# An Extension of RankBoost for semi-supervised Learning of Ranking Functions

Faïza Dammak
Laboratoire MIRACL – ISIMS
SFAX
Sfax, Tunisia
faiza.dammak@gmail.com

Hager Kammoun
Laboratoire MIRACL – ISIMS
SFAX
Sfax, Tunisia
hager.kammoun@isd.rnu.tn

Abdelmajid Ben Hamadou
Laboratoire MIRACL – ISIMS
SFAX
Sfax, Tunisia
abdelmajid.benhamadou@isimsf.rnu.tn

*Abstract*—**The purpose of this paper was a semi-supervised learning method of alternatives ranking functions. This method extends the supervised RankBoost algorithm to combines labeled and unlabeled data. RankBoost is a supervised boosting algorithm adapted to the ranking of instances. Previous work on ranking algorithms has focused on supervised learning (i.e. only labeled data is available for training) or semi-supervised learning of instances. We are interested in semi-supervised learning, which has as objective to learn in the presence of a small quantity of labeled data, simultaneously a great quantity of unlabeled data, to generate a ranking method of alternatives. The goal is to understand how combining labeled and unlabeled data may change the ranking behavior, and how RankBoost can with its character inductive improve ranking performance.**

***Keywords-learning to rank; ranking functions; semi-supervised learning; RankBoost algorithm.***

## I. INTRODUCTION

Learning to rank is a relatively new research area which has emerged rapidly in the past decade. It plays a critical role in information retrieval. Learning to rank is to learn a ranking function by assigning a weight to each document feature, then using this obtained ranking function to estimate relevance scores for each document, and finally ranking these documents based on the estimated relevance scores [1][2]. This process has recently gained much attention in learning, due to its large applications in real problems such as information retrieval (IR). In learning to rank, the performance of a ranking model is strongly affected by the number of labeled examples in the training set, therefore, labeling large examples may require expensive human resources and time-consuming, especially for ranking problems. This presents a great need for the semi-supervised learning approaches [3] in which the model is constructed with a small number of labeled instances and a large number of unlabeled instances. Semi-supervised learning is a well-known strategy to label unlabeled data using certain techniques and thus increase the amount of labeled training data [5].

Ranking is the central problem for many information retrieval (IR) applications. It aims to induce an ordering or preference relations over a predefined set of labeled instances. This is for example the case of Document Retrieval (DR), where the goal is to rank documents from a collection based on their relevancy to a user's query. This type of problem is known under the name of ranking for alternatives [1]. The ranking of instances is another type of ranking which comes from the IR such as routing information [6].

Since obtaining labeled examples for training data is very expensive and time-consuming, it is preferable to integrate unlabeled data in training base.

Most semi-supervised ranking algorithms are graph-based transductive techniques [4]. These techniques can not easily extend to new test points outside the labeled and unlabeled training data. Induction has recently received increasing attention.

For an effective use of the semi-supervised learning on large collections data, [6] presents a boosting based algorithm for learning a bipartite ranking function (BRF) for instances. This an extended version of the RankBoost algorithm [7] that optimizes an exponential upper bound of a learning criterion which combines the misordering loss for both parts of the training set. We propose an adaptation of the supervised RankBoost algorithm on partially labeled data of alternatives which can be applied to some applications such as web search. Our algorithm based on pairwise approach [8] which takes query-document pairs as instances in learning.

Our contribution is to develop a semi-supervised ranking algorithm for alternatives. The proposed algorithm has an inductive character since it is able to infer an ordering on new examples that were not used for its training [5]. The unlabeled data will be initially labeled by a transductive method such as the *K* nearest neighbours *KNN.*

The rest of the paper is organized as follows : Section 2 provides a brief literature review to the related work, we introduce the principle learning to rank and its interest into the IR. We also detail the problem of ranking of alternatives, the RankBoost algorithm and the principle of semi-supervised learning. In sections 3, we present our proposal for semi-supervised method. The collections used and experimental results are detailed in Section 4. Finally, Section 5 concludes the paper and gives directions for future work.

