Direct Optimization of Evaluation Measures in Learning to Rank using Particle Swarm

Juan M. Fernández-Luna, Juan F. Huete
Departamento de Ciencias de la Computación e I.A.
E.T.S.I. Informática y de Telecomunicación. CITIC-UGR
Universidad de Granada
Granada, Spain
{jmfluna,jhg}@decsai.ugr.es

Óscar Alejo
Informatic Faculty, University of Cienfuegos
Cienfuegos, Cuba
alejo@ucf.edu.cu

Ramiro Pérez-Vázquez
Center of Informatic Studies, Central University Las Villas
Santa Clara, Cuba
rperez@uclv.edu.cu

Abstract— **One of the central issues in Learning to Rank (L2R) for Information Retrieval is to develop algorithms that construct ranking models by directly optimizing evaluation measures used in IR such as Precision at *n*, Mean Average Precision and Normalized Discounted Cumulative Gain. In this work we propose a new learning-to-rank method, referred as RankPSO. This algorithm is based on Particle Swarm Optimization. It builds a ranking model able to directly optimize evaluation measures used in Information Retrieval. To evaluate performance of RankPSO, we have compared it with other methods referenced in literature. We have carried out an experimental study using Letor OHSUMED dataset. The obtained results were analyzed statistically, demonstrating that RankPSO has significant improvement in precision compared to RankSVM, RankBoost and Regression methods; nevertheless, it does not have significant differences with AdaRank-MAP, AdaRank-NDCG, ListNet and FRank. The results show the advantages to use Particle Swarm Optimization as bio-inspired algorithm for learning to rank.**

Keywords Information Retrieval, Learning to Rank, Particle Swarm Optimization.

I. INTRODUCTION

Ranking is the central problem for many IR applications. These include document retrieval, collaborative filtering, key term extraction, definition finding, important email routing, sentiment analysis, product rating, and anti web spam, among others. Specifically, in the Document Retrieval field [1], the ranking problem consists of defining a representative order among the documents, taking into account relevant degree between each document and the user's query, obtaining the retrieval list, in which the relevant documents are in the highest positions with regard to less relevant document or irrelevant at all.

This ranking problem is considered as a standing topic of research inside the branches of Artificial Intelligence and IR, the *Learning to Rank* (L2R) problem. Many methods of L2R have been proposed (i.e. [2], [3], [4], [5]).

Most of the existing methods used for text retrieval are designed to optimize loss functions loosely related to the IR performance measures. Ideally, a learning algorithm should train a ranking model that optimizes directly evaluation measures according to training data.

Recently, direct optimization of performance measures in learning has become a hot research topic. Several methods for classification [4] and ranking [3] have been proposed.

In fact, in this work a new method of L2R, named RankPSO, is introduced. It can directly optimize any evaluation measures used in IR. This method is based on Particle Swarm Optimization (PSO) [6]. The main contribution of this paper is the application of such optimization technique to the problem of L2R (to the best of our knowledge, it has not been applied to this problem), obtaining good results compared with other approaches.

In order to present the algorithm itself, as well as the experimentation to evaluate its performance, the rest of the paper is organized as follows. In the Section II, it is presented a brief description of the problem of L2R. The PSO Classic algorithm is detailed in Section III in order to contextualize our approach. In the Section IV, the RankPSO algorithm is formally described. Section V is in charge of introducing the settings of the experimental study that allows evaluating the performance of the new proposed algorithm, as well as the results obtained, comparing them statistically with those from state-of-the-arte methods referenced in literature. Finally, in Section VI we conclude this work and point out some directions for future research.

II. LEARNING TO RANK

Recently, a large number of studies have been conducted on L2R and its application to IR. The aim is to automatically create a ranking model by using labeled training data and machine learning techniques. A typical setting in L2R is that

feature vectors and ranks (ordered categories) are given as training data.

Existing methods for L2R fall into three categories: the Pointwise Approach [7], which transforms ranking to classification or regression on single documents; the Pairwise Approach [8], which formalizes ranking as classification on document pairs; and the Listwise Approach [9][10], which directly minimizes a loss function defined on document lists. In this category two main alternatives are presented: Probabilistic models for ranking and direct optimization of evaluation measures.

