Variance Maximization via Noise Injection for Active Sampling in Learning to Rank

Wenbin Cai
Shanghai Jiao Tong University
Shanghai, China
cai-wenbin@sjtu.edu.cn

Ya Zhang*
Shanghai Jiao Tong University
Shanghai, China
ya_zhang@sjtu.edu.cn

ABSTRACT

Active learning for ranking, which is to selectively label the most informative examples, has been widely studied in recent years. In this paper, we propose a general active learning for ranking strategy called Variance Maximization (VM). The algorithm relies on noise injection to perturb the original unlabeled examples and generate the rank distribution of each example. Using a DCG-like gain function to measure each ranked list sampled from the rank distribution, Variance Maximization selects the unlabeled example with the largest variance in the gain. The VM strategy is applied at both the query level and the document level, and a two-stage active learning algorithm is further derived. Experimental results on both the LETOR 4.0 dataset and a real-world Web search ranking dataset have demonstrated the effectiveness of the proposed active learning approach.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: INFORMATION STORAGE AND RETRIEVAL—*Information Search and Retrieval*; I.2.6 [**Computing Methodologies**]: ARTIFICIAL INTELLIGENCE—*Learning*

General Terms

Algorithms, Experimentation, Theory

Keywords

Active Learning, Variance Maximization, Noise Injection, Learning to Rank

1. INTRODUCTION

Learning to rank is to automatically generate ranking functions through supervised learning. It has been widely applied to many information retrieval (IR) applications, such as Web search and recommendation. Like many other supervised learning tasks, training a high quality ranking function is usually at the cost of a large number of labeled data. However, in many real-world learning-to-rank applications, it is very expensive to collect a large volume of labeled training data.

To reduce the cost of labeling, active learning has been applied to ranking [1, 13, 2, 8, 10]. Compared to traditional active learning tasks, active learning for ranking is more complex due to a unique *query-document* structure. Most of the existing active learning for ranking algorithms are at either the query level or the document level [1, 13, 2, 10]. In recent year, Long *et al.* proposed a novel two-stage active learning for ranking framework [8], Expected Loss Optimization (ELO), to integrate the query level active learning and the document level active learning.

Uncertainty sampling is a popular active learning strategy in classification and regression task. Traditional uncertainty strategy selects the unlabeled example with the highest uncertainty in predicted scores. However, the predicted scores are not directly related to the orders in ranking task. In this paper, we explore how to effectively measure the uncertainty in ranking. We propose a general active learning strategy for Web search ranking called Variance Maximization (VM). The underlying motivation is that variation in ranking are highly correlated with uncertainty in ranking. Similar to SoftRank [11], we assume that each ranking score is non-deterministic and is sampled from a certain score distribution, and the score distribution can be transformed to rank distribution. The difference between the two studies lies in the fact that we leverage noise injection to generate the score distribution while the score distribution is generated by smoothing the score with equal variance Gaussian distribution in the case of [11]. Using a DCG-like gain function to measure each ranked list sampled from the rank distribution, Variance Maximization selects the unlabeled example with the largest variance in the gain. Considering the *query-document* structure in Web search ranking, we investigate VM at both the query level and the document level and extend to a two-stage active learning algorithm. Experimental results on both the LETOR 4.0 data set and a real-world Web search ranking data set have demonstrated the effectiveness of our approach.

The reminder of this paper is organized as follows: We first review the related work in Section 2. Section 3 describes the process to generate score distribution by noise injection and to transform score distribution to rank distribution. Our approach, Variance Maximization, is presented in Section

*Author for correspondence.

CIKM'12, October 29–November 2, 2012, Maui, HI, USA.

4. Section 5 discusses experiments and results. Finally, we conclude the paper in Section 6.

2. RELATED WORK

So far, various active learning strategies have been proposed. One common strategy is called *uncertainty sampling*. The uncertainty sampling selects the unlabeled example about which it is most uncertain how to label. Another typical active learning strategy is *query by committee* (QBC) algorithm [4]. The QBC algorithm generates a committee of models and selects the unlabeled data instance about which the models disagree the most. A comprehensive active learning survey can be found in [9].

