficiency and to keep RCV1 pool size close to that of CLEF.

5.3.2 CLEF EMED 2019

The CLEF EMED datasets were made publicly available[5] for the TAR in EMED tasks ran from 2017 to 2019. The goal of the task was to assess TAR algorithms aimed at supporting the production of systematic reviews in empirical medicine. Following (Li and Kanoulas 2020), we use the Diagnostic Test Accuracy (DTA) reviews part of the dataset, working with the abstract relevance assessments (i.e., the first phase of the review, where the physician only assesses abstracts): the dataset consists of 72 "Training" topics and 8 "Testing" topics.

The texts of the reviewed documents are not available for download on the GitHub platform: they have to be downloaded from PubMED. While an HTTP API is available, the full text of documents are often under a paywall: hence the choice of focusing on the abstract reviews only. Moreover, we have encountered several issues in downloading some of the abstracts and we were unable to retrieve the whole dataset: in total we have retrieved abstracts for 60 topics (between "Training" and "Testing"), downloading a collection of 264,750 documents. Since we do not need to distinguish between a training and testing phase with different topics (e.g., for transfer learning), we merge together the "Training" and "Testing" subcollections.[6]

5.3.3 Jeb Bush Email collection

The Jeb Bush's emails collection consists of 290,099 emails sent and received by the former governor of Florida Jeb Bush. We used the subset published by Grossman et al.[7]: the sample consists of 9 topics, with 50,000 documents. For each document and topic, a relevance judgment is available. We do not run experiments on this sample, but only use it to set the hyperparameter α (see Sect. 4.1).

5.4 Evaluation measures

We evaluate the tested methods using three measures:

1. The Mean Square Error (MSE) between the recall at stopping R_s and the target recall R:

[5] https://github.com/CLEF-TAR/tar.
[6] We published the dataset at (Molinari 2022).
[7] https://github.com/hical/sample-dataset.

$$\text{MSE} = (R - R_s)^2 \tag{14}$$

2. The Relative Error (RE):

$$\text{RE} = \frac{|R - R_s|}{R} \tag{15}$$

3. The "idealized" cost (IC) presented by Yang et al. (2021b). This measure specifically evaluates a stopping method on its capability of saving effort and money when stopping the TAR process.

The first two measures are well-known and used in many different tasks in IR and machine learning literature, and they have been used to evaluate TAR tasks (Li and Kanoulas 2020; Yang et al. 2021a).

The IC metric is defined as follows: it uses a *cost structure* (similar to what was previously done in Oard et al. (2018)), a four-tuple $s = (\alpha_p, \alpha_n, \beta_p, \beta_n)$. Subscript p and n indicate the cost of reviewing positive (relevant) and negative (non-relevant) documents; α and β represents the costs of reviewing a document in a first or second phase: in a one-phase TAR process, this "second" phase is referred to as the *failure penalty*. That is, it would be the cost of continuing the review with an optimal second phase, by ranking documents with the model trained in the first one.

Let Q be the minimum number of documents to review to reach the recall target R. Say we review batches of size b and we stop at iteration t: let Q_t be the number of documents we reviewed before the method stopped the review. If $Q_t < Q$, we have a deficit of $Q - Q_t$ positive documents; let ρ_t be the number of documents that need be reviewed,[8] following the ranking, to find the additional $Q - Q_t$ positive documents. The total cost of our review is then:

$$\begin{aligned} \text{IC} = &\alpha_p Q_t + \alpha_n(bt - Q_t) + I[Q_t < Q](\beta_p(Q - Q_t) \\ &+ \beta_n(\rho_t - Q + Q_t)) \end{aligned} \tag{16}$$

Where $I[Q_t < Q]$ is 0 if Q documents were found in the first t iterations and 1 otherwise.

Following both Yang et al. (2021b) and Oard et al. (2018), we evaluate our results with three cost structures:

- a uniform cost structure, where $s = (1, 1, 1, 1)$, which we call Cost$_u$. This assumes that there is no difference between the different phases of review, and that reviewing positive and negative documents have the same cost (we keep this latter assumption in all our cost structures). As argued in Yang et al. (2021b, §4.1), this cost structure is common in many review scenarios;
- the expensive training cost structure, where $s = (10, 10, 1, 1)$, which we call Cost$_e$. This assumes that reviewing a document in the first phase is 10 times

[8] Note that the value of ρ_t can be computed, at evaluation time, since we know the relevance labels of all documents in the pool in the experimental setting.