## II. LEARNING TO RANK

Ranking a set of retrieval documents according to their relevance for a given query is a popular problem at the intersection of web search, machine learning, and information retrieval. Over the past decade, a large number of learning to rank algorithms has been proposed [9]. In learning to rank, a number of queries are provided, each query is associated with a perfect ranking list of documents, a ranking function assigns a score to each document, and ranks the documents in descending order of the scores [7]. The ranking order represents relative relevance of documents with respect to the query. In a problem related to learning to rank, an instance is a set of objects and a label is a sorting applied over the instance. Learning to rank aims to construct a ranking model from training data.

Many applications of learning to rank involve a large number of unlabeled examples and a few labeled examples, as expensive human effort is usually required in labeling examples [7].

The issue of effectively exploiting the information in the unlabeled instances to facilitate supervised learning has been extensively studied known as the name semi-supervised learning [2]. We are interested to apply the supervised RankBoost algorithm with this type of learning. Indeed, RankBoost has an inductive character; it is thus able to order a list of examples not seen during the phase of training by inferring an order on this list. In the following, we present the principle of the ranking for alternatives, the RankBoost algorithm as well as the principle of semi-supervised ranking algorithm.

### *A. Ranking of Alternatives*

Learning to rank is a newly popular topic in machine learning. When it is applied to DR, it can be described as the following problem : assume that there is a collection of alternatives which called documents in DR. In retrieval, giving a query, the ranking function assigns a score to each pair query-document, and ranks the documents in descending order of these scores. The ranking order represents the relevance of documents according to the query. The relevance scores can be calculated by a ranking function constructed with machine learning. This type of ranking is known as of ranking of alternatives [1].

### *B. RankBoost Algorithm*

RankBoost is a supervised learning algorithm of instances designed for ranking problems. It builds a document ranking function by combining a set of ranking features of a set of document pairs [3].

More precisely, RankBoost learns a ranking feature $f_t$ on each iteration, and maintains a distribution $D_t$ over the ranked pairs. The final ranking function $F$ is a linear combination of these ranking features that, in our context , defined by:

$$F = \sum_{t=1}^{T} \alpha_t f_t(x_i, k) \,. \tag{1}$$

where $x_i$ is the query and $k$ its vector of alternatives associated.

Each ranking feature $f_t$ is uniquely defined by an input feature $j_t \in \{1...d\}$ and a threshold $\theta_t$:

$$f_t(x) = \begin{cases} 1, \, if \; \varphi_{jt}(x_i, k) > \theta_t \\ 0, \, si \; non \end{cases} . \tag{2}$$

where $\varphi_{jt}(x_i, k)$ is the $j^{th}$ feature characteristic of $x_i$.

Assume that for all example pairs, one knows which example should be ranked above the other one. The learning criterion to be minimized in RankBoost is the number of example pairs whose relative ranking as computed by the final combination is incorrect.

### *C. Semi-supervised Ranking*

Semi-supervised ranking has a great interest in machine learning because it can readily use available unlabeled data to improve supervised learning tasks when the labeled data are scarce or expensive. Semi-supervised ranking also shows potential as a quantitative tool to understand human category learning, where most of the input is self-evidently unlabeled.

The majority of the semi-supervised ranking algorithms are transductive techniques based on valuated and non-oriented graph [10]. The latter is formed by connecting gradually the nearest points until the graph becomes connected. The nodes are consisted of the examples labeled and unlabeled of training base and the weights reflect the similarity between the neighboring examples. This graph is built with a method, such as $k$ nearest neighbors, which allows finding the labels of the unlabeled examples by exploiting the graph directly by propagating for example the labels of the data labeled with their unlabeled neighbors. It thus affects a score for each instance, 1 for the positive instances and 0 for the others. The scores are then propagated through the graph until the convergence. At the end, the scores obtained make it possible to induce an order on the whole of the unlabeled instances [5]. We chose this method in our context to label the unlabeled data in the training set.
These data will be used with the labeled as inputs in our proposal that have the advantage of both the inductive and transductive approaches. We thus propose a semi-supervised algorithm which it is able to infer an ordering on new pairs query-alternative that were not used for its training. We detail this proposal in the following section.

