Incorporating Density in Active Learning with Application to Ranking

Wenbin Cai, and Ya Zhang

Institute of Image Communication & Networking
Shanghai Jiao Tong University
\{cai-wenbin,ya_zhang\}@sjtu.edu.cn

Abstract. Active learning aims to achieve high performance using as few labeled training set as possible, thereby minimizing the cost of data labeling. In Web search ranking applications, learning to rank is an important task which is to automatically build a ranking function through supervised learning. Like many supervised learning tasks, a large amount of labeled training data is required to train a high quality ranking function. Meanwhile, in many real-world learning-to-rank applications, data labeling is usually very expensive and time-consuming. To reduce the labeling cost, there have been many studies on applying active learning to ranking, which aim to select the most informative example for labeling manually. However, existing works certainly ignore the information about prior data density which can be useful for active learning. In this paper, we use the classical Kernel Density Estimation (KDE) method to infer information about data density. Then, under the Generalization Error Reduction (GER) framework, we propose a novel active learning strategy to select the most informative example that minimizes the generalization error. The proposed strategy is applied at the query level, the document level, and further at query-document level with a two-stage active learning algorithm. Experimental results on a real-world Web search ranking dataset have demonstrated the effectiveness of the proposed active learning algorithms.

Keywords: Active Learning, Ranking, Kernel Density Estimation, Generalization Error Reduction

1 Introduction

In a typical machine learning setting, the performance of a learning model is often highly correlated with the amount of training data available. The widely used approach for data collection is called passive learning, where examples are randomly selected from the underlying distribution and annotated by editors. However, it is always the case that there are not enough training data to ensure the performance of the learning model due to the high cost associated with the data labeling process. To reduce the cost of data labeling, active learning comes...
as a general machine learning paradigm, which aims to select the most informative examples for labeling. The key idea behind active learning is that if the learner is allowed to choose the data from which it learns, it can achieve higher performance with fewer labeled instances. So far, various active learning strategies have been proposed for classification and regression task such as uncertainty sampling [15] and query by committee (QBC) [5]. However, the drawback of such active learning methods is that it may fail by selecting outliers. To address the outlier problem, several density-weighted active learning approaches [11, 14, 19, 13] have been proposed to trade-off the uncertainty and the data density. The corresponding reported results have indicated that the density-weighted active learning strategies are superior to the methods that do not consider the information about density.

In Web search ranking applications, learning to ranking is an important task which is to automatically construct ranking functions through supervised learning. Existing learning to rank approaches can be categorized into three groups: pointwise approaches [6], pairwise approaches [17] and listwise approaches [1]. Like many other supervised learning tasks, the performance of a ranking function is highly affected by the number of training data. Due to the complexity of ranking, training a high quality ranking function usually requires a large number of labeled training data. However, in the case of Web search ranking, labeling the example is usually very expensive since the editor is required to carefully assess not only the relevance between the document and the query but also the preference order among various documents related to the query. Therefore, in many real-world Web search ranking applications, while it is relatively easy to collect a large volume of unlabeled examples, it is very expensive to label the examples.

Recently, active learning with application to ranking has been widely studied. Compared to the traditional supervised learning setting, active learning for ranking is more complex due to a unique query-document structure. Considering the query-document structure, most of the existing active learning for ranking algorithms are at either the query level [22, 3] or the document level [23, 4, 21]. In recent year, Long et al [10] proposed a novel two-stage active learning framework for ranking, Expected Loss Optimization (ELO), to address the data dependency issue due to the structure: each query is independent of each other and the query-document pairs are conditionally independent given a query. However, existing works on active learning for ranking certainly ignore the underlying data density, which can be useful to improve the active learning.