Table 2 MSE and RE results on RCV1. For each target recall, bold indicates the best result, underline indicates the second best

	All		Very Low		Low		Medium	
	MSE	RE	MSE	RE	MSE	RE	MSE	RE
Recall = 0.8								
BudgetKnee	0.032	0.222	0.038	0.242	0.030	0.214	0.029	0.211
CHM	0.037	0.240	0.039	0.248	0.038	0.243	0.034	0.228
IPP	0.090	0.300	0.030	0.204	0.057	0.253	0.184	0.444
Knee	0.035	0.231	0.040	0.250	0.035	0.233	0.029	0.211
QBCB	**0.022**	0.174	0.038	0.243	**0.014**	**0.145**	**0.013**	0.135
Quant	0.039	0.248	0.040	0.250	0.040	0.249	0.038	0.243
QuantCI	0.040	0.249	0.040	0.250	0.040	0.250	0.039	0.246
SAL_τ	0.025	**0.167**	0.040	0.200	0.021	0.173	0.014	**0.129**
SAL_τ^R	0.026	0.188	**0.028**	**0.186**	0.027	0.200	0.022	0.177
SAL_τCI	0.031	0.213	0.040	0.250	0.038	0.243	0.016	0.146
Recall = 0.9								
BudgetKnee	**0.007**	0.087	0.009	0.104	0.006	0.079	0.005	0.077
CHM	0.009	0.108	0.010	0.111	0.010	0.109	0.009	0.103
IPP	0.107	0.242	**0.008**	**0.089**	0.066	0.190	0.248	0.446
Knee	0.008	0.095	0.010	0.111	0.008	0.096	0.005	0.077
QBCB	0.007	0.091	0.010	0.110	0.006	0.087	0.005	0.076
Quant	0.010	0.111	0.010	0.111	0.010	0.111	0.010	0.109
QuantCI	0.010	0.111	0.010	0.111	0.010	0.111	0.010	0.110
SAL_τ	0.010	**0.077**	0.022	0.110	**0.004**	**0.064**	**0.004**	**0.057**
SAL_τ^R	0.007	0.080	0.009	0.089	0.005	0.075	0.005	0.076
SAL_τCI	0.009	0.104	0.010	0.111	0.010	0.111	0.008	0.090
Recall = 0.95								
BudgetKnee	**0.001**	**0.035**	**0.002**	**0.046**	**0.001**	**0.031**	**0.001**	**0.027**
CHM	0.002	0.051	0.003	0.053	0.002	0.052	0.002	0.048
IPP	0.123	0.230	0.004	0.054	0.077	0.178	0.287	0.457
Knee	0.002	0.041	0.002	0.052	0.002	0.042	0.001	0.027
QBCB	0.003	0.053	0.003	0.053	0.003	0.053	0.003	0.053
Quant	0.002	0.052	0.003	0.053	0.003	0.053	0.002	0.052
QuantCI	0.002	0.053	0.003	0.053	0.003	0.053	0.002	0.053
SAL_τ	0.006	0.051	0.015	0.079	0.001	0.035	0.003	0.039
SAL_τ^R	0.002	0.039	0.002	0.047	0.001	0.036	0.001	0.033
SAL_τCI	0.002	0.051	0.003	0.053	0.003	0.053	0.002	0.049

more expensive than in the second phase. According to Yang et al. (2021b, §4.2) this is fairly common in systematic reviews in empirical medicine;

- a MINECORE-like cost structure, where $s = (1, 1, 5, 5)$, which we call Cost_m. This cost structure reflects MINECORE cost structure 2 (Oard et al. 2018, Table 5), where we assume that reviewing in the second stage (in MINECORE case, it was reviewing by privilege) is 5 times as expensive as in the first stage.