## III. PROPOSAL FOR SEMI-SUPERVISED METHOD

In training, a set of queries $X = \{x_1, x_2, .., x_m\}$ and a set of alternatives $Y$ is given. Each query $x_i \in X$ is associated with a list of retrieved alternatives of variable size *mi*, $y_i = (y_i^1, ..., y_i^{m_i})$, with $y_i^k \in IR$. $y_i^k$ represents the degree of relevance of the alternative $k$ from $x_i$. A feature vector $\varphi_j$ $(x_i, k)$ is created from each query-document pair $(x_i, k)$ [6].

The ranking function $f_t$ allows associating a score for this vector. We propose thus a labeled learning base $S = \{(x_i, y_i)\}_{i=1}^m$ and an unlabeled learning base formed with all parts of queries unlabeled $S_U = \{(x_i^{'})\}_{i=m+1}^{m+n}$.

In this paper, we demonstrate a semi-supervised learning method could worth exploring in ranking functions of alternatives. The principal motivation to this led to find an effective ranking function. And it is necessary to have a base of learning which often requires on the one hand the manual labeling alternatives and on the other hand the unlabeled alternatives. The goal is to find the best entered to label to reduce to the maximum the number of labeled data. For an effective use of the semi-supervised learning on large collections, we adapted a modification of the supervised ranking RankBoost algorithm, and we presented the model suggested and described its functionalities as well as the choices of implementation.

In the following part, we detail the operation of the RankBoost algorithm applied to our context.

### A. Adapation of RankBoost algorithm to semi-supervised ranking of alternatives

The adaptation of RankBoost is given in the algorithm 1: we dispose a labeled training set $S = \{(x_1, y_1), .., (x_m, y_m)\}$, where each example $x_i$ is associated with a vector of relevance judgment $y_i = (y_i^1, ..., y_i^{m_i})$ where $y_i^k \in IR$. $m_i$ denotes the number of alternatives for $x_i$.

$S' = \{(x_i^{'}, y_i^{'});\ i \in \{m+1, .., m+n\}\}$ is the second labeled subset obtained from unlabeled set $S_U$ by using the nearest neighbours (*NN*) algorithm.

At each iteration, the algorithm maintains a distribution $\lambda_t$ (resp. $\lambda_t'$) on the examples of the learning base S (rep. S'), a distribution $\nu_t^i$ (resp. $\nu_t^{i}{'}$) on the alternatives associated with the example $x_i$ (resp. $x_i'$) and a distribution $D_t^i$ (resp. $D_t^{i}{'}$) over the pairs (query, alternative), represented by a distribution on couples $(k, l)$ (resp. $(k', l')$) such as $y_i^k \in Y_+$ (resp. $y_i^{k}{'} \in Y_+'$) and $y_i^l \in Y_-$ (resp. $y_i^{l}{'} \in Y_-'$) for each example $x_i$ (resp. $x_i'$).

$\forall$ i $\in \{1,..,m\}$, $\forall$ $(k, l) \in \{1,.., m_i\}^2$ such as $y_i^k \in Y_+$, $y_i^l \in Y_-$,

$$D_t^i(\kappa,\lambda) = \lambda_t^i\ \nu_t^i(k)\ \nu_t^i(l). \tag{3}$$

$\forall$ i $\in \{m+1,.., m+n\}$, $\forall$ $(k', l') \in \{1,.., m_i'\}^2$ such as $y_i^{k}{'} \in Y_+'$, $y_i^{l}{'} \in Y_-'$:

$$D_t^{i}{'}(k',l') = \lambda_t^{i}{'}\ \nu_t^{i}{'}(k')\ \nu_t^{i}{'}(l'). \tag{4}$$

These distributions are updated due to the scoring function $f_t$, selected from the semi-supervised learning of ranking features algorithm (algorithm 2) which will return the resulting value of the threshold $\theta_{res}$ associated with each characteristic and the possible values which can be associated with $f_t$, such as:

$$f_t(x_i, k) = \begin{cases} 1 & si\ \varphi_j(x_i, k) > \theta_{res} \\ 0 & si\ \varphi_j(x_i, k) \le \theta_{res} \end{cases}. \tag{5}$$

where $x_i$ is the query of index i and $k$ is the index of the alternative associated with $x_i$.

