Mach Learn (2015) 101:415–435
DOI 10.1007/s10994-014-5442-3

DEARank: a data-envelopment-analysis-based ranking method

Chunheng Jiang · Wenbin Lin

Received: 7 November 2013 / Accepted: 6 April 2014 / Published online: 27 June 2014

Abstract A new weak-ranker construction method based on **D**ata **E**nvelopment **A**nalysis technique is presented. Each weak ranker represents a feature subset drawn from the complete feature space. Two linear programming models are formulated, both of which treat the documents to be ranked as the decision making units. By solving the models, we construct a pool of weak-ranker candidates from the optimal weight vectors, and then develop DEARank algorithm based on Boosting technique. We conduct extensive experiments on LETOR 3.0 and LETOR 4.0 collections, with twelve well-known algorithms as the baselines. The experimental results indicate that DEARank is a competitive learning to rank algorithm.

Keywords Learning to rank · Listwise · Data envelopment analysis · Boosting

1 Introduction

Ranking is a core problem in information retrieval, specially in web search engine. Recently, learning to rank, a rapidly developing branch of machine learning, has shown its strong ability for the ranking task. When being applied to document retrieval, learning to rank first establishes a ranking model learned from the training data sets, which consist of queries and associated documents with relevance grades. For a new query and a set of associated documents, the trained ranker (model) assigns each document a score, and then sorts them by their scores. To evaluate the ranker's accuracy, evaluation measures such as Mean Average Precision (MAP) (Baeza-Yates et al. 1999) and Normalized Discounted Cumulative Gain (NDCG) (Järvelin and Kekäläinen 2000) can be utilized.

Editors: Vadim Strijov, Richard Weber, Gerhard-Wilhelm Weber, and Süreyya Ozogur Akyüz.

C. Jiang
School of Mathematics, Southwest Jiaotong University, Chengdu, China
e-mail: chiangchunheng@my.swjtu.edu.cn

W. Lin (✉)
School of Physical Science and Technology, Southwest Jiaotong University, Chengdu, China
e-mail: wl@swjtu.edu.cn

Recent works on learning to rank can be categorized into three main paradigms: pointwise, pairwise, and listwise. The pointwise approach, such as Pranking (Crammer and Singer 2001) and McRank (Li et al. 2007), formulates ranking as an ordinal regression or classification problem. Unfortunately, this approach fails to handle pairwise preferences and the orders of retrieved documents. The pairwise approach, such as Ranking SVM (Smola 2000; Joachims 2002), RankBoost (Freund et al. 2003) and RankNet (Burges et al. 2005), deals with the ranking problem by treating documents pairs as training instances, and trains models via the minimization of related risks, such as the number of misordered pairs and the cross-entropy loss (Burges et al. 2005). However, in this approach, the ground truth labels with respect to partial orders of documents are ignored. The listwise approach, such as ListNet (Cao et al. 2007), AdaRank (Xu and Li 2007), ListMLE (Xia et al. 2008) and RankCosine (Qin et al. 2008), views list of documents as a whole in the training, and focus on recovering the true permutation of the retrieved documents.

Previous works (Cao et al. 2007; Lan et al. 2009; Liu 2011; Niu et al. 2012) demonstrated that the listwise approach can capture enough information, and therefore can address the ranking problem in a very natural way. However, the majority of existing algorithms depend heavily upon the human-elicited relevance grades, and ignore the intrinsic relationships between documents and ranking features (Moon et al. 2010).

The aim of our work is to fill the gap, and shows a different way of constructing weak learners to improve ranking performance. We employ Data Envelopment Analysis (DEA) (Charnes et al. 1978; Seiford and Thrall 1990) technique to recover the information which the conventional listwise approaches lost. Since the 1970s, DEA had been widely used in evaluating the relative efficiencies of a set of homogenous Decision Making Units (DMUs) with multiple inputs and outputs. Without explicit specification of the functional relation between the inputs and outputs, DEA assigns each unit an efficiency score. One unit which gets a higher score is considered to be more efficient. A score of one implies an ideal efficient unit. The efficient DMUs form a piecewise linear frontier to assist in measuring the degree of efficiency for units. DEA has the natural ability for dichotomized classification and ranking, and it is straightforward to group DMUs into two sets: efficient and inefficient. It has been applied to deal with tasks such as classification (Troutt et al. 1996; Seiford and Zhu 1998; Emel et al. 2003; Xu and Wang 2009; Pendharkar 2011; Yan and Wei 2011), clustering (Po et al. 2009) and association rule mining (Chen 2007; Toloo et al. 2009). To assist decision making, a variety of methods have also been proposed for DMUs ranking in the literature of DEA, and a comprehensive survey can be found in the works (Adler et al. 2002; Jahanshahloo et al. 2008). All the DMUs ranking approaches in DEA context, such as cross-efficiency method (Sexton et al. 1986) and super-efficiency method (Andersen and Petersen 1993), directly use the calculated efficiency scores to rank the given DMUs. However, they are unable to make prediction for other unknown DMUs.

In this paper, two DEA models, namely CCR-I and CCR-O are derived to evaluate documents' *relative efficiency*. Both models are based on the assumption that each document is a DMU, but there is a difference in the role of documents' feature vectors. In the model of CCR-I, documents are represented as units with multiple outputs (the features) and the same input (a query). In the model of CCR-O, each document is represented as a unit with multiple inputs (the features) and one output (a human-elicited relevance score).

On the basis of these two DEA models, we propose a novel weak construction method to build a pool of weak ranker candidates. Each candidate in the pool is an optimal weight solved from CCR-I or CCR-O. Inspired by the works (Freund et al. 2003; Xu and Li 2007; Qin et al. 2008), we incorporate Boosting technique (Schapire 1990) into the rank learning.

The combination of the objective evaluation of DEA and the weak learnability of Boosting leads to an improved rank learning method called DEARank.

The rest of this paper is organized as follows. In Sect. 2, we give a brief review of DEA technique, Boosting technique and AdaRank algorithm, as well as some notations for readers' convenience. The DEARank method is presented in Sect. 3, and the experimental results are reported in Sect. 4. In Sect. 5, we discuss the effect of error diversity on ranking performance, and present an alternative approach to weight the weak rankers, then end the work with a conclusion.

2 Background

2.1 Data envelopment analysis

Data envelopment analysis (DEA) is a typical non-parametric LP approach for evaluating DMU's performance by a relative efficiency score, which is defined as a ratio of the sum of weighted outputs to the sum of weighted inputs. It allows DMUs to search the most advantageous weight vectors in the calculations of relative efficiency (Kao and Hung 2005). When clear from context, we use the terms “DMU”, “unit”, “document”, and “document unit” interchangeably.

Suppose there are N DMUs to be evaluated, and the nth DMU consumes M-inputs $X_n \in R^M$ and produces S-outputs $Y_n \in R^S$. We also assume that the inputs and the outputs are nonnegative, i.e., $X_n \geq 0$ and $Y_n \geq 0$. Let $w_n \in R^M$ denote the weight vector of the inputs X_n, and $u_n \in R^S$ be the weight vector of the outputs Y_n.

The first classical DEA model CCR is pioneered by Charnes et al. (1978). The model aims at maximizing the relative efficiency of the nth unit, with the constraints that the relative efficiency of each DMU with the optimal weights w_n and u_n is not greater than one. The model can be formulated as follows

$$\begin{aligned} \max_{w_n, u_n} \quad & u_n^T Y_n / w_n^T X_n \\ s.t. \quad & u_n^T Y_i / w_n^T X_i \leq 1, \quad i = 1, 2, \ldots, N \\ & w_n \geq 0, u_n \geq 0. \end{aligned} \tag{1}$$

With the Charnes-Cooper transformation

$$\begin{cases} t = 1/w_n^T X_n, \\ \nu = t w_n, \\ \mu = t u_n, \end{cases} \tag{2}$$

we have $\nu^T X_n = t w_n^T X_n = 1$, and

$$u_n^T Y_n / w_n^T X_n = \mu^T Y_n \ . \tag{3}$$

Therefore, Eq. (1) can be solved via an equivalent LP model

$$\begin{aligned} \max_{\mu, \nu} \quad & \mu^T Y_n \\ s.t. \quad & \mu^T Y_i - \nu^T X_i \leq 0, \quad i = 1, 2, \ldots, N \\ & \nu^T X_n = 1 \\ & \nu \geq 0, \mu \geq 0. \end{aligned} \tag{4}$$

The optimal solution (μ^*, ν^*) to Eq. (4) represents the nth DMU's most advantageous preference weight pair, which may vary for different DMUs. It's the diversity of preferences that results in a very competitive performance in contrast to the weak feature rankers used in (Freund et al. 2003; Xu and Li 2007).