In this paper, we attempt to improve the active learning with application to ranking by incorporating the data probability density\(^1\). We explore the information about the prior data density by using the classical Kernel Density Estimation (KDE) method [18]. Due to the query-document structure, we estimate the probability density at the query level and the document level, respectively. For the query density estimation, we first compress the query into a query vector,  

\(^1\) We interchangeably use the terms density, probability and probability density in this paper.
Incorporating Density in Active Learning with Application to Ranking

which aggregates all the documents related to the query. Then, the probability density of the query is directly estimated by using KDE. For the document density estimation, considering the data dependence relationship, we calculate the document probability by multiplying the probability of the document given the related query and the query probability. Then, under the Generalization Error Reduction (GER) framework, we propose a novel active learning strategy to select the most informative example which aims to minimize the generalization error. The proposed strategy is applied at both the query level and the document level, and further at query-document level with a two-stage active learning algorithm. Experimental results on a real-world Web search ranking dataset have demonstrated the effectiveness of the proposed active learning algorithms.

The reminder of this paper is organized as follows: We first review the related work in Section 2. Section 3 describes the details of probability density estimation. Our active learning algorithm is presented in Section 4. Section 5 discusses experiments and results. Finally, we conclude the paper and propose for our future work in Section 6.

2 Related Work

So far, various active learning strategies have been proposed. A comprehensive active learning survey can be found in [12]. Among various types of strategies for active learning, uncertainty sampling [15] and query by committee (QBC) [5] are the two major active learning schemes. The uncertainty sampling selects the unlabeled example about which it is most uncertain how to label, and is usually straightforward for probabilistic model using entropy as the uncertainty measure. The QBC method generates a committee of models and selects the unlabeled data instance about which the models disagree the most. A popular approach to quantify the disagreement is vote entropy.

Compared to the traditional supervised learning setting, active learning for ranking is more complex due to the unique query-document structure. Considering the structure and dependence relationship, existing active learning for ranking can 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 [22] empirically showed that having more queries but shallow documents performed better than having less queries but deep documents. Cai et al. [3] proposed a query selection strategy by combining domain adaptation and QBC-based active learning. For the document level active learning, Yu [23] proposed an algorithm based on RankSVM [8], which chooses the document pairs with the most similar predicted relevance scores. Another document selection strategy was proposed by Donmez et al. [4]. They treat the documents that are expected to change the current model mostly as the most informative example, and the base ranking functions are RankSVM and RankBoost [24]. Both of the above two methods are based on pairwise approach. Silva et al. [21] recently proposed a novel document sampling based on association rules, and this active learning method does not rely on any initial training seed. The query level ac-
tive learning selects all documents related to a query. In the Web search ranking setting, it may include some non-informative documents when there are a large number of documents related to the selected query. The document level active learning selects documents individually. However, this sampling process ignores the conditional dependence between query-document pairs given a query, and hence may lead to undesirable results. To address the above problems, Long et al. [10] proposed a two-stage active learning framework by integrating 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. However, one common limitation of these works is that the information about the data density is ignored, which can be useful to improve the active learning. That is the motivation of this work.

Actually, in previous works on active learning for classification task, several density-weighted strategies have been proposed to balance the uncertainty and the density, and the corresponding reported results have demonstrated that the density-weighted active learning strategies perform better than the methods that do not consider the information about the data density. Xu et al [11] proposed a representative sampling method which uses K-means to cluster the unlabeled examples within the margin of an SVM classifier, and queries the cluster centroids for labeling. Zhu et al [14] presented a method called K-Nearest-Neighbor-based density measure, which quantifies the density by the average similarity between the unlabeled example and the K nearest neighbors, and weighted the entropy-based uncertainty by the KNN density. McCallum et al [13] proposed a density-weighted QBC algorithm, which choose the examples with the largest committee disagreement in predicted label weighted by the density. Nguyen and Smeulders [19] suggested a probabilistic framework that incorporated clustering information into the active learning process, and the clustering structure is used to estimate the data density based on a Gaussian mixture model. They argued that the examples lying close to the decision boundary with higher density are more informative.