For each example, the weight $\alpha_t$ is defined by [3]:

$$\alpha_t = \frac{1}{2}\ln\left(\frac{1+r_t}{1-r_t}\right). \tag{6}$$

where

$$\begin{aligned} r_t &= \sum\nolimits_{k,l} D_t^i(k,l)\left(f_t(x_i,k) - f_t(x_i,l)\right) \\ &+ \beta \sum\nolimits_{k',l'} D_t^{i}{'}(k',l')\left(f_t(x_i',k') - f_t(x_i',l')\right) \end{aligned}. \tag{7}$$

$\beta$ is a discount factor. When this factor is zero, we will find the situation of supervised learning.

**Algorithm 1.** RankBoost algorithm adapted to ranking of alternatives

```
Entry : A labeled learning set S= $\{(x_i, y_i);\ i \in \{1,..,m\}\}$
A labeled learning set S'= $\{(x_i^{'}, y_i^{'});\ i \in \{m+1,.., m+n\}\}$
obnained by KNN method.
Initialisation :
```

```
$\forall$ i $\in \{1, ..., m\}, \lambda_1^i = \frac{1}{m}, \; \nu_1^i(k) = \begin{cases} \frac{1}{p_i} \; si \; y_i^k \in Y_+ \\ \frac{1}{n_i} \; si \; y_i^k \in Y_- \end{cases}$

$\forall$ i $\in \{m+1, .., m+n\}, \; \lambda_1^i{}' = \frac{1}{n}, \; \nu_1^i{}'(k) = \begin{cases} \frac{1}{p_i'} \; si \; y_i^k{}' \in Y_+' \\ \frac{1}{n_i'} \; si \; y_i^k{}' \in Y_-' \end{cases}$

For $t$ : = 1,…, T do
  - Select the ranking feature $f_t$ from $D_t$ and $D_t'$
  - Calculate $\alpha_t$ using formula (6)
  - $\forall$ i $\in$ {1, .., $m$}, $\forall$ ($k,l$) $\in$ {1, ..., $m_i$}$^2$ such as
      $y_i^k \in Y_+$, $y_i^l \in Y_-$, update $D_{t+1}^i(k, l)$ :
      $D_{t+1}^i(k, l) = \lambda_{t+1} \, \nu_{t+1}^i(k) \, \nu_{t+1}^i(l)$
  - $\forall$ i $\in$ {$m$+1, .., $m$+$n$}, $\forall$ ($k'$, $l'$) $\in$ {1, ..., $m_i'$}$^2$ such
      as $y_i^k{}' \in Y_+'$, $y_i^l{}' \in Y_-'$, update $D_{t+1}^i{}'(k', l')$ :
          $D_{t+1}^i{}'(k', l') = \lambda_{t+1}' \, \nu_{t+1}^i{}'(k') \, \nu_{t+1}^i{}'(l')$
  - $\forall$ i $\in \{1, .., m\}, \; \lambda_{t+1}^i = \frac{\lambda_t^i Z_t^{1i}}{Z_t}$,
      $\nu_{t+1}^i(k) = \begin{cases} \frac{\nu_t^i(k)\exp(-\alpha_t f_t(x_i, k))}{Z_t^{1i}} \; si \; y_i^k \in Y_+ \\ \frac{\nu_t^i(k)\exp(\alpha_t f_t(x_i, k))}{Z_t^{-1i}} \; si \; y_i^k \in Y_- \end{cases}$
    where $Z_t^{1i}$, $Z_t^{-1i}$ and $Z_t$ are defined by :
      $Z_t^{1i} = \sum_{k:y_i^k \in Y_+} \nu_t^i(k)\exp(-\alpha_t f_t(x_i, k))$,
      $Z_t^{-1i} = \sum_{l:y_i^l \in Y_-} \nu_t^i(l)\exp(\alpha_t f_t(x_i, l))$, $Z_t = \sum_{i=1}^{m} \lambda_t^i Z_t^{-1i} Z_t^{1i}$
  - $\forall$ i $\in \{m+1, .., m+n\}, \; \lambda_{t+1}^i{}' = \frac{\lambda_t^i{}' Z_t^{-1i}{}' Z_t^{1i}{}'}{Z_t'}$,
      $\nu_{t+1}^i{}'(k') = \begin{cases} \frac{\nu_t^i{}'(k')\exp(-\alpha_t f_t(x_i', k'))}{Z_t^{1i}} \; si \; y_i^k{}' \in Y_+' \\ \frac{\nu_t^i{}'(k')\exp(\alpha_t f_t(x_i', k'))}{Z_t^{-1i}{}'} \; si \; y_i^k{}' \in Y_-' \end{cases}$
    where $Z_t^{1i}{}'$, $Z_t^{-1i}{}'$ and $Z_t'$ are defined by :
      $Z_t^{1i}{}' = \sum_{k':y_i^k{}' \in Y_+'} \nu_t^i{}'(k')\exp(-\alpha_t f_t(x_i', k'))$
      $Z_t^{-1i}{}' = \sum_{l':y_i^l{}' \in Y_-'} \nu_t^i{}'(l')\exp(\alpha_t f_t(x_i', l'))$
      $Z_t' = \sum_{i=1+m}^{m+n} \lambda_t^i{}' Z_t^{-1i}{}' Z_t^{1i}{}'$
end
Output : The final ranking function $F = \sum_{t=1}^{T} \alpha_t f_t$
```

