Information Retrieval Journal (2021) 24:371–399
https://doi.org/10.1007/s10791-021-09396-2

Combining semi-supervised and active learning to rank algorithms: application to Document Retrieval

Faiza Dammak[1] · **Hager Kammoun**[1]

Received: 8 January 2020 / Accepted: 13 August 2021 / Published online: 4 October 2021

Abstract
Generally, the purpose of learning to rank methods is to combine the results from existing ranking models that within a single ranking function, applied to order the documents as efficiently as possible, improving the quality lists of results returned. However, learning to rank has several limitations namely the creation and size of the labeled database. We have considered the two frameworks of semi-supervised and active learning in order to look for solutions to these problems. We have been interested in semi-supervised, active and semi-active learning to rank algorithms for Document Retrieval (DR) which is a ranking application of alternatives. A good balance between exploration and exploitation has a positive impact on the performance of the learning. Thus, we have focused firstly on two active learning to rank algorithms that use supervised learning and semi-supervised learning as auxiliaries and use an automatic method for the labeling of unlabeled pairs selected. These algorithms are named "Semi-Active Learning to Rank: SAL2R" and "Active-Semi-Supervised Learning to Rank: ASSL2R". We have been particulary interested in providing efficient and effective algorithms to handle a large set of unlabeled data. Second, we have considered improvement of these semi-active SAL2R and ASSL2R algorithms using a multi-pair in the selection step. Our contribution lies particulary in the in depth experimental study of the performance of these algorithms and precisely the influence of certain fixed parameters on the learned ranking function.

Keywords Learning to rank algorithms · Supervised learning · Semi-supervised learning · Active learning · Information Retrieval · Document Retrieval · Evaluation

Faiza Dammak
faiza.dammak@gmail.com

Hager Kammoun
hager.kammoun@gmail.com

[1] MIRACL Laboratory, Institute of Computer Science and Multimedia of Sfax, Technology Center of Sfax, Sfax University, Tunis Road Km 10, B.P. 242, 302 Sfax, Tunisia

1 Introduction

In recent years, learning to rank has been successfully applied to a wide variety of applications in Information Retrieval (IR) such as Document Retrieval (DR) (Duh & Kirchhoff, 2008; Liu, 2011). Learning to rank can be seen as a task of a supervised learning (Chapelle & Chang, 2011; Liu, 2011) whose applications usually require the use of a large set of labeled data to accurately train a model. Since these labels might be costly to acquire, active learning (Ailon, 2012; Brinker, 2004; Long et al., 2014; Settles, 2010) and semi-supervised learning (Amini et al., 2008; Duh & Kirchhoff, 2008; Li et al., 2009; Zhu, 2005) technologies aim to reduce manual labeling workload. The model of each method is constructed with a small set of labeled examples and a large set of unlabeled ones. The semi-supervised learning method is more focused on exploiting the data while the active learning method is dedicated to the exploration of these data. The latter interacts with the expert (oracle) to label the selected most informative examples; which is usually expensive and time-consuming. Consequently, using the active learning or semi-supervised learning independently may lead to a poor performance in some cases. Moreover, in literature there are few studies that combine these two learning methods using selectively sampled and automatically labeled data; and to our knowledge, this method has not been studied in the context of learning to rank classifiers (Dammak et al., 2017a).

Therefore, our first aim is to be interested in the evaluation part of the two active learning to rank algorithms of alternatives that combine an active learning to rank method (Dammak et al., 2015) with semi-supervised learning to reduce the labeling effort for DR. We want to give a particular interest to the adjustment of their respective parameters and particulary to the impact on their performances. Our motivation is still to learn with a small set of labeled data in order to reduce the time and to take advantage of both types of learning (active and semi-supervised) and avoid some problems caused by employing only active or semi-supervised learning. Our second aim consists in considering some parameter setting related especially to the number of labeled examples and those to be labeled for the automatic labeling method. This method will be used in the labeling phase of these two algorithms and hence exclude the intervention of the expert for labeling. We expected that the combination will bring a gain in time (efficiency) and will improve the experimental results (effectiveness) by granting a particular interest to learning parameter settings that will be presented in the rest of this paper. These algorithms, referred to as “Semi-Active Learning to Rank” (SAL2R) and “Active-Semi-Supervised Learning to Rank” (ASSL2R), can deal with the most informative examples for improving the performance of the training (Dammak et al., 2017a). The idea, in these algorithms, is to select only the most informative query-document unlabeled pair at each round and specify, afterwards, if the document is relevant or not in relation with this query.

At last, we would like to reconsider the algorithms proposed in Dammak et al. (2017b) to further consolidate the validation part by analyzing the respective influence of each parameter of the learning model. These algorithms select at each round more than query-document pair from the unlabeled training data instead of choosing only one pair. These algorithms, referred to as “Semi-Active List Learning to Rank” (SALL2R) and “Active-Semi-Supervised List Learning to Rank” (ASSLL2R), can deal with a list of most informative query-document pairs for improving the selection strategy and thereby improve their performances and reducing selection time (Dammak et al., 2017b).

In this paper, DR is considered as an application for learning to rank. In this framework, when some queries are given with the associated labeled documents in training, the learning to rank system commonly focuses on learning an effective ranking function which assigns a score to each document, and ranks the documents with respect to the query in a descending order of their scores. Each query-document pair is characterized by a feature vector.

The rest of the paper is organized as follows. Section 2 discusses some related works in the domain of IR and briefly introduces the active and semi-supervised learning to rank literature. Section 3 describes in details the learning to rank algorithms object of our interest in this paper by emphasizing on some parameters subject of the evaluation part. We want to carry on an experimental study to identify which parameters influence the performance of the learned model better than others. Section 4 displays and analyses the experimental results related to each algorithm defined by its fixed parameters. The main conclusions of this research study are drawn and some potential perspectives are suggested in Sect. 5.

2 Related works

The central problem in IR is to extend the Information Retrieval Systems (IRS) with efficient and fast models, taking into account the user's information need. Thus, many works have focused on the proposal of for automatically optimizing the ranking of the search results returned by the IRS.

In recent years, more and more machine learning technologies have been used to form ranking models. A new area of research called Learning to Rank (discriminative learning) has gradually emerged as an attractive technique. This latter is dedicated to the optimization of ranking results and based on automatic learning techniques. The capacity of combining a large number of features is a very important advantage of learning to rank.

There are two major approaches to learning to rank, referred to as *pairwise* approach (Burges et al., 2005), and *listwise* approach (Cao et al., 2007). These approaches learn to rank in different ways and have been successively applied to IR. In the pairwise approach (Cao et al., 2007; Burges et al., 2005), pairs of documents for a given query are considered as input to the learning system. The objective here is to determine which document is more relevant than another. This approach can take into account the order of relationships between pairs of documents. In the learning to rank literature, several pairwise ranking algorithms have been proposed, based on boosting (Freund et al., 2003), neural network (Burges et al., 2005), support vector machines (Joachims, 2002) and other learning machines.

The listwise approach (Xia et al., 2008) takes all of the documents associated with a query in the learning data and predicts their labels. The input space of this approach contains a set of documents related to a query. The output space contains the ordered list (or permutation) of the documents according to their relevance or the list of their relevancy scores.

In general, the performance of ranking models is greatly affected by labeled examples' number in the training set (Chapelle & Chang, 2011; Duh & Kirchhoff, 2008; Liu, 2011). Since these labels might be expensive to acquire as labeling is usually scarce and costly to get in many applications (Li, 2011; Settles, 2010), semi-supervised learning and active learning technologies (Settles & Craven, 2008) try to tackle the same issue of getting and economizing the number of unlabeled data to learn a specific model. Their ranking algorithms have attracted a greater deal of research interest (Liu, 2011; Pan et al., 2013).

Generally, the semi-supervised learning concentrates more on the exploitation of unlabeled data. It tries to label examples by the machine itself (Li et al., 2009; Liu, 2011; Zhu, 2005). Furthermore, it selects the example that has the highest confidence in each round and adds the predicted label by the machine without any human involvement (Chapelle et al., 2006; Zhu, 2005).

Transductive and inductive learning are two useful and complementary paradigms whose arise from the semi-supervised learning. They look for labeling any unlabeled data to improve the performance of semi-supervised learning algorithms.

The transductive framework is only interested in unlabeled instances of the training set. It is then unable to order new data absent in the learning phase. Whereas the inductive framework has a very different purpose: the aim is to be able to order any data set. It consists firstly of finding a function (model) from the training data, and then applying this function to the new test data. The inductive learning is therefore able to order new data that are absent in the learning phase. In fact, inductive and transuctive methods seek to label any unlabeled data. The disadvantage is that, these methods are extremely complex to efficiently deal with a large amount of unlabeled data. In order to improve the performance of learning to rank, active methods, based on active learning have been proposed. The active learning approach (Freund et al., 1997; Roy & McCallum, 2001; Settles, 2010; Tong, 2001) selects the example that has the lowest confidence for labeling as the most informative one in each round (Ailon, 2012; Brinker, 2004; Long et al., 2014; Settles, 2010). It needs human involvement for the labeling and incorporates the obtained information to select new examples.

On the one hand, this type of learning typically reduces the number of unlabeled data that needs to be labeled (Kuwadekar and Neville, 2011). Indeed, the learner can impact the choice of learning examples which should be selected for labeling. Therefore, this paradigm is more dedicated to the exploration of unlabeled data (Settles, 2010; Huang et al., 2010). Thus, active learning can significantly improve the model's performance and accelerate the convergence's speed. On the other hand, it proposes to the user some optimal selection strategies for the ranking of alternatives in order to construct the training set of the model (Ailon, 2012) and determine which alternatives are most informative (Truong, 2009; Settles & Craven, 2008). The most well-known strategies are uncertainty sampling, Query By Committee (QBC) (Seung et al., 1992) and expected error reduction (Truong, 2009).

Although the advantages of both active and semi-supervised learning methods in saving efficiently the number of labeled data, there is little research focusing on combining them and dealing with an automatic step to label the most informative examples for learning to rank. However, there is a good deal of research on combining these techniques in different fields related to IR (Huang et al., 2010; Gu et al., 2014; Song et al., 2011; Krithara et al., 2011; Muslea et al., 2002; Leng et al., 2013).

Furthermore, other research studies suggested to introduce a step of automatic labeling the unlabeled data (Tur et al., 2005; Zhou et al., 2006) and proved that these methods have improved the performance of their results. In the same way, Dammak et al. (2017a) have proposed two new inductive learning to rank algorithms for DR which combine active and semi-supervised learning to assign the relevance scores to an unlabeled set of document-query pairs and Dammak et al. (2017b) have improved these inductive algorithms by using a multi-pairs query-document in the selection stage. The results obtained have proved the performance of these previous algorithms and have shown that this is a promising line of research that enhance the labeling process of the unlabeled data and thus increases the efficiency of costly human labelers. In this paper, we would like to further consolidate the evaluation part.

Techniques and methodologies have been proposed to construct learning to rank data sets that lead to an efficient learning to rank with a reduced cost of obtaining relevance judgments. These methods face the challenge of how to select the appropriate queries, the appropriate documents to be judged and the evaluation metric, for an efficient, reliable and effective evaluation and learning to rank. The major goal of these methods is to select only a small subset of documents. The document selection though should be done in a way that not harms the effectiveness of learning.

