Marie Curie CIG Project 322155

Efficient Effective Learning to Rank

Final Report

Summary

Ranking sits at the core of information retrieval. Given a query, a collection of documents has to be ranked based on their relevance with respect to the query. Most modern search technologies are based on machine learning algorithms that learn to rank documents given a query. This approach is commonly referred to as "learning to rank". Constructing training data for learning to rank is an expensive procedure as it requires a significant amount of human effort for labelling training examples. Hence, methods that can be used to build efficient and effective training datasets for learning to rank, or learning to rank algorithms that can be trained with fewer labelled examples are needed.

The goal of this project has been to build new methodologies that can be used to increase the efficiency and effectiveness of learning to rank. To realize this goal, the primary objectives of this project were to:

(1) construct low cost training data for learning to rank and release datasets.

(2) increase the efficiency and effectiveness of learning to rank algorithms.

(3) develop techniques for creating reusable/adaptable training data

Due to the early termination of this project, the main focus of the work performed during the duration of the project has mainly focused on one of these three objectives, increasing the efficiency and effectiveness of learning to rank algorithms.

Sampling has recently become a commonly used method for reducing judgment effort needed to create test and training collections [2, 3]. The sampling method is based on (1) creating a sampling distribution that estimates the probability of relevance of each document, and (2) sampling documents to be labelled according to this sampling distribution. The standard approach used in learning to rank is then based on using the labelled documents in the training

data during the training phase, completely ignoring the sampling distribution that was used to select the labelled documents.

The main focus of our work has been to improve the efficiency and effectiveness of learning to rank algorithms when the training data is created by the aforementioned sampling procedure. Since most document collections contains many more nonrelevant documents compared to the relevant documents, usually distributions that assign higher probability values to documents that are more likely to be relevant are used. Therefore, the documents sampled using these distributions, are not a representative sample of the complete document collection. Therefore, the sampling distribution used to select the labelled documents need to be utilised in order to train the learning to rank algorithm with respect to the entire document collection.

In this project, our main focus has been on creating learning to rank algorithms that can incorporate the sampling distribution used to select the labelled data during training. We showed that through this approach it is possible to improve the efficiency and effectiveness of the learning to rank algorithms with respect to the entire document collection.

Impact

The techniques described in this report will allow individual researchers and engineers to develop better search systems even though the amount of training data is the same. This will directly affect the search experience of end users, enabling them to reach relevant information faster. Given that the proposed methods can be used to create training datasets for a variety of different document collections/tasks, the proposed research can affect the search experience of a broad set of users ranging from patent office workers, journalists, students and academics, medical professionals and every day users of the web and social networks.

Due to the cost of constructing training data for learning to rank, most research in learning to rank and building search engines have been performed in large commercial companies. The methods presented here can help academics build search systems based on learning to rank with much less cost or fewer judgments. Similarly, the methods presented here can also help smaller search engine companies that do not have a big budget for creating large scale training datasets as well as companies that can deploy their search systems into different environments (e.g., enterprise search). This would enable academics and researchers from smaller industrial institutions to conduct research in learning to rank, leading to enhance the research done in search engine construction.

Work Performed Since the Beginning of the Project

One of the main objectives of the proposed research in the proposal "Efficient and Effective Learning to Rank" was to increase the efficiency and effectiveness of learning to rank algorithms, which has been the main focus of the research performed during the project.

In particular, we focused on the situations in which training data is limited and hence incomplete. has been created based on sampling and focusing on SoftRank [3] as the main learning to rank algorithm, we showed that significantly better test performance can be achieved when the sampling distribution is incorporated during the training phase.

The effect of incomplete judgments in learning to rank was previously investigated by He et al. [4] However, the authors do not propose a solution to the problem that can be applicable given any objective metric and any sampling distribution that was used to generate the incomplete judgments. They assume that the incomplete judgments are a random subset of complete judgments and they use previous evaluation metrics that are shown to be more robust to incomplete judgments given this setup [5]. Our proposed approach is applicable to optimize for *any* evaluation metric, for *any* sampling distribution that was used to create the training data.

Methodology

Most learning to rank algorithms in information retrieval are based on optimizing evaluation for metrics such as average precision, NDCG, etc. These evaluation metrics are assumed to evaluate user satisfaction with respect to the entire document collection that the machine learning algorithm will be tested on. However, when the training data is generated according to a distribution that is different than random, the evaluation metrics used as objectives in the learning to rank procedure do not represent the performance of the system with respect to the entire document collection. Aslam et al. [1] proposed a method that can be used to reliably estimate the value of the performance metric with respect to the entire document collection when the labelled data is generated based on sampling.