In each iteration t, $\alpha_t$ is selected in order to minimize the normalization factors $Z_t$ and $Z_t'$.

Our goal in this algorithm is finding a function *F*, which minimizes the average numbers of irrelevant alternatives scored better than relevant ones in S and S' separately. We call this quantity the average ranking loss for alternatives, $Rloss(F, S \cup S')$ defined as:

$$Rloss(F, S \cup S') = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{n_i p_i}\sum_{k:y_i^k \in Y_+}\sum_{l:y_i^k \in Y_-}[[f(x_i, k) - f(x_i, l) \le 0]] + \frac{\beta}{n}\sum_{i=1}^{m}\frac{1}{n'_i \, p'_i}\sum_{k':y'^k_i \in Y_+'}\sum_{l':y_i^k{}' \in Y_-'}[[f(x_i', k') - f(x_i', l') \le 0]] \tag{8}$$

where $p_i$ (resp. $n_i$) is the number of relevant alternatives (resp. not relevant) for example $x_i$ in S and $p'_i$ (resp. $n'_i$) is the number of relevant alternatives (resp. not relevant) for example $x'_i$ in S'. And the expression [[P]] is defined to be 1 if predicate P is true and 0 otherwise.

## *B. Adaptation of the Algorithm of selection of ranking features*

The algorithm of selection of ranking features or functions (Algorithm 2) makes it possible to find, with a linear complexity in a number of alternatives, a function $f_t$ which minimizes $r_t$ in a particular case where the function $f_t$ is in {0, 1} and is created by thresholded characteristics associated to the examples.

Let us suppose that each query $x_i$ (resp. $x_i'$) has a set of characteristics provided by functions $\varphi_j$, j = 1... d. For each j, $\varphi_j(x_i, k)$ (resp. $\varphi_j(x_i', k')$) is a real value. Thus, it is a question of using a thresholding of the characteristic $\varphi_j$ to create binary values. All the basic functions are created by defining a priori a set of thresholds $\{\theta_q\}_{q=1}^{Q}$ with $\theta_1 > ... > \theta_q$. Generally, these thresholds depend on the characteristic considered.