2.2 Efficiency reference set

Definition 1 (*Efficiency Reference Set*) Suppose (u_n^*, w_n^*) be an optimal weight vector pair of unit n. The Efficiency Reference Set (ERS) of unit n is a set defined as

$$\text{ERS}_n = \left\{ i \,\middle|\, \frac{(u_n^*)^T Y_i}{(w_n^*)^T X_i} = 1, \forall i = 1, \ldots, N \right\},$$

with a common weight vector pair (u_n^*, w_n^*).

Theorem 1 *The common optimal weight vector pair (u_n^*, w_n^*) of ERS_n, is also an optimal solution of the problem CCR_i associated to every unit $i \in ERS_n$.*

Proof According to CCR model, the pair (u_n^*, w_n^*) satisfies the constraints

$$\frac{(u_n^*)^T Y_j}{(w_n^*)^T X_j} \leq 1, \; j = 1, \ldots, N.$$

which implies that it is a feasible solution to the programming CCR_i. Since unit i is a member of ERS_n, we have

$$\frac{(u_n^*)^T Y_i}{(w_n^*)^T X_i} = 1 \; ,$$

i.e., the pair (u_n^*, w_n^*) is also an optimal solution of the problem CCR_i, and the ith unit is efficient. □

Each CCR model is represented by a LP problem, and corresponds to one DMU. Suppose there are N units, in order to examine the relative efficiency of each unit, we have to run LP solving procedure N times. Although LPs can be solved in polynomial-time, the repeated solving of LPs tends to be computationally intensive and time consuming, especially when large data sets are involved. According to Theorem 1, the units in a same *Efficiency Reference Set* share a common optimal weight. This property helps to alleviate the computational burden. Also, we can find different approaches (Ali 1993; Barr and Durchholz 1997; Zhu 2003; Emrouznejad and Shale 2009) (detailed discussion of which is beyond the scope of this work) to address this issue.

2.3 Boosting technique

Boosting is a general method based on the idea of creating a highly accurate model by combining many relatively weak rules. Boosting has its roots in the PAC model. Schapire (1990) is the first to show that the *weak* learning algorithms which perform just slightly better than coin flip can be combined to create an arbitrarily accurate *strong* learning algorithm. The proof gives birth to the first polynomial-time Boosting procedure. Later, Freund (1990) develops a *boost-by-majority* version. Both algorithms share a common limitation, they require the losses of weak learners are bounded. Fortunately, AdaBoost (Freund and Schapire 1995) removes the inapplicable constraint and has been successfully applied to different learning tasks.

Due to its advantages of simplicity and ease of implementation, Boosting has been extensively used to improve the performance of different learning algorithms. Several Boosting-based methods, such as RankBoost (Freund et al. 2003), AdaRank (Xu and Li 2007), RankCosine (Qin et al. 2008), PermuRank (Xu et al. 2008) and cascade ranking model (Wang et al. 2011) have been developed for rank learning. Among them, AdaRank is most suitable for combining weak rankers that are generated independently, and also it can directly optimize the evaluation metrics for ranking.

2.4 AdaRank algorithm

AdaRank is one of the state-of-the-art listwise learning to rank algorithms, which extends the famous AdaBoost (Freund and Schapire 1995) philosophy from binary classification to rank learning. It takes queries (with associated document lists) as instances, and directly optimizes an exponential loss which bounds a specific IR measure (e.g., MAP, NDCG). With the maintained weight distribution over the training queries, AdaRank minimizes the loss function through continuously re-weighting the weight distribution over the queries. At each round, AdaRank selects the weak ranker that performs best in terms of the specific measure, and assigns it a weight according to its ranking accuracy, and combines it with the current learned model. Simultaneously, the algorithm increases the weight of each query that has not been ranked well by the current model, and decreases the weights of each query on which the selected weak shows good performance, so that the training procedure in the next round can focus on the hard-to-learn queries. AdaRank fits a forward stage-wise additive model (i.e., the final ranking function) represented in the form of a weighted combination of the selected weak rankers.

2.5 Notations

Suppose the training data set S contains $|Q|$ queries $Q = \{q_1, \ldots, q_{|Q|}\}$. For any query $q_i \in Q$, S provides a set of associated documents $D_i = \{d_{i1}, \ldots, d_{in_i}\}$, where n_i denotes the number of documents in D_i. A query-document pair (q_i, d_{ij}) can be represented by one m-dimensional feature vector x_{ij}, and each query corresponds to one vector set $x_i = \{x_{i1}, \ldots, x_{in_i}\}$, where m denotes the dimension of features. The relevance grades are denoted by $y_i = \{y_{i1}, \ldots, y_{in_i}\}, i = 1, \ldots, |Q|$.

The learned ranking function F assigns scores to documents, e.g., $s_{ij} = F(x_{ij})$, and its ranking performance can be evaluated with a generic measure denoted by $E(x_i, y_i, F)$. Given a well-defined loss function L, the ranker can also be evaluated by loss $L(x_i, y_i, F)$. The notation rules are summarized in Table 1.

3 DEARank method

In this section, we first formulate two modified DEA models CCR-I and CCR-O, then introduce DEARank algorithm. To our knowledge, this is the first time that the DEA technique is employed in the process of documents rank learning.

3.1 Modified DEA models

It is crucial for DEA to select the input and output variables, which may severely affect the efficiency scores of DMUs. However, DEA does not provide a general rule for the

Table 1 Summary of notations

Notation	Description
S	Training data set
$Q = \{q_i, \dots, q_{\|Q\|}\}$	Query set in training data set
$D_i = \{d_{i1}, \dots, d_{in_i}\}$	Document set associated with query q_i
$x_i = \{x_{i1}, \dots, x_{in_i}\}$	Feature vector set of documents associated with q_i
$y_i = \{y_{i1}, \dots, y_{in_i}\}$	Relevance grades for D_i
$x_{ij} \in R^m$	Feature vector of document d_{ij}
$y_{ij} \in R$	Relevance label of document d_{ij}
Φ	Pool of weak ranker candidates
F_t	Combined ranker at epoch t
h_t	Weak ranker selected at epoch t
$E(x_i, y_i, F)$	Performance measure of F on document set D_i
$L(x_i, y_i, F)$	Loss of ranker F on document set D_i

specification of the inputs and outputs, rather, it is left to the analysts' judgments. In the rank learning, the variables are the blended document features. We propose two different variable-selection methods, according to the relationships among queries, features, relevance grades, and particularly, documents.

Given a query q, suppose there are n associated documents $D = \{d_1, \dots, d_n\}$. We take all documents as the units with a constant input (e.g., 1) and multiple outputs. The document features are all benefit, and used as the output variables. For two documents d_i and d_j, if one feature value of d_i is higher than that of d_j (e.g., page rank score (Brin and Page 1998)), while the other features being the same, then d_i should be preferred and deserves a higher relevancy score. In DEA, a larger relevancy score results in a higher relative efficiency level. With the aforementioned assumption, we formulate a degenerated DEA model with one input and multiple outputs (Lovell and Pastor 1999; Liu et al. 2011; Kostrzewa et al. 2011), and named CCR-I:

$$\begin{aligned} \max_{\mu} \quad & \mu^T x_k \\ s.t. \quad & \mu^T x_i \le 1, \quad i = 1, \dots, n \\ & \mu \ge 0. \end{aligned} \tag{5}$$

In document ranking problem, the relevance grades of documents for the training set are assumed to be given. Here, we explore the human-judgement procedure: how relevance grades of documents with respect to the associated queries are generated. We assume that, in the procedure, the input features characterizing a document are the major factors dominating assessors' decision, and the units (i.e., the documents) *invest* multiple input features and *harvest* from the assessors the relevance scores. According to the model CCR-I, a document with absolutely higher features deserves a higher score. Alternatively, we may assume that the assessor gives the document a lower score. This counterintuitive assumption is not well self-explained. For instance, one document with duplicate keywords has higher feature values, e.g., term frequency, sometimes is suspicious of black-hat-backed spam pages, therefore, may be punished by search engines (Ntoulas et al. 2006). In order to cope with this case, we propose to use another model named CCR-O:

$$\begin{aligned} \min_{\nu} \quad & \nu^T x_k \\ s.t. \quad & \nu^T x_i \geq \varphi(y_i), \quad i = 1, 2, \ldots, n \\ & \nu \geq 0, \end{aligned} \tag{6}$$

where φ is a continuous real-valued, order-preserving function.

The documents formulated via CCR-O model are viewed as the units with multiple inputs (features) and one output (the relevance score). The documents under CCR-I model, however, are viewed as the units with one input (a constant, e.g., 1) and multiple outputs (features).

Both models are standard LP problems, and can be efficiently solved by the well-known simplex method or interior-point method. To seek a high relative efficiency score, DMUs tend to search a perfect weighting structure on features. Each optimal weight vector μ^* in (5) or ν^* in (6) corresponding to unit k represents the underlying importance it places on the features. In economical words, the weight vector is a *pricing list*, the sum of weighted inputs/outputs indicates a intrinsic value of the resources. These optimal weights, therefore can be used as *weak* rankers on documents and make up the building blocks for our method.