These methods are based on random sampling, by considering statistical methods to estimate the values of measures (Inferred Average Precision (InfAP) Yilmaz & Aslam, 2006; Aslam et al., 2006). Other methods utilize stratified sampling (Statistical Average Precision (StatAP) sampling Pavlu, 2008) or a greedy online algorithm (Minimal Test Collection (MTC) (Carterette et al., 2006). They try to test whether low cost methods produce reliable evaluation when used to select documents and how many queries are necessary and needed to draw robust conclusions.

In the next section, we present the algorithms (Dammak et al., 2017a, 2017b) as well as the parameters that we set and deem relevant to consider in the experimental study.

3 Learning to rank algorithms

In this section, we consider two inductive learning to rank algorithms which combine active and semi-supervised learning in order to build ranking models. These learning to rank algorithms are well adapted to DR (Truong 2009), where documents are considered as alternatives and queries as entries or observations. These algorithms focus on the active learning to rank in the context of alternatives (Dammak et al., 2015) to select the appropriate query-document pairs and consider the supervised and semi-supervised learning as auxiliaries to learn ranking functions. The main idea of combining them efficiently can further reduce the task of the manual labeling and takes advantage of their frameworks. Moreover, the original idea that we assume is to use a labeling algorithm instead of resorting to an expert for the labeling process.

In both algorithms a small data set of labeled examples denoted $S_L = \{(x_i, y_i); i \in \{1, \dots, m\}\}$ and a large data set of unlabeled examples denoted $S_U = \{(x_i^{'}); i \in \{1 + m, \dots, n + m\}\}$ are considered. Each y_i is a vector of variable size m_i, where m_i is the number of candidate alternatives for x_i, thus $y_i = (y_i^1, y_i^2, \dots, y_i^{m_i})$, where y_i^k expresses the degree of relevance of the k^{th} alternative. In this setting, the process is generally given a set of queries (observations or entries) $X = (x_1, x_2, \dots, x_n)$, a set of documents (alternatives) A and a set of labels (real output (scores)) $Y(y_i^k \in Y)$. We assume that each query $x_i \in X$ is related to a subset of known alternatives $A_{x_i} \subset A$ considered with labels grouped as a variable-size vector y_i. The y_i vector specifies the order that is to be predicted on alternatives. The score function h that should predict this order, considers an input pair (x_i, k) and returns a real score which reflects the similarity between an observation and an alternative, x_i represents an observation (query) and k represents an index of candidate alternative (document) for x_i.

Unlike the semi-supervised framework, we notice that there are two types of strategies of labeling for the active learning to rank: the first deals only with one entry and one alternative, whereas the second deals with all the alternatives related to the entry. The first

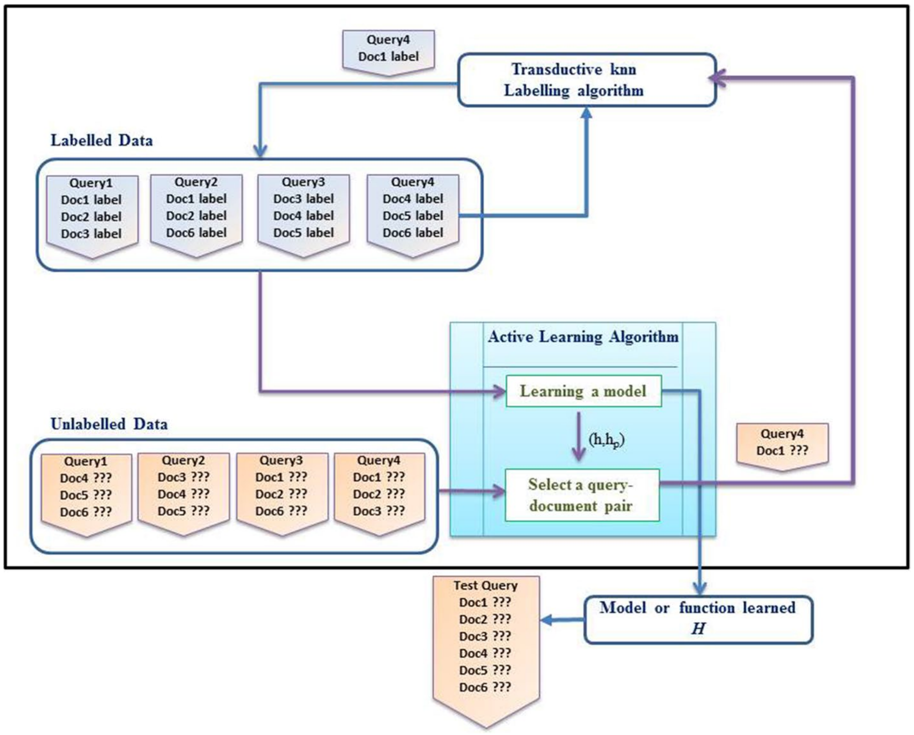


Fig. 1 Active approach for inductive learning with a transductive knn

methods use an uncertainty measure while the most recent select the examples that seem to change the most current model. Experimentally, these latter methods seem to be more competitive but suffer from greater complexity (Truong, 2009).

The basic idea, in these algorithms, is to select only one entry-alternative (query-document) pair at each round and determine, afterwards, if the alternative is relevant or not according to the considered entry. In this context, the algorithms employ the effective QBC selection strategy (Melville & Mooney, 2004) to select the pair which puts in conflict most of the members of all models called "committee models" (Freund et al., 1997). This strategy is typical one which maintains a committee of models. All models are trained only once on the initial labeled set, but represent challenging hypotheses. Each committee model is then considered in order to choose the appropriate pair. The goal of this strategy is minimizing the version space. Hence, the algorithms start by learning P ranking models called representative committee models $\{h_p\}_{p\in\{1,\ldots,P\}}$ and then learn a ranking model h (score function). Thereafter, they randomly select a model h_p among the P representative committee models.

Subsequently, they select the most informative query-document pair from the unlabeled dataset in each round. The pair corresponds to the one having the maximum measure of disagreement between the representative committee model h_p and the model h. Once the pair is selected, the labeling process is carried out automatically with a labeling algorithm. Nevertheless, these algorithms will proceed in two distinct ways (Fig. 1).

In what follows, we detail the particularities of each one.

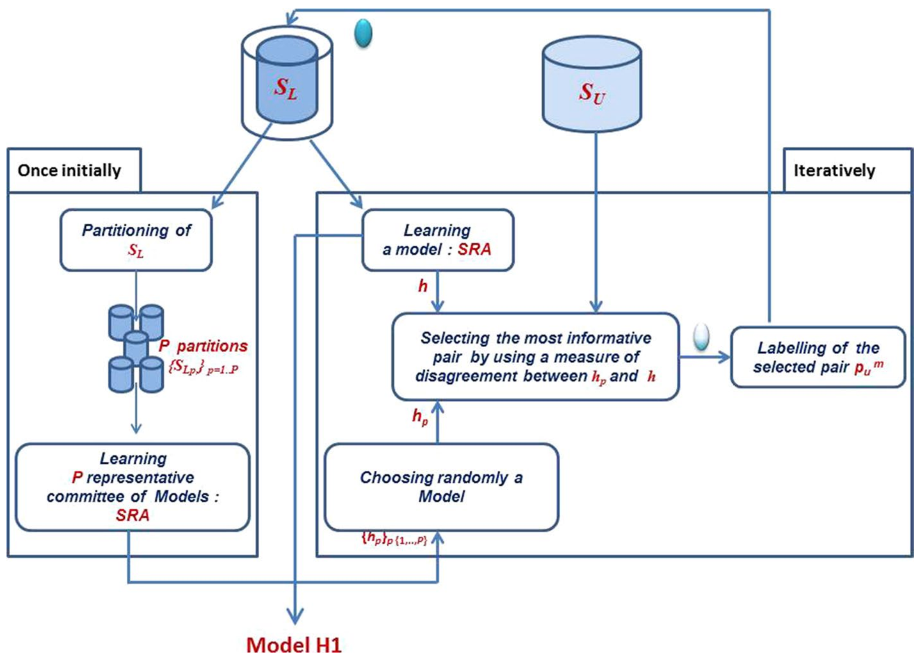

S_L
S_U
Once initially
Iteratively
Partitioning of S_L
P partitions
$\{S_{Lp}\}_{p=1..P}$
Learning P representative committee of Models : SRA
Learning a model : SRA
h
Selecting the most informative pair by using a measure of disagreement between h_p and h
Labelling of the selected pair p_u^m
h_p
Choosing randomly a Model
$\{h_p\}_{p\{1,...,P\}}$
Model H1

Fig. 2 Semi-active learning to rank proposition: SAL2R

3.1 Semi-active learning to rank algorithm: SAL2R

The SAL2R algorithm (Algorithm 2), described in Fig. 2, involves a supervised learning algorithm since the initial training set includes a small set of labeled query-document pairs in addition to the unlabeled ones.

SAL2R deals with two auxiliary algorithms:

- A supervised learning to rank algorithm (SRA) to learn the P representative committee models $\{h_p\}_{p\in\{1,\ldots,P\}}$ on S_L.
- A Transductive-knn labeling algorithm (Algorithm 1) whose role is to attribute the adequate label of the selected query-document pair which is considered as the most informative one.

At first, SAL2R includes an initial phase which consists in learning P representative committee models on the currently labeled pairs. For that, S_L is subdivided in P partitions for which the supervised learning algorithm is applied to generate P ranking committee models $\{h_p\}_{p\in\{1,\ldots,P\}}$. Each one is defined by a ranking function. Then, iteratively, the algorithm will randomly choose a model h_p among the learned P models. The effectiveness of this algorithm depends on the learning of the committee models which must be varied enough and representative of the entries space, as well as the choice of the measure of disagreement. As well, SAL2R applies, iteratively, the same supervised learning algorithm to learn a ranking model h, characterized by a ranking function from S_L. This function is updated, at each iteration, since the labeled set is increased by the newly selected labeled pair. Once the models are learned, SAL2R selects the most informative query-document pair from the unlabeled data set S_U. This pair $(x_{max}, kmax)$ defined as p_U^{max} corresponds to the one that maximizes the measure of disagreement between the representative committee model h_p chosen randomly and the model h for each unabeled query-document pair $(x_i^{'}, k)$ where $x_i^{'} \in S_U$. This measure is defined as follows:

$$(x_{max}, kmax) = argMax_{((x_i', k)} d_c(h, h_p, (x_i^{'}, k)) \tag{1}$$

$$d_c(h, h_p)_{(x_i', k)} = \sqrt{|(h(x_i', k) - h_p(x_i', k)|} \tag{2}$$

The basic idea at this stage is to introduce a labeling algorithm to label the selected pair p_U^{max}, referred to as transductive-knn algorithm (Algorithm 1). The main idea, inspired by knn algorithm, is to seek for the k-nearest labeled query-document pairs to the more informative selected pair. After that, we choose the label L predominantly represented for the k nearest labeled pairs (belonging to S_L). Finally, L is assigned as a label to the selected pair p_U^{max}.