Compared to traditional supervised learning setting, a unique *query-document* structure exists in learning to rank setting. Considering the structure, existing active learning for ranking may be categorized into two types: the query level active learning and the document level active learning. For the query level active learning, Yilmaz and Robertson [12] empirically showed that having more queries but shallow documents performed better than having less queries but deep documents. Cai *et al.* [1] proposed a query selection strategy by combining domain adaptation and QBC-based active learning. Yu [13] proposed a document level active learning algorithm, which treats the document pairs with similar predicted relevance scores as the most informative example. The algorithm is applied to RankSVM [6]. Another state-of-art document selection strategy was proposed in [2]. They choose the documents which are expected to change the current model mostly. The learning to rank algorithms are RankSVM and RankBoost [3]. Silva *et al.* [10] proposed a novel document level active sampling algorithm based on association rules, which does not rely on any initial training seed. Recently, Long *et al.* [8] proposed a two-stage active learning framework to integrate the query level active learning and the document level active learning. Under the Bayesian framework, the Expected Loss Optimization (ELO) principle is introduced for active learning.

3. GENERATING RANK DISTRIBUTION BY NOISE INJECTION

Similar to SoftRank [11], we assume each ranking score is non-deterministic and is sampled from a certain score distribution. We propose to approximate the score distribution by noise injection. The method first perturbs the original unlabeled example by injecting random noises and generates a set of noisy 'replicates'. The distribution of the corresponding ranking scores, predicted by the current ranking model, is treated as an approximation of the score distribution, which is then transformed to the rank distribution.

3.1 Smoothing Score by Noise Injection

Intuitively, if a data example is close to the decision boundary, a small perturbation in feature value may make it cross the decision boundary and result in changes in the corresponding predicted ranking scores. On the contrary, if a data example is far away from the decision boundary, the predicted scores under reasonable perturbation will remain consistent. Based on the above intuition, we propose to smooth the score of a data example by noise injection. Let $e \in [e_{min}, e_{max}]$ be the feature vector of an unlabeled data example. Noise injection distorts the original data example

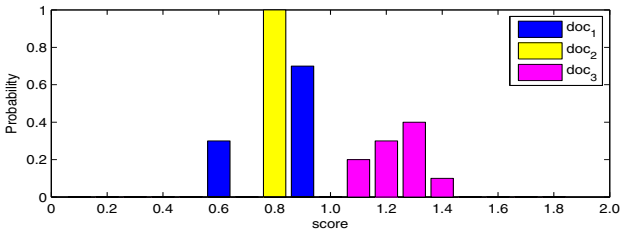

Figure 1: The score distributions of three example documents. doc_1 may be close to a decision boundary because is assigned two predicted scores after a small perturbation. Similarly, doc_2 may be far away from any decision boundary and doc_3 may be close to the intersection of multiple decision boundaries.

e by adding some random noise η to the features of e and generates m noisy data examples around the original data example. We formulate noise injection as follows:

$$e^i = e + \eta \quad (i = 1, 2, ..., m) \tag{1}$$

where $\eta \sim N(\mu, \Sigma)$.

Given a ranking function, let $[f(e^1), \cdots, f(e^m)]$ represents the predicted score vector for the data instance e after perturbation. We smooth the predicted ranking score of the original data instance e using the corresponding score vector $[f(e^1), \cdots, f(e^m)]$. Figure 1 shows the score distributions of three documents after noise injection. Under perturbation, the data example doc_1 is assigned two ranking scores with a probability of 0.3 and 0.7, respectively. This implies that doc_1 may be close to a decision boundary and hence a small perturbation leads its 'replicates' to cross the boundary. Similarly, the perturbation of doc_3 results in four predicted ranking scores, suggesting that doc_3 may be close to the intersection of multiple decision boundaries. On the contrary, doc_2 may be far away from any decision boundary because its ranking scores are very consistent under perturbation.

3.2 Generating Rank Distribution

In learning to rank, we are not directly interested in the absolute value of the predicted score, but rather the resulted ranked list ordered by the predicted ranking score. Therefore, the score distribution is transformed to the rank distribution. Considering the *query-document* structure in Web search ranking, we consider the rank distribution at two levels: the query level rank distribution and the document level rank distribution.