Table 3 MSE and RE results on CLEF. For each target recall, bold indicates the best result, underline indicates the second best

	All		Very Low		Low		Medium	
	MSE	RE	MSE	RE	MSE	RE	MSE	RE
Recall = 0.8								
BudgetKnee	0.035	0.230	0.035	0.229	0.031	0.219	0.038	0.242
CHM	0.038	0.245	0.039	0.248	0.038	0.242	0.038	0.244
IPP	0.047	0.223	0.047	0.225	0.058	0.231	0.037	0.212
Knee	0.038	0.245	0.039	0.248	0.037	0.241	0.039	0.246
QBCB	0.027	0.198	0.032	0.216	0.022	0.177	**0.027**	**0.199**
Quant	0.039	0.247	0.040	0.249	0.039	0.248	0.038	0.244
QuantCI	0.040	0.249	0.040	0.250	0.040	0.250	0.039	0.248
SAL$_\tau$	**0.027**	**0.174**	0.034	0.174	**0.017**	**0.146**	0.029	0.201
SAL$_\tau^R$	0.027	0.190	**0.026**	**0.173**	0.026	0.193	0.030	0.205
SAL$_\tau$CI	0.036	0.235	0.040	0.250	0.034	0.224	0.035	0.230
Recall = 0.9								
BudgetKnee	0.008	0.094	0.008	0.093	0.006	0.084	0.009	0.104
CHM	0.010	0.109	0.010	0.111	0.010	0.109	0.010	0.108
IPP	0.041	0.137	0.037	0.135	0.061	0.159	0.025	0.119
Knee	0.009	0.106	0.010	0.109	0.009	0.103	0.009	0.107
QBCB	0.008	0.096	0.009	0.101	0.007	0.087	**0.008**	**0.098**
Quant	0.010	0.110	0.010	0.111	0.010	0.110	0.010	0.108
QuantCI	0.010	0.111	0.010	0.111	0.010	0.111	0.010	0.111
SAL$_\tau$	0.013	0.089	0.021	0.100	**0.005**	**0.069**	0.013	0.100
SAL$_\tau^R$	**0.007**	**0.087**	**0.007**	**0.079**	0.006	0.084	0.008	0.099
SAL$_\tau$CI	0.010	0.110	0.010	0.111	0.010	0.109	0.010	0.111
Recall = 0.95								
BudgetKnee	**0.002**	**0.042**	**0.002**	**0.050**	**0.001**	**0.031**	**0.002**	**0.047**
CHM	0.002	0.052	0.003	0.053	0.002	0.052	0.002	0.051
IPP	0.044	0.113	0.038	0.107	0.068	0.140	0.026	0.091
Knee	0.002	0.048	0.002	0.051	0.002	0.045	0.002	0.049
QBCB	0.003	0.053	0.003	0.053	0.003	0.053	0.003	0.053
Quant	0.002	0.052	0.003	0.053	0.003	0.053	0.002	0.052
QuantCI	0.003	0.053	0.003	0.053	0.003	0.053	0.003	0.053
SAL$_\tau$	0.009	0.062	0.017	0.086	0.003	0.039	0.007	0.061
SAL$_\tau^R$	0.002	0.047	0.004	0.057	0.001	0.035	0.002	0.048
SAL$_\tau$CI	0.003	0.053	0.003	0.053	0.003	0.053	0.003	0.053

6 Results

We run each of the experiments defined in the previous section 20 times, using a different randomly generated seed set S each time (the same random seed set for all methods compared); we set $\alpha = 0.3$ (Sect. 4.1), as this was the best-performing value in a hyperparameter search conducted on the Jeb Bush dataset. Therefore, the

Table 4 IC measure on RCV1. For each target recall, bold indicates the best result, underline indicates the second best