At last, SAL2R withdraws the selected pair p_U^{max} from S_U and adds it to S_L. These steps are repeated until reaching the desired number of the data to be labeled. As output, the algorithm provides the model $H1$ characterized by the required score function.

Algorithm 1 *Transductive-knn Labeling Algorithm*

```
    Inputs
    Labeled pairs from $S_L$
    The most informative unlabeled query-document pair selected from $S_U$ : $p_U^{max}$
    Begin
    Calculate the scores $\{sc_L^i\}_{i \in \{1,\dots,m\}}$ of labeled query-document pairs by the learned function
    Calculate the score $sc_U^{max}$ of $p_U^{max}$
    Calculate the difference between the score of the unlabeled pair and all scores of labeled pairs
    Search the k nearest labeled pairs from $p_U^{max}$
    Select the label $L$ predominantly represented for the $k$ nearest pairs
    Assign this label $L$ to the unlabeled pair $p_U^{max}$
    End
    Output : Selected pair labeled : $p_L^{max}$
```

Many learning to rank algorithms have considered pairs of entries in the learning process. They are referred to as pairwise approaches (Cao et al., 2007), such as the supervised algorithms RankBoost (Freund et al., 2003) and LambdaMART (Qiang et al., 2010). Other learning to rank algorithms, have been proposed to solve the problem of ranking by minimizing a loss function defined on object lists. They are referred to as listwise approaches (Cao et al., 2007), such as the supervised algorithm AdaRank (Xu & Li, 2007). Therefore, these pairwise and listwise approaches may consider different input and output spaces, deal with different hypotheses, and are based on different loss functions (Liu, 2011). In DR, the input space of the pairwise approach is characterized by pairs of documents according to a given query, both represented by feature vector. The output space is represented by the pairwise preference (which takes values from $\{-1, +1\}$) between each pair of documents.

However, the input space of the listwise approach is characterized by the entire set of documents associated with a query in the training data. The output space of this approach is represented by the ranked list of the documents. During these years, each approach has shown higher empirical ranking performance as for its use in the IR field (Xia et al., 2008; Liu, 2011). We recommend to choose, as supervised algorithm the well-known boosting algorithms in the DR: RankBoost (Freund et al., 2003), AdaRank (Xu & Li, 2007) and LambdaMART (Qiang et al., 2010). The resulting algorithms are referred as SAL2R-RankBoost, SAL2R-AdaRank and SAL2R-LambdaMART respectively.

In the following, we give the SAL2R algorithm:

Algorithm 2 Semi-Active Learning to Rank algorithm: SAL2R

```
Inputs
Small set of labeled data $S_L = \{(x_i, y_i); i \in \{1, \dots, m\}\}$
Large set of unlabeled data $S_U = \{(x_i^{'}); i \in \{1 + m, \dots, n + m\}\}$
Supervised learning to rank algorithm SRA : RankBoost \ AdaRank \ LambdaMART
Labeling algorithm: Transductive knn
Number of $S_L$ partitions: $P$
Number of required examples to be labeled: $NbLab$
Begin
Learn $P$ committee models $\{h_p\}_{p\in\{1,\dots,P\}}$ with SRA
$nbIter \leftarrow 1$
While $nbIter <= NbLab$ do
Learn a ranking function $h$ with SRA on $S_L$
Choose randomly a committee model $h_p$
Select the most informative query-document pair $(p_U^{max}(x_{max}, kmax))$ from $S_U$ which maximizes the measure of disagreement $d_c(h, h_p)_{(x_i',k)\in S_U}$
Label the selected pair with the labeling algorithm
Withdraw this pair from $S_U$ and add it to $S_L$
$nbIter \leftarrow nbIter + 1$
End while
End
Output: Model H1
```

At this stage, we think that the following parameters are relevant and may have an influence on the performance of the learned ranking function.

- the considered supervised learning to rank algorithm: SRA.
- the number of S_L partitions: P.
- the number of labeled examples: S_L.
- the number of examples to be labeled $NbLab$. We notice here that $NbLab < n = |S_U|$.

3.2 Active-semi-supervised learning to rank algorithm: ASSL2R

The ASSL2R algorithm (Algorithm 3) differs from the SAL2R algorithm by the way that it achieves the learning of the ranking functions (Fig. 3). Indeed, we further assume that we consider a partially labeled pairs in the training set. But, the idea introduced here is to add a subset S_{U1}, extracted from the large set of the unlabeled data S_U, to the small amounts of labeled training set S_L. ASSL2R deals with three auxiliary algorithms:

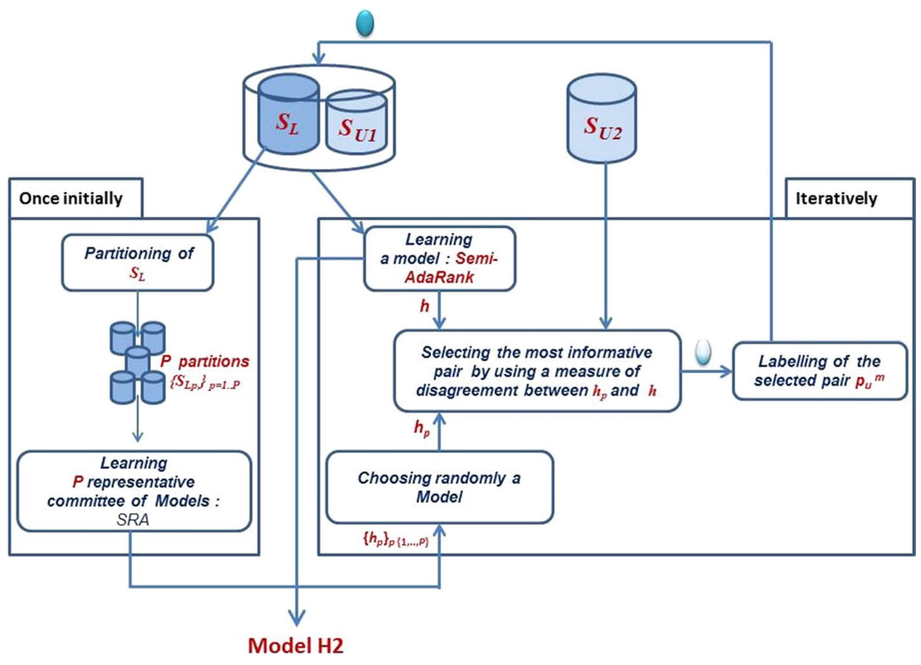

S_L
S_{U1}
S_{U2}
Once initially
Iteratively
Partitioning of S_L
P partitions $\{S_{Lp}\}_{p=1..P}$
Learning P representative committee of Models : SRA
Learning a model : Semi-AdaRank
h
Selecting the most informative pair by using a measure of disagreement between h_p and h
Labelling of the selected pair $p_u{}^m$
h_p
Choosing randomly a Model
$\{h_p\}_{p\{1,...,P\}}$
Model H2

Fig. 3 Active-semi-supervised learning to rank proposition: ASSL2R

- A supervised learning to rank algorithm (SRA) to learn the P representative committee models $\{h_p\}_{p\in\{1,\dots,P\}}$ on S_L.
- A semi-supervised learning to rank algorithm (SSRA) to learn a model h on $S_L \cup S_{U1}$.
- A Transductive-knn labeling algorithm (Algorithm 1).

As in the SAL2R algorithm, the most informative unlabeled pair p_U^{max} is the one that maximizes the measure of disagreement between the model h and the model h_p randomly chosen at each iteration, nevertheless this pair will be selected from the remaining dataset S_{U2}.

The Semi-AdaRank algorithm (Dammak et al., 2017a) is a two-staged algorithm that combines the label propagation process (Zhu & Ghahramani, 2002) and a regularized version of AdaRank (Xu & Li, 2007). It is based on the LP-AdaRank algorithm proposed by Miao and Tang (2013). The key concern of this work is articulated around the label-propagation phase, which consists in labeling just the S_{U1} subset, and then learns the model h on $S_L \cup S_{U1}$. As for the regularized version of AdaRank, Semi-AdaRank optimizes a novel performance measure and provides a flexible framework that shares the advantages of theoretical soundness, efficiency in training and high performance in testing. The label propagation process proposed in Dammak et al. (2017a) is a graph-based semi-supervised learning framework (Fujiwara & Irie, 2014). The considered idea is to propagate the relevant labels from labeled examples to other unlabeled ones, so that more training data will be available to learn the ranking function. In the graph nodes represent the training data and edges represent similarities between them. The similarities are given by a weight matrix W (with n rows and c columns). In the following, we present the ASSL2R algorithm.

Algorithm 3 Active-Semi-Supervised Learning to Rank algorithm: ASSL2R

```
Inputs
Small set of labeled data $S_L = \{(x_i, y_i); i \in \{1, \dots, m\}\}$
Large set of unlabeled data $S_U = \{(x_i^{'}); i \in \{1 + m, \dots, n + m\}\} = S_{U1} \cup S_{U2}$
Supervised learning to rank algorithm SRA: RankBoost \ AdaRank \ LambdaMART
Semi-supervised learning to rank algorithm SSRA: Semi-AdaRank
Labeling algorithm: Transductive knn
Number of $S_L$ partitions: $P$
Number of required examples to be labeled: $NbLab$
Begin
Learn $P$ committee models $\{h_p\}_{p \in \{1,\dots,P\}}$ with SRA
$nbIter \leftarrow 1$
While $nbIter <= NbLab$ do
Learn a ranking function $h$ with Semi-AdaRank on $S_L \cup S_{U1}$
Choose randomly a committee model $h_p$
Select the most informative query-document pair from $S_{U2}$ which maximizes the measure of disagreement $d_c(h, h_p)_{(x_i^{'},k) \in S_L \cup S_{U1}}$
Label the selected pair with the labeling algorithm
Withdraw this pair from $S_{U2}$ and add it to $S_L$
$nbIter \leftarrow nbIter + 1$
End while
End
Output: Model H2
```

For this algorithm, we consider as parameters, on the one hand the number of partitions P which corresponds to the number of representative committee models as well as the number of labeled examples S_L an those to be labeled *NbLab*. On the other hand, we focus particulary on the distribution of S_U between S_{U1} and S_{U2}.

Selecting the most useful features within the ranking functions and decreasing execution times are issues in learning to rank. We present in the next two sections an improvement of the two previous algorithms SAL2R and ASSL2R by using a multi-pairs in the selection phase, in order to accelerate this phase. These algorithms, denoted as "Semi-Active List Learning to Rank" (SALL2R) and " Active-Semi-Supervised List Learning to Rank" (ASSLL2R) (Dammak et al., 2017b), use a list of most informative query-document pairs from the unlabeled training data instead of selecting a single pair, at each iteration, and then look for their relevance by an automatic labeling method.

3.3 Semi-active list learning to rank algorithm: SALL2R

We describe in this section the SALL2R algorithm (Algorithm 4), that considers more than one document-query pair to be labeled at each round, compared to SAL2R algorithm.

As SAL2R algorithm, SALL2R uses two auxiliary algorithms:

- A listwise supervised learning to rank algorithm to learn for once P representative committee models $\{h_p\}_{p \in \{1,\dots,P\}}$ on S_L, then to learn iteratively a model h on S_L increased by the new selected pairs being labeled.
- A Transductive-knn labeling algorithm (Algorithm 1).

