Machine Learning Ranking and INEX'05

Jean-Noël Vittaut and Patrick Gallinari

Laboratoire d'Informatique de Paris 6
8, rue du Capitaine Scott, F-75015 Paris, France
{vittaut, gallinari}@poleia.lip6.fr

Abstract. We present a Machine Learning based ranking model which can automatically learn its parameters using a training set of annotated examples composed of queries and relevance judgments on a subset of the document elements. Our model improves the performance of a baseline Information Retrieval system by optimizing a ranking loss criterion and combining scores computed from doxels and from their local structural context. We analyze the performance of our algorithm on CO-Focussed and CO-Thourough tasks and compare it to the baseline model which is an adaptation of Okapi to Structured Information Retrieval.

1 Introduction

Different studies and developments have been recently carried out on ranking algorithms in the machine learning community. In the field of textual documents, they have been successfully used to combine features or preferences relations in tasks such as meta search [1] [2] [3], passage classification, automatic summarization [4] and recently for the combination of different sources of evidence in Information Retrieval (IR) [5]. One of the challenges of this paradigm is to reduce the complexity of the algorithms which is in the general case quadratic in the number of samples. This is why most real data applications of ranking are based on two-classes problems. Nevertheless, under some conditions, fast rates of convergence are achieved with this class of methods [6].

Ranking algorithms work by combining features which characterize the data elements to be ranked. In our case, these features will depend on the doxel itself and on its structural context. Ranking algorithms will learn to combine these different features in an optimal way according to a specific loss function using a set of examples. It is hoped that ranking algorithms may help to improve the performance of existing techniques.

The paper is organized as follows, in section 2 we present the ranking model, in section 3 we show how we adapted it to CO-Focussed and CO-Thorough tasks. In section 4 we comment the results reached by our model and compare it to a baseline Okapi method adapted for SIR.

2 Ranking model

We present in this section a general model of ranking which can be adapted to IR or SIR. The idea of the ranking algorithms proposed in the machine learning

community is to learn a total order on a set $\mathcal{X}$, which allows to compare any element pair in this set. Given this total order, we are able to order any subset of $\mathcal{X}$ in a ranking list. For instance in IR, $\mathcal{X}$ can be the set of couples (document, query), and the total order is the natural order on the document scores.

As for any machine learning technique, one needs a training set of labeled examples in order to learn how to rank. This training set will consist in ordered pairs of examples. This will provide a partial order on the elements of $\mathcal{X}$. The ranking algorithm will use this information to learn a total order on the elements of $\mathcal{X}$ and after that will allow to rank new elements. For plain IR, the partial ordering may be provided by human assessments on different documents for a given query.

2.1 Notations

Let $\mathcal{X}$ be a set of elements with a partial order $\prec$ defined on it. This means that some of the element pairs in $\mathcal{X}$ may be compared according to the $\prec$ relation. For Structured Information retrieval $\mathcal{X}$ will be the set of couples (doxel, query) for all doxels and queries in the document collection. This set is partially ordered according to the existing relevance judgments for each query.

2.2 Ranking

Let f be a function from $\mathcal{X}$ to the set of real numbers. We can associate a total order to f such that:

$$x \prec x' \Leftrightarrow f(x) < f(x') \ . \tag{1}$$

Clearly, learning the f function is the same as learning the total order.

An element of $\mathcal{X}$ will be represented by a real vector of features:

$$x = (x_1, x_2, ..., x_d).$$

In our case, the features will be local scores computed on different contextual elements of a doxel. In the following, f will be a linear combination of x's features:

$$f_\omega(x) = \sum_{j=1}^{d} \omega_j x_j \tag{2}$$

where $\omega = (\omega_1, \omega_2, ..., \omega_d)$ are the parameters of the combination to be learned.

Ranking loss. f_ω is said to respect $x \prec x'$ if $f_\omega(x) < f_\omega(x')$. In this case, couple (x, x') is said to be well ordered by f_ω. The ranking loss [3] measures how much f_ω respects $\prec$.