	All			Very Low			Low			Medium		
	C_u	C_e	C_m	C_u	C_e	C_m	C_u	C_e	C_m	C_u	C_e	C_m
Recall = 0.8												
BudgetKnee	2933	29,330	2933	5069	50,686	5069	1333	13,331	1333	2397	23,975	2397
CHM	3882	38,818	3882	6080	60,802	6080	2710	27,102	2710	2855	28,552	2855
IPP	2124	19,648	2831	4437	44,352	4444	**740**	**7137**	854	**1196**	**7456**	3196
Knee	4459	44,594	4460	7987	79,867	7987	2960	29,597	2960	2432	24,317	2433
QBCB	9163	91,625	9163	19,123	191,231	19,123	5770	57,699	5770	2595	25,946	2595
Quant	6373	63,725	6373	7669	76,687	7669	6515	65,150	6515	4934	49,338	4934
QuantCI	8719	87,192	8719	10,000	100,000	10,000	9855	98,549	9855	6303	63,028	6303
SAL_τ	**982**	**9391**	**1175**	**688**	**5996**	1083	792	7923	**793**	1467	14,255	1649
SAL_τ^R	1186	11,612	1297	739	6678	**1058**	1006	10,056	1006	1813	18,101	1828
SAL_τCI	6667	66,644	6677	10,000	100,000	10,000	8402	84,020	8402	1598	15,912	**1628**
Recall = 0.9												
BudgetKnee	2933	29,330	2933	5069	50,686	5069	1333	13,331	1333	2397	23,975	2397
CHM	5325	53,250	5325	8193	81,933	8193	4090	40,905	4090	3691	36,912	3691
IPP	2289	20,070	3543	4496	44,673	4626	**845**	**7533**	1253	**1526**	**8003**	4750
Knee	4460	44,594	4460	7987	79,867	7987	2960	29,597	2960	2432	24,317	2434
QBCB	9672	96,716	9672	19,190	191,899	19,190	6389	63,892	6389	3436	34,356	3436
Quant	7772	77,720	7772	8841	88,414	8841	7994	79,935	7994	6481	64,811	6481
QuantCI	9691	96,915	9691	10,000	100,000	10,000	10,000	100,000	10,000	9074	90,745	9074
SAL_τ	**1146**	**10,430**	**1602**	**832**	**6428**	**1675**	911	8997	**959**	1694	15,866	**2173**
SAL_τ^R	1531	14,905	1709	1293	11,915	1744	1079	10,730	1105	2220	22,070	2279
SAL_τCI	8742	87,375	8759	10,000	100,000	10,000	10,000	100,000	10,000	6225	62,126	6278
Recall = 0.95												
BudgetKnee	2967	29,364	3103	5069	50,686	5069	1377	13,375	1554	2455	24,032	**2687**

Table 4 (continued)

	All			Very Low			Low			Medium		
	C_u	C_e	C_m	C_u	C_e	C_m	C_u	C_e	C_m	C_u	C_e	C_m
CHM	6727	67,273	6727	9641	96,411	9641	6064	60,644	6064	4476	44,764	4476
IPP	2524	20,433	4662	4596	44,903	5068	**1031**	**7864**	2116	**1946**	**8533**	6804
Knee	4484	44,618	4582	7987	79,867	7987	2974	29,611	3034	2490	24,376	2725
QBCB	15,336	153,360	15,336	19,987	199,868	19,987	15,056	150,564	15,056	10,965	109,647	10,965
Quant	8699	86,991	8699	9475	94,747	9475	8917	89,172	8917	7705	77,055	7705
QuantCI	9929	99,294	9929	10,000	100,000	10,000	10,000	100,000	10,000	9788	97,881	9788
SAL$_\tau$	**1387**	**11,542**	**2422**	**1013**	**6870**	**2460**	1092	9983	1507	2057	17,773	3299
SAL$_\tau^R$	2234	21,642	2543	2873	27,842	3269	1271	12,247	**1475**	2557	24,837	2884
SAL$_\tau$CI	9675	96,699	9695	10,000	100,000	10,000	10,000	100,000	10,000	9024	90,097	9085

Table 5 IC measure on CLEF. For each target recall, bold indicates the best result, underline indicates the second best