Firstly, SALL2R consists in learning P representative committee models on S_L denoted $\{h_p\}_{p\in\{1,\dots,P\}}$. Then, SALL2R will randomly choose at each round a model h_p among the learned P models.

Secondly, SALL2R uses the same supervised learning algorithm at each round to learn a ranking model h, characterized by a ranking function from S_L. Once the models are learned, SALL2R selects a list of most informative query-document pairs from the unlabeled data set S_U which maximize the measure of disagreement (equation 1.1) between the representative committee model h_p chosen randomly and the model h. The ranking function is updated iteratively since S_L is increased by the new selected labeled pairs.

As supervised algorithm, we propose to choose the well-known boosting AdaRank algorithm (Xu & Li, 2007). It is among the first listwise learning to rank algorithms of alternatives.

Algorithm 4 Semi-Active List Learning to Rank algorithm: SALL2R

```
Inputs
Small set of labeled data $S_L = \{(x_i, y_i); i \in \{1,\dots,m\}\}$
Large set of unlabeled data $S_U = \{(x_i^{'}); i \in \{1+m,\dots,n+m\}\}$
Supervised learning to rank algorithm SRA : AdaRank \LambdaMART
Labeling algorithm: Transductive knn
Number of $S_L$ partitions: $P$
Number of required examples to be labeled: $NbLab$
Number of most informative query-document pairs: $Nbp$
Begin
Learn $P$ committee models $\{h_p\}_{p\in\{1,\dots,P\}}$ with SRA
$nbIter \leftarrow 1$
While $nbIter <= NbLab$ do
Learn a ranking function $h$ with AdaRank on $S_L$
Choose randomly a committee model $h_p$
Select $Nbp$ of most informative query-documents pairs from $S_U$ which maximize the measure of disagreement $d_c(h, h_p)_{(x_i^{'},k)\in S_U}$
Label the $Nbp$ selected pairs with the labeling algorithm
Remove these pairs from $S_U$ and add them to $S_L$
$nbIter \leftarrow nbIter + 1$
End while
End
Output: Model H
```

We consider for this algorithm the same parameters for the experimental study as for the Algorithm 2. We focus particulary on the number of pairs to be labeled in one given round (*Nbp*).

3.4 Active-semi-supervised list learning to rank algorithm: ASSLL2R

The "Active-Semi-Supervised List Learning to Rank" (ASSLL2R) algorithm (Algorithm 5) constitutes an improvement of the ASSL2R algorithm by selecting, at each round, a list of most informative query-document pairs from the unlabeled training data S_{U2} which maximize the measure of disagreement $d_c(h, h_p)_{(x_i^{'},k)}$. ASSLL2R algorithm uses a partially labeled pairs in the initial training set as it considers a subset S_{U1}, extracted from the big set of unlabeled data S_U, with the small amounts of labeled training set S_L.

As ASSL2R algorithm (Algorithm 3), ASSLL2R uses three auxiliary algorithms:

- A supervised AdaRank algorithm for learning of P representative committee models $\{h_p\}_{p\in\{1,\dots,P\}}$ on S_L.
- A semi-supervised Semi-AdaRank algorithm to learn a model h on $S_L \cup S_{U1}$.

In the following, we give the ASSLL2R algorithm.

Algorithm 5 Active-Semi-Supervised List Learning to Rank algorithm: ASSLL2R

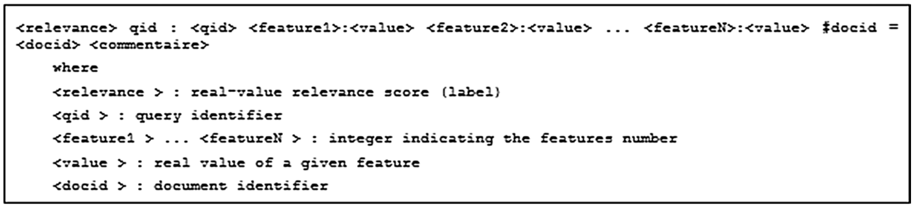
```
**Inputs**
Small set of labeled data $S_L = \{(x_i, y_i); i \in \{1,\dots,m\}\}$
Large set of unlabeled data $S_U = \{(x_i'); i \in \{1+m,\dots,n+m\}\} = S_{U1} \cup S_{U2}$
Supervised learning to rank algorithm SRA : AdaRank \ LambdaMART
Semi-supervised learning to rank algorithm SSRA: Semi-AdaRank
Labeling algorithm: Transductive knn
Number of $S_L$ partitions : $P$
Number of required examples to be labeled: $NbLab$
Number of most informative query-document pairs: $Nbp$
**Begin**
Learn $P$ committee models $\{h_p\}_{p\in\{1,\dots,P\}}$ with SRA
$nbIter \leftarrow 1$
**While** $nbIter <= NbLab$ **do**
Learn a ranking function $h$ with Semi-AdaRank on $S_L \cup S_{U1}$
Choose randomly a committee model $h_p$
Select $Nbp$ of most informative query-document pairs from $S_{U2}$ which maximizes the measure of disagreement $d_c(h, h_p)_{(x_i',k)\in S_{U2}}$
Label the $Nbp$ selected pairs with the labeling algorithm
Remove these pairs from $S_{U2}$ and add them to $S_L$
$nbIter \leftarrow nbIter + 1$
**End while**
**End**
**Output**: Model H
```

The two crucial parameters to consider for this algorithm are the respective sizes of S_{U1} and S_{U2} in addition to *Nbp*.

In the following section, we discussed the experimental part of this paper.

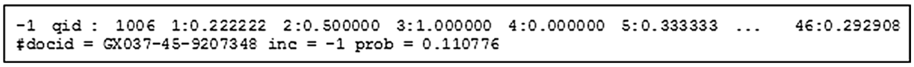
4 Experimental study

We chose DR (Liu, 2011) as an experimental framework to validate our proposed learning to rank algorithms and their improvements and to show the interest of semi-supervised and active learning to rank in improving results. We realized a number of empirical experiments, by varying several parameters, in order to compare the different results found and to evaluate the importance of the unlabeled data to learn the ranking functions in the proposed algorithms.



```
<relevance> qid : <qid> <feature1>:<value> <feature2>:<value> ... <featureN>:<value> #docid =
<docid> <commentaire>
    where
    <relevance > : real-value relevance score (label)
    <qid > : query identifier
    <feature1 > ... <featureN > : integer indicating the features number
    <value > : real value of a given feature
    <docid > : document identifier
```

Fig. 4 Query-document fields corresponding to a line of the MQ2007-semi and MQ2008-semi collections



```
-1 qid : 1006 1:0.222222 2:0.500000 3:1.000000 4:0.000000 5:0.333333 ...  46:0.292908
#docid = GX037-45-9207348 inc = -1 prob = 0.110776
```

Fig. 5 Example of a line extracted from the MQ2007-semi collection

4.1 Experiment setup

Experiments are conducted on the standard benchmark for learning to rank LETOR (LEarning TO Rank) (Liu et al., 2007), released by Microsoft Research Asia. LETOR is widely used in IR which was constructed based on multiple data corpora and query sets. We mainly exploited “MQ2007”, “MQ2008”, “MQ2007-semi” and “MQ2008-semi” (Million Query track) collections from LETOR 4.0.[1] These selected collections are conducted on the .GOV2 corpus using respectively the TREC 2007 and the TREC 2008, which are extracted from Web sites in the .gov domain.[2] Each subset of these collections is partitioned into five parts, denoted as S1, S2, S3, S4 and S5, in order to perform five-fold cross validation experiments (Liu et al., 2007). For each fold, there are three subsets for learning: training set, validation set and testing set. There are about 1700 queries in MQ2007 dataset with labeled documents, and about 70,000 query-document pairs, while MQ2008 has 800 queries and about 15,000 query-document pairs. MQ2007-semi and MQ2008-semi contain a small set of the labeled query-document pairs and a large amount of unlabeled query-document pairs (in training set but not in validation and testing set). There are about 2000 queries in these datasets. On average, each query is related with about 40 labeled documents and about 1000 unlabeled documents. Each query-document pair in the dataset is given a relevance level (−1, 0, 1 or 2) where −1 means “unlabeled” and a greater number means more relevance. Each query-document pair is represented by a feature vector that contains 46 features such as TF−IDF, BM25 and LMIR (Figs. 4, 5).

Normalised Discounted Cumulative Gain ($NDCG@n$), Precision at position n ($P@N$) and Mean Average Precision (MAP) (Järvelin & Kekäläinen, 2000) are used as a standard ranking performance measures to evaluate the retrieval effectiveness of our experiments on LETOR.

Mean Average Precision (MAP) is a measure for assessing the quality of ranking a list of results in the case of binary judgments relevance (relevant documents vs irrelevant documents). It is defined by using the Precision at position n ($P@n$)). $P@n$ for a query q

[1] http://www.bigdatalab.ac.cn/benchmark/bm/Domain?domain=Information%20Retrieval.

[2] http://research.microsoft.com/en-us/um/beijing/projects/letor/.

corresponds to the ratio of relevant documents among the top n documents returned in the ranking results for this query.

$$P@n = \frac{\#relevant\ documents\ in\ top\ n\ results}{n} \tag{3}$$

$$MAP = \frac{1}{Q}\sum_{q=1}^{Q}\frac{\sum_{n=1}^{N}(P@n * rel(n))}{\#total\ relevants\ documents\ of\ this\ query} \tag{4}$$

Q is the number of queries in the considered collection.

The Normalized Discounted Cumulative Gain (*NDCG*) at position *n* (*NDCG@n*) is calculated by the following equation. *r*(*j*) is the degree of relevance of the document at position *j* in the ranking list.

$$NDCG@n = \frac{1}{Q}\sum_{q=1}^{Q}\frac{1}{Z_n}\sum_{j=1}^{n}\frac{2^{r(j)}-1}{log(1+j)} \tag{5}$$

In the next section, we present a series of experiments in order to compare the proposed learning to rank algorithms with reference algorithms on which they are based.

4.2 Experimental results

The first part of our experimental study (Sect. 4.2.1) concerns the two algorithms SAL2R and ASSL2R. Our principal objective is to evaluate them according to some supervised (RankBoost, AdaRank and LambdaMART), semi-supervised (Semi-RankBoost and Semi-AdaRank) and active learning to rank algorithms (Active-RankBoost and Active-AdaRank Dammak et al., 2015). Furthermore, we plan to compare them mutually by considering the different supervised learning to rank algorithms SRA as auxiliaries which leads to the following variants:

- SAL2R-RankBoost\ -AdaRank\ -LambdaMART
- ASSL2R-RankBoost\ -AdaRank\ -LambdaMART

The evaluation was carried out on the MQ2007, MQ2008 (Tables 1, 2, 3, 4), MQ2007-semi and MQ2008-semi (Tables 5, 6, 7) based on *NDCG@n*, *P@n* and *MAP* measures. Moreover, we want to study the impact of varying the number of labeled data considered for training $|S_L|$ (Figs. 4, 5), the number of S_L partitions: *P* (the number of partitions of committee model) (Fig. 7) and the number of required examples to be labeled *NbLab* (Fig. 6) on the effectiveness of the ranking function trained. Finally, we focus on the distribution of the unlabeled examples S_U between S_{U1} and S_{U2} (Fig. 8).