To generate the query level rank distribution, assuming there are n documents associated with a query, we randomly sample a score from the score distribution of each document to generate a score vector $[f(d^1), \cdots, f(d^n)]$. Then we sort the documents according to the score vector to generate a ranked list for the query. By performing the above process multiple times, we get an approximation of the rank distribution.

Given a query-document pair, to generate its document level rank distribution, the predicted ranking scores of the other documents related to the given query are fixed to be the predicted scores without noise perturbation. We then randomly sample the score distribution of the given document to generate a score vector $[f(d^1), \cdots, f(d^n)]$ and sort it to produce a ranked list for the document. Again, the

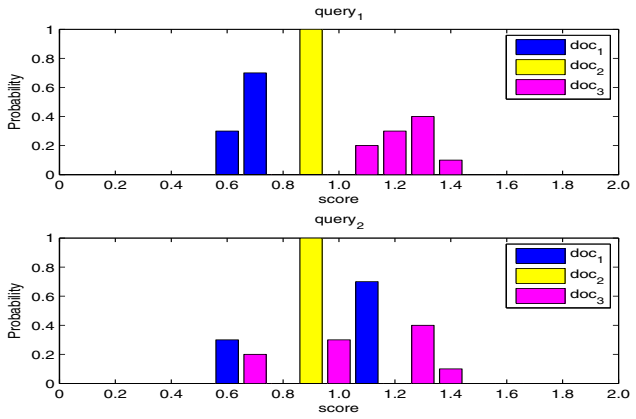

Figure 2: The score distributions of two example queries. The ranking model f is certain about $query_1$ because its ranked list is stable and is least certain with $query_2$ because there are multiple ranked lists.

sampling and sorting process is performed multiple times to generate an approximation of the rank distribution.

4. VARIANCE MAXIMIZATION

In the case of learning to rank, change in ranking scores does not necessarily leads to variation in the final ranking. Since we are mainly concerned with the ranking of the example rather than the absolute ranking scores, we compute the variance in terms of ranking rather than the variance of the predicted ranking scores. We provide the details of VM in the following section.

4.1 VM at Query Level

As mentioned above, if the ranked list of a query is stable after perturbation, it implies that the ranking model is certain about the query in ranking. Otherwise, the ranking model is uncertain about the query, and we treat it as the informative query. Figure 2 shows two example queries, either of which has three associated documents. While the two queries seem to have similar score distributions, their rank distributions turn out to be very different. The first query, $query_1$, has a stable ranked list no matter which score value is used to generate the ranked list, suggesting that the current ranking model f is certain with $query_1$. On the contrary, the query $query_2$ has multiple possible ranked lists, indicating that the ranking model f is less certain about the query $query_2$.

Given the rank distribution, VM aims to select unlabeled examples with the largest variation in ranking. Inspired by DCG function [7], we define the gain function g of the ranked list as:

$$g(list) = \sum_{r=1}^{n} (2^{s(r)} - 1)/log(1 + r) \quad (2)$$

Where $s(r)$ is the predicted ranking score of the document without noise perturbation at rank r in the list, and n is the number of documents related to the list.

We represent the variation of the ranking associated with a query as the gain variance of the query and approximate it with the gain variance of the ranked lists sampled from the corresponding rank distribution. If a query has a stable ranked list, its corresponding gain values will be consistent, and the variance in the gain will be zero. Otherwise, it will yield a large variance in the gain. Therefore, the query selection criteria can be expressed as:

$$q^* = \underset{q \in pool}{\operatorname{argmax}}\, var(g(list, q)) \quad (3)$$

Where $pool$ represents the large size unlabeled data set, and $var(g(list, q))$ denotes the variance in the gain by sampling the query level rank distribution of the query q.

4.2 VM at Document Level

The query level sampling selects all documents associated with the least certain query. However, a least certain query may still contain documents that the ranking model is certain about. Hence we aim to select only documents that the ranking model is least certain as the informative examples. Take the above example query $query_2$ (Figure 2) for instance, no matter how we sample the document level rank distribution of doc_2, the ranking of the document doc_2 stays the same. Therefore, although the ranking model is not certain about the ranking of the query $query_2$, it is certain sure about the rank of the document doc_2. Thus we may choose the documents doc_1 and doc_3 as the informative documents.