Our method is based on using the statistical estimation method by Aslam et al. [1] to estimate the values of evaluation metrics with respect to the entire document collection, and altering learning to rank algorithms to optimize for these evaluation metrics. Focusing on average precision as the evaluation metric and SoftRank as the learning to rank algorithm, below we first describe how the statistical estimation method can be used to estimate the value of average

precision with respect to the entire document collection, and we then extend how SoftRank can be extended to optimize for the estimated values via this method. Finally, we show that incorporating sampling distribution in training results in better performance than solely using the labelled documents.

We would like to note that even though we focus on average precision as the evaluation metric and SoftRank as the learning to rank algorithm, the method can be used to compute unbiased estimates of any evaluation metric that can be defined as an expectation and any learning to rank algorithm that is devised to optimize for an evaluation metric.

Statistical Estimation Method for Estimating Average Precision

Average precision is defined as the sum of precisions at relevant documents, divided by the total number of relevant documents in the collection (R). Let SP be the sum of precisions at relevant documents. Aslam et al. [1] compute the estimate of AP ($\hat{AP}$) by first computing the estimate of SP ($\hat{SP}$) and dividing this value by the estimate of R ($\hat{R}$).

Abusing the notation used by Aslam et al., let N be the total number of documents in the complete collection and let rel(i) be the binary relevance of the document at rank i such that rel(i) = 0 when document at rank i is nonrelevant and rel(i) = 1 if it is relevant. Then, SP can be written as:

$$SP = \sum_{i=1}^{N} rel(i) \cdot PC(i)$$
$$= \sum_{i=1}^{N} rel(i) \sum_{j=1}^{N} rel(j)/i = \sum_{1 \le j \le i \le N} \frac{1}{i} rel(i) rel(j)$$

The approach is based on viewing the above formula as an expectation over pairs of ranks (i, j) by viewing $rel(i) \cdot rel(j)$ as a random variable and the associated weight 1/i as the associated probability distribution (appropriately normalized so that the sum equals to 1).

Let X be a random variable corresponding to the product of relevances $rel(i) \cdot rel(j)$ and let JD be a probability distribution $JD(i, j) = \frac{1/i}{H_N}$, where $H_N = \sum_{i=1}^{N} \sum_{j=1}^{N} 1/i$ is the normalization

factor used to convert the weights into a probability distribution. Given this definition, SP can be computed as $SP = N \cdot E[X]$, where the expectation is computed with respect to the joint distribution JD. Based on this, if U is a multiset of pairs drawn according to JD, SP can be estimated as:

$$\hat{SP} = N \cdot \frac{1}{|U|} \sum_{(i,j) \in U} rel(i) \cdot rel(j)$$

Usually, the relevance judgments are generated according to some distribution that does not necessarily match the distribution necessary to estimate the desired expectation. In the case of training for example, the ranked lists returned in response to a query often change during each training epoch. Hence, during each epoch, there would be a different sampling distribution required to compute the above expectation. Ideally, given a fixed sampling distribution and the relevance judgments generated according to this sampling distribution, we would like to use this distribution to compute the above expectation regardless of the distribution required.

Hence, we would like to compute the expected value above using a sample drawn according to a different distribution than the one necessary to estimate the desired expectations. In order to do this, Aslam et al. [1] employ scaling factors that are used to correct for the differences between the actual and the required sampling distributions.

Let M(i) be the effective sampling distribution that is used to sample the individual judged documents and let JD(i; j) be the joint distribution that is required to estimate the proper expectation, where JD(i; j) is now defined over documents i and j as opposed to ranks (i.e., JD(i; j) is the joint distribution over documents i and j). Then, SP can be estimated as:

$$\hat{SP} = N \cdot \frac{1}{|U|} \sum_{(i,j) \in U} rel(i) \cdot rel(j) \cdot SF(i, j)$$

where SF(i; j) is the scaling factor that corresponds to the ratio between the required and the effective sampling distributions. Let K be the total number of sampled documents. Then, the scaling factors can be computed as:

$$SF(i, j) = \frac{JD(i, j)}{I(i, j)}$$

and I(i,j) can be computed as:

$$I(i, j) = \frac{K-1}{K} M(i)M(j) \quad \text{if } i \neq j$$

$$I(i,i) = \frac{1}{K} M(i)(1 + (K-1)M(i) + (1-M(i))^{(K-1)})$$

Based on this idea, it is easy to show that if S is the set of judged documents sampled according to distribution M, the total number of relevant documents in the collection, R can be estimated as $\hat{R} = \frac{N}{|S|} \sum_{i \in S} rel(i) \cdot SF(i)$, where $SF(i) = \frac{1/N}{M(i)}$