From 1994, it has been developed various ordinal regression approaches based on machine learning techniques, which can be grouped into three categories: (1) Learning of multiple thresholds, (2) Learning of multiple classifiers and (3) Optimizing pair wise preferences.

In 2005, the methods that directly optimize the performance in terms of an IR measure have captured the interest of the scientific community. In this line, three new categories on L2R approaches can be found in the specialized literature: First, the minimization of loss functions upper bounding, considering these loss functions defined on IR measures [10][3]. Second, the approximation of IR measures by means of an easy-to-handle function [11]. Third, and finally, specially designed technologies for optimizing non-smooth IR measures. In this category, the researchers have worked on three main subcategories: (1) Smooth Approaches [12], (2) Smooth Approaches using Genetic Programming [5] and (3) Smooth Approaches for descending gradient [13].

III. CLASSIC PSO

Particle Swarm Optimization (PSO) is a population-based stochastic optimization technique developed by Russell C. Eberhart and James Kennedy in 1995 [6], inspired by social behavior of bird flocking or fish schooling.

Basically, individuals in PSO are named particles. Each particle *i* is composed of a position vector, x_i, (coordinates in the searching space), a vector of velocity, v_i, which defines the displacement of that position, and a memory of the best solution found by the particle p_i. Particularly, the velocity of any particle is determined as p_i as well as g_{best}, the global memory of the swarm δ (for example, the best among the particles). There are models that utilize the best solution of one specific neighborhood that is considered only one part of the swarm. Those models are known as l_{best}. The global approach is considered in our proposal.

PSO equations to update the velocity and position of each particle *i* are the following:

$$v_i^{(t+1)} = v_i^{(t)} + c_1 n_1 o(p_i - x_i^{(t)}) + c_2 n_2 o(g_{best} - x_i^{(t)}) \quad (1)$$

$$x_i^{(t+1)} = x_i^{(t)} + v_i^{(t+1)}, \quad (2)$$

where n_1 and n_2 are unidimensional vectors formed at random numbers in [0,1]. On the other hand, c_1 and c_2 are real numbers known as coefficients of acceleration. The operator "*o*" specifies a Hadamard product among the matrixes formed by coordinates of vectors, that is, element by element.

The expression of velocity encloses itself the principal PSO contribution that permits it to be classified as a paradigm of intelligence with swarm [14].

IV. DESCRIPTION OF RANKPSO APPROACH

A. General L2R Framework and Notation

Let $\boldsymbol{Y} = \{r_1, r_2, \ldots, r_k\}$ the set of ranks, where *k* denotes the number of ranks. There exists a total order between the them, i.e. $r_k > r_{k-1} > \ldots > r_1$, where > is the order relationship. Suppose that $\boldsymbol{Q} = \{q_1, q_2, \ldots, q_m\}$ is the set of queries in the training set. Each query q_i is represented by a list of terms $\{t_1, t_2, \ldots, t_{h(qi)}\}$, where $h(q_i)$ is the number of them in the i^{th} query, and it is associated to a list of retrieved documents $\mathbf{d}_i = \{d_{i1}, d_{i2}, \ldots, d_{in(qi)}\}$ and a list of labels $\mathbf{y}_i = \{y_{i1}, y_{i2}, \ldots, y_{in(qi)}\}$, where $n(q_i)$ denotes the sizes of lists $\mathbf{d}_i$ and $\mathbf{y}_i$, $\mathbf{d}_i \subseteq \boldsymbol{D}$ (the set of all rankings for all the queries in $\boldsymbol{Q}$) and $\mathbf{y}_i \subseteq \boldsymbol{Y}$ for the query $q_i \in \boldsymbol{Q}$. $d_{ij} \in \mathbf{d}_i$ denotes the j^{th} document in $\mathbf{d}_i$, and $y_{ij} \in \mathbf{y}_i$ is the label of document d_{ij}. A feature vector $\phi(q_i,d_{ij})$ is created from each query-document pair (q_i, d_{ij}), $i=1, 2, \ldots, m$; $j=1, 2, \ldots, n(q_i)$. Finally, the training set is noted as $S = \{(q_i, d_i, y_i)\}_{i=1}^{m}$.