Similar to the query level active learning, we use the variance in the gain values to measure the stability of each document regarding to the document level rank distributions. The document selection criteria can be represented as:

$$d^* = \underset{d \in pool}{\operatorname{argmax}}\, var(g(list, d)) \quad (4)$$

Where $var(g(list, d))$ denotes the variance in the gain by sampling the document level rank distribution of the document d.

4.3 VM at Two-stage

The query level active learning selects all documents associated with a query. Actually, it may include some non-informative documents because there are usually a large number of documents associated with a selected query. Since the quality of a ranking model is mainly scored by the top-k documents, most of them are non-informative. The document level active learning selects documents individually. However, this sampling strategy ignores the *query-document* structure and the dependency among the documents given a query and hence may not be optimal.

To address the problem, Long *et al* [8] proposed a two-stage active learning algorithm, which first selects the most informative queries at the query level and then selects the most informative documents associated with the selected queries. We follow this two-stage active learning strategy in designing the proposed algorithm.

5. EXPERIMENT

5.1 Dataset and Experimental Setting

We use two learning to rank data sets to validate the proposed active learning algorithms. The first one is the LETOR 4.0 data set, a benchmark data set on learning to rank for information retrieval. The query-document pairs are labeled with a three-level relevance judgment: {Bad, Fair, Good}. The second dataset is the Web search data set from a commercial search engine (denoted as WEB-

Table 1: The statistics of the two data sets.

Data Set	AL set	#queries	#documents
LETOR 4.0	base set	60	2,000
	pool set	1940	66,383
	test set	297	10,262
WEB-SEARCH	base set	200	4,102
	pool set	3000	60,609
	test set	564	11,363

SEARCH). The relevance score is labeled with five-level relevance scheme: {Bad, Fair, Good, Excellent, Perfect}. All the features from both of the two datasets have been normalized. Both of the two datasets are randomly split into three parts at the query level: base set, pool set, and test set. We use the base set as the small labeled data set to train the initial base ranking models. The pool set is used as a large size unlabeled data set to select the most informative examples. The test set is used to evaluate different active learning strategies. The statistics of the two data sets are listed in Table 1.

For the base learner, we use Gradient Boosting Decision Tree (GBDT) [5] to train the ranking models. We first experiment on noise injection to determine the parameter η for Gaussian noise. Then, we compare the proposed VM algorithms with several other active learning algorithms. The algorithms select the top k informative examples. In this study, the active learning process iterates 10 rounds. In each round of active selection, 50 queries were selected at the query level and 500 documents were selected at the document level respectively. For the two-stage active learning, we simply fix the number of documents selected for each query to be 10 based on the result from [12]. We repeat each experiment for 10 times and report the average DCG at the rank 10 (DCG@10).

5.2 Noise Injection

In this section, we experiment on noise injection to empirically determine the optimal parameters for Gaussian noise. There are several important parameters in the Gaussian noise injection process: the mean μ, the covariance matrix Σ, and the number of noisy 'replicates' m, respectively. In this study, we set $\mu = \mathbf{0}$ and $\Sigma = s^2\mathbf{I}$ and empirically fix the number of noisy 'replicates' m to be 20. We experiment with four values for s: s=0.000001, s=0.00001, s=0.0001, s=0.001, denoted as s-1, s-2, s-3, and s-4, respectively.

Figure 3 shows the percentage of examples(documents) that cross the decision boundary after noise injection with different standard deviation. We observe that the percentage increases monotonically with the value of s . The results agree with the following intuition. If a data example is very close to the decision boundary, a small perturbation will make it cross the boundary. Otherwise, a larger perturbation is required. The experimental results of the VM with different values of s show that s-1 consistently outperforms the other three s-$i(i = 2, 3, 4)$ on both datasets. Based on the above experimental results, we set s to be 0.000001 for Gaussian noise injection in the rest of our experiments.

5.3 Query Level Active Learning

In this section, we compare the query level VM algorithm (denoted by VM-Q) with other two query level active learning algorithms. One is the query level ELO-DCG (denoted by ELO-DCG-Q) algorithm, representing one of state-of-art. The other is random query selection (denoted by RANODM-Q), representing a baseline.