By definition, the ranking loss measures the number of mis-ordered couples in $\mathcal{X}^2$:

$$R(\mathcal{X}, \omega) = \sum_{\substack{(x,x') \in \mathcal{X}^2 \\ x \prec x'}} \chi(x, x') \tag{3}$$

where $\chi(x, x') = 1$ if $f_\omega(x) > f_\omega(x')$ and 0 otherwise.

Ranking aims at learning ω for minimizing (3).

Exponential loss. In practice, this expression is not very useful since χ is not differentiable, ranking algorithms use to optimize another loss criterion called the exponential loss:

$$R_e(\mathcal{X}, \omega) = \sum_{\substack{(x,x') \in \mathcal{X}^2 \\ x \prec x'}} e^{f_\omega(x) - f_\omega(x')}. \tag{4}$$

If is straightforward that $R(\mathcal{X}, \omega) \leq R_e(\mathcal{X}, \omega)$. (4) is differentiable and convex, and then can be minimized using standard optimization techniques. Minimizing (4) will allow to minimize $R(\mathcal{X}, \omega)$.

We can compute a gradient descent. The components of the gradient of R_e are:

$$\frac{\partial R_e}{\partial \omega_k}(\mathcal{X}, \omega) = \sum_{\substack{(x,x') \in \mathcal{X}^2 \\ x \prec x'}} (x_k - x'_k) e^{f_\omega(x) - f_\omega(x')}. \tag{5}$$

With no more hypothesis, the computation of (5) is in $O(|\mathcal{X}|^2)$.

3 Application to CO tasks

3.1 Definitions

Let denote $\mathcal{D}$ is the set of doxels for all the documents in the collection and $\mathcal{Q}$ the set of CO queries. $\mathcal{X} = \mathcal{Q} \times \mathcal{D}$ is the set of elements we want to order.

We suppose that there exists a partial order $\prec$ on $\mathcal{X} = \mathcal{Q} \times \mathcal{D}$, this partial order will reflect for some queries, the evidence we have about preferences between doxels provided via manual assessments. Note that these relevance assessments are only needed for a few queries and doxels in the collection. We consider here the task which consists in producing a ranked list of doxels which answer the query $q \in \mathcal{Q}$. For that, we will train the ranking model to learn a total strict order on $\mathcal{X}$.

3.2 Vector Representation

Each element $x \in \mathcal{X}$ is represented by a vector $(x_1, x_2, ..., x_d)$ were x_i represents some feature which could be useful to order elements of $\mathcal{X}$. Let denote $\mathcal{L}$ the set of doxel types, which are defined according to the DTD of the document collection: article, abstract, sections, paragraphs, lists...

We used the following combination:

$$f_w(x) = \omega_1^l + \omega_2^l Okapi(x) + \omega_3^l Okapi(parent(x)) + \omega_4^l Okapi(document(x))$$

where l is the node type of x and $Okapi$ is the SIR adapted Okapi model [7] described in [8]. This adaptation consists in using doxels rather than documents for computing the term frequencies, and using as normalization factor for each doxel, the mean size of the doxels with the same node type.

This combination take into account the information provided by the context of the doxel and the structural information given by the node type of the doxel.

This combination leads to the following vector representation:

$$x = \left((x_1^{l_1}, x_2^{l_1}, x_3^{l_1}, x_4^{l_1}), (x_1^{l_2}, x_2^{l_2}, x_3^{l_2}, x_4^{l_2}), ..., (x_1^{l_{|\mathcal{L}|}}, x_2^{l_{|\mathcal{L}|}}, x_3^{l_{|\mathcal{L}|}}, x_4^{l_{|\mathcal{L}|}}) \right)$$

where $|\mathcal{L}|$ is the number of different doxel types in the collection.

In the above expression all vector components of the form $(x_1^{l_i}, x_2^{l_i}, x_3^{l_i}, x_4^{l_i})$ are equal to $(0,0,0,0)$ except for one where l_i is the doxel type of x which is equal to $(1, Okapi(x), Okapi(parent(x)), Okapi(document(x)))$.

3.3 Reduction of complexity

In this section, we use some properties of SIR in order to decrease the complexity of the computation of (4) and (5).