3.2 Feature subset weak construction: DEARank

AdaRank directly uses the raw features as the weak rankers. Instead of using the raw features, we apply DEA technique to create weak ranker candidates. Specifically, given a query and its associated documents, we formulate a CCR-I or CCR-O model for each document, and then solve it. Within the query, each document has at least one optimal solution to the corresponding derived DEA model. The solution, representing a weighting strategy adopted by the unit under consideration, is used as a weak ranker candidate. With a pool of weak ranker candidates, we start the training procedure, as described in Algorithm 1. Notice that MAP or NDCG (to be defined in Sect. 4.2) is used as the evaluation measure throughout this paper.

Algorithm 1 DEARank

```
Solve models **CCR-I** or **CCR-O** to populate $\Phi$ on $S$
**Input:** Training set $S$, weak ranker candidates $\Phi$, initialized queries probability distribution $P_1(i) = 1/|Q|$
  and combined ranker $F_0 = \varnothing$
  **for** $t = 1, \ldots, T$ **do**
    1. Learn a weak ranker $h_t$ with distribution $P_t$ on $Q$ using Eq. (8).
    2. Weight the selected weak ranker $h_t$ using Eq. (9).
    3. Combine the learned weak ranker $h_t$ with the current ensemble ranker $F_{t-1}$
         $F_t = F_{t-1} + \beta_t h_t$ .
    4. Re-weight the queries with the distribution
         $P_{t+1}(i) = \frac{\exp\{-E(x_i, y_i, F_t)\}}{\sum_{j=1}^{|Q|} \exp\{-E(x_j, y_j, F_t)\}}$ .
  **end for**
**Output:** The final ranker $F_T = \sum_{t=1}^{T} \beta_t h_t$.
```

At each round, DEARank examines all the weak ranker candidates in the pool in step 1, and the one who performs best on the whole training queries with the weight distribution will be selected. Then, the selected weak ranker is assigned with a weight which is obtained by Eq. (9) in step 2, and it is combined with the current learned model to make the empirical loss decrease in step 3. To focus on the hard-to-learn queries, DEARank decreases the weights of queries on which the current learned model performs well, and increases the weights of queries that are not ranked well by the learned model. The weight distribution over queries is updated (step 4) according to this adaptive strategy.

Another modification is made due to the fact that AdaRank may be dominated by the weak ranker which performs well for most training queries, and the weak learning procedure cannot further improve the performance of the ranking model consequently (Cartright et al. 2009). We adopt the strategy given by Cartright et al. (2009) to deal with this domination problem in DEARank.

In the algorithm, we minimize an upper bound of the exponential loss of the combiner F_t on the training set to maximize the ranking accuracy with respect to the generic IR measure E

$$\min_{h\in\Phi,\beta\in R}\sum_{i=1}^{|Q|}\exp\left\{-E(x_i,y_i,F_{t-1}+\beta h)\right\}. \tag{7}$$

To minimize the surrogate loss function, we select a weak ranker

$$h_t = \arg\max_{h\in\Phi}\sum_{i=1}^{|Q|}P_t(i)E(x_i,y_i,h), \tag{8}$$

and assign it a weight

$$\beta_t = \frac{1}{2}\ln\frac{1+\sum_{i=1}^{|Q|}P_t(i)E(x_i,y_i,h)}{1-\sum_{i=1}^{|Q|}P_t(i)E(x_i,y_i,h)}, \tag{9}$$

according to its performance $\sum_{i=1}^{|Q|}P_t(i)E(x_i,y_i,h)$ based on the training set with the query weight distribution P_t at round t, as indicated in (Xu and Li, 2007).

In summary, DEARank first constructs the candidates of weak rankers via the optimal weights of documents obtained by DEA. Each optimal weight represents the most advantageous preference on the inputs/outputs for one document. With the boosting technique, DEARank trains the ranking model by repeatedly selecting weak rankers, which may come from different queries. The variety of the weak ranker, representing the collective intelligence of units, helps to improve the ranking performance of the whole set of documents.

4 Experimental evaluation

4.1 Data sets

To examine the performance of DEARank, we conduct experiments on LETOR (Qin et al. 2010), which is a collection of benchmark data sets for research on learning to rank. LETOR contains standard features, relevance judgments, as well as the results of dozen state-of-the-art learning to rank algorithms. The versions LETOR 3.0 and LETOR 4.0 are used in this work. LETOR 3.0 contains seven data sets: HP2003, HP2004, NP2003, NP2004, TD2003, TD2004 and OHSUMED. LETOR 4.0 contains two data sets: MQ2007 and MQ2008.

LETOR 3.0 is composed of two document corpora: the Gov corpus and the OHSUMED corpus. The OHSUMED corpus (Hersh et al. 1994) is a set of medical publications, and contains millions of records from 270 medical journals. There are 106 queries used to retrieve records in OHSUMED. The Gov web page collection is searched using three query sets: topic distillation (TD), home page finding (HP) and named page finding (NP). While NP is related to the traditional information retrieval task, TD and HP are related to more navigational tasks, in which only one document is the right answer to a query. LETOR 4.0 is created based on the Gov2 web page collection and two query sets from Million Query track of TREC 2007 and TREC 2008, respectively. There are about 1,700 queries in MQ2007 and about 800 queries in MQ2008 with labeled documents.

Each query-document pair in LETOR is represented by standard features, including the traditional information retrieval features, such as tf $\times$ idf, document title length as well as some link-based features, e.g., PageRank (Brin and Page 1998) and HostRank (Xue et al. 2005). Each of these data sets generates five fold partitions for cross validation, with about the same number of queries. In each fold, there are three subsets for learning: training set (3/5), validation set (1/5, for model selection) and testing set (1/5). All experimental results reported are those averaged over the five trails.

4.2 Evaluation measures

Two popular information retrieval measures: MAP (Baeza-Yates et al. 1999) and NDCG (Järvelin and Kekäläinen 2000) are used for evaluation in our experiments.

4.2.1 Mean average precision

MAP is a measure on precision of ranking results. It is assumed that there are two relevance levels for each item: *relevant* and *irrelevant*. Given a ranking list π, precision at k measures the accuracy of top k results according to the ground truth labels

$$P@k = \frac{1}{k}\sum_{i=1}^{k} y_{\pi^{-1}}(i) \ , \tag{10}$$

where $\pi^{-1}(i)$ represents the item ranked at the position i in the list π, and $y_{\pi^{-1}}(i)$ denotes the item's ground truth label. The average precision (AP) of the ranking list π is calculated based on the precision at k

$$AP = \frac{1}{m_r}\sum_{k=1}^{n} y_{\pi^{-1}(k)} P@k \ , \tag{11}$$

where n is the total number of items in the ranking list, m_r denotes the number of relevant items. The measure MAP is defined as the mean value of AP over all the test queries. The relevance degrees of documents with respect to the queries in the data sets of MQ2007, MQ2008 and OHSUMED, are judged by human on three levels: definitely relevant, partially relevant, or irrelevant. We define 'definitely relevant' and 'partially relevant' as *relevant* for MAP calculation.

4.2.2 Normalized discounted cumulative gain

NDCG is designed to evaluate ranking quality in multiple level graded ranking applications (Busa-Fekete et al. 2012). The higher NDCG is, the better the ranking quality is. Given a ranking list π, DCG at position k can be computed via

$$DCG@k = \sum_{i=1}^{k} \frac{2^{y_{\pi^{-1}(i)}} - 1}{\log(i+1)}, \tag{12}$$

where the numerator denotes a gain, and the denominator serves as a discount. To balance the influence of individual queries, the acknowledged measure NDCG is used. It divides the DCG by the ideal discounted cumulative gain, which is the maximum possible DCG until position k, and it is defined as

$$NDCG@k = \frac{DCG@k}{IDCG@k}. \tag{13}$$

4.3 Experimental results

Depending on which DEA variant (CCR-I or CCR-O) is used to construct the weak rankers, we denote DEARank as DEARank-I or DEARank-O. Furthermore, during the weak learning process, both MAP and NDCG can be used as evaluation measure to select weak rankers. Correspondingly, DEARank-I or DEARank-O is given the suffix MAP or NDCG. For convenience, we denote DEARank-I-MAP, DEARank-I-NDCG, DEARank-O-MAP, DEARank-O-NDCG in abbreviated forms: **DIM, DIN, DOM, and DON**, respectively. Both DIN and DON use NDCG@5 for weak learning. The number of iterations in DEARank is fixed at 200, we select models on the validation set basing on the mean value of MAP and NDCG@1. The implementation of DEARank has two components: the computation of DEA models, and the learning procedure of boosting weak models to a stronger one. The transformation function $\varphi(x) = \ln(1+x)$ is used for the CCR-O model. The source codes for DEARank algorithm are available at the Google code repository (https://code.google.com/p/l2r/).