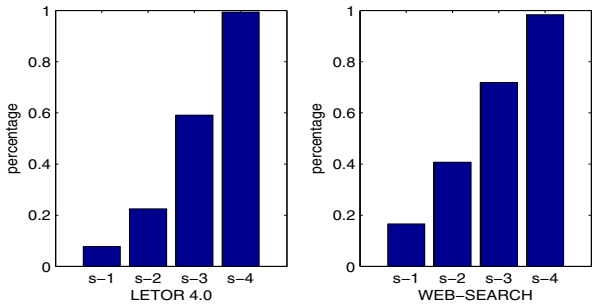

Figure 3: The percentage of examples(documents) that cross the decision boundary after perturbation, where s-$i(i = 1, ..., 4)$ denotes the standard deviation for noise injection.

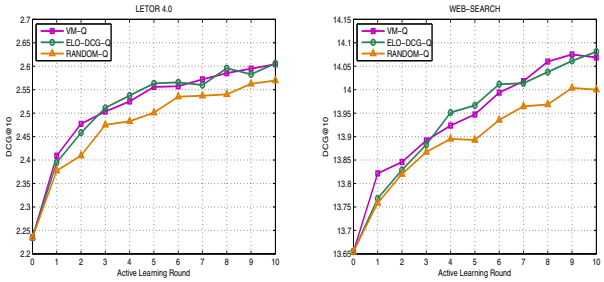

Figure 4: Comparison Results at Query Level.

Figure 4 shows the learning curves of the three query level active learning algorithms. We observe that both VM-Q and ELO-DCG-Q perform better than RANDOM-Q. The results may be based on the following explanation. The ELO-DCG-Q selects the queries with the largest expected DCG loss that is directly related to the objective function DCG@10, and the VM-Q chooses the most uncertain queries to improve the ranking model performance effectively. Furthermore, VM-Q performs as well as ELO-DCG-Q. T-test shows that VM-Q is statistically equivalent to ELO-DCG-Q and significantly better than RANDOM-Q ($p<0.05$) most of the times.

5.4 Document Level Active Learning

In this section, we show that document level VM algorithm (denoted by VM-D) effectively selects the most informative documents. We compare VM-D with the document-level ELO-DCG (denoted by ELO-DCG-D) algorithm , the traditional uncertainty sampling for regression (denoted by UNCERTAINTY-R) [9] and random document selection (denoted by RANODM-D).

The results of the four document level algorithms are plotted in Figure 5. We observe that VM-D consistently performs better than the other three methods. The performance of ELO-DCG-D is better than UNCERTAINTY-R in most of the cases, and RANDOM-D performs the worst. Here we are particularly interested in the performance gap between VM-D and UNCERTAINTY-R since both of the two algorithms aim to select the examples with uncertain-

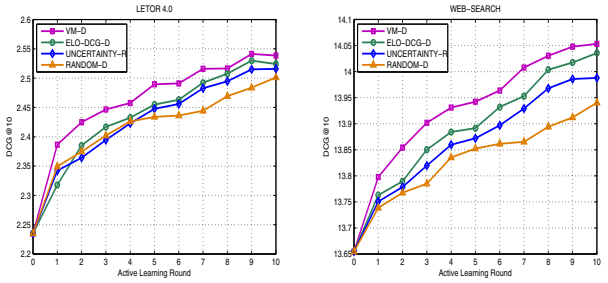

Figure 5: Comparison Results at Document Level.

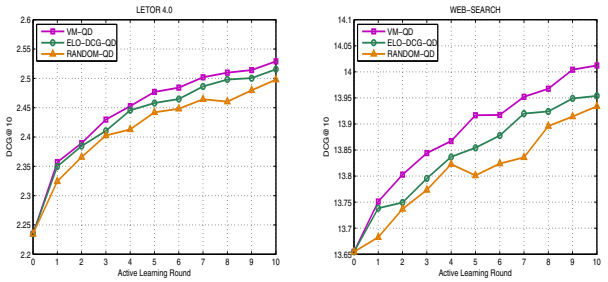

Figure 6: Comparison Results at Two-Stage.

ty. The results may be based on the possible explanation. While the classical UNCERTAINTY-R simply selects the documents with the highest uncertainty in predicted ranking scores, the VM-D selects the documents with the most uncertainty in ranking, and such examples may contribute more to the performance of the ranking model. T-test shows that VM-D performs statistically better than ELO-DCG-D, UNCERTAINTY-R and RANDOM-D ($p<0.05$) in most of the cases.