Queries. Comparing elements from different queries has no sense. We can define a partition $\mathcal{X} = \bigcup_{q \in \mathcal{Q}} \mathcal{X}_q$, where

$$\mathcal{X}_q = \{x = (d, q') \in \mathcal{X} / q' = q\}$$

and we can rewrite (4):

$$R_e(\mathcal{X}, \omega) = \sum_{q \in \mathcal{Q}} \left\{ \sum_{\substack{(x,x') \in \mathcal{X}_q \times \mathcal{X}_q \\ x \prec x'}} e^{f_\omega(x)} e^{-f_\omega(x')} \right\}. \tag{6}$$

Assessments. For each subset $\mathcal{X}_q$, the preferences among doxels are expressed according to a several discrete dimensions. We have:

- an information of exhaustivity, which measures how much a doxel answers the totality of an information need (0 not exhaustive, ..., 3 fully exhaustive)
- an information of specificity, which measures how much a doxel answers only the information need (0 not specific, ..., 3 means fully specific)

There is no preference between doxels sharing the same value of exhaustivity and specificity.

An assessment is a couple (exhaustivity, specificity). Let denote $\mathcal{A}$ the set of assessments and $A(x)$ the assessment of element x. We can define a partition $\mathcal{X}_q = \bigcup_{a \in \mathcal{A}} \mathcal{X}_q^a$, where

$$\mathcal{X}_q^a = \{x \in \mathcal{X}_q / A(x) = a\}.$$

We can rewrite (6):

$$R_e(\mathcal{X},\omega)=\sum_{q\in\mathcal{Q}}\sum_{a\in\mathcal{A}}\left\{\left(\sum_{x\in\mathcal{X}_q^a}e^{f_\omega(x)}\right)\left(\sum_{\substack{b\in\mathcal{A}\\ \mathcal{X}_q^b\prec\mathcal{X}_q^a}}\sum_{x\in\mathcal{X}_q^b}e^{-f_\omega(x)}\right)\right\}. \tag{7}$$

where $\mathcal{X}_q^b \prec \mathcal{X}_q^a$ means that the assessments of the elements of $\mathcal{X}_q^a$ are better than those of $\mathcal{X}_q^b$. An possible order between assessments is represented in figure 1.

The complexity for computing this expression is $O(|\mathcal{Q}|.|\mathcal{X}|)$ whereas it is $O(|\mathcal{X}|^2)$ for (4).

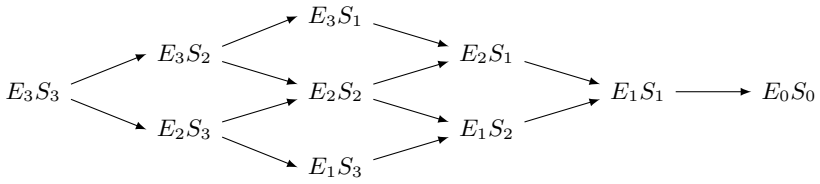

Fig. 1. Graph representing the order between elements for a given query, according to the two dimensional discrete scale of INEX. Doxels labeled E_3S_3 must be the highest ranked, and doxels labeled E_0S_0 the lowest ranked.

3.4 Gradient descent

Since (7) is convex, we can use a gradient descent technique to minimize it. The components of the gradient has the following form:

$$\begin{aligned}\frac{\partial R_e}{\partial \omega_k}(\mathcal{X},\omega)=\sum_{q\in\mathcal{Q}}\sum_{a\in\mathcal{A}}\Bigg\{&\left(\sum_{x\in\mathcal{X}_q^a}x_k e^{f_\omega(x)}\right)\left(\sum_{\substack{b\in\mathcal{A}\\ \mathcal{X}_q^b\prec\mathcal{X}_q^a}}\sum_{x\in\mathcal{X}_q^b}e^{-f_\omega(x)}\right)\\ &+\left(\sum_{x\in\mathcal{X}_q^a}e^{f_\omega(x)}\right)\left(\sum_{\substack{b\in\mathcal{A}\\ \mathcal{X}_q^b\prec\mathcal{X}_q^a}}\sum_{x\in\mathcal{X}_q^b}-x_k e^{-f_\omega(x)}\right)\Bigg\}.\end{aligned} \tag{8}$$

The complexity for computing the gradient is the same ($O(|\mathcal{Q}|.|\mathcal{X}|)$) as that of (7).