**Algorithm 2. Algorithm of selection de ranking features**

```
Entry :
▪ ∀ i ∈ {1,…, m}, (k, l) ∈ {1,…, $m_i$ } such as $y_i^k \in Y_+$ and
    $y_i^l \in Y_-$ :
    A distribution $D_t^i(k,l) = \lambda_t^i \, v_t^i(k) \, v_t^i(l)$ on the training set S.
▪ ∀ i ∈ {m+1, .., m+n}, ∀ (k', l') ∈ {1, ..., $m_i'$ }$^2$ such as
    $y_i^k{}' \in Y_+^{'}$, $y_i^l{}' \in Y_-^{'}$ :
  A distribution $D_{t+1}^i{}'(k', l') = \lambda_{t+1}^{'} \, v_{t+1}^i{}'(k') \, v_{t+1}^i{}'(l')$ on the training subset S'.
▪ Set of characteristics $\{\varphi_j(x_i,k)\}_{j=1}^{d}$
▪ For each $\varphi_j$ , a set of thresholds $\{\theta_q\}_{q=1}^{Q}$ such as $\theta_1 > \ldots > \theta_q$
Initialisation :
▪ ∀ i ∈ {1,…, m}, (k, l) ∈ {1,…, $m_i$ },
    $\pi(x_i, k) = y_i^k \lambda_1^i v_1^i(k) \sum_{l: y_i^l \neq y_i^k} v_1^i(l)$
▪ ∀ i ∈ {m+1,..,m+n}, (k', l') ∈ {1,…, $m_i'$ },
    $\pi'(x_i', k') = y_i^k{}' \lambda_1^i{}' v_1^i{}'(k') \sum_{l': y_i^{l'} \neq y_i^k{}'} v_1^i(l')$
 r*←0
For j :=1,…, d do
    - L← 0
    For q :=1,…, Q do
      L← L + $\sum_{i=1}^{m} \sum_{k:\varphi_j(x_i,k)} \pi(x_i, k)$
      + $\sum_{i=m+1}^{m+n} \sum_{k':\varphi_j(x_i',k')} \pi'(x_i', k')$
    if |L|>|r*| then
         r*← L
         j*←j
         $\theta^* \leftarrow \theta_q$
         k*←k
     end
   end
 end
Output : ($\varphi_j$ *, $\theta$ *, k*)
```

## IV. EXPERIMENTS

We used the MQ2008-semi (Million Query track) dataset in LETOR4.0 (*LEarning TO Rank*) [1] in our experiments, because it contains both labeled and unlabeled data. There are about 2000 queries in this dataset. On average, each query is associated with about 40 labeled documents and about 1000 unlabeled documents.

MQ2008-semi is conducted on the .GOV2 corpus using the TREC 2008, which is crawled from Web sites in the .gov domain. There are 25 million documents contained in the .GOV2 corpus, including HTML documents, plus the extracted text of PDF, Word and postscript files [1].

Each subset of the collection MQ2008-semi is partitioned into five parts, denoted as S1, S2, S3, S4, and S5, in order to conduct five-fold cross validation. The results reported in this section are the average results over multiple folds. For each fold, three parts are used : one part for training, one part for validation, and the remaining one for testing. The training set is used to learn the ranking model, the validation set is used to tune the parameters of the ranking model, such as the number of iterations in RankBoost. And the test set is used to report the ranking performance of the model.

In order to compare the performance of the algorithm we evaluate our experimental results using a set of standard ranking measures such as Mean Average Precision MAP, Precision at N, and normalised Discounted Cumulative Gain (NDCG).

$$MAP = \frac{\sum_{n=1}^{N} (P@n * rel(n))}{\# total\ relevants\ \ docs\ for\ this\ query} \qquad (14)$$

$$P@n = \frac{\# relevantdocs\ in\ top\ n\ results}{\mathrm{n}} \qquad (15)$$

$$N(n) = Z_n \sum_{j=1}^{n} \frac{2^{r(j)} - 1}{\log(1+j)} \qquad (16)$$

The value of the discount factor, which provided the best ranking performance for these training sizes, is $\beta = 1$. We therefore use this value in our experiments.