3 Probability Density Estimation

As discussed above, in the classification setting, most of the existing studies on density-weighted active learning use a clustering-based technique for density estimation. However, the performance of the clustering-based methods highly rely on the quality of clustering and hence the methods suffer from the following two drawbacks. First, estimation of the density may not be accurate if there is no clear cluster structure in the data set. Secondly, it is difficult to determine the proper number of clusters in a specific active learning task.

To avoid the above problems, we adopt the classical Kernel Density Estimation (KDE) [18], also known as Parzen Windows, to estimate the data density. The KDE method, which is widely used for nonparametric density estimation, constructs the probability density by locating kernels at each of the observed data point. Considering the query-document structure in learning-to-rank, we
Incorporating Density in Active Learning with Application to Ranking

estimate the density at the query level and the document level respectively. The details of the probability density estimation are provided in the rest of this section.

3.1 Estimating the Query Probability Density

In the case of ranking, there exists a unique query-document structure. Thus, each query $q$ is not a single example, but a set of documents. Assuming a query $q$ retrieves $n$ documents, we denote it by $q=\{d_1, d_2, ..., d_n\}$, and represent each document $d_i \in \mathbb{R}^{p_d}$ as a list of $p_d$ feature values, i.e. $d_i = \langle f_1, f_2, ..., f_{p_d} \rangle$.

In practice, the problem to estimate the query probability directly by KDE is that the KDE method can only estimate the density for a single vector whereas the query is a set of document vectors. A simple solution for this problem is to sum over the related document probability to calculate the query probability:

$$p(q) = \sum_{d_i \in q} p(d_i) \quad \text{for all } q \in U$$

where $p(d_i)$ is the probability of the $i$-th related document estimated by KDE. However, using this summation to compute the query probability may result in undesirable results since the data dependency relationship is ignored. Moreover, another critical issue with the summation process is that the computational cost is extremely high because we need to estimate the probability density of each unlabeled document directly with KDE, and there are usually a very large number of documents.

Inspired by the recent work on local ranking [20] and ranking adaptation [2], we first compress the query into a query feature vector by aggregating all the associated documents. Then, the probability of the query can be estimated directly by using the KDE method. We define three types of aggregation to construct the query vector, i.e. the mean vector (denoted by $m$), the variance vector (denoted by $v$), and the skewness vector (denoted by $s$). Then, the query vector $q$ can be represented as $q=\langle m, v, s \rangle \in \mathbb{R}^{p_q}$ (note that $p_q=3p_d$), and $m$, $v$ and $s$ can be computed as:

$$m_j = \frac{1}{n} \sum_{d_i \in q} d_{ij} \quad (j = 1, 2, ..., p)$$  \hspace{1cm} (2)$$

$$v_j = \frac{1}{n} \sum_{d_i \in q} (d_{ij} - m_j)^2 \quad (j = 1, 2, ..., p)$$  \hspace{1cm} (3)$$

$$s_j = \frac{1}{n\sigma_j^3} \sum_{d_i \in q} (d_{ij} - m_j)^3 \quad (j = 1, 2, ..., p)$$  \hspace{1cm} (4)$$

where $d_{ij}$ denotes the $j$-th feature from $i$-th document associated with the query $q$, and $\sigma_j$ is the standard deviation of the $j$-th feature related to the query $q$.

Once we construct all the query vectors by feature aggregation, we first normalize the query vectors to avoid extreme differences of spread in the various
directions. Then, we utilize the KDE method to estimate the probability density of the query $q$. The probability estimation of the query $q$ is given as:

$$p(q) = \frac{1}{|Q|} \sum_{i=1}^{\left|Q\right|} K\left(\frac{1}{\lambda} (q - q_i)\right)$$  \hspace{1cm} (5)$$

where $|Q|$ represents the number of queries in the pool set, and $\lambda$ is the kernel width (the smoothing parameter). We choose the kernel function $K$ as the Gaussian kernel:

$$K(x) = (2\pi)^{-p/2}\exp\left(-\frac{1}{2} x^T x\right)$$  \hspace{1cm} (6)$$

which is a symmetric kernel with its value smoothly decaying away from the kernel center. Combining the equations (5) and (6), the density estimation for the query $q$ can be expressed as:

$$p(q) = \frac{1}{|Q|(2\pi \lambda^2)^{p/2}} \sum_{i=1}^{\left|Q\right|} \exp\left(\frac{\|q - q_i\|^2}{2\lambda^2}\right)$$  \hspace{1cm} (7)$$