Considering the patterns of the described formulation in [3], it is supposed that π_i is the prediction made by the ranking model on $\mathbf{d}_i$ in terms of the query q_i. We use Π_i to denote the set of all possible predictions on $\mathbf{d}_i$, and use $\pi_i(j)$ to denote the position of item *j* (i.e. d_{ij}). The ranking process would concentrate on obtaining a prediction $\pi_i \in \Pi_i$ for the given query q_i and the associated list of documents $\mathbf{d}_i$ using the ranking model.

This ranking model, *f*, is a real-valued function of features, more specifically, a document level function, which is a linear combination of the features in a feature vector $\phi(q_i,d_{ij})$:

$$f(q_i,d_{ij}) = w^T \phi(q_i,d_{ij}), \quad (3)$$

where *w* denotes the weight vector. In the ranking of query q_i, we assign a score to each of the documents using $f(q_i,d_{ij})$ and sort out the documents based on their scores. We then obtain a prediction π_i.

In this context of L2R, evaluation measures are used to measure the goodness of a ranking model, which are usually query-based. By this term, we mean that the measure is defined on a ranking list of documents with respect to the query. These include Mean Average Precision (MAP) [15], Normalized Discounted Cumulative Gain (NDCG) [16] and Precision at n (P@n) [15] (see Section V.B).

In this research, a general function $E(\pi_i, \mathbf{y}_i) \in [0,1]$ is used to represent the evaluation measures. The first argument of *E* is the prediction π_i created using the ranking model. The second argument is the list of ranks $\mathbf{y}_i$ given as ground truth. *E* measures the agreement between π_i e $\mathbf{y}_i$. Most evaluation measures return real values in [0,1]. We note the ideal prediction as π_i^*. Note that there may be more than one ideal prediction for a query, and we use Π_i^* to note the set of all

possible ideal predictions for query q_i. For $\pi_i^* \in \Pi_i^*$, we have $E(\pi_i^*, \mathbf{y}_i) = 1$.

Ideally, we would create a ranking model that maximize the accuracy in terms of an IR measure on training data, or equivalently, minimizes the loss function [3] defined as follows:

$$R(f) = \sum_{i=1}^{m} (E(\pi_i^*, y_i) - E(\pi_i, y_i)) = \sum_{i=1}^{m} (1 - E(\pi_i, y_i)), \qquad (4)$$

where π_i is the prediction determined for query q_i by ranking model *f*.

B. Algorithm

In this section, a new method of L2R for IR is formally described. This method is based on PSO and is able to optimize any evaluation measure used in IR. As mentioned before, the algorithm is named RankPSO and it is shown in Figure 1.

RankPSO takes as input a training set *S*, a performance measure *E* and a number of iterations *T*. First, RankPSO creates a specific swarm of particles and initiates each of them randomly; always updating the position vector g_{best} in each iteration. Then, the swarm begins its evolution stage. *T* rounds are executed in which the particles propagated in σ dimension, with the purpose of finding the best position vector g_{best} obtained in the searching space.

Finally, the ranking model *f* is built with the position vector g_{best} obtained in the last round. Equation (4) represents the fitness function used to evaluate the position of each particle. This fitness function uses the performance measure *E* and the prediction π_i obtained from the application in *S* of the expression (3).

```
1:  Input: S = {(q_i, d_i, y_i)}_{i=1}^m, E and T
2:  for each particle i do
3:    Randomly initialize v_i, x_i = p_i
4:    Update g_best
5:  end for each
6:  for t = 1,...,T
7:    for each particle i do
8:      Update i with expressions (5) and (2)
9:      Evaluate x_i on S, with expressions (3) and (4)
10:     Update p_i
11:     Update g_best
12:   end for each
13: end for
14: Build the ranking model f with the position vector g_best
15: Output: f
```

Figure 1. The RankPSO Algorithm.

The σ value is determined by the number of features considered in the ranking models.

To update the velocity of the particle the classic expression (1) is considered, adding an inertial weight *w* proposed by Eberhart and Shi in [17]. The expression would be the following:

$$v_i^{(t+1)} = w\, v_i^{(t)} + c_1\, n_1\, o(p_i - x_i^{(t)}) + c_2\, n_2\, o(g_{best} - x_i^{(t)}) \qquad (5)$$

A proposal is to use $w < 1.0$, to ensure a decrease of the velocity with the time that is impossible if $w > 1.0$. In [17], it is recommended its usage in such a way that decreases with the time from 0.9 to 0.4.