The second part of our experimental study (Sect. 4.2.2) treats the evaluation of the two algorithms SALL2R and ASSLL2R by according a particular interest to the variation of *Nbp* (the number of most informative query-document pairs to be labeled for a given iteration). The results obtained for the MQ2007 and MQ2008 collections in terms of *NDCG@n*

Table 1 NDCG@n measures on the MQ2007 collection

Algorithms	NDCG@1	NDCG@3	NDCG@5	NDCG@7	NDCG@10
RankBoost	0.4126	0.4147	0.4185	0.4191	0.4191
SAL2R-RankBoost	**0.4436**	**0.4512**	**0.4614**	**0.4713**	**0.482**
ASSL2R-RankBoost	**0.4478**	**0.4573**	**0.4626**	**0.4784**	**0.4943**
Improv-SAL2R-RankBoost	7.51%	7.81%	10.09%	10.17%	8.51%
Improv-ASSL2R-RankBoost	8.53%	9.27%	10.38%	11.83%	11.28%
AdaRank	0.3913	0.3963	0.4021	0.4091	0.4125
SAL2R-AdaRank	**0.4384**	**0.4495**	**0.4663**	**0.4714**	**0.4845**
ASSL2R-AdaRank	**0.4418**	**0.4525**	**0.4681**	**0.4772**	**0.4917**
Improv-SAL2R-AdaRank	6.25%	7.41%	11.26%	10.19%	9.07%
Improv-ASSL2R-AdaRank	7.08%	8.12%	11.69%	11.55%	10.69%
LambdaMART	0.4122	0.4085	0.4115	0.4141	0.4185
SAL2R-LambdaMART	**0.4437**	**0.4452**	**0.4487**	**0.4612**	**0.4812**
ASSL2R-LambdaMART	**0.4521**	**0.462**	**0.4684**	**0.4784**	**0.4891**
Improv-SAL2R-LambdaMART	7.54%	6.38%	7.06%	7.81%	8.33%
Improv-ASSL2R-LambdaMART	9.57%	10.39%	11.76%	11.83%	10.11%

Bold values indicate statistically significant

Table 2 P@n and MAP measures on the MQ2007 collection

Algorithms	P@1	P@3	P@5	P@7	P@10	MAP
RankBoost	0.4799	0.444	0.4113	0.395	0.38	0.4624
SAL2R-RankBoost	**0.5297**	**0.4886**	**0.4543**	**0.4352**	**0.4234**	**0.5176**
ASSL2R-RankBoost	**0.5327**	**0.4912**	**0.4623**	**0.4418**	**0.4312**	**0.5263**
Improv-SAL2R-RankBoost	10.10%	10.05%	10.45%	10.18%	11.42%	11.12%
Improv-ASSL2R-RankBoost	10.73%	11.76%	10.26%	10.01%	11.88%	12.99%
AdaRank	0.448	0.4253	0.4073	0.3928	0.3741	0.4602
SAL2R-AdaRank	**0.5196**	**0.4815**	**0.4576**	**0.4395**	**0.4315**	**0.5142**
ASSL2R-AdaRank	**0.5241**	**0.4871**	**0.4628**	**0.451**	**0.4276**	**0.521**
Improv-SAL2R-AdaRank	9.48%	11.59%	10.67%	10.04%	12.58%	10.39%
Improv-ASSL2R-AdaRank	10.43%	10.83%	10.37%	12.30%	10.95%	11.85%
LambdaMART	0.4811	0.4395	0.4193	0.4016	0.3854	0.4658
SAL2R-LambdaMART	**0.5022**	**0.4815**	**0.4451**	**0.4287**	**0.3954**	**0.4975**
ASSL2R-LambdaMART	**0.5274**	**0.4923**	**0.4542**	**0.4391**	**0.4133**	**0.5217**
Improv-SAL2R-LambdaMART	4.39%	9.56%	6.15%	6.75%	2.59%	6.81%
Improv-ASSL2R-LambdaMART	9.90%	10.88%	10.43%	11.16%	8.76%	12.82%

Bold values indicate statistically significant

will be spread out in Tables 10 and 11 and in term of *MAP* in Table 12. Those obtained for MQ2007-semi and MQ2008-semi in term of *MAP* will be presented in Table 13.

There were two learning parameters to be tuned. We tested several values of the k nearest neighbors' parameter of the labeling algorithm: $k = 5$, 7, 10 and 12. We noted

Table 3 NDCG@n measures on the MQ2008 collection

Algorithms	NDCG@1	NDCG@3	NDCG@5	NDCG@7	NDCG@10
RankBoost	0.3665	0.4281	0.468	0.4919	0.2289
SAL2R-RankBoost	**0.4185**	**0.4682**	**0.5174**	**0.4412**	**0.2534**
ASSL2R-RankBoost	**0.4218**	**0.4742**	**0.5214**	**0.5482**	**0.2575**
Improv-SAL2R-RankBoost	11.87%	9.19%	10.34%	10.51%	10.70%
Improv-ASSL2R-RankBoost	12.75%	10.77%	11.41%	11.45%	12.49%
AdaRank	0.372	0.4289	0.4759	0.4947	0.2276
SAL2R-AdaRank	**0.4195**	**0.4697**	**0.5162**	**0.5457**	**0.2506**
ASSL2R-AdaRank	**0.4221**	**0.4735**	**0.5271**	**0.5483**	**0.2534**
Improv-SAL2R-AdaRank	12.14%	9.54%	10.09%	10.31%	10.11%
Improv-ASSL2R-AdaRank	13.47%	10.40%	10.76%	10.83%	11.34%
LambdaMART	0.3741	0.4288	0.4689	0.4903	0.2275
SAL2R-LambdaMART	**0.4067**	**0.4548**	**0.4972**	**0.5231**	**0.2413**
ASSL2R-LambdaMART	**0.4151**	**0.4684**	**0.5175**	**0.5451**	**0.2567**
Improv-SAL2R-LambdaMART	8.71%	6.06%	6.04%	6.69%	6.07%
Improv-ASSL2R-LambdaMART	10.96%	9.24%	10.36%	11.18%	12.84%

Bold values indicate statistically significant

Table 4 P@n and MAP measures on the MQ2008 collection

Algorithms	P@1	P@3	P@5	P@7	P@10	MAP
RankBoost	0.4413	0.3962	0.348	0.3063	0.2508	0.4758
SAL2R-RankBoost	**0.4972**	**0.4452**	**0.3897**	**0.3403**	**0.2827**	**0.5285**
ASSL2R-RankBoost	**0.4987**	**0.4511**	**0.4015**	**0.3424**	**0.2954**	**0.5332**
Improv-SAL2R-RankBoost	10.76%	12.37%	11.98%	11.10%	11.10%	11.08%
Improv-ASSL2R-Rankboost	11.09%	13.86%	15.37%	11.79%	17.78%	12.06%
AdaRank	0.4374	0.3848	0.3431	0.2992	0.2459	0.4783
SAL2R-AdaRank	**0.4982**	**0.4372**	**0.3854**	**0.3422**	**0.2828**	**0.5296**
ASSL2R-AdaRank	**0.4976**	**0.4432**	**0.3913**	**0.3488**	**0.2952**	**0.5354**
Improv-SAL2R-AdaRank	10.98%	10.35%	10.75%	11.72%	12.76%	11.31%
Improv-ASSL2R-AdaRank	10.85%	11.86%	12.44%	13.88%	17.70%	12.53%
LambdaMART	0.4489	0.3843	0.3405	0.2971	0.244	0.4753
SAL2R-LambdaMART	**0.4582**	**0.4215**	**0.3734**	**0.3256**	**0.2764**	**0.5025**
ASSL2R-LambdaMART	**0.4854**	**0.4432**	**0.3855**	**0.3467**	**0.2894**	**0.524**
Improv-SAL2R-LambdaMART	2.07%	9.68%	9.66%	9.59%	13.28%	5.72%
Improv-ASSL2R-LambdaMART	8.13%	15.33%	13.22%	16.69%	18.61%	10.25%

Bold values indicate statistically significant

that this variation had no big influence on the results found. This parameter was fixed to 10 ($k = 10$) in our series of experiments.

Table 5 NDCG@n measures on the MQ2007-semi collection

Algorithms	NDCG@1	NDCG@3	NDCG@5	NDCG@7	NDCG@10
Active-RankBoost	0.3366	0.4073	0.4351	0.4545	0.4783
SAL2R-RankBoost	**0.3769**	**0.4097**	0.4217	**0.4625**	**0.4749**
ASSL2R-RankBoost	**0.3517**	**0.4093**	**0.4471**	**0.4628**	**0.4857**
Active-AdaRank	0.3264	0.3684	0.4318	0.4477	0.4658
SAL2R-AdaRank	**0.3562**	**0.4557**	**0.4925**	**0.5081**	**0.4862**
ASSL2R-AdaRank	**0.4433**	**0.4394**	**0.4727**	**0.5048**	**0.5178**

Bold values indicate statistically significant

Table 6 NDCG@n measures on the MQ2008-semi collection

Algorithms	NDCG@1	NDCG@3	NDCG@5	NDCG@7	NDCG@10
Semi-RankBoost	0.4630	0.4550	0.4490	0.4120	0.2300
Active-RankBoost	0.4707	0.4543	0.4525	0.4126	0.2419
SAL2R-RankBoost	**0.4723**	**0.4565**	**0.4528**	**0.4424**	**0.2527**
ASSL2R-RankBoost	**0.4732**	**0.4966**	**0.5072**	**0.4894**	**0.2733**
LP-AdaRank	0.4840	0.4620	0.4510	0.4050	0.2470
Active-AdaRank	0.4817	0.4568	0.4536	0.4345	0.2503
SAL2R-AdaRank	**0.4861**	**0.4623**	**0.4579**	**0.4417**	**0.2596**
ASSL2R-AdaRank	**0.4924**	**0.4763**	**0.4612**	**0.4462**	**0.2608**

Bold values indicate statistically significant

Table 7 MAP measures on MQ2007-semi and MQ2008-semi

Algorithms	MAP on MQ2007-semi	MAP on MQ2008-semi
Active-RankBoost	0.4647	0.4729
SAL2R-RankBoost	0.455	**0.6003**
ASSL2R-RankBoost	**0.4807**	0.4402
Active-AdaRank	0.4188	0.4716
SAL2R-AdaRank	**0.4328**	**0.5427**
ASSL2R-AdaRank	**0.4521**	**0.6227**

Bold values indicate statistically significant

4.2.1 Evaluation of SAL2R and ASSL2R

Tables 1 and 2 show improvement in the results obtained by SAL2R-RankBoost\ -AdaRank\ -LambdaMART and ASSL2R-RankBoost\ -AdaRank\ -LambdaMART for the MQ2007 dataset (Improvement $\simeq$ 10.5%) as compared to RankBoost\ AdaRank\ LambdaMART. The overall results are very similar to those obtained with the MQ2008 dataset in Tables 3 and 4 (Improvement $\simeq$ 11.5%). From these results, we can see clear performance improvements brought by the algorithms on all the metrics. This series of experiments validate the idea of combining the active (active-semi-supervised) approach with a step of automatic labeling.