5.5 Two-Stage Active Learning

In this section, we compare our two-stage VM algorithm (denoted by VM-QD) with two other two-stage active learning algorithms. One is two-stage ELO-DCG algorithm (denoted by ELO-DCG-QD). The other is two-stage random selection (denoted by RANDOM-QD), i.e. random query selection followed by random document selection for each selected query.

Figure 6 shows the comparison results for the three two-stage algorithms on the LETOR 4.0 data set and the WEB-SEARCH data set, respectively. We observe that among the three methods, VM-QD achieves the highest DCG@10 scores, and ELO-DCG-QD performs the second. The results indicate that both the VM-QD and ELO-DCG-QD can select more informative queries and more informative documents than RANDOM-QD. T-test demonstrates that VM-QD performs significantly better than ELO-DCG-QD ($p<0.05$) at some of the check points and statistically outperforms RANDOM-QD ($p<0.05$) most of the times.

6. CONCLUSION

In this paper, we propose a general active learning strategy, Variance Maximization, with application to learning to rank. Noise injection is employed to generate the score distribution. We transform the score distribution to the rank distribution and then adopt the variance of DCG-like gain to measure the model's uncertainty for each unlabeled example. The proposed active learning strategy is applied at both the query level and the document level, and a two-stage active learning algorithm is further extended. Experimental results on both the LETOR 4.0 data set and a real-world Web search data set have demonstrated the effectiveness of the proposed algorithms.

7. ACKNOWLEDGMENTS

This research was supported by National Natural Science Foundation of China (No. 61003107), National Significant Science and Technology Projects (No. 2011ZX01042-001-001), Shanghai Science and Technology Rising Star Program (No. 11QA1403500), Shanghai Talent Development Fund (No. 2010002), and STCSM. (No. 12DZ2272600).

8. REFERENCES

[1] P. Cai, W. Gao, A. Zhou, and K. F. Wong. Relevant knowledge helps in choosing right teacher: Active query selection for ranking adaptation. In *Proceedings of SIGIR '11*, pages 115–124, 2011.

[2] P. Donmez and J. G. Carbonell. Optimizing estimated loss reduction for active sampling in rank learning. In *Proceedings of ICML '08*, pages 248–255, 2008.

[3] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, pages 933–969, 2003.

[4] Y. Freund, H. S. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. *Machine Learning*, pages 133–168, 1997.

[5] J. Friedman. Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, pages 1189–1232, 2001.

[6] R. Herbrich, T. Graepel, and K. Obermayer. Large margin rank boundaries for ordinal regression. In *Advances in Large Margin Classifiers*. MIT Press, 2000.

[7] K. Jarvelin and J. Kekalainen. IR evaluation methods for retrieving highly relevant documents. In *Proceedings of SIGIR '00*, pages 41–48, 2000.

[8] B. Long, O. Chapelle, Y. Zhang, Y. Chang, Z. Zheng, and B. Tseng. Active learning for ranking through expected loss optimization. In *Proceedings of SIGIR '10*, pages 267–274, 2010.

[9] B. Settles. Active learning literature survey. Computer sciences technical report, University of Wisconsin–Madison, 2009.

[10] R. Silva, M. A. Gonçalves, and A. Veloso. Rule-based active sampling for learning to rank. In *Proceedings of ECML PKDD '11*, pages 240–255, 2011.

[11] M. Taylor, J. Guiver, S. Robertson, and T. Minka. SoftRank: Optimizing non-smooth rank metrics. In *Proceedings of WSDM '08*, pages 77–86, 2008.

[12] E. Yilmaz and S. Robertson. Deep versus shallow judgments in learning to rank. In *Proceedings of SIGIR '09*, pages 662–663, 2009.

[13] H. Yu. SVM selective sampling for ranking with application to data retrieval. In *Proceedings of KDD '05*, pages 354–363, 2005.