Tables 1 and 2 show the results on testing set generated by an assessment tool associated with the benchmark Letor [1].

TABLE I. P@N AND MAP MEASURES ON THE MQ2008-SEMI COLLECTION

| Algorithmes | P@1 | P@3 | P@5 | P@7 | P@10 | MAP |
|---|---|---|---|---|---|---|
| RankBoost | 0. 457 | 0.391 | 0.340 | 0.302 | 0.248 | 0.477 |
| RankSVM | 0.427 | 0.390 | 0.347 | 0.302 | 0.249 | 0.469 |
| Algorithme 1 | 0.450 | 0.393 | 0.341 | 0.302 | 0.252 | 0.479 |

TABLE II. NDCG@N MEASURES ON THE MQ2008-SEMI COLLECTION

| Algorithmes | NDCG@1 | NDCG@3 | NDCG@5 | NDCG@7 | NDCG@10 |
|---|---|---|---|---|---|
| RankBoost | 0.463 | 0.455 | 0.449 | 0.412 | 0.430 |
| RankSVM | 0.495 | 0.420 | 0.416 | 0.413 | 0.414 |
| Algorithme 1 | 0. 465 | 0.453 | 0.438 | 0.414 | 0.434 |

These results illustrate how the unlabeled data affect the performance of ranking in the proposed algorithm. We notice a slight improvement in using the criterion P @ n (resp. NDCG) for n = 3 and n =10 (resp. for n = 1, n=7 and n = 10). The results also show that our proposed algorithm has an average precision (MAP) better than that found by RankBoost and RankSVM. These results prove the interest of integrating unlabeled data in ranking functions with semi-supervised learning.

## V. CONCLUSION

In this paper, we proposed a semi-supervised learning algorithm for learning ranking functions for alternatives. This algorithm has the advantages of both transductive and inductive approaches, and can be applied in semi-supervised and supervised ranking setups. In fact, this algorithm is able to infer an ordering on new pairs query-alternative that were not used for its training. The advantage of this proposition is that it is able to advantageously exploit the unlabeled alternatives. We propose in the following to supplement the experimental part and to integrate other methods such as active learning which select most informative examples for ranking learning.

## REFERENCES

[1] T.-Y. Liu, J. Xu, T. Qin, W.-Y. Xiong, and H. Li, LETOR: Benchmark dataset for research on learning to rank for information retrieval. SIGIR, 2007.

[2] J. Xu, and H. Li, AdaRank : a boosting algorithm for information retrieval. In Kraaij, W., de Vries, A. P. Clarke, C. L. A. Fuhr, N. Kando, N. editors, SIGIR, pp. 391-398. ACM, 2007.

[3] X. Zhu, Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison, 2005.

[4] D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Schölkopf, Ranking on data manifolds. NIPS. MIT Press, 2003.

[5] K. Duh, and K. Kirchhoff, Learning to rank with partially-labeled data. In Myaeng, S.-H. Oard, D. W. Sebastiani, F. Chua, T.-S., and Leong, M.-K., editors, SIGIR, pp. 251-258. ACM. 2008.

[6] M.-R Amini, V. Truong, and C. Goutte: A boosting algorithm for learning bipartite ranking functions with partially labeled data. SIGIR 2008: pp. 99-106. 2008.

[7] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer, "An efficient boosting algorithm for combining preferences". Journal of Machine Learning Research, pp. 933-969, 2003.

[8] F. Xia, T. Liu, J. Wang, W. Zhang and H. Li, Listwise approach to learning to rank: theory and algorithm. In ICML '08, pp. 1192-1199, New York, NY, USA, ACM 2008.

[9] T. Y. Liu, Learning to Rank for Information Retrieval. Now Publishers, 2009.

[10] S. Agarwal, Transductive Ranking on Graphs, Computer Science and Artificial Intelligence Laboratory Technical Report, CSAIL-TR-2008-051, 2008.