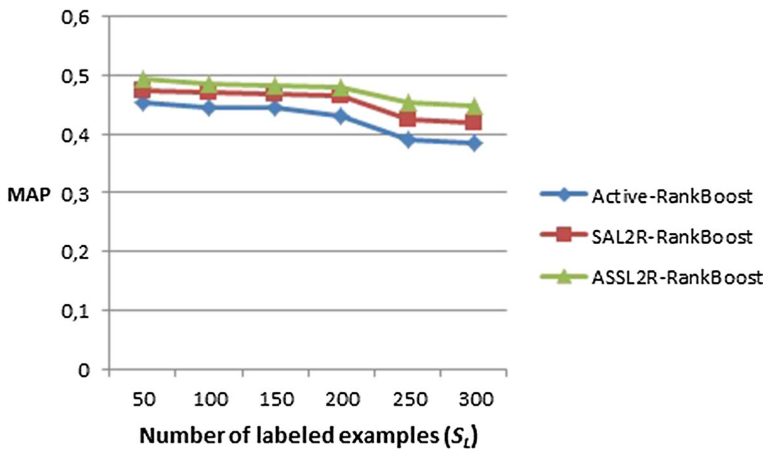


Fig. 6 Variation of MAP as a function of the number of labeled examples (Active-RankBoost, SAL2R-RankBoost, ASSL2R-RankBoost) on MQ2008-semi

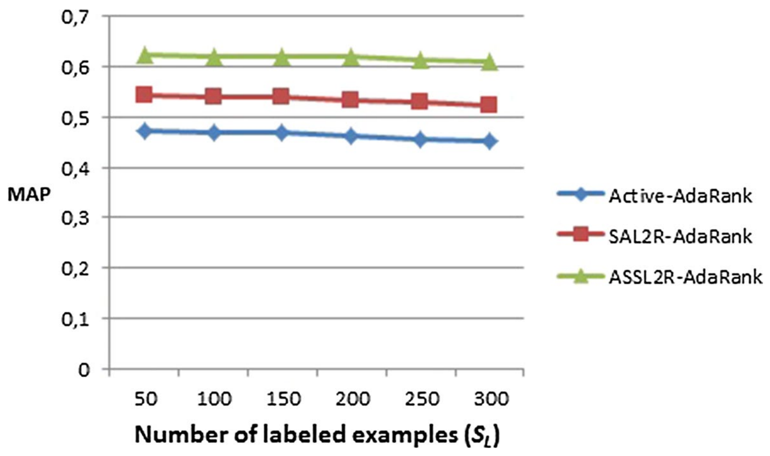


Fig. 7 Variation of MAP as a function of the number of labeled examples (Active-AdaRank, SAL2R-AdaRank, ASSL2R-AdaRank) on MQ2008-semi

From Tables 5 and 7, we found that the performance in terms of *NDCG@n* and *MAP* of SAL2R-RankBoost and ASSL2R-RankBoost algorithms in the MQ2007-semi collection was better than those obtained by Active-RankBoost except *NDCG@*5 of SAL2R-RankBoost. Moreover, we notice that the results of SAL2R-AdaRank and ASSL2R-AdaRank were better than those obtained by Active-AdaRank in terms of *NDCG@n* and *MAP* measures (Tables 6 and 7) (Improvement $\simeq$ 14.5%). Nevertheless, *MAP* of ASSL2R-RankBoost on the MQ2008-semi collection were inferior to those obtained by SAL2R-RankBoost and Active-RankBoost. Contrary, we observe that the *MAP* measure of ASSL2R-AdaRank is very important (0.6227) according to the Table 7. Table 6 shows, on the MQ2008-semi collection, that ASSL2R-RankBoost algorithm has a better performance than Active-RankBoost algorithms (Improvement $\simeq$ 13.5%). It displays also that ASSL2R-AdaRank is more

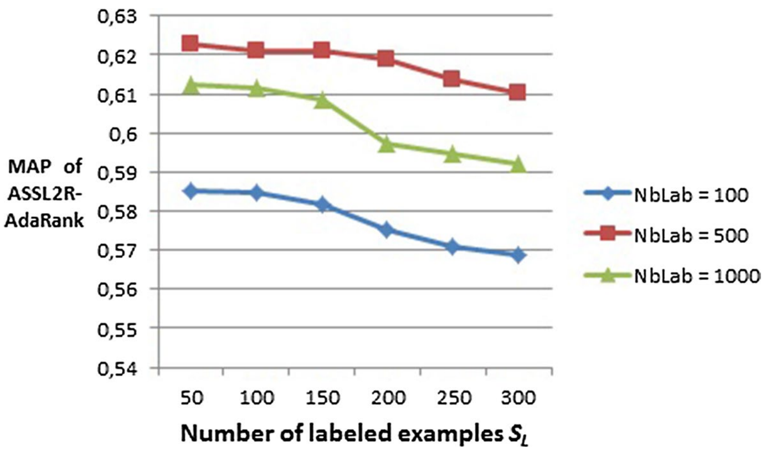


Fig. 8 Variation of MAP as a function of the number of labeled examples S_L ($NbLab$ = 100, 500 and 1000) on ASSL2R-adaRank algorithm

Table 8 AP measures of three queries extracted from the MQ2008-semi collection

Algorithms	AP Q1	AP Q2	AP Q3
SAL2R-RankBoost	**0.5856**	**0.5522**	**0.51375**
ASSL2R-RankBoost	**0.6124**	**0.5768**	**0.5244**
SAL2R-AdaRank	**0.5341**	**0.5415**	**0.5184**
ASSL2R-AdaRank	**0.5572**	**0.5774**	**0.5216**
SAL2R-LambdaMART	**0.5486**	**0.50172**	**0.4963**
ASSL2R-LambdaMART	**0.5832**	**0.5214**	**0.517**

Bold values indicate statistically significant

effective than Active-AdaRank in terms of all measures excluding NDCG@3 (Improvement $\simeq$ 12%).

From these experiments, we also compared SAL2R and ASSL2R algorithms by examining respectively the results obtained by SAL2R-RankBoost\ -AdaRank\ -LambdaMART and ASSL2R-RankBoost\ -AdaRank\ -LambdaMART. These results show improvements indicating that ASSL2R algorithm is more effective than SAL2R (Improvement $\simeq$ 2.5%). This observation supports the idea of using partially labeled pairs in the training set, more precisely the addition of a subset S_{U1}, extracted from the large set of the unlabeled data S_U, to the small amounts of labeled training set S_L. And obviously the use of the semi-supervised learning algorithm to learn a model h on $S_L \cup S_{U1}$.

To deepen our experimental study we have looked for some examples of queries for which we obtain Ap measures better than the MAP obtained on the set of queries (Table 8).

We conclude from these tables that in general the use of unlabeled data constantly improves the effectiveness in terms of $NDCG@n$, $P@n$ and MAP of the proposed algorithms. These results show improvements indicating that these algorithms are more effective than the supervised \ semi-supervised learning to rank algorithms and the active learning to rank algorithms : Active-RankBoost and Active-AdaRank.

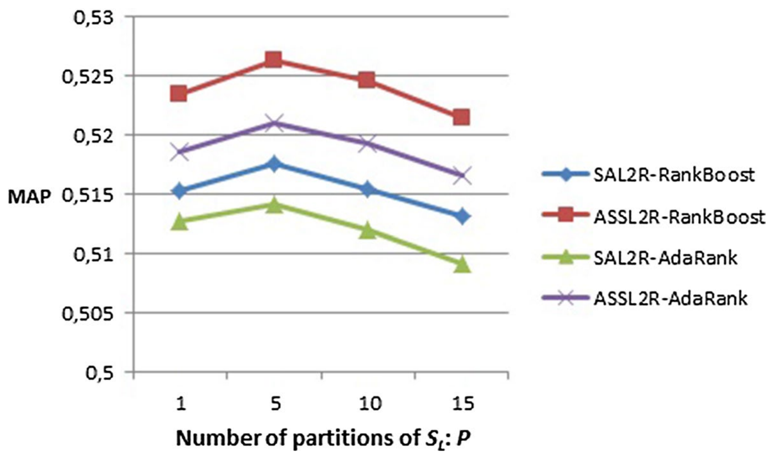


Fig. 9 Variation of MAP as a function of the number of S_L partitions P on MQ2007

The effect of varying the number of labeled examples S_L We plan here to study the effect of increasing the number of labeled examples on the MAP measure first by running SAL2R-RankBoost, ASSL2R-RankBoost, Active-RankBoost algorithms, then by running the Active-AdaRank, SAL2R-AdaRank and ASSL2R-AdaRank algorithms. We looked for *MAP* variation on the MQ2008-semi collection based on the obtained results. Figures 4 and 5 indicate that the curves of these algorithms have a decreasing appearance with the addition of the labeled examples in the learning phase.

The effect of increasing the number of labeled examples S_L by varying NbLab In this subsection, we tested the effect of increasing the number of examples to be labeled *NbLab* and that of the labeled ones S_L on ASSL2R-AdaRank algorithm. We looked for the variation of the *MAP* on the MQ2008-semi collection by varying the values of *NbLab* and S_L. Figure 6 indicates that the curves of the three algorithms have a decreasing appearance with increasing the labeled examples. This observation is valid for the three curves corresponding to the different values fixed for *NbLab* (*NbLab* = 100, 500 and 1000). Figure 6 illustrates performing ASSL2R-AdaRank when *NbLab* = 500 and $|S_L| = 50$ (10% of *Nblab*) which represents the following percentages: 90.91% for S_U and 9.09% for *NbLab* relative to the entire learning set. It is clear from Figs. 4, 5 and 6 that the values of *MAP* measures decrease the number of labeled pairs increase. This justifies the idea of choosing to learn with small set of labeled examples and a large set of unlabeled ones.

The effect of varying the committee size P We include experiments on MQ2007 collection that test the effect of varying the number of S_L partitions: P on the *MAP* measure. We looked for the results obtained on SAL2R-RankBoost\ -AdaRank and ASSL2R-RankBoost\ -AdaRank algorithmes. Figure 7 illustrates that the best results are obtained according to the chosen algorithms when $P = 5$.

The effect of varying the distribution of S_U between S_{U1} and S_{U2} We focus particulary on the distribution of S_U between S_{U1} and S_{U2}. Figure 8 shows that the curves of these algorithms have a decreasing appearance when the values of S_{U1} exceed the values of S_{U2}. Results in this figure achieve the best performance when $S_{U1} << S_{U2}$. This reinforces the idea of learning with a small set of labeled examples and a large set of unlabeled ones is more effective.