In the kernel density estimation, the choice of the kernel width $\lambda$ has an important effect on the probability estimation. The optimal kernel width size has been extensively studied in the previous literature. An optimal kernel width $\lambda^*$ can be determined through the minimization of mean square integrated error (MSIE). For the Gaussian kernel function, we choose the $\lambda$ based on the results proposed in [18]. More complicated methods such as cross-validation method for determining the kernel width can also be found in [18].

Remark 1. The time to estimate the probability of a query, $p(q)$, is $O(p_q|Q|)$. Therefore, for all the unlabeled queries, the total time cost is $O(p_q|Q|^2)$. However, estimating all the unlabeled documents directly by KDE to calculate the query probability will take $O(p_d|D|^2)$ time. In Web search ranking applications, $|Q|$ is much smaller than $|D|$, i.e. $|Q| \ll |D|$. Therefore, the computational cost $O(p_q|Q|^2)$ is much lower than $O(p_d|D|^2)$.

3.2 Estimating the Document Probability Density

Considering the data dependency relationship, i.e. the query-document pairs are conditionally independent given a query, we estimate the probability density of the document $d$ as:

$$p(d) = p(d,q) = p(d|q)p(q)$$  \hspace{1cm} (8)$$

where

$$p(d|q) = \frac{1}{|D_q|(2\pi \lambda^2)^{p/2}} \sum_{i=1}^{\left|D_q\right|} \exp\left(\frac{\|d - d_i\|^2}{2\lambda^2}\right)$$  \hspace{1cm} (9)$$

and $|D_q|$ denotes the number of documents related to the query $q$.

---

$ ^2$ As the amount of the unlabeled data (denoted by pool) is overwhelming over the labeled data, we estimate the density with the pool set.  

$ ^3$ $|D|$ denotes the number of documents in the pool set.
Remark 2. The time to estimate $p(d|q)$ is $O(p_d|D_q|)$. Thus, the total time for all the unlabeled documents in the pool set is $O(p_d|D_q|D^2|Q|)$ plus the time $O(p_q|Q|^2)$ costed by the query density estimation. As mentioned above, directly estimating all the unlabeled documents will take $O(p_d|D|^2)$ time. Because both $|D_q|$ and $|Q|$ are much smaller than $|D|$, i.e. $|D_q|\ll |D|$ and $|Q|\ll |D|$, the time cost $O(p_d|D_q|^2|Q|)+O(p_q|Q|^2)$ is much lower than $O(p_d|D|^2)$.

4 Generalization Error Reduction for Active Learning

In this section, we first introduce the framework of Generalization Error Reduction (GER) for active learning. We then apply the GER framework to learning-to-rank to derive the active learning algorithm. More specifically, the GER framework is applied to the query level and the document level to derive the query level active learning algorithm and the document level active learning algorithm respectively. By integrating the query level algorithm and the document level algorithm, a two-stage active learning algorithm is further extended.

4.1 The Framework of GER

So far, a number of strategies [16, 19] have been proposed for active learning, which aim to select the example to minimize the generalization error:

$$ \text{Err} = \int_x \text{EL}[y(x), \hat{y}(x)|x]p(x)dx $$

(10)

where $y(x)$ is the true label of $x$, and $\hat{y}(x)$ is the predicted label of $x$. EL[.|x], called expected error or expected loss, denotes the expectation regarding to a loss function.