	All			Very Low			Low			Medium		
	C_u	C_e	C_m	C_u	C_e	C_m	C_u	C_e	C_m	C_u	C_e	C_m
Recall = 0.8												
BudgetKnee	1661	16,606	1661	2988	29,880	2988	1129	11,288	1129	865	8651	865
CHM	2443	24,435	2443	4773	47,726	4773	1736	17,358	1736	822	8220	822
IPP	1198	11,327	1486	2454	23,838	2764	663	**6070**	912	**476**	**4074**	782
Knee	3006	30,064	3006	5725	57,250	5725	2283	22,834	2283	1011	10,108	1011
QBCB	5238	52,377	5238	11,611	116,108	11,611	3018	30,178	3018	1084	10,844	1084
Quant	3329	33,292	3329	6512	65,122	6512	2576	25,758	2576	900	8995	900
QuantCI	5145	51,450	5145	10,095	100,954	10,095	4172	41,715	4172	1168	11,681	1168
SAL_τ	**600**	**5678**	**745**	**680**	**6447**	837	**632**	6234	**673**	489	4355	725
SAL_τ^R	724	7183	751	774	7641	**819**	850	8497	850	549	5410	**583**
SAL_τCI	4657	46,440	4716	10,095	100,954	10,095	3132	31,306	3136	745	7059	917
Recall = 0.9												
BudgetKnee	1661	16,606	1661	2988	29,880	2988	1129	11,288	1129	865	8651	865
CHM	3271	32,715	3271	6433	64,328	6433	2432	24,322	2432	949	9493	949
IPP	1343	11,676	2120	2721	24,314	4009	**762**	**6407**	1299	**545**	**4307**	1053
Knee	3006	30,064	3006	5725	57,250	5725	2283	22,834	2283	1011	10,108	1011
QBCB	5965	59,590	5990	13,217	132,003	13,292	3378	33,782	3378	1298	12,984	1298
Quant	4187	41,870	4187	8053	80,534	8053	3397	33,968	3397	1111	11,108	1111
QuantCI	5524	55,238	5524	10,095	100,954	10,095	4956	49,556	4956	1520	15,204	1520
SAL_τ	**770**	**6472**	1316	**974**	**7371**	**2026**	775	7335	**958**	561	4710	964
SAL_τ^R	1011	9471	**1296**	1290	11,010	2132	1044	10,428	1048	699	6975	**707**
SAL_τCI	5336	53,361	5336	10,095	100,954	10,095	4393	43,926	4393	1520	15,204	1520
Recall = 0.95												
BudgetKnee	1734	16,680	**2026**	3205	30,097	**4072**	1131	11,291	**1142**	865	8651	865

Table 5 (continued)

	All			Very Low			Low			Medium		
	C_u	C_e	C_m	C_u	C_e	C_m	C_u	C_e	C_m	C_u	C_e	C_m
CHM	4015	40,152	4015	7846	78,465	7846	3108	31,081	3108	1091	10,910	1091
IPP	1547	11,966	3104	3127	24,795	6006	**902**	**6669**	1947	**611**	**4434**	1357
Knee	3006	30,064	3006	5725	57,250	5725	2283	22,835	2284	1011	10,108	1011
QBCB	8892	88,919	8892	17,656	176,556	17,656	7032	70,316	7032	1988	19,884	1988
Quant	4767	47,671	4767	9010	90,100	9010	4022	40,215	4022	1270	12,698	1270
QuantCI	5537	55,371	5537	10,095	100,954	10,095	4995	49,954	4995	1520	15,204	1520
SAL$_\tau$	**998**	**7230**	2221	**1434**	**8220**	4156	920	8212	1357	640	5259	1149
SAL$_\tau^R$	1311	11,361	2086	1975	14,596	4268	1150	11,422	1184	807	8067	**807**
SAL$_\tau$CI	5537	55,371	5537	10,095	100,954	10,095	4995	49,954	4995	1520	15,204	1520