We compare the ranking performance of DEARank with twelve learning to rank baselines,[1] including two pointwise algorithms (Linear Regression and Ridge Regression), five pairwise algorithms (RankSVM, RankSVM-Primal (Chapelle and Keerthi 2010), RankSVM-Struct (Joachims 2006), RankBoost and FRank (Tsai et al. 2007)), and five listwise algorithms (SVM-MAP (Yue et al. 2007), AdaRank-MAP, AdaRank-NDCG, ListNet and SmoothRank (Chapelle and Wu 2010)). For ease of reference, they are denoted in order as **LR, RR, RS, RSP, RSS, RB, FR, SM, ARM, ARN, LN** and **SR**, respectively. Except FRank and RankBoost, all methods are linear ranking functions. At the LETOR website, only five reported baseline algorithms (including RB, RSS, LN, ARM and ARN) are available for LETOR 4.0.

We conduct experiments on all data sets in LETOR collections, and report the comparison results in Tables 2, 3, 4, and 5. According to the experimental results, several observations can be made: (1) Different algorithms perform differently on different data sets and no algorithm can always give the best performance on all data sets, with respect to different measures. For instance, RankBoost outperforms all the other methods on TD2004, but it tends to be nearly the worst one on NP2004. (2) It's generally believed that the listwise learning to rank approach is superior to the pairwise and pointwise approaches. However, the results in Table 3 seems do not support this viewpoint, and the simplistic pointwise Ridge Regression can also yield impressive results on NP2003. (3) DEARank learns the boosted ranking model using weak ranker candidates, which are created from two DEA variants: CCR-I or CCR-O. Compared with CCR-O model, CCR-I utilizes less information about the training set, since it does not take into account the human-elicited labels of the documents.

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

Table 2 Ranking performances (data bolded and italicized indicate the best results, and data bolded indicate the second best results w.r.t a particular measure) on HP2003 and HP2004

Method	N@1	N@2	N@3	N@4	N@5	N@6	N@7	N@8	N@9	N@10	MAP
HP2003											
LR	0.420	0.470	0.510	0.535	0.546	0.551	0.569	0.571	0.580	0.594	0.497
RR	0.693	0.787	0.809	0.815	0.814	0.815	0.815	0.816	0.816	0.822	0.749
RS	0.693	0.747	0.775	0.788	0.795	0.802	0.805	0.807	0.807	0.808	0.741
RSP	0.740	0.767	0.791	0.802	0.808	0.813	0.816	0.816	0.817	0.818	0.765
RSS	0.740	0.770	0.791	0.800	0.807	0.810	0.812	0.816	0.816	0.816	0.763
RB	0.667	0.770	0.792	0.793	0.803	0.807	0.814	0.816	0.817	0.817	0.733
FR	0.653	0.730	0.743	0.763	0.778	0.788	0.789	0.789	0.791	0.797	0.710
SM	0.713	0.757	0.779	0.790	0.792	0.797	0.799	0.799	0.799	0.799	0.742
ARM	0.733	***0.800***	0.805	0.816	0.825	0.828	0.832	0.835	0.836	**0.838**	0.771
ARN	0.713	0.767	0.790	0.795	0.801	0.804	0.805	0.805	0.805	0.806	0.748
LN	0.720	**0.797**	***0.813***	***0.824***	***0.830***	**0.832**	**0.835**	0.835	0.836	0.837	0.766
SR	0.713	0.780	0.808	**0.821**	**0.829**	0.831	0.831	0.831	0.831	0.833	0.763
DIM	***0.753***	0.793	**0.810**	0.818	***0.830***	***0.834***	***0.838***	***0.841***	***0.841***	***0.843***	***0.780***
DIN	**0.747**	0.783	0.802	0.810	0.822	0.829	0.834	**0.836**	**0.838**	**0.838**	**0.775**
DOM	0.733	0.763	0.790	0.799	0.813	0.819	0.823	0.823	0.826	0.827	0.762
DON	0.720	0.753	0.783	0.800	0.807	0.813	0.817	0.823	0.824	0.825	0.753
HP2004											
LR	0.387	0.540	0.575	0.607	0.613	0.629	0.629	0.635	0.643	0.647	0.526
RR	0.533	0.620	0.655	0.686	0.698	0.703	0.713	0.713	0.719	0.719	0.630
RS	0.573	0.680	0.715	0.740	0.751	0.759	0.759	0.763	0.765	0.769	0.668
RSP	0.573	0.687	0.713	0.741	0.753	0.768	0.771	0.771	0.772	0.772	0.671
RSS	0.587	0.673	0.725	0.747	0.752	0.765	0.765	0.767	0.767	0.767	0.678
RB	0.507	0.660	0.699	0.704	0.721	0.726	0.726	0.734	0.743	0.743	0.625
FR	0.600	0.707	0.729	0.737	0.749	0.755	0.755	0.755	0.758	0.762	0.682
SM	0.627	0.740	0.754	0.795	0.801	0.806	0.806	0.806	0.806	0.806	0.718
ARM	0.613	**0.773**	***0.816***	***0.825***	***0.828***	***0.833***	***0.833***	***0.833***	***0.833***	**0.833**	0.722
ARN	0.587	0.733	0.751	0.786	0.792	0.797	0.797	0.802	0.804	0.806	0.691
LN	0.600	0.687	0.721	0.762	0.769	0.780	0.785	0.785	0.785	0.785	0.690
SR	0.613	0.753	**0.796**	0.804	0.817	**0.822**	0.822	0.822	0.822	0.822	0.717
DIM	***0.667***	0.740	0.787	0.811	0.811	0.811	0.813	0.815	0.819	0.827	***0.747***
DIN	**0.640**	***0.780***	0.794	**0.819**	**0.822**	**0.822**	**0.824**	**0.824**	**0.829**	***0.837***	**0.732**
DOM	0.547	0.647	0.689	0.722	0.739	0.750	0.750	0.756	0.758	0.758	0.656
DON	0.547	0.653	0.695	0.731	0.742	0.763	0.763	0.763	0.765	0.765	0.657

Here, NDCG@m is denoted as N@m

However, on average, DEARank-I method performs better than DEARank-O method, as shown in Fig. 1. We evaluate every raw feature ranker in terms of NDCG@10 in each data set, and compute the standard variance of their performances. We discover that, DEARank-I outperforms DEARank-O with 2–10 % gains in HP2003, HP2004, NP2003 and NP2004, whose standard variances are about three times greater than those in TD2003, TD2004 and

Table 3 Ranking performances (data bolded and italicized indicate the best results, and data bolded indicate the second best results) on NP2003 and NP2004

Method	N@1	N@2	N@3	N@4	N@5	N@6	N@7	N@8	N@9	N@10	MAP
NP2003											
LR	0.447	0.557	0.614	0.635	0.642	0.653	0.660	0.660	0.663	0.666	0.564
RR	0.547	0.727	***0.773***	***0.783***	***0.789***	***0.794***	**0.799**	***0.796***	**0.799**	**0.803**	0.682
RS	0.580	0.723	0.765	0.774	0.782	0.785	0.794	0.792	0.794	0.800	0.696
RSP	0.573	0.700	0.763	0.775	0.775	0.777	0.786	0.781	0.785	0.789	0.688
RSS	0.553	0.717	0.750	0.770	0.779	0.780	0.787	0.787	0.789	0.796	0.679
RB	**0.600**	0.730	0.764	0.770	0.782	0.787	0.798	**0.795**	***0.803***	***0.807***	**0.707**
FR	0.540	0.697	0.726	0.749	0.760	0.766	0.768	0.768	0.768	0.776	0.664
SM	0.560	***0.740***	**0.767**	**0.782**	**0.788**	**0.791**	0.798	0.793	0.795	0.798	0.687
ARM	0.580	0.703	0.729	0.735	0.748	0.755	0.757	0.757	0.759	0.764	0.678
ARN	0.560	0.687	0.716	0.736	0.745	0.756	0.762	0.759	0.765	0.767	0.668
LN	0.567	0.720	0.758	0.773	0.784	0.788	***0.800***	***0.796***	0.798	0.802	0.690
SR	0.580	**0.733**	0.754	0.778	***0.789***	***0.794***	0.797	0.792	**0.799**	0.799	0.696
DIM	0.593	0.723	0.753	0.771	0.775	0.783	0.788	0.783	0.792	0.797	0.700
DIN	***0.613***	**0.733**	***0.773***	0.777	0.779	0.790	0.797	0.790	0.798	0.800	***0.714***
DOM	0.580	0.683	0.734	0.744	0.752	0.756	0.757	0.755	0.759	0.761	0.678
DON	0.567	0.683	0.725	0.735	0.744	0.749	0.753	0.754	0.756	0.756	0.669
NP2004											
LR	0.373	0.513	0.555	0.602	0.614	0.639	0.654	0.654	0.654	0.654	0.514
RR	***0.573***	**0.700**	0.735	0.775	0.777	0.790	0.800	0.800	0.804	0.804	***0.687***
RS	0.507	***0.727***	**0.750**	***0.793***	**0.796**	0.796	**0.801**	0.801	0.805	0.806	0.659
RSP	**0.560**	**0.700**	0.724	0.766	0.772	0.782	0.786	0.791	0.795	0.795	0.676
RSS	**0.560**	**0.700**	0.732	0.775	0.775	0.784	0.789	0.794	0.798	0.798	**0.677**
RB	0.427	0.553	0.627	0.643	0.651	0.667	0.681	0.685	0.685	0.691	0.564
FR	0.480	0.600	0.643	0.670	0.687	0.699	0.709	0.713	0.722	0.730	0.601
SM	0.520	**0.700**	0.749	0.785	0.787	0.795	0.799	0.804	0.804	**0.808**	0.662
ARM	0.480	0.660	0.698	0.714	0.731	0.741	0.743	0.750	0.750	0.750	0.622
ARN	0.507	0.613	0.672	0.712	0.712	0.723	0.727	0.738	0.738	0.738	0.627
LN	0.533	***0.727***	**0.759**	**0.788**	***0.797***	***0.804***	***0.808***	***0.813***	***0.813***	***0.813***	0.672
SR	0.547	**0.700**	0.744	0.780	0.783	**0.798**	***0.808***	**0.808**	**0.808**	**0.808**	0.676
DIM	0.540	0.680	0.712	0.737	0.744	0.746	0.756	0.751	0.753	0.758	0.656
DIN	0.540	0.690	0.713	0.743	0.757	0.763	0.770	0.765	0.767	0.770	0.662
DOM	0.527	0.623	0.653	0.679	0.702	0.708	0.717	0.713	0.715	0.715	0.626
DON	0.507	0.653	0.683	0.706	0.715	0.717	0.726	0.725	0.727	0.727	0.626