SoftRank for Optimizing Average Precision

Most learning to rank algorithms in information retrieval are based on optimizing evaluation metrics such as average precision, NDCG, etc. However, most information retrieval metrics tend to be non-smooth as they depend on the ranks of documents retrieved. Hence, learning to rank algorithms need to be able to handle non-smooth metrics in IR.

The main idea used in SoftRank for optimizing non-smooth IR metrics is based on defining smooth versions of information retrieval metrics by assuming that the score s_j of each document j is a value generated according to a Gaussian distribution with mean equal to s_j and shared smoothing variance σ. Based on this, Taylor et al. [3] then defined $\pi_{ij,}$ the probability that document i will beat document j and the rank distribution $p_j(r)$, the probability that document j will be at rank r. These distributions can then be used to define smooth versions of IR metrics as expectations over these rank distributions.

Given these definitions, we now extend SoftRank to optimize for average precision by defining SoftAP, the expected average precision with respect to these distributions and compute the gradient of SoftAP.

In order to define soft average precision, consider the random experiment corresponding to average precision, as defined by Yilmaz and Aslam [5]:

1. Pick a relevant document at random. Let the rank of this document be r.
2. Pick a document that is ranked at or above r, at random.
3. Output the relevance of this document.

Given the distribution π_{ij}, the probability that document i will beat document j, SoftAP can then be defined as:

$$SoftAP = \frac{1}{R}\sum_{j=1}^{N}\sum_{k=1}^{N} rel(j).\frac{rel(j) + \sum_{i=,1,i\neq j}^{N}\pi_{ij} rel(i)}{\sum \pi_{ij} + 1}$$

where R is the number of relevant documents in the query and N is the total number of documents in the collection. Using the same approach as in SoftRank and using the chain rule for obtaining derivatives, the derivative of SoftAP with respect to the score of document m ($\bar{s}_m$) can be written as

$$\frac{\partial SoftAP}{\partial \bar{s}_m} = \sum_{j=1}^{N}\sum_{k=1}^{N} \frac{\partial SoftAP}{\partial \pi_{jk}} \cdot \frac{\partial \pi_{jk}}{\partial \bar{s}_m} + \frac{\partial SoftAP}{\partial \pi_{kj}} \cdot \frac{\partial \pi_{kj}}{\partial \bar{s}_m}$$

$$\frac{\partial SoftAP}{\partial \pi_{kj}} = \frac{rel(j)}{R} \cdot \frac{rel(k)}{\sum_{i=1,i\neq j}^{N}\pi_{ij} + 1} - \frac{rel(j)}{R} \cdot \frac{rel(j) + \sum_{i=1,i\neq j}^{N}\pi_{ij} rel(i)}{\left(\sum_{i=1,i\neq j}^{N}\pi_{ij} + 1\right)^2}$$

and $\frac{\partial \pi_{kj}}{\partial \bar{s}_m}$ can be computed as described by Taylor et al. [3].

Incorporating Sampling Distribution to Learning to Rank

Above we described the method by Aslam et al. [1] that can be used to compute unbiased estimates of evaluation metrics with respect to a complete collection given a limited number of relevance judgments and the sampling distribution that was used to generate these relevance judgments. We will now show how the method can be used for training purposes using SoftRank as the learning algorithm and AP as the evaluation metric. Due to the properties of the method, the same approach is applicable for optimizing any evaluation metric given any sampling distribution.

Using the approach described above to estimate the value of average precision, given a document collection D, a set of documents S sampled from D nd the associated sampling distribution M, one can show that SoftAP can be estimated as follows:

Let $\overline{PC}_j$ be

$$\overline{PC}_j = \frac{rel(j)SF(j,j) + \sum_{i \in S, i \neq j} \pi_{ij} rel(i) SF(i,j)}{\sum_{i \in D, i \neq j} \pi_{ij} + 1}$$

Then, $\overline{SoftAP}$ can be written as:

$$\overline{SoftAP} = \frac{1}{\hat{R}} \sum_{j \in S} rel(j) \cdot PC_j$$

, where SF(i; j) can be computed as before.