4 Experiments

4.1 Learning base

We used the series of topics and assessments from the INEX 2003 and 2004 collections as a learning base. We will comment the results on 2005 collection.

4.2 Filtering

In CO-Focussed task, overlapping doxels were not allowed. In order to suppress all overlapping elements from the lists computed by the ranking algorithm, we used a strategy which consists in removing all elements which are overlapping with an element ranked higher in the list.

As for Okapi model, we used the same strategy exept that biggest doxels like articles or bdy's were not allowed in the final ranking list to reach better performance.

4.3 Results

We comment the results obtained with the ncXG official metric with generalized quantization which is more related to the ranking loss criterion and the different levels of assessment we have used in our model.

CO-Focussed. We have plotted in figure 2 the evaluation of the lists produced by the ranking algorithm and by the modified Okapi when overlap is not authorized. We can see that the ranking algorithm performs better than Okapi. In some parts of the plot, the difference between the two models is not large: this is due to the post filtering of the lists. The ranked lists had not been optimized for non overlapping doxels since there is no notion of overlap in the exponential loss.

The table 1 shows that the ranking model is always significantly better than its baseline Okapi model, and that is quite good to retrieve the most informative doxels in the begining of the list.

Table 1. Rank of Okapi and ranking models among all participant submissions using MAncXG metric for CO-Focussed task

	@1	@2	@3	@4	@5	@10	@15	@25	@50	@100	@500	@1000	@1500
Okapi	21	20	19	19	18	18	19	19	19	18	20	20	20
Ranking	1	1	1	1	2	7	11	13	15	14	10	14	13

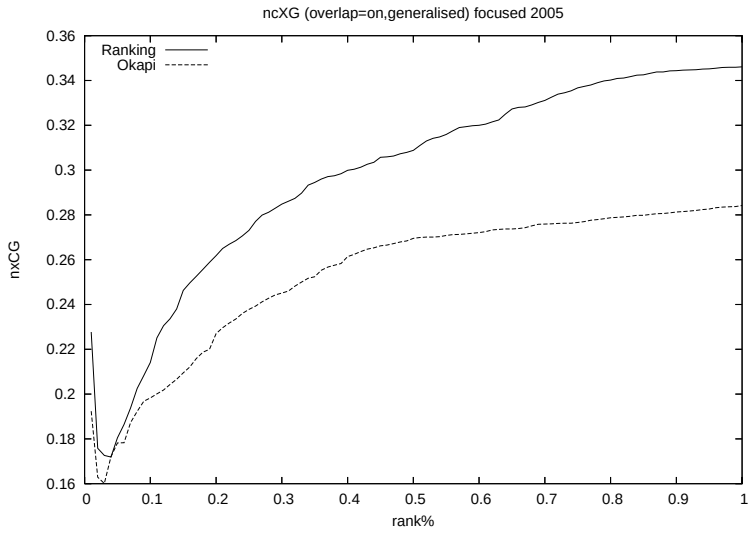

Fig. 2. Performance of ranking and Okapi models for CO-Focussed task evaluated with the cumulated gain based metric ncXG.

CO-Thorough. Figure 3 show the evaluation of the lists produced by the ranking algorithm and modified Okapi when overlap is authorized. We can see that the ranking algorithm performs clearly better than Okapi and the difference in performance is superior than in the Focussed task.

The table 2 shows that the ranking model is always significantly better than its baseline Okapi model, and that is quite good to retrieve the most informative doxels in the begining of the list. This can be explained by the expression of the ranking loss which penalize more a irrelevant doxel when it is located in the begining of the list.