Here, NDCG@m is denoted as N@m

MQ2007. DEARank-O wins DEARank-I with only 0.5–4 % gains in the latter three data sets. These facts indicate that CCR-O is vulnerable to noisy features (Gomes et al. 2013) and duplicate features (Geng et al. 2007). Moreover, when the training set contains many noises, CCR-I is more suitable to formulate the relationships between features and documents. (4) It is very important for a ranking model to assign higher scores to the most relevant items. DEARank performs best in terms of NDCG@1 and MAP on two thirds of all the data sets.

Table 4 Ranking performances (data bolded and italicized indicate the best results, and data bolded indicate the second best results) on TD2003 and TD2004

Method	N@1	N@2	N@3	N@4	N@5	N@6	N@7	N@8	N@9	N@10	MAP
TD2003											
LR	0.320	0.320	0.307	0.308	0.298	0.311	0.316	0.324	0.325	0.326	0.241
RR	0.340	0.360	**0.357**	0.340	0.338	0.326	0.332	0.332	0.328	0.330	0.243
RS	0.320	0.330	0.344	0.353	0.362	0.355	0.346	0.344	0.345	0.346	0.263
RSP	0.320	**0.370**	0.355	**0.363**	***0.366***	***0.363***	***0.358***	***0.352***	***0.355***	***0.357***	0.265
RSS	0.340	0.340	0.343	0.358	**0.365**	**0.358**	**0.356**	0.348	**0.350**	0.347	0.271
RB	0.280	0.300	0.325	0.321	0.315	0.320	0.313	0.313	0.312	0.312	0.227
FR	0.300	0.280	0.267	0.257	0.247	0.254	0.254	0.255	0.265	0.269	0.203
SM	0.320	0.340	0.320	0.338	0.332	0.327	0.325	0.323	0.323	0.328	0.245
ARM	0.260	0.310	0.307	0.297	0.303	0.308	0.311	0.309	0.306	0.307	0.228
ARN	0.360	0.280	0.291	0.296	0.294	0.295	0.303	0.301	0.303	0.304	0.237
LN	0.400	0.340	0.337	0.325	0.339	0.348	0.354	0.348	**0.350**	**0.348**	0.275
SR	0.380	0.360	0.332	0.335	0.335	0.333	0.336	0.336	0.334	0.337	0.270
DIM	0.400	***0.380***	0.346	0.343	0.351	0.353	0.350	**0.350**	0.347	0.347	0.259
DIN	***0.460***	**0.370**	0.349	0.344	0.343	0.337	0.328	0.329	0.337	0.339	0.273
DOM	**0.440**	***0.380***	***0.366***	***0.368***	0.352	0.350	0.345	0.340	0.342	**0.348**	***0.284***
DON	***0.460***	**0.370**	0.344	0.345	0.335	0.327	0.325	0.324	0.340	0.341	**0.278**
TD2004											
LR	0.360	0.340	0.335	0.328	0.326	0.313	0.308	0.309	0.306	0.303	0.208
RR	0.293	0.320	0.304	0.296	0.292	0.290	0.290	0.286	0.284	0.283	0.199
RS	0.413	0.347	0.347	0.341	0.324	0.318	0.316	0.311	0.306	0.308	0.224
RSP	0.307	0.307	0.313	0.303	0.306	0.300	0.295	0.292	0.294	0.291	0.206
RSS	0.347	0.347	0.337	0.329	0.319	0.313	0.310	0.315	0.308	0.309	0.220
RB	***0.507***	***0.433***	***0.430***	***0.405***	***0.388***	***0.377***	***0.364***	***0.359***	***0.353***	***0.350***	***0.261***
FR	**0.493**	0.407	0.388	0.358	0.363	0.355	0.347	0.339	0.333	0.333	**0.239**
SM	0.293	0.307	0.304	0.300	0.301	0.294	0.298	0.293	0.291	0.291	0.205
ARM	0.413	0.393	0.376	0.368	0.360	0.353	0.344	0.336	0.334	0.329	0.219
ARN	0.427	0.380	0.369	0.352	0.351	0.338	0.330	0.328	0.321	0.316	0.194
LN	0.360	0.347	0.357	0.347	0.333	0.327	0.325	0.321	0.319	0.318	0.223
SR	0.400	0.420	0.383	0.370	0.356	0.343	0.344	0.343	0.339	0.334	0.233
DIM	0.413	0.353	0.337	0.325	0.314	0.304	0.306	0.303	0.298	0.301	0.214
DIN	0.427	0.400	0.368	0.352	0.341	0.338	0.330	0.329	0.332	0.327	0.221
DOM	0.400	0.353	0.331	0.327	0.321	0.315	0.312	0.309	0.310	0.310	0.219
DON	0.480	**0.427**	**0.390**	**0.381**	**0.364**	**0.362**	**0.351**	**0.349**	**0.349**	**0.345**	0.232

Here, NDCG@m is denoted as N@m

To give an overall rank for all the algorithms in the experiments, we use one pairwise comparison method (Liu 2011) to count their winning numbers (the number of one algorithm beats others) with respect to all measures over all data sets:

$$W_A = \sum_B \sum_E \sum_S I\{perf(A, E, S) > perf(B, E, S)\}, \tag{14}$$

Table 5 Ranking performances (data bolded and italicized indicate the best results, and data bolded indicate the second best results) on OHSUMED, MQ2007 and MQ2008