Note that even though the numerator of the above formula only depends on the sampled documents, the denominator depends on all the documents in the collection D. Therefore, the unjudged documents also have an effect in the computation even though their relevance values are not used.

The gradient of $\overline{SoftAP}$ with respect to the distribution π_{kj} can then be computed as follows:

If both documents k and j are sampled, then the gradient with respect to π_{kj} is:

$$\frac{\partial \overline{SoftAP}}{\partial \pi_{kj}} = \frac{rel(j)}{\hat{R}} \left[\pi_{kj} \frac{rel(k) \cdot SF(k,j)}{\sum_{i \in D, i \neq j} \pi_{ij} + 1} - \frac{PC_j}{\sum_{i \in D, i \neq j} \pi_{ij} + 1} \right]$$

If either document k or document j is not sampled, then

$$\frac{\partial \overline{SoftAP}}{\partial \pi_{kj}} = \begin{cases} 0 & \text{if } j \notin S \\ -\frac{rel(j)}{\hat{R}} \cdot \frac{PC_j}{\sum_{i \in D, i \neq j} \pi_{ij} + 1} & \text{if } j \in S, k \notin S \end{cases}$$

and $\frac{\partial \pi_{kj}}{\partial \bar{s}_m}$ can again be computed as described by Taylor et al. [3].

Given these definitions, we can now employ SoftRank to train a ranking function over a partially judged collection (where the partial judgments are generated using a known sampling distribution), while optimizing for the estimated value of an evaluation metric with respect to the complete collection.

Experimental Results

We compare the quality of the resulting ranking function trained by SoftRank over a partially judged collection using the afore-described approach of measure estimation, with the quality of the ranking function when optimized by the mere use of the incompletely judged data (ignoring the sampling distribution).

In our experiments, we use the TREC collection that was used by Aslam et al. [6] to compare the effect of different document selection methodologies for learning to rank. The document corpus, the queries and the relevance judgments in this collection are obtained from TREC 6, 7 and 8 Adhoc retrieval track. Collectively, the dataset contains 150 queries. We split the data into five folds to apply five fold cross validation. For each fold create training sets of different sizes (ranging from 1% to 10% of the complete judgments) by sampling documents from the collection using the sampling distribution described by Aslam et al. [1].

We then employ our proposed technique for training using the partially judged collection by utilizing the estimated value of the AP with respect to the complete collection as the objective metric (referred to as Estimated AP). We then compare this with the results obtained by the standard method of training solely using the labelled documents (ignoring the sampling distribution) (referred to as Sampled Pool AP). Figure 1 illustrates how the two methods compare. It can be seen that optimizing for Estimated AP consistently outperforms optimizing for Sampled Pool AP, and the difference is statistically significant for smaller sampling percentages ($p <= 0.05$).

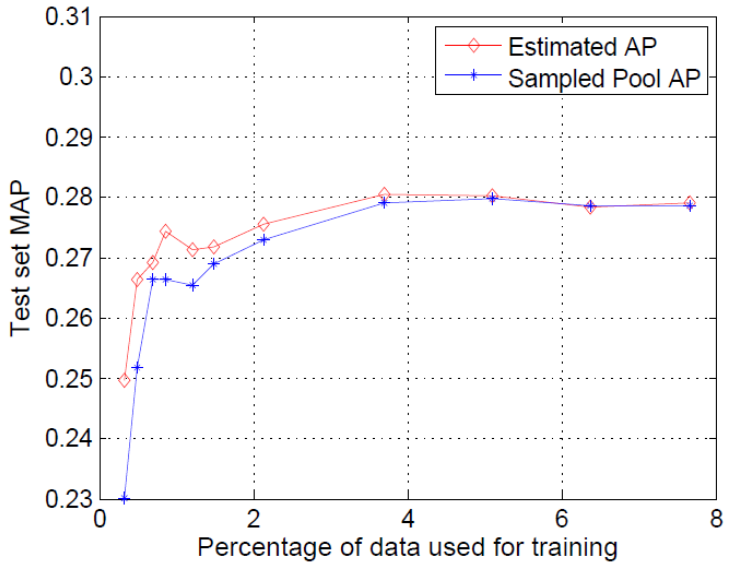

Figure 1: Test set mean average precision values when SoftRank is training using Estimated AP versus Sampled Pool AP as the objective metric.

A commonly used method when using partially labelled data for training is based on assuming unjudged documents are nonrelevant. Figure 2 shows how our proposed method compares with the performance obtained when unjudged documents are assumed to be nonrelevant. Similar to before, training with Estimated AP consistently performs better than this approach, with the improvements being significant when smaller training datasets are used.