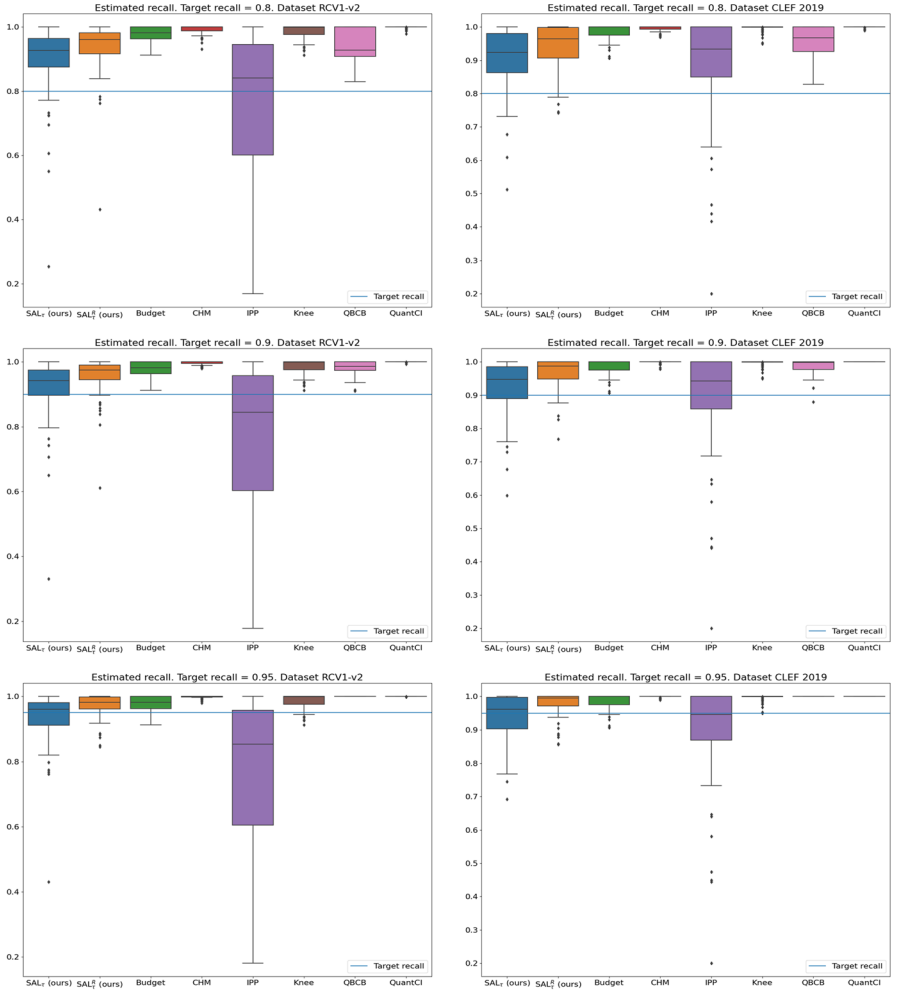

Fig. 4 Box plots of actual recall reached by the methods, given a target recall: 0.8 (top), 0.90 (center), 0.95 (bottom). Left column is RCV1, right column is CLEF

reported results for a dataset represent the average evaluation measure applied to all generated subsets across all classes within that dataset. We also define three equal-sized bins sorted by class prevalence (Very low, Low, and Medium) to show how the different methods performed with different level of imbalance between relevant and non-relevant documents. The average prevalence for the classes in the three bins of RCV1 is 0.002, 0.012, and 0.084, and for CLEF is 0.005, 0.027, and 0.117.

Tables 2 and 3 report the MSE and RE values, Tables 4 and 5 report the IC values. The most competitive models are SAL$_\tau$, SAL$^R_\tau$, Budget, QBCB and the IPP method. For the MSE and RE metrics our SAL$_\tau$ and SAL$^R_\tau$ stopping rules performs on par, or better, than the other state-of-the-art techniques. More in details, for lower

target recalls ($R = 0.8$ and $R = 0.9$) the above-mentioned methods show a comparable performance, both for RCV1-v2 and CLEF datasets. When $R = 0.95$, the Budget method performs slightly better instead.

With respect to the IC measure, SAL_τ shows the lowest costs on average (and for the Very Low prevalence bin) for all proposed cost structures, closely followed by SAL_τ^R and IPP. Notice that the IPP method, despite achieving good cost results, was not as performant when previously compared on the MSE and RE metrics. On the other hand, the QBCB method, which achieves state-of-the-art performance on MSE and RE, incurs very high costs due to the annotation of the pre-review random sample required by the method. Furthermore, the most relevant reduction of cost shown by SAL_τ and SAL_τ^R, with respect to the compared methods, is for the Very Low prevalence bin. This is an important result, as the "needle in a haystack" scenario is the most common in TAR. The Budget method has comparable costs for high target recall values and in higher prevalence bins.

Overall, we argue that our SAL_τ^R method seems to strike the best trade-off between proper recall targeting (i.e., actually stopping the review at the given recall target) and low costs.

Indeed, as we anticipated, SAL_τ tends to stop the TAR process too early and, as a result, cannot be consistently used in all scenarios despite achieving lower costs. This can be seen in the box plots in Fig. 4, which show at which real recall values the different stopping rules decided to halt the review process. The average recall value for SAL_τ is always higher than the target recall, i.e. the expected value of recall satisfies the target recall requirement. Yet, it is evident how for higher recall targets, the distribution of recall values produced by SAL_τ goes under the target not only for the tail of the distribution, as it happens for Budget and Knee, but also for a portion of the center part of the distribution. Most TAR tasks require to match the target recall in a much larger portion of the cases. By simply adding a margin that is a function of the target recall, SAL_τ^R shifts the distribution of the reached recall values up. Its distribution is similar to the Budget method's, with slightly increasing costs with respect to SAL_τ but still lower than the other tested methods.

Finally, notice that the IPP method, despite achieving good costs in some cases (and good MSE/RE values in some other), cannot stop the review process at the right moment, often greatly overshooting the recall estimate.