Table 2. Rank of Okapi and ranking models among all participant submissions using MAncXG metric for CO-Thorough task

	@1	@2	@3	@4	@5	@10	@15	@25	@50	@100	@500	@1000	@1500
Okapi	26	22	26	26	26	31	34	37	38	38	35	32	32
Ranking	1	1	1	2	2	3	3	4	11	12	5	5	6

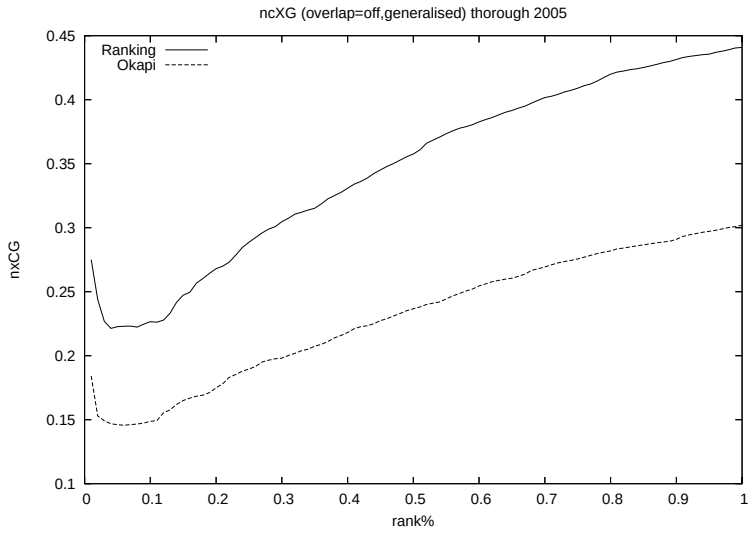

Fig. 3. Performance of ranking and Okapi models for CO-Thorough task evaluated with the cumulated gain based metric ncXG.

5 Conclusion

We have described a new model for CO tasks which relies on a combination of scores from the Okapi model and takes into account the document structure. This score combination is learned from a training set by a ranking algorithm.

For both tasks, the ranking algorithm has been able to increase by a significant amount the performance of the baseline Okapi. Ranking methods thus appear as a promising direction for improving SIR search engine performance. It remains to perform tests with additional features (for example the scores of additional IR systems).

References

1. Cohen, W.W., Schapire, R.E., Singer, Y.: Learning to order things. In Jordan, M.I., Kearns, M.J., Solla, S.A., eds.: Advances in Neural Information Processing Systems. Volume 10., The MIT Press (1998)
2. Bartell, B.T., Cottrell, G.W., Belew, R.K.: Automatic combination of multiple ranked retrieval systems. In: Research and Development in Information Retrieval. (1994) 173–181
3. Freund, Y., Iyer, R., Schapire, R.E., Singer, Y.: An efficient boosting algorithm for combining preferences. In Shavlik, J.W., ed.: Proceedings of ICML-98, 15th International Conference on Machine Learning, Madison, US, Morgan Kaufmann Publishers, San Francisco, US (1998) 170–178
4. Amini, M.R., Usunier, N., Gallinari, P.: Automatic text summarization based on word-clusters and ranking algorithms. In: ECIR. (2005) 142–156

5. Craswell, N., Robertson, S., Zaragoza, H., Taylor, M.: Relevance weighting for query independent evidence. In: SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, ACM Press (2005) 416–423
6. Auer, P., Meir, R., eds.: Learning Theory, 18th Annual Conference on Learning Theory, COLT 2005, Bertinoro, Italy, June 27-30, 2005, Proceedings. In Auer, P., Meir, R., eds.: COLT. Volume 3559 of Lecture Notes in Computer Science., Springer (2005)
7. Robertson, S.E., Walker, S., Hancock-Beaulieu, M., Gull, A., Lau, M.: Okapi at TREC. In: Text REtrieval Conference. (1992) 21–30
8. Vittaut, J.N., Piwowarski, B., Gallinari, P.: An algebra for structured queries in bayesian networks. In: Advances in XML Information Retrieval, Third International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2004, Dagstuhl Castle, Germany, December 6-8, 2004. Volume 3493 of Lecture Notes in Computer Science., Springer (2005)