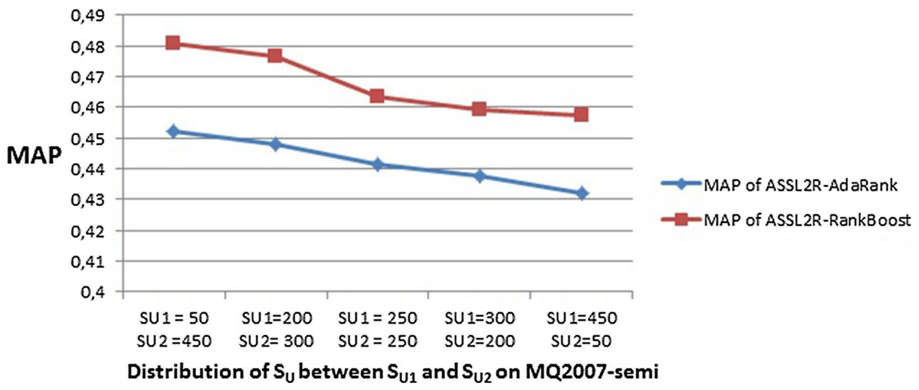


Fig. 10 Variation of MAP as a function of the distribution of S_U between S_{U1} and S_{U2} on MQ2007-semi

Table 9 NDCG@n measures on the MQ2007 collection

Algorithms	NDCG@1	NDCG@3	NDCG@5	NDCG@7	NDCG@10
AdaRank	0.3913	0.3963	0.4021	0.4091	0.4125
Active-AdaRank	0.4052	0.4136	0.4372	0.4477	0.4558
SAL2R-AdaRank	0.4384	0.4495	0.4663	0.4714	0.4845
ASSL2R-AdaRank	0.4418	0.4525	0.4681	0.4772	0.4917
SALL2R-3	**0.4452**	**0.4562**	**0.4725**	**0.4825**	**0.5102**
SALL2R-7	**0.4463**	**0.4584**	**0.4751**	**0.4863**	**0.5128**
SALL2R-10	**0.4472**	**0.4586**	**0.4764**	**0.4884**	**0.5137**
SALL2R-15	**0.4472**	**0.4587**	**0.4766**	**0.4885**	**0.5137**
ASSLL2R-3	**0.4488**	**0.4583**	**0.4859**	**0.5262**	**0.5338**
ASSLL2R-7	**0.4547**	**0.4733**	**0.4947**	**0.5402**	**0.5566**
ASSLL2R-10	**0.4553**	**0.4734**	**0.4959**	**0.5410**	**0.5568**
ASSLL2R-15	**0.4553**	**0.4734**	**0.4961**	**0.5411**	**0.5568**

Bold values indicate statistically significant

Table 10 NDCG@n measures on the MQ2008 collection

Algorithms	NDCG@1	NDCG@3	NDCG@5	NDCG@7	NDCG@10
AdaRank	0.372	0.4289	0.4759	0.4947	0.2276
SAL2R-AdaRank	0.4195	0.4697	0.5162	0.5457	0.2506
ASSL2R-AdaRank	0.4221	0.4735	0.5271	0.5483	0.2534
SALL2R-3	**0.4265**	0.4722	**0.5245**	**0.5542**	**0.5102**
SALL2R-7	**0.4313**	**0.4837**	**0.5322**	**0.5593**	**0.2573**
SALL2R-10	**0.4324**	**0.4856**	**0.5341**	**0.5601**	**0.2607**
SALL2R-15	**0.4331**	**0.4862**	**0.5347**	**0.5614**	**0.2612**
ASSLL2R-3	**0.4386**	**0.4886**	**0.5302**	0.5462	**0.2575**
ASSLL2R-7	**0.4467**	**0.4948**	**0.5481**	**0.5513**	**0.2616**
ASSLL2R-10	**0.4471**	**0.4965**	**0.5496**	**0.5542**	**0.2628**
ASSLL2R-15	**0.4478**	**0.4967**	**0.5504**	**0.5548**	**0.2631**

Bold values indicate statistically significant

Table 11 MAP measures on MQ2007 and MQ2008

Algorithms	MAP on MQ2007	MAP on MQ2008
AdaRank	0.4602	0.4783
Active-AdaRank	0.4628	0.4806
SAL2R-AdaRank	0.5142	0.5296
ASSL2R-AdaRank	0.521	0.5354
SALL2R-3	**0.5147**	**0.5851**
SALL2R-7	**0.5158**	**0.6162**
SALL2R-10	**0.5168**	**0.6223**
SALL2R-15	**0.5174**	**0.6228**
ASSLL2R-3	**0.5217**	**0.6368**
ASSLL2R-7	**0.5236**	**0.6412**
ASSLL2R-10	**0.5265**	**0.6419**
ASSLL2R-15	**0.5267**	**0.6424**

Bold values indicate statistically significant

Table 12 MAP measures on MQ2007-semi and MQ2008-semi

Algorithms	MAP on MQ2007-semi	MAP on MQ2008-semi
Active-AdaRank	0.4188	0.4716
SAL2R-AdaRank	0.4328	0.5427
ASSL2R-AdaRank	0.4521	0.6227
SALL2R-3	**0.4347**	**0.5651**
SALL2R-7	**0.4745**	**0.5821**
SALL2R-10	**0.4756**	**0.5823**
SALL2R-15	**0.4756**	**0.5823**
ASSLL2R-3	**0.4605**	**0.6368**
ASSLL2R-7	**0.4818**	**0.6412**
ASSLL2R-10	**0.4831**	**0.6419**
ASSLL2R-15	**0.4834**	**0.6419**

Bold values indicate statistically significant

In the next section, we present a series of experiments with the aim of the evaluation of SALL2R and ASSLL2R algorithms (Figs. 9, 10).

4.2.2 Evaluation of SALL2R and ASSLL2R

Our goal through these series of experiments is to evaluate the behavior of the algorithms (SALL2R, ASSLL2R) according to the number of pairs (*Nbp*) to be labeled at each round in the training sets. We choose to use *Nbp* = 3, 7, 10 and 15. The evaluation results of SALL2R-n and ASSLL2R-n (n = 3, 7, 10 and 15) on testing set are summarized in Tables 9, 10, 11 and 12. From these tables, we note that the results progress in terms of NDCG@n and MAP measures on MQ2008 collection. We found also that

Table 13 NDCG@10 measures on MQ2007, MQ2008, MQ2007-semi and MQ2008-semi collections

	RankBoost	SAL2R-RankBoost	Difference
MQ2007	0.4442	0.482	0.0378
MQ2008	0.2289	0.2534	0.0245
MQ2007-semi	0.446	0.4749	0.0289
MQ2008-semi	0.225	0.2527	0.0277
p-value	0.0009381		

Table 14 NDCG@10 measures on MQ2007, MQ2008, MQ2007-semi and MQ2008-semi collections

	RankBoost	ASSL2R-Rank-Boost	Difference
MQ2007	0.4442	0.4943	0.0501
MQ2008	0.2289	0.2575	0.0286
MQ2007-semi	0.446	0.4857	0.0397
MQ2008-semi	0.225	0.2733	0.0483
p-value	0.001721		

the performance of SALL2R-n (n = 3, 7, 10 and 15) are better than those obtained by Active-AdaRank and by SAL2R-AdaRank.

According to the Table 11, we note that the results of ASSLL2R-n ($n = 3$, 7, 10 and 15) on MQ2008 collection are more effective than those of Active-AdaRank and ASSL2R-AdaRank and we notice improvement in results. On another side, from these experiments, we note that the SALL2R-n1 results are better than those of SALL2R-n2 if n1 > n2. We can deduce that when the number of pairs increase the results are better. However, we have noted that the variation between the values of ASSLL2R-3, ASSLL2R-7 (respectively SALL2R-3 and SALL2R-7) is of 0.1% on the other hand the variation between ASSLL2R-10 and ASSLL2R-15 (respectively SALL2R-10 and SALL2R-15) is of 0.01% that's why we didn't test any other value for *Nbp*. This validates the advantage of learning to rank algorithms based on the listwise approach with more than one query-document pair which appears as a promising direction to improve the performance of the SAL2R and ASSL2R algorithms.

The experiments we carried out demonstrate improvements in MQ2007\ MQ2007-semi and MQ2008\ MQ2008-semi collections and highlight the importance of combining the active and the semi-supervised types of learning.

In general, both of the SAL2R and ASSL2R algorithms have shown their performance relatively to some other supervised, semi-supervised and active learning to rank algorithms. These results prove the utility of introducing unlabeled data in the learning to rank process with the combination of active learning and semi-supervised learning methods. Also, both of the SALL2R and ASSLL2R algorithms have shown their performance, by growing number of pairs chosen. This justifies the advantages of introducing the listwise approach with more than one query-document pair to be labeled which appears as a promising direction to improve the performance of SALL2R and ASSLL2R. Moreover, introducing an automatic labeling method in the active learning to rank can potentially improve the evaluation results and might be necessary in some cases, particulary when the training set contains a small set of labeled examples.

Table 15 NDCG@10 measures on MQ2007, MQ2008, MQ2007-semi and MQ2008-semi collections

	AdaRank	SAL2R-AdaRank	Difference
MQ2007	0.4364	0.4845	0.0481
MQ2008	0.2276	0.2506	0.023
MQ2007-semi	0.437	0.4862	0.0492
MQ2008-semi	0.231	0.2596	0.0206
p-value	0.005746		

Table 16 NDCG@10 measures on MQ2007, MQ2008, MQ2007-semi and MQ2008-semi collections

	AdaRank	ASSL2R-AdaRank	Difference
MQ2007	0.4364	0.4917	0.0553
MQ2008	0.2276	0.2534	0.0258
MQ2007-semi	0.437	0.5178	0.0808
MQ2008-semi	0.231	0.2608	0.0298
p-value	0.01648		

To verify the improvement hypothesis of the algorithms that we proposed, we selected to use the paired parametric T-Test (student's t test) (Demšar, 2006) as well as its non-parametric alternative Wilcoxon signed-ranks Test (Wilcoxon, 1992). The first test checks whether the average difference in the performances of given two algorithms to be compared over the data sets is significantly different from zero. The second test ranks the difference in performances ignoring the signs and compares the ranks for positive and negative differences. The purpose is to try to reject the null hypothesis that both algorithms perform equally well.

The Tables 13, 14, 15 and 16 show the evaluation results obtained by ASSL2R and SAL2R algorithms to be compared with RankBoost and AdaRank algorithms. In these tables, we propose to add the *p*-value which allows to statistically compare the SAL2R and ASSL2R algorithms proposed (with as auxiliaries for the SRA: RankBoost and AdaRank) with RankBoost and AdaRank respectively.

If we consider the SAL2R algorithm, we observe that for the RankBoost and AdaRank variants we obtain the respective *p*-values 0.000981 and 0.005746 (< 0.05). The collections considered are MQ2007\ MQ2007-semi and MQ2008\ MQ2008-semi; and the effectiveness metric is NDCG@10. Moreover, for the ASSL2R algorithm we obtain the respective *p*-values 0.00172 and 0.01648 for the same variants (< 0.05).

For a confidence level of $\alpha = 0.05$ and N = 4 datasets, we need to form a null hypothesis and determine whether we can reject the null hypothesis. When the significance level (*p*-value) is low, we can feel comfortable in rejecting the null hypothesis. . This hypothesis is validated with the t-test but not with the Wilcoxon test. This can be explained by the fact that the Wilcoxon test is more sensible (Demšar, 2006). According to the values in the Tables 13, 14, 15 and 16 ASSL2R and SAL2R are significantly better than RankBoost and AdaRank with *p*-value < 0.05, we therefore reject the null hypothesis.