In that way, the algorithm of learning is able to construct a ranking model optimizing directly one of the evaluation measures used in IR.

V. Experimentation and Evaluation

In this section we present the experimental results of the evaluation stage of our proposal in terms of effectiveness on OHSUMED collection, a standard collection to test L2R algorithms. This analysis is based on direct comparison with the main methods that conforms the state-of-the-art in this field. As measures for evaluation, we have used also the standard MAP, P@n and NDCG at the positions of 1 to 10.

A. Letor OHSUMED dataset

The OHSUMED dataset is a subset of MEDLINE, which is a database on medical publications. There are 106 queries in the collection. For each query, there are a number of associated documents. The relevance degrees of documents with respect to the queries are judged by humans, on three levels: *definitely relevant*, *partially relevant*, or *not relevant*. There are 16,140 query-document pairs with relevance labels, and 45 extracted features. We extracted 4 features from each query-document pair, also standard in the literature. Table 1 gives a list of the features.

TABLE I. Features used in the experiments on OHSUMED

ID	Feature Description
2	$\sum_{q_i \in q \cap d} \log(c(q_i, d) + 1)$ *in´title´*
4	$\sum_{q_i \in q \cap d} \log(\frac{c(q_i, d)}{\lvert d \rvert} + 1)$ *in´title´*
8	$\sum_{q_i \in q \cap d} \log(\frac{c(q_i, d)}{\lvert d \rvert} \cdot \log(\frac{\lvert C \rvert}{df(q_i)}) + 1)$ *in´title´*
3	LMIR with DIR smoothing 'title + abstract'

c(qi,d) being the frequency of the query term q_i in document d, C the collection, df(·) the frequency of a term in a document, and |·| the cardinality of the corresponding set.

B. Evaluation Criteria

In this evaluation, we have used P@n, MAP and NDCG as performance measures because they are widely used in IR. Their definitions are as follows:

Precision at position *n* (P@n). Precision at *n* measures the relevance of the top *n* documents in the resulting ranking with respect to a given query:

$$P@n = \frac{\#relevant\ docs\ in\ top\ n\ results}{n}.$$

For example, if the top 10 documents returned for a query are {relevant, irrelevant, irrelevant, relevant, relevant, relevant, irrelevant, irrelevant, relevant, relevant}, then P@1 to P@10 values will be {1, 1/2, 1/3, 2/4, 3/5, 4/6, 4/7, 4/8, 5/9, 6/10} respectively [15].

Mean Average Precision (MAP). For a single query, average precision is defined as the average of the P@n values for all relevant documents:

$$AP = \frac{\sum_{n=1}^{N}(P@n * rel(n))}{\#total\ relevant\ docs\ for\ this\ query},$$

where N is the number of retrieved documents, and *rel(n)* is a binary function on the relevance of the n-th document:

$$rel(n) = \begin{cases} 1, \text{ if } n^{th} \text{ doc is relevant} \\ 0, \text{ otherwise} \end{cases}.$$

Similar to mean P@n, over a set of queries, we get MAP by averaging the AP values of all the queries [15].

Normalized Discounted Cumulative Gain (NDCG). Recently, a new evaluation measure called Normalized Discounted Cumulative Gain [16] has been proposed, which can handle multiple levels of relevance judgments. While evaluating a ranking list, NDCG follows two rules:

- Highly relevant documents are more valuable than marginally relevant documents.
- The lower ranking position of a document (of any relevance level), the lesser value for the user, because it is less likely to be examined by her.

According to the above rules, the NDCG value of a ranking list at position n is calculated as follows:

$$NDGC(n) \equiv Z_n \sum_{j=1}^{n} \frac{2^{r(j)}-1}{\log(1+j)},$$

where *r(j)* is the rating of the j-th document in the ranking list, choosing the normalization constant Z_n in such a way that the ideal list gets a NDCG score of 1.

C. Evaluation

For evaluating the performance of the proposed method and compare it with those obtained by the main approaches found in this research field, it was followed the experimental setting published in LETOR website.