Method	N@1	N@2	N@3	N@4	N@5	N@6	N@7	N@8	N@9	N@10	MAP
OHSUMED											
LR	0.446	0.453	0.443	0.437	0.428	0.422	0.422	0.419	0.414	0.411	0.422
RR	0.536	0.478	0.474	0.460	0.457	0.454	0.445	0.444	0.443	0.444	0.444
RS	0.496	0.433	0.421	0.424	0.416	0.416	0.413	0.407	0.412	0.414	0.433
RSP	0.546	0.501	0.486	0.477	0.469	0.455	0.453	0.450	0.449	0.450	0.445
RSS	0.552	0.500	0.485	0.482	0.473	0.458	0.457	0.459	0.457	0.452	0.448
RB	0.463	0.450	0.456	0.454	0.449	0.444	0.441	0.436	0.433	0.430	0.441
FR	0.530	0.501	0.481	0.469	0.459	0.455	0.453	0.448	0.446	0.443	0.444
SM	0.523	0.491	0.466	0.463	0.452	0.441	0.434	0.433	0.430	0.432	0.445
ARM	0.539	0.479	0.468	0.472	0.461	0.458	0.456	0.451	0.446	0.443	0.449
ARN	0.533	0.492	0.479	0.469	0.467	0.460	0.460	0.458	0.454	0.450	0.450
LN	0.533	0.481	0.473	0.456	0.443	0.440	0.441	0.446	0.446	0.441	0.446
SR	0.558	0.515	0.496	0.485	0.478	0.470	0.467	0.461	0.456	0.457	0.447
DIM	0.548	0.497	0.484	0.483	0.476	0.468	0.466	0.461	0.457	0.456	0.450
DIN	0.520	**0.524**	**0.502**	***0.495***	***0.486***	***0.480***	***0.476***	**0.468**	***0.468***	***0.465***	***0.454***
DOM	***0.590***	***0.530***	***0.504***	**0.490**	**0.481**	**0.474**	**0.475**	***0.469***	**0.463**	**0.462**	**0.451**
DON	**0.568**	0.480	0.468	0.475	0.466	0.460	0.449	0.449	0.448	0.447	0.447
MQ2007											
RSS	**0.410**	**0.407**	0.406	0.408	0.414	**0.420**	0.425	0.431	0.436	**0.444**	0.464
RB	***0.413***	***0.409***	0.407	**0.412**	***0.418***	***0.423***	**0.427**	**0.432**	**0.439**	***0.446***	***0.466***
ARM	0.382	0.390	0.398	0.402	0.407	0.411	0.415	0.422	0.428	0.434	0.458
ARN	0.388	0.397	0.404	0.407	0.410	0.416	0.420	0.426	0.432	0.437	0.460
LN	0.400	0.406	**0.409**	***0.414***	**0.417**	***0.423***	***0.428***	***0.433***	**0.438**	**0.444**	**0.465**
DIM	0.393	0.397	0.405	0.410	0.415	0.418	0.422	0.427	0.433	0.439	0.461
DIN	0.400	0.406	***0.411***	0.411	0.415	0.419	0.424	0.429	0.434	0.440	0.461
DOM	0.393	0.397	0.404	0.411	0.415	**0.420**	0.424	0.429	0.434	0.440	0.462
DON	0.403	0.403	**0.409**	**0.412**	0.416	**0.420**	0.426	0.430	0.434	0.439	0.461
MQ2008											
RSS	0.363	0.398	0.429	0.451	0.470	0.485	0.491	0.456	0.224	0.228	0.470
RB	**0.386**	0.399	0.429	0.448	0.467	0.482	0.490	0.457	0.221	0.226	0.478
ARM	0.375	0.414	**0.437**	0.461	0.479	**0.492**	0.497	0.461	0.225	0.229	0.476
ARN	0.383	***0.421***	***0.442***	***0.465***	***0.482***	***0.495***	**0.499**	**0.464**	0.227	0.231	**0.482**
LN	0.375	0.411	0.432	0.457	0.475	0.489	0.498	0.463	0.227	0.230	0.478
DIM	0.385	0.410	0.432	0.460	0.479	**0.492**	**0.499**	0.462	0.227	0.231	0.480
DIN	***0.395***	**0.416**	0.436	**0.464**	**0.481**	***0.495***	***0.500***	***0.466***	***0.231***	***0.234***	***0.484***
DOM	**0.386**	0.412	**0.437**	0.458	0.477	0.490	0.496	0.460	0.226	0.229	0.479
DON	**0.386**	0.414	0.436	0.460	0.478	0.491	0.497	0.461	**0.229**	**0.232**	0.481

Here, NDCG@m is denoted as N@m

where A and B denote the index of algorithms, E denotes the index for measures (NDCG@1, …, NDCG@10 or MAP), and S is the index of data sets. $perf(A, E, S)$ represents the ranking performance of algorithm A with respect to the measure E on the data set S. Because

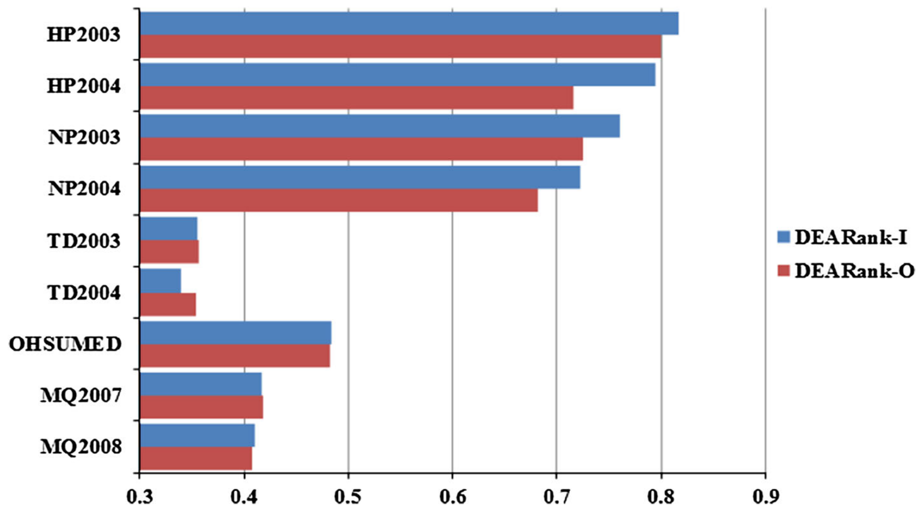

Fig. 1 Average performances of DEARank-I and DEARank-O. Each bar represents the average performance of two DEARank methods (DIM and DIN for the bars in *blue*, DOM and DON for the bars in *red*) with respect to the mean value of NDCG@1–NDCG@10 and MAP (Color figure online)

Table 6 Winning numbers on LETOR 3.0 (including HP2003, HP2004, NP2003, NP2004, TD2003, TD2004 and OHSUMED) (data bolded indicate the best result)

Method	LR	RR	RS	RSP	RSS	RB	FR	SM
W	190	1006	1034	1210	1274	900	704	1014
Method	ARM	ARN	LN	SR	DIM	DIN	DOM	DON
W	1210	900	1502	1744	1538	**1750**	1112	1056

Table 7 Winning numbers on LETOR 4.0 (including MQ2007 and MQ2008) (data bolded indicate the best result)

Method	RSS	RB	ARM	ARN	LN	DIM	DIN	DOM	DON
W	122	178	78	164	212	142	**244**	140	204

only five baselines are used for comparisons in LETOR 4.0, we present the winning numbers separately, as shown in Tables 6 and 7. From the two tables, we notice that DIN is the best method in both collections, and the second best is SmoothRank in LETOR 3.0, and ListNet in LETOR 4.0.

4.4 Experiments with a reduced pool

In DEARank, at each round, all the weak ranker candidates in the pool should be examined, and the one who performs best on the training set with the weight distribution over queries, is selected. The training process requires amount of time for weak learning. It is clearly, as the iteration proceeds, many candidates perform so badly that they have little chances of being selected. Therefore, we keep only top $K(\ll N$, the total number of documents in the training

Table 8 Performance of DEARank training with a reduced candidates' pool

	N@1	N@2	N@3	N@4	N@5	N@6	N@7	N@8	N@9	N@10	MAP
HP2003											
DIM	**0.747**	**0.800**	0.812	0.820	**0.833**	**0.837**	**0.840**	**0.840**	**0.843**	**0.844**	**0.777**
DIN	***0.753***	***0.803***	***0.817***	**0.821**	***0.835***	***0.838***	***0.845***	***0.845***	***0.849***	***0.850***	***0.785***
DOM	0.707	0.743	0.768	0.783	0.796	0.799	0.811	0.815	0.816	0.817	0.743
DON	0.700	0.743	0.776	0.795	0.801	0.807	0.814	0.821	0.822	0.823	0.740
P. C. (%)	−1.58	−0.11	−0.37	−0.29	−0.22	−0.41	−0.08	−0.04	0.04	0.04	−0.79
NP2003											
DIM	0.587	0.703	0.733	0.754	0.762	0.764	0.776	0.769	0.778	0.783	0.688
DIN	***0.613***	0.727	**0.767**	0.770	0.773	0.786	0.790	0.785	0.791	0.792	***0.711***
DOM	0.513	0.643	0.694	0.712	0.728	0.731	0.733	0.731	0.735	0.739	0.635
DON	0.527	0.667	0.713	0.731	0.740	0.746	0.748	0.745	0.750	0.752	0.648
P. C. (%)	−4.82	−2.95	−2.65	−1.95	−1.61	−1.68	−1.55	−1.70	−1.66	−1.56	−2.91
TD2003											
DIM	0.380	**0.370**	0.339	0.327	0.333	0.333	0.334	0.340	0.337	0.341	0.253
DIN	**0.420**	0.360	0.353	0.331	0.323	0.328	0.323	0.315	0.316	0.319	0.256
DOM	***0.440***	***0.380***	***0.366***	***0.368***	0.352	0.350	0.345	0.340	0.342	**0.348**	***0.284***
DON	***0.440***	**0.370**	0.339	0.347	0.335	0.327	0.324	0.323	0.338	0.338	**0.275**
P. C. (%)	−4.55	−1.33	−0.57	−1.85	−2.67	−2.06	−1.59	−1.95	−2.36	−2.14	−2.26
OHSUMED											
DIM	0.548	0.502	0.499	***0.496***	0.485	**0.476**	0.472	0.468	0.467	***0.469***	**0.459**
DIN	0.539	**0.520**	0.502	**0.491**	***0.493***	***0.485***	**0.478**	**0.473**	**0.469**	0.466	0.456
DOM	***0.596***	***0.528***	***0.509***	***0.496***	**0.491**	***0.485***	***0.483***	***0.478***	***0.470***	**0.468**	***0.462***
DON	**0.565**	0.518	**0.503**	0.488	0.476	0.468	0.461	0.464	0.460	0.460	0.456
P. C. (%)	1.00	1.76	2.78	1.43	1.85	1.70	1.42	1.94	1.65	1.88	1.75
MQ2008											
DIM	**0.388**	0.404	0.427	0.459	0.478	0.491	0.496	0.461	0.226	0.231	0.479
DIN	***0.391***	**0.416**	**0.437**	**0.463**	***0.483***	***0.495***	***0.500***	0.463	**0.229**	**0.233**	***0.483***
DOM	0.385	0.410	0.436	0.456	0.476	0.490	0.497	0.461	0.226	0.230	0.480
DON	0.386	0.414	**0.437**	0.460	0.480	**0.494**	**0.499**	***0.465***	***0.230***	***0.234***	0.481
P. C. (%)	−0.14	−0.44	−0.16	−0.22	0.10	0.16	−0.05	0.04	−0.15	0.05	−0.11