5 Conclusion

To take benefit of the active and semi-supervised learning methods, we have considered an enhanced experimental study of two inductive learning to rank algorithms SAL2R and ASSL2R, for DR which combine the two methods of learning to rank and compensate the small number of labeled data by the information in a large dataset of unlabeled data. Thus, we focus on the number of unlabeled examples (*NbLab*) and the distribution of S_U between S_{U1} and S_{U2}. In fact, these algorithms are based on the principle of active learning to rank of alternatives and use supervised and semi-supervised learning as auxiliaries to learn ranking functions in order to select only the most informative query-document unlabeled pair at each round and specify, afterwards, if the document is relevant or not in relation with this query. For this purpose, we propose different variants of the algorithms according to the SRA applied in order to compare them. We focused also for the proposed algorithms on the QBC selection strategy and the use of an automatic labeling algorithm for the labeling process instead of resorting to an expert. Consequently the committee number *P* can be considered as a parameter which could influence the learning performance to which we must attribute an interest. At last, we deal with SALL2R and ASSLL2R algorithms which used a list of most informative unlabeled query-document pairs (*Nbp*) instead of opting for the most informative one, which showed its effectiveness through a set of experiments. To evaluate the impact of varying all these parameters, we use collections from the standard benchmark LETOR4.0 and *P@n*, *NDCG@n* and *MAP* metrics. The obtained results corresponding to SAL2R-RankBoost\ -AdaRank\ -LambdaMART, ASSL2R-RankBoost\ -AdaRank\ -LambdaMART algorithms were compared to those corresponding to supervised, semi-supervised and active learning to rank algorithms. Our experiments demonstrate significant improvements in MQ2007, MQ2008, MQ2007-semi and MQ2008-semi collections and highlight the importance of combining the active and the semi-supervised types of learning. This justifies the use of unlabeled data for ranking. Moreover, the automatic labeling in the active learning to rank can potentially improve the evaluation results and might be necessary in some cases, particulary when the training set contains very little labeled examples. These results also demonstrate the influence of the pairwise and the listwise approaches on these algorithms. The main conclusion that emerges from our study of the combination of these parameters is that a semi-supervised listwise algorithm as auxiliary gives the best performances compared to the pairwise algorithms if we consider in addition a greater proportion of S_U (*NbLab*) against S_L (10% of S_L against 90% of S_U). Further more, by setting number of partitions *P* to 5 and the number of pairs (*Nbp*) to 10 or to 15 we obtain the best performances.

A further perspective consists in implementing another selection strategy for active learning through integrating either the transductive-knn algorithm or the Semi-AdaRank algorithm. Coupling weak learning with deep learning can constitute an interesting research avenue for processing unlabeled data for learning.

References

Ailon, N. (2012). An active learning algorithm for ranking from pairwise preferences with an almost optimal query complexity. *Journal of Machine Learning Research,13*(Jan), 137–164.

Amini, M. R., Truong, T. V., & Goutte, C. (2008). A boosting algorithm for learning bipartite ranking functions with partially labeled data, In *Proceedings of the 31st annual international ACM SIGIR conference on research and development in information retrieval* (pp. 99–106).

Aslam, J. A., Pavlu, V., & Yilmaz, E. (2006). A statistical method for system evaluation using incomplete judgments. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 541–548).

Brinker, K. (2004). Active learning of label ranking functions. In *Proceedings of the twenty-first international conference on Machine learning* (p. 17).

Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., & Hullender, G. N. (2005). Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on machine learning (ICML-05)* (pp. 89–96).

Cao, Z., Qin, T., Liu, T.-Y., Tsai, M.-F., & Li, H. (2007). Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning* (pp. 129–136).

Carterette, B., Allan, J., & Sitaraman, R. (2006). Minimal test collections for retrieval evaluation. In *Proceedings of the 29th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 268–275).

Chapelle, O., & Chang, Y. (2011). Yahoo! learning to rank challenge overview. In *Proceedings of the learning to rank challenge* (pp. 1–24).

Chapelle, O., Schölkopf, B., & Zien, A. (2006). *Semi-supervised learning, ser. Adaptive computation and machine learning*. The MIT Press.

Dammak, F., Gabsi, I., Kammoun, H., & Hamadou, A. B. (2015). Active learning to rank for documents retrieval. In *The tenth international conference on internet and web applications and services (ICIW)* (pp. 16–21).

Dammak, F., Kammoun, H., & Hamadou, A. B. (2017). Improving pairwise learning to rank algorithms for document retrieval, 2017. In *IEEE symposium series on computational intelligence (SSCI)* (pp. 1–8). IEEE.

Dammak, F., Kammoun, H., Hmid, S. B., & Hamadou, A. B. (2017). Semi-active learning to rank algorithms for document retrieval. *International Journal of Intelligent Information and Database Systems, 10*(3–4), 289–313.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research,7,* 1–30 (JMLR. org).

Duh, K., & Kirchhoff, K. (2008). Learning to rank with partially-labeled data. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 251–258).

Freund, Yoav, I., Raj, S., Robert, E., & Singer, Y. (2003). An efficient boosting algorithm for combining preferences. *Journal of machine learning research,4*(Nov), 933–969.

Freund, Y., Seung, H. S., Shamir, E., & Tishby, N. (1997). Selective sampling using the query by committee algorithm. *Machine Learning, 28*(2–3), 133–168.

Fujiwara, Y., & Irie, G. (2014). Efficient label propagation. In *International conference on machine learning* (pp. 784–792).

Gu, Y., Jin, Z., & Chiu, S. C. (2014). Combining active learning and semi-supervised learning using local and global consistency. *International conference on neural information processing* (pp. 215–222). Springer.

Huang, S.-J., Jin, R., & Zhou, Z.-H. (2010). Active learning by querying informative and representative examples. In *Advances in neural information processing systems* (pp. 892–900).

Järvelin, K., & Kekäläinen, J. (2000). IR evaluation methods for retrieving highly relevant documents. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 41–48). ACM.

Joachims, T. (2002). Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 133–142).

Kanoulas, E. (2009). *Building reliable test and training collections in information retrieval*. Northeastern University Boston.

Kleinberg, J. M. (1999). Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM), 46*(5), 604–632.

Krithara, A., Amini, M. R., Goutte, C., & Renders, J.-M. (2011). Learning aspect models with partially labeled data. *Pattern Recognition Letters, 32*(2), 297–304.

Kuwadekar, A., & Neville, J. (2011). Relational active learning for joint collective classification models. In *Proceedings of the 28th international conference on machine learning (icml-11)* (pp. 385–392).

Leng, Y., Xu, X., & Qi, G. (2013). Combining active learning and semi-supervised learning to construct SVM classifier. *Knowledge-Based Systems, 44,* 121–131.

Li, H. (2011). A short introduction to learning to rank. *IEICE Transactions on Information and Systems, 94*(10), 1854–1862.

Li, M., Li, H., & Zhou, Z.-H. (2009). Semi-supervised document retrieval. *Information Processing & Management, 45*(3), 341–355.

Liu, T.-Y. (2011). *Learning to rank for information retrieval*. Springer Science & Business Media.

Liu, T. Y., Xu, J., Qin, T., Xiong, W., & Li, H. (2007). LETOR: Benchmark dataset for research on learning to rank for IR, L R4IR.

Long, B., Bian, J., Chapelle, O., Zhang, Y., Inagaki, Y., & Chang, Y. (2014). Active learning for ranking through expected loss optimization. *IEEE Transactions on Knowledge and Data Engineering, 27*(5), 1180–1191.

Melville, P., & Mooney, R. J. (2004). Diverse ensembles for active learning. In *Proceedings of the twenty-first international conference on Machine learning*. ACM 74.

Miao, Z., & Tang, K. (2013). Semi-supervised ranking via list-wise approach. In *International conference on intelligent data engineering and automated learning* (pp. 376–383). Springer.

Muslea, I., Minton, S., & Knoblock, C. A. (2002). Active+ semi-supervised learning= robust multi-view learning. *ICML, 2,* 435–442.

Page, L., Brin, S., Motwani, R., & Winograd, T. (1999). *The pagerank citation ranking: Bringing order to the web*. Stanford InfoLab.

Pan, Z., You, X., Chen, H., Tao, D., & Pang, B. (2013). Generalization performance of magnitude-preserving semi-supervised ranking with graph-based regularization. *Information Sciences, 221,* 284–296.

Pavlu, V. (2008). *Large scale ir evaluation*. ProQuest LLC.

Ponte, J. M., & Croft, W. B. (1998). A language modeling approach to information retrieval. In *Proceedings of the 21st annual international ACM SIGIR conference on research and development in information retrieval* (pp. 275–281).

Qiang, W., Burges, C. J. C., Svore, K. M., & Gao, J. (2010). Adapting boosting for information retrieval measures. *Information Retrieval Journal, 13*(53), 254–270.

Robertson, S., & Zaragoza, H. (2009). The probabilistic relevance framework: BM25 and beyond. *Foundations and trends in information retrieval* (pp. 333–389).

Roy, N., & McCallum, A. (2001). *Toward optimal active learning through monte carlo estimation of error reduction* (pp. 441–448). ICML.

Settles, B. (2010). *Active Learning Literature Survey* (p. 1648). Comput. Sci. Technol. Rep.

Settles, B., & Craven, M. (2008). An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the conference on empirical methods in natural language processing* (pp. 1070–1079). Association for Computational Linguistics.

Seung, H. S., Opper, M., & Sompolinsky, H. (1992). Query by committee. In *Proceedings of the fifth annual workshop on Computational learning theory* (pp. 287–294). ACM.

Song, M., Yu, H., & Han, W.-S. (2011). Combining active learning and semi-supervised learning techniques to extract protein interaction sentences. *BMC bioinformatics,12*(12), S4. BioMed Central.

Tong, S. & Koller, D. (2001). Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research,2*(Nov), 45–66.

Truong, T. V. (2009). *Learning functions ranking with little labeled examples*. PhD thesis, University of Pierre and Marie Curie—Paris.

Tur, G., Hakkani-Tür, D., & Schapire, R. E. (2005). Combining active and semi-supervised learning for spoken language understanding. *Speech Communication, 45*(2), 171–186.

Wilcoxon, F. (1992). *Individual comparisons by ranking methods, Breakthroughs in statistics* (pp. 196–202). Springer.

Xia, F., Liu, T.-Y., Wang, J., Zhang, W., & Li, H. (2008). Listwise approach to learning to rank: theory and algorithm. In *Proceedings of the 25th international conference on Machine learning* (pp. 1192–1199).

Xu, J., & Li, H. (2007). Adarank: A boosting algorithm for information retrieval. In *Proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 391–398). ACM.

Yilmaz, E., & Aslam, J. A. (2006). Estimating average precision with incomplete and imperfect judgments. In *Proceedings of the 15th ACM international conference on Information and knowledge management* (pp. 102–111).

Zhou, Z.-H., Chen, K.-J., & Dai, H.-B. (2006). *ACM Transactions on Information Systems (TOIS),24*(2), 219–244 (ACM).

Zhu, X. J. (2005). *Semi-supervised learning literature survey*. University of Wisconsin-Madison Department of Computer Sciences.

Zhu, X., & Ghahramani, Z. (2002). *Learning from labeled and unlabeled data with label propagation*. Citeseer.