For the learning or training process, 5-fold cross-validation experiments were performed. These prefixed 5-folds in OHSUMED were taken from the version "*QueryLevelNorm*". This allows making direct comparisons among published algorithms in terms of precision. More specifically, RankPSO has been compared in terms of performance with those algorithms that have got their assigned scores for each ranking function applied to each query-document published in the LETOR website (Pointwise approach: Regression; Pairwise approaches: RankSVM, RankBoost, FRank; Listwise approaches: ListNet, with loss minimization, and AdaRank, with direct optimization of IR measures).

The whole experiments were performed taking into account MAP as performance measure in the expression (4).

Fig. 2 graphically shows the obtained efficiency in the ranking on OHSUMED, considering MAP as performance measure. These results show the good performance of RankPSO. Otherwise, the Figure 3 shows the behaviour of the values of precision in the ranking for OHSUMED under NDCG@1, NDCG@5 and NDCG@10 terms. Table 2 contains the values obtained in the ranking on OHSUMED, considering P@n as performance measure, from 1 to 10 positions.

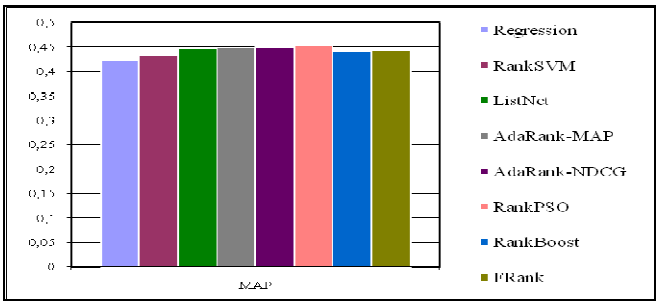

Figure 2. Performance on OHSUMED, considering the MAP measure

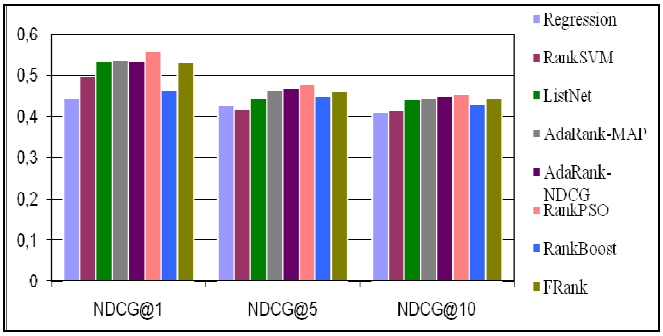

Figure 3. Performace on OHSUMED considering the NDCG measure

Algorithms	P@1	P@2	P@3	P@4	P@5
Regression	0.597	0.601	0.577	0.561	0.534
RankSVM	0.597	0.549	0.543	0.544	0.532
ListNet	0.652	0.609	**0.602**	0.575	0.550
AdaRank-MAP	0.634	0.596	0.590	0.589	0.567
AdaRank-NDCG	**0.672**	**0.624**	0.598	0.584	0.577
RankBoost	0.558	0.548	0.561	0.558	0.545
FRank	0.643	0.620	0.593	0.584	0.564
RankPSO	**0.672**	0.619	0.593	**0.593**	**0.579**

TABLE II. PERFORMANCE CONSIDERING P@1,..., P@5

Algorithm	P@6	P@7	P@8	P@9	P@10
Regression	0.505	0.500	0.484	0.475	0.467
RankSVM	0.525	0.510	0.493	0.492	0.486
ListNet	0.537	0.527	0.524	0.514	0.498
AdaRank-MAP	0.557	0.539	0.524	0.508	0.498
AdaRank-NDCG	0.556	**0.551**	**0.535**	**0.521**	**0.509**
RankBoost	0.530	0.524	0.513	0.502	0.497
FRank	0.552	0.545	0.525	0.515	0.502
RankPSO	**0.558**	0.547	0.533	**0.521**	0.506

TABLE III. PERFORMANCE CONSIDERING P@6,..., P@10

After obtaining the results of each studied algorithms, statistic tests were applied to determine the significance in the precision at query level. In this sense, non-parametric tests were applied for two related samples, applying the Wilcoxon test, even for *k* related samples, using Friedman test. In the tests analysis, the statistic significance was considered with p-value<0.05.