The performance changes w.r.t a measure are abbreviated as "**P. C.**" in the table, in which a positive value implies an improvement, and a negative one means the performance is deteriorated, comparing with the corresponding performance of DEARank using a complete pool (data bolded and italicized indicate the best results, and data bolded indicate the second best results)

set) candidates which perform best with respect to an IR measure (e.g., MAP, NDCG) on the whole training set.

We conduct experiments on five representative data sets: HP2003, NP2003, TD2003, OHSUMED and MQ2008, with a reduced pool of candidates. Here, we fix $K = 100$, and use the training measure (MAP for DIM and DOM, NDCG@5 for DIN and DON) to filter candidates for the training process. The experimental results on the reduced pool are reported in Table 8.

It can be seen from the results that,on average, the performance change of DEARank ranges from the worst 4.82 % loss on NP2003 with respect to NDCG@1, to the best 2.78 %

gain on OHSUMED in terms of NDCG@3. When compared with the baselines, however, the rank of our method in terms of winning number does not hurt. Taking HP2003 for instance, DIM dominates the performance in terms of NDCG@1, NDCG@5 to NDCG@10, and MAP on the complete pool. With the reduced pool of weak ranker candidates, DIN obtains the highest scores on all measures except NDCG@4, on which DIN is ranked at the second best place. After filtering the candidates, DEARank can be more efficient without much loss of performance.

5 Discussions and conclusions

The proposed DEARank shows to be a useful rank learning method. In this section, we would like to discuss the impact of one special local property of the weak candidates and a new weak weighting strategy on the ranking performance of the method.

5.1 Local overfitting and error diversity

To learn a ranking model, DEARank requires to construct plenty of weak candidates from the training set. Each weak candidate has a strong relationship with a query and its associated documents, and may perform better on its associated query over other queries. We call this phenomena as *local overfitting*, and the corresponding query and its associated documents are referred to as the *home-set*.

At the weak learning stage, the weak rankers are selected based on their average performances. If local overfitting occurs, the selection will be biased and may deteriorate the overall ranking accuracy.

To speculate whether there exists local overfitting on the used data sets, all the constructed candidates are examined. Firstly, all the candidates are used as linear rankers to sort the associated document of the queries, and measured by MAP. Secondly, the overall performance is averaged and compared with the performance on the home-set. Finally, we simply count the number of rankers that attain better performance on their home-sets than the overall average performances, and the percentages are then calculated, as shown in Fig. 2.

The two groups, one is CCR-I based, the other is CCR-O based, report the percentages of home-set biased rankers on all data sets. For example, on average, about 64.3 % of the candidates which are built upon CCR-O may produce biased ranking accuracy towards the home-set in MQ2007. With higher proportion (seven out of nine data sets), the candidates based on CCR-O may produce biased ranking accuracy. We notice that, the home-set biased ranker candidates can produce higher ranking accuracy on their home-set queries. If they do not generalize well for other queries, the local overfitting may deteriorate the ranking performance, as shown by the results on HP2003, HP2004, NP2003, NP2004, TD2003 and MQ2008 (see Fig. 1). On the other side, according to previous works (Cunningham and Carney 2000; Tsymbal et al. 2005), the base learners who have error diversity in their predictions can effectively improve the ensemble's accuracy. If the home-set biased ranker candidates can also make accurate predictions for other queries, the local overfitting may bring beneficial effects on increasing the error diversity of the ensemble model, and results in an improved overall performance, as indicated by the results on TD2004, OHSUMED and MQ2007 in Fig. 1.

To further improve the combined ranker's overall performance, we should carry out a series thorough experiments on the local overfitting phenomenon, and investigate its effect on the error diversity of the ensemble model.

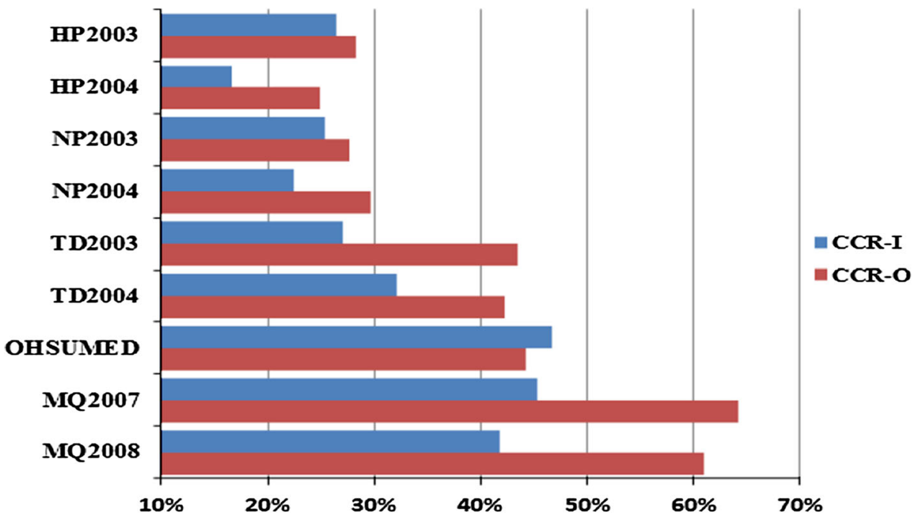

Fig. 2 Local diversity with local overfitting

5.2 Feature subset weighting strategy

In this work, we adopt the strategy (see Eqs. (8) and (9)) given in AdaRank to minimize the empirical risk on training set. In this case, all the selected features are treated equally, and the contribution from each single feature is ignored. Actually, each weak ranker can be decomposed into multiple single feature rankers, and thus each selected feature can be assigned a weight proportional to its contribution to the overall ranking performance

$$\beta_{tk} \propto \sum_{i=1}^{|Q|} E(x_i, y_i, f_k), \quad k = 1, \ldots, m, \tag{15}$$

where f_k represents the kth feature selected by the weak ranker h_t. This kind of weighting scheme deservers further critical appraisal.

5.3 Conclusions

The key contribution of this work is to introduce data envelopment analysis (DEA) into the field of learning to rank and propose DEARank algorithm. Making use of DEA's powerful potential in capturing the intrinsic characteristics of documents, we construct the weak ranker candidates using the optimal weights of features for units. The optimal weights are all solved from the DEA variant: CCR-I or CCR-O, and experimental results demonstrate that DEARank provides a promising alternative for rank learning. The incorporation of DEA into rank learning also opens up many challenges and possibilities, e.g., the computational complexity, the generalization ability, the performance of other kinds of DEA models (e.g., BCC (Banker et al. 1984), additive model (Charnes et al. 1985)), and the relationship between the local overfitting phenomenon and the error diversity. We plan to explore these issues in further detail in our future works.

Acknowledgments The authors would like to thank the anonymous reviewers for valuable comments and many constructive suggestions on improving the quality of this paper. This work was supported in part by the Program for New Century Excellent Talents in University (Grant No. NCET-10-0702).

References

Adler, N., Friedman, L., & Sinuany-Stern, Z. (2002). Review of ranking methods in the data envelopment analysis context. *European Journal of Operational Research*, *140*(2), 249–265.

Ali, A. I. (1993). Streamlined computation for data envelopment analysis. *European Journal of Operational Research*, *64*(1), 61–67.

Andersen, P., & Petersen, N. C. (1993). A procedure for ranking efficient units in data envelopment analysis. *Management Science*, *39*(10), 1261–1264.

Baeza-Yates, R., Ribeiro-Neto, B., et al. (1999). *Modern information retrieval* (Vol. 463). New York: ACM press.

Banker, R. D., Charnes, A., & Cooper, W. W. (1984). Some models for estimating technical and scale inefficiencies in data envelopment analysis. *Management Science*, *30*(9), 1078–1092.