To summarize, SAL_τ enables the use of SLD in AL and TAR processes, solving the issues observed in Esuli et al. (2022) and Molinari et al. (2023). SAL_τ brings consistent and substantial improvements, especially for medium/high (0.8, 0.9) target recalls; for higher targets, it tends to stop too early. SAL_τ^R solves this issue without a significant increase in annotation costs with respect to SAL_τ, finding a very good trade-off between annotation costs and proper target recall matching. In future works we propose to investigate more informed methods to mitigate SAL_τ target recall overestimation, as well as testing SAL_τ and SAL_τ^R with other sampling techniques (e.g., Li and Kanoulas 2020).

7 Conclusions

In this paper, we introduced a new method called SAL_τ (and its variations SAL_τ^R and $SAL_\tau CI$) to improve posterior probabilities and prevalence estimates in an active learning review process; more specifically, we tested our method as a "when-to-stop" stopping rule for TAR tasks. SAL_τ is a variant of the well-known SLD algorithm (Saerens et al. 2002): our algorithm was designed to enable the use of SLD in AL and TAR processes, solving the issues observed in Esuli et al. (2022) and Molinari et al. (2023). Experiments have shown that SAL_τ still tends to slightly overestimate the true recall as it gets close to 1, stopping the process too early. SAL_τ^R solves this issue, without significantly increasing the review cost compared to SAL_τ. In the experiments, SAL_τ^R has consistently improved over state-of-the-art methods by improving the estimation of prevalence and thus stopping the TAR process much earlier than other methods, while still achieving the target recall.

For future work, we propose to investigate more informed methods to mitigate the overestimation of target recall of SAL_τ, as well as to test SAL_τ and SAL_τ^R with other sampling techniques (e.g., Li and Kanoulas 2020).

Author contributions Both authors equally contributed to the definition of the proposed method and in writing the paper. Alessio Molinari wrote the python implementation of the method used in the experiments.

Data availability The implementation of the proposed method and the code to replicate the experiments is published at https://github.com/levnikmyskin/salt. The version of the CLEF EMED 2019 dataset used in the experiments of this paper is available at https://doi.org/10.5281/zenodo.7142639 (Molinari 2022).

Declarations

Conflict of interest Authors declare to have no competing interests.

Ethical approval Not applicable.

References

Cormack GV, Grossman MR (2020) Systems and methods for a scalable continuous active learning approach to information classification. Google Patents. US Patent 10:671–675

Callaghan MW, Müller-Hansen F (2020) Statistical stopping criteria for automated screening in systematic reviews. Syst Rev 9(1):1–14

Cormack GV, Grossman MR (2015) Autonomy and reliability of continuous active learning for technology-assisted review. CoRR abs/1504.06868

Cormack GV, Grossman MR (2016) Engineering quality and reliability in technology-assisted review. In: Proceedings of the 39th ACM conference on research and development in information retrieval (SIGIR 2016), Tokyo, JP, pp 75–84. https://doi.org/10.1145/2911451.2911510

Cormack GV, Grossman MR (2016) Scalability of continuous active learning for reliable high-recall text classification. In: Proceedings of the 25th ACM conference on information and knowledge management (CIKM 2016), pp 1039–1048. https://doi.org/10.1145/2983323.2983776

Cormack GV, Grossman MR, Hedin B, Oard DW (2010) Overview of the TREC 2010 legal track. In: Proceedings of the 19th text retrieval conference (TREC 2010)

Dasgupta S, Hsu D (2008) Hierarchical sampling for active learning. In: Proceedings of the 25th international conference on machine learning (ICML 2018), Stockholm, SE, pp 208–215

Esuli A, Molinari A, Sebastiani F (2022) Active learning and the Saerens–Latinne–Decaestecker algorithm: an evaluation. In: CIRCLE 2022: 2nd joint conference of the information retrieval communities in Europe

Grossman MR, Cormack GV (2011) Technology-assisted review in e-discovery can be more effective and more efficient than exhaustive manual review. Richmond J Law Technol 17(3):5

Huang S, Jin R, Zhou Z (2014) Active learning by querying informative and representative examples. IEEE Trans Pattern Anal Mach Intell 36(10):1936–1949. https://doi.org/10.1109/TPAMI.2014.2307881