The obtained results for LETOR OHSUMED allow to confirm that RankPSO has significant improvement in terms of precision compared to RankSVM, RankBoost and Regression methods; nevertheless, it do not have significant differences with AdaRank-MAP, AdaRank-NDCG, ListNet y FRank.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed a new method called RankPSO for Learning to Rank. This approach is based on Particle Swarm Optimization and allows direct optimization of evaluation measures used in IR.

Analyzing the results of the experimentation and comparing the RankPSO performance with respect to the evaluation measures achieved by state-of-the-art approaches, we could conclude that RankPSO is just as good as the similar direct optimization methods.

Finally, we affirm that, for LETOR OHSUMED dataset, the methods based on direct optimization of evaluation measures can always outperform conventional methods. However, no significant difference exists among the performances of the direct optimization methods themselves. It may be that these methods hit a ceiling here (ceiling effect), so that one cannot expect to be much better without putting more problem knowledge into the algorithms.

As future work, we plan to address the following issues:

- To compare our approach with other state-of-the-art methods.
- We wish to conduct more experiments with medium and large scale datasets, to further verify the performance of RankPSO.
- To propose new learning-to-rank models based on Multi-objective Particle Swarm Optimization.
- To search for the application of new bio-inspired algorithms at L2R for IR.
- To conceive new ranking models taking into account not only the queries, the associated list of documents for these queries and relevant judgments, but also the context where the queries are formulated.

ACKNOWLEDGMENT

This work has been supported by the Spanish Ministerio de Ciencia e Innovación by means of the Project TIN2008-06566-C04-01.

REFERENCES

[1] Y. Cao , J.X., T-Y Liu , H. Li , Y. Huang, H-W Hon, "Adapting ranking SVM to document retrieval", In Proceedings of SIGIR, 2006.

[2] H. Valizadegan, R. Jin, R. Zhang, J. Mao, "Learning to Rank by Optimizing NDCG Measure", In Proceedings of NIPS, 2009.

[3] J. Xu, T.-Y.L., M. Lu, H. Li, W-Y. Ma, "Directly Optimizing Evaluation Measures in Learning to Rank", In Proceedings of SIGIR 2008.

[4] T. Joachims, "A support vector method for multivariate performance measures", In Proceedings of ICML 2005.

[5] J.-Y. Yeh, J.-Y.L., H.-R. Ke, W.-P. Yang, "Learning to rank for information retrieval using genetic programming", In Proceedings of SIGIR 2007.

[6] R. Eberhart, "A new optimizer using particle swarm theory", In Proceedings of the Sixth International Symposium on Micro Machine and Human Science MHS95, 1995.

[7] R. Nallapati, "Discriminative models for information retrieval", In Proceedings of SIGIR 2004.

[8] Y. Freund, R.I., R. Schapire, Y. Singer: "An efficient boosting algorithm for combining preferences", In Proceedings of JMLR 2003.

[9] Z. Cao, T.Q., T.-Y. Liu, M.-F. Tsai, H. Li, "Learning to rank: from pairwise approach to listwise approach", In Proceedings of ICML 2007.

[10] Y. Yue, T.F., F. Radlinski, T. Joachims, "A support vector method for optimizing average precision", In Proceedings of SIGIR 2007.

[11] M. Taylor, J.G., S. Robertson, T. Minka, "Softrank: Optimising non-smooth rank metrics", In Proceedings of SIGIR 2007.

[12] C. Burges, R.R., Q. Le: "Learning to rank with nonsmooth cost functions", In Proceedings of NIPS 2006.

[13] E. Snelson, J.G.: "Learning to Rank with SoftRank and Gaussian Processes", In Proceedings of SIGIR 2008.

[14] C. Blum, X.L.: 'Swarm intelligence in optimization', 2008.

[15] T.-Y. Liu, J.X., T. Qin, W.-Y. Xiong, H. Li: 'Letor: Benchmark dataset for research on learning to rank for information retrieval', In Proceedings of SIGIR, 2007.

[16] K. Jarvelin, "Cumulated Gain-Based Evaluation of IR Techniques", In Proceedings of ACM Transactions on Information Systems, 2002.

[17] R. Eberhart, Y.Shi., "Comparing inertia weights and constriction factors in particle swarm optimization", 2000.