Barr, R. S., & Durchholz, M. L. (1997). Parallel and hierarchical decomposition approaches for solving large-scale data envelopment analysis models. *Annals of Operations Research*, *73*, 339–372.

Brin, S., & Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, *30*(1), 107–117.

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

Busa-Fekete, R., Szarvas, G., Elteto, T., Kégl, B. et al. (2012). An apple-to-apple comparison of learning-to-rank algorithms in terms of normalized discounted cumulative gain. In *20th European Conference on Artificial Intelligence (ECAI 2012): Preference Learning: Problems and Applications in AI Workshop*.

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).

Cartright, M.-A., Seo, J., & Lease, M. (2009). Umass amherst and ut austin@ the TREC 2009 relevance feedback track. In *Proceedings of the 18th Retrieval Conference (TREC 2009)*. NIST.

Chapelle, O., & Keerthi, S. S. (2010). Efficient algorithms for ranking with SVMs. *Information Retrieval*, *13*(3), 201–215.

Chapelle, O., & Mingrui, W. (2010). Gradient descent optimization of smoothed information retrieval metrics. *Information Retrieval*, *13*(3), 216–235.

Charnes, A., Cooper, W. W., & Rhodes, E. (1978). Measuring the efficiency of decision making units. *European Journal of Operational Research*, *2*(6), 429–444.

Charnes, A., William, W., Cooper, B. G., Seiford, L., & Stutz, J. (1985). Foundations of data envelopment analysis for Pareto–Koopmans efficient empirical production functions. *Journal of Econometrics*, *30*(1), 91–107.

Chen, M. C. (2007). Ranking discovered rules from data mining with multiple criteria by data envelopment analysis. *Expert Systems with Applications*, *33*(4), 1110–1116.

Crammer, K., & Singer, Y. (2001). Pranking with ranking. *Advances in Neural Information Processing Systems*, *14*, 641–647.

Cunningham, P., & Carney, J.. (2000). Diversity versus quality in classification ensembles based on feature selection. In *Machine learning: Ecml 2000*. (pp. 109–116). Springer.

Emel, A. B., Oral, M., Reisman, A., & Yolalan, R. (2003). A credit scoring approach for the commercial banking sector. *Socio-Economic Planning Sciences*, *37*(2), 103–123.

Emrouznejad, A., & Shale, E. (2009). A combined neural network and dea for measuring efficiency of large scale datasets. *Computers & Industrial Engineering*, *56*(1), 249–254.

Freund, Y. (1990). Boosting a weak learning algorithm by majority, vol. 90. In *Colt* (pp. 202–216).

Freund, Y., & Schapire, R. (1995). A desicion-theoretic generalization of on-line learning and an application to boosting. In *Computational learning theory*. (pp. 23–37). Springer.

Freund, Y., Iyer, R., Schapire, R. E., & Singer, Y. (2003). An efficient boosting algorithm for combining preferences. *The Journal of Machine Learning Research*, *4*, 933–969.

Geng, X., Liu, T.-Y., Qin, T., & Li, H. (2007). Feature selection for ranking. In *Proceedings of the 30th annual international acm sigir conference on research and development in information retrieval* (pp. 407–414).

Gomes, G. de C. M., de Oliveira, V. C., de Almeida, J. M., & Gonçalves, M. A. (2013). Is learning to rank worth it? a statistical analysis of learning to rank methods. arXiv:1303.2277.

Hersh, W., Buckley, C., Leone, T. J., & Hickam, D. (1994). OHSUMED: An interactive retrieval evaluation and new large test collection for research. In *Sigir94* (pp. 192–201). New York: Springer.

Jahanshahloo, G. R., Hosseinzadeh Lotfi, F., Sanei, M., & Jelodar, M. Fallah. (2008). Review of ranking models in data envelopment analysis. *Applied Mathematical Sciences*, *2*(29), 1431–1448.

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).

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).

Joachims, T. (2006). Training linear svms in linear time. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 217–226).

Kao, C., & Hung, H. T. (2005). Data envelopment analysis with common weights: The compromise solution approach. *Journal of the Operational Research Society*, *56*(10), 1196–1203.

Kostrzewa, K., Okninski, A., & Radziszewski, B. (2011). *Data envelopment analysis without input or output*. Krakow, Poland: AGH University of Science and Technology Press.

Lan, Y., Liu, T.-Y., Ma, Z., & Li, H. (2009). Generalization analysis of listwise learning-to-rank algorithms. In *Proceedings of the 26th Annual International Conference on Machine Learning* (pp. 577–584).

Li, P., Burges, C. J. C., Wu, Q. (2007). Mcrank: Learning to rank using multiple classification and gradient boosting. In *Nips'07*.

Liu, T. Y. (2011). *Learning to rank for information retrieval* (Vol. 13). Berlin: Springer.

Liu, W. B., Zhang, D. Q., Meng, W., Li, X. X., & Xu, F. (2011). A study of DEA models without explicit inputs. *Omega*, *39*(5), 472–480.

Lovell, C. A., & Pastor, J. T. (1999). Radial DEA models without inputs or without outputs. *European Journal of Operational Research*, *118*(1), 46–51.

Moon, T., Smola, A., Chang, Y., & Zheng, Z. (2010). IntervalRank: Isotonic regression with listwise and pairwise constraints. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining* (pp. 151–160).

Niu, S., Guo, J., Lan, Y., & Cheng, X. (2012). Top-k learning to rank: Labeling, ranking and evaluation. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 751–760).

Ntoulas, A., Najork, M., Manasse, M., & Fetterly, D. (2006). Detecting spam web pages through content analysis. In *Proceedings of the 15th International Conference on World Wide Web* (pp. 83–92).

Pendharkar, P. C. (2011). A hybrid radial basis function and data envelopment analysis neural network for classification. *Computers & Operations Research*, *38*(1), 256–266.

Po, R. W., Guh, Y. Y., & Yang, M. S. (2009). A new clustering approach using data envelopment analysis. *European Journal of Operational Research*, *199*(1), 276–284.

Qin, T., Liu, T. Y., Xu, J., & Li, H. (2010). Letor: A benchmark collection for research on learning to rank for information retrieval. *Information Retrieval*, *13*(4), 346–374.

Qin, T., Zhang, X.-D., Tsai, M.-F., Wang, D.-S., Liu, T.-Y., & Li, H. (2008). Query-level loss functions for information retrieval. *Information Processing & Management*, *44*(2), 838–855.

Schapire, R. E. (1990). The strength of weak learnability. *Machine Learning*, *5*(2), 197–227.

Seiford, L. M., & Thrall, R. M. (1990). Recent developments in DEA: The mathematical programming approach to frontier analysis. *Journal of Econometrics*, *46*(1), 7–38.

Seiford, L. M., & Zhu, J. (1998). An acceptance system decision rule with data envelopment analysis. *Computers & Operations Research*, *25*(4), 329–332.

Sexton, T. R., Silkman, R. H., & Hogan, A. J. (1986). Data envelopment analysis: Critique and extensions. *New Directions for Program Evaluation*, *1986*(32), 73–105.

Smola, A. J. (2000). *Advances in large margin classifiers*. Cambridge, MA: MIT Press.

Toloo, M., Sohrabi, B., & Nalchigar, S. (2009). A new method for ranking discovered rules from data mining by DEA. *Expert Systems with Applications*, *36*(4), 8503–8508.

Troutt, M. D., Rai, A., & Zhang, A. (1996). The potential use of DEA for credit applicant acceptance systems. *Computers & Operations Research*, *23*(4), 405–408.

Tsai, M.-F., Liu, T.-Y., Qin, T., Chen, H.-H., & Ma, W.-Y. (2007). Frank: A ranking method with fidelity loss. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 383–390).

Tsymbal, A., Pechenizkiy, M., & Cunningham, P. (2005). Diversity in search strategies for ensemble feature selection. *Information Fusion*, *6*(1), 83–98.

Wang, L., Lin, J. J., & Metzler, D. (2011). A cascade ranking model for efficient ranked retrieval, vol. 11. In *SIGIR* (pp. 105–114).

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).

Xu, J., Liu, T.-Y., Lu, M., Li, H., & Ma, W.-Y. (2008). Directly optimizing evaluation measures in learning to rank. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 107–114).

Xu, X., & Wang, Y. (2009). Financial failure prediction using efficiency as a predictor. *Expert Systems with Applications*, *36*(1), 366–373.

Xue, G.-R., Yang, Q., Zeng, H.-J., Yu, Y., & Chen, Z. (2005). Exploiting the hierarchical structure for link analysis. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 186–193).

Yan, H., & Wei, Q. (2011). Data envelopment analysis classification machine. *Information Sciences*, *181*(22), 5029–5041.

Yue, Y., Finley, T., Radlinski, F., & Joachims, T. (2007). A support vector method for optimizing average precision. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 271–278).

Zhu, J. (2003). Imprecise data envelopment analysis (idea): A review and improvement with an application. *European Journal of Operational Research*, *144*(3), 513–529.