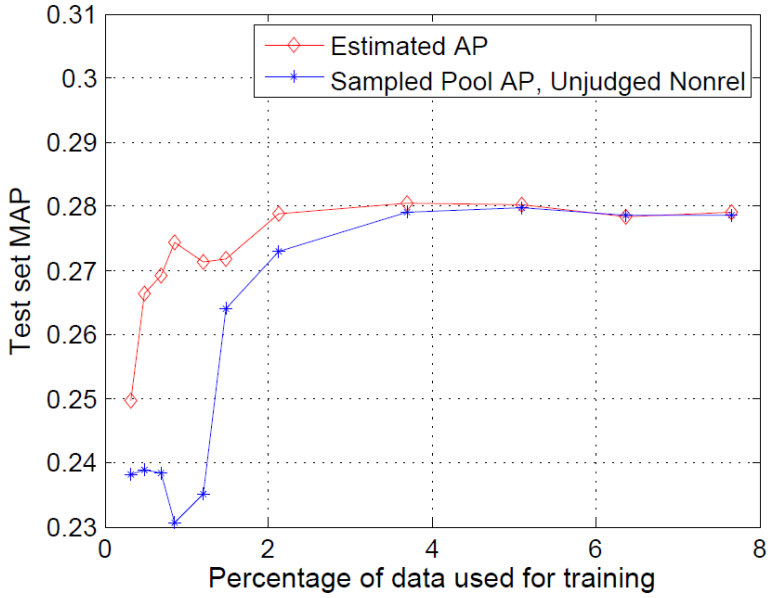

Figure 2: Test set mean average precision values when SoftRank is training using Estimated AP versus Sampled Pool AP assuming unjudged documents are nonrelevant as the objective metric

Dissemination Activities

This project had to be terminated after the first year due to Dr. Yilmaz leaving the host university. Therefore, most of the work proposed in this project is still ongoing. However, one of the objectives of the proposed research, increasing the efficiency and effectiveness of learning to rank algorithms (Objective 2) has been mostly completed. This work is currently under preparation for a submission to the ACM CIKM Conference on Information and Knowledge Management, 2014. The current version of the paper is provided in the annex of this report.

In terms of dissemination activities, during the first year of the project, Dr. Yilmaz attended the CIKM Conference on Knowledge Management and the SIGIR Conference on Research and Development in Information Retrieval. Dr. Yilmaz presented a paper in the CIKM Conference. The work described here has also been presented at a seminar at Koc University, where the students were presented with the notion of learning to rank, as well as important open problems in the area of learning to rank.

Dr. Yilmaz also presented some of the work described here as part of a course in Information Retrieval, where students from different backgrounds (from BSc, MSs, and PhD) were presented with these ideas.

Part of the work presented here has been done in collaboration with Dr. Evangelos Kanoulas from Google Research in Zurich.

References

[1] J. A. Aslam, V. Pavlu, and E. Yilmaz. A statistical method for system evaluation using incomplete judgments. In SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval, pages 541{548, New York, NY, USA, 2006. ACM.

[2] Tie-Yan Liu, Jun Xu, Tao Qin2, Wenying Xiong3, and Hang Li1. LETOR: Benchmark Dataset for Research on Learning to Rank for Information Retrieval. In SIGIR Workshop on Learning to Rank for IR (LR4IR), 2007.

[3] M. Taylor, J. Guiver, S. Robertson, and T. Minka. Softrank: optimizing non-smooth rank metrics. In WSDM '08: Proceedings of the international conference on Web search and web data mining, pages 77-86, New York, NY, USA, 2008. ACM.

[4] B. He, C. Macdonald, and I. Ounis. Retrieval sensitivity under training using different measures. In SIGIR '08: Proceedings of the 31th annual international ACM SIGIR conference on Research and development in information retrieval, pages 67-74. ACM, 2008.

[5] E. Yilmaz and J. A. Aslam. Estimating average precision with incomplete and imperfect judgments. In CIKM '06: Proceedings of the 15th ACM international conference on Information and knowledge management, pages 102-111, New York, NY, USA, 2006. ACM.

[6] J. A. Aslam, E. Kanoulas, V. Pavlu, and E. Yilmaz. Document selection methodologies for efficient and effective learning-to-rank. In SIGIR '09: Proceedings of the 32nd annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, 2009. ACM.