Kanoulas E, Li D, Azzopardi L, Spijker R (2019) CLEF 2019 technology assisted reviews in empirical medicine overview. In: Working notes of the conference and labs of the evaluation forum (CLEF 2019), Lugano, CH

Konyushkova K, Sznitman R, Fua P (2017)Learning active learning from data. Adv Neural Inf Process Syst 30

Krishnan R, Sinha A, Ahuja N, Subedar M, Tickoo O, Iyer R (2021) Mitigating sampling bias and improving robustness in active learning. arXiv preprint arXiv:2109.06321

Lease M, Cormack GV, Nguyen AT, Trikalinos TA, Wallace BC (2016) Systematic review is e-discovery in doctor's clothing. In: Proceedings of the SIGIR 2016 medical information retrieval workshop (MedIR 2016), Pisa, IT

Lewis DD, Gale WA (1994) A sequential algorithm for training text classifiers. In: Proceedings of the 17th ACM international conference on research and development in information retrieval (SIGIR 1994), Dublin, IE, pp 3–12. https://doi.org/10.1007/978-1-4471-2099-5_1

Lewis DD, Yang Y, Rose TG, Li F (2004) RCV1: a new benchmark collection for text categorization research. J Mach Learn Res 5:361–397

Lewis DD, Yang E, Frieder O (2021) Certifying one-phase technology-assisted reviews. In: Proceedings of the 30th ACM international conference on information and knowledge management. CIKM '21. Association for Computing Machinery, New York, NY, USA, pp 893–902. https://doi.org/10.1145/3459637.3482415

Li D, Kanoulas E (2020) When to stop reviewing in technology-assisted reviews: sampling from an adaptive distribution to estimate residual relevant documents. ACM Trans Inf Syst 38(4):41–14136. https://doi.org/10.1145/3411755

Michelson M, Reuter K (2019) The significant cost of systematic reviews and meta-analyses: a call for greater involvement of machine learning to assess the promise of clinical trials. Contemp Clin Trials Commun 16:100443

Molinari A (2022) CLEF EMED 2019 dataset. Zenodo. https://doi.org/10.5281/zenodo.7142640

Molinari A, Esuli A, Sebastiani F (2023) Improved risk minimization algorithms for technology-assisted review

Moreno-Torres JG, Raeder T, Alaız-Rodrıguez R, Chawla NV, Herrera F (2012) A unifying view on dataset shift in classification. Pattern Recognit 45(1):521–530. https://doi.org/10.1016/j.patcog.2011.06.019

O'Mara-Eves A, Thomas J, McNaught J, Miwa M, Ananiadou S (2015) Using text mining for study identification in systematic reviews: a systematic review of current approaches. Syst Rev 4(5):1–22

Oard DW, Sebastiani F, Vinjumur JK (2018) Jointly minimizing the expected costs of review for responsiveness and privilege in e-discovery. ACM Trans Inf Syst 37(1):11–11135. https://doi.org/10.1145/3268928

Saerens M, Latinne P, Decaestecker C (2002) Adjusting the outputs of a classifier to new a priori probabilities: a simple procedure. Neural Comput 14(1):21–41. https://doi.org/10.1162/089976602753284446

Satopaa V, Albrecht J, Irwin D, Raghavan B (2011) Finding a" kneedle" in a haystack: Detecting knee points in system behavior. In: 2011 31st International conference on distributed computing systems workshops, pp 166–171. IEEE

Shemilt I, Khan N, Park S, Thomas J (2016) Use of cost-effectiveness analysis to compare the efficiency of study identification methods in systematic reviews. Syst Rev 5(140):1–13. https://doi.org/10.1186/s13643-016-0315-4

Sneyd A, Stevenson M (2021) Stopping criteria for technology assisted reviews based on counting processes. In: Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval, pp 2293–2297

Wang S, Scells H, Mourad A, Zuccon G (2022) Seed-driven document ranking for systematic reviews: a reproducibility study. In: European conference on information retrieval, pp 686–700. Springer

Yang E, Lewis DD, Frieder O (2021) Heuristic stopping rules for technology-assisted review. In: Proceedings of the 21st ACM symposium on document engineering (DocEng 2021), Limerick, IE, pp 31–13110. https://doi.org/10.1145/3469096.3469873

Yang E, Lewis DD, Frieder O (2021) On minimizing cost in legal document review workflows. In: Proceedings of the 21st ACM symposium on document engineering, pp 1–10