Since the computation of the generalization error in equation (10) is usually complicated. Nguyen et al [19] suggested to select the unlabeled example $x$ that has the largest contribution to the current error instead of choosing the example yielding the smallest generalization error, and the selection criterion is:

$$ x^* = \arg\max_{x \in \text{pool}} \text{EL}[.|x]p(x) $$

(11)

It shows that the example which aims to minimize the generalization error is the one with the largest expected loss weighted by the probability density. Furthermore, if $p(x)$ is uniform, the example with the largest expected loss can be treated as the most informative example. When $p(x)$ is non-uniform, the information about the data density can be used to select better data.

4.2 GER for Ranking

Compared to the traditional GER framework, GER for ranking is a bit more complex due to the unique query-document structure. In the case of ranking, the
Algorithm 1: Query Level Active Learning

Given: the small labeled set \( L \), the large unlabeled pool set

Query Sampling:
1: For \( q = 1 \cdots |Q| \) in pool
2: \( m_j \leftarrow \frac{1}{n} \sum_{d_i \in q} d_{ij} \)
3: \( v_j \leftarrow \frac{1}{n} \sum_{d_i \in q} (d_{ij} - m_j)^2 \)
4: \( s_j \leftarrow \frac{1}{n \sigma_3} \sum_{d_i \in q} (d_{ij} - m_j)^3 \)
5: construct the query vector \( q \leftarrow <m, v, s> \)
6: \( p(q) \leftarrow \text{Eq}.7 \)
7: \( \text{EL}[.|q] \leftarrow \text{Eq}.13 \)
8: current error of \( q \leftarrow \text{EL}[.|q] \times p(q) \)
9: End For

Output: The query \( q^* \) with the largest current error

generalization error has two levels, i.e. the query level generalization error and the document level generalization error. Thus, we aim to select the unlabeled query and the unlabeled document to minimize the query level generalization error and the document level generalization error respectively.

Query Level Active Learning To minimize the query level generalization error, our active learning strategy at the query level is to select the unlabeled query \( q \) that maximizes the current query level error. The query selection criterion can be expressed as:

\[
q^* = \arg\max_{q \in \text{pool}} \text{EL}[.|q] p(q) \quad (12)
\]

where

\[
\text{EL}[.|q] = \min_{\phi} \int y l(\phi, y) p(y|q; L) dy \quad (13)
\]

and \( y \) represents the predicted relevance score vector of the query \( q \), and \( l(\phi, y) \) denotes the loss function.

In the case of Web search ranking, if the ranking metric is Discounted Cumulative Gain (DCG), the associated loss is the difference between the DCG for that ranking and the ranking with the largest DCG:

\[
l(\phi, y) = \max_{\phi^*} \text{DCG}(\phi^*, y) - \text{DCG}(\phi, y) \quad (14)
\]

where

\[
\text{DCG}(\phi, y) = \sum_i \frac{2^{y_i} - 1}{\log(1 + \phi(i))} \quad (15)
\]

and \( \phi(i) \) denotes the rank of \( i \)-th document. The expected DCG loss for a given query can be calculated by combining the equations (13), (14) and (15). More practical implementation details about the expected loss can be found in [10].

Combining the equations (7) and (12), we can derive the query level active learning algorithm. The corresponding pseudo-code for query sampling is shown in algorithm 1.
Document Level Active Learning The document level active learning selects document individually. Similar to the query sampling strategy, our document level active learning algorithm aims to select the unlabeled document $d$ that maximizes the current document level error. The document selection criterion can be represented as:

$$d^* = \arg\max_{d \in \text{pool}} \text{EL}[|d|]p(d)$$

(16)

where

$$\text{EL}[|d|] = \int_{y'} \min_{\phi} \int_{y_i} l(\phi, y)p(y|q; L)dy_i dy'_i$$

(17)

and $y_i$ denotes the predicted relevance score of $i$-th document associated with the query $q$, and $y'$ is the vector $y$ after removing $y_i$. More details about the expected loss of the document can be found in [10].

Combining the equations (8), (9) and (16), the document level active learning algorithm can be derived. The pseudo-code for document sampling is given in algorithm 2 (Note that the steps 2 to 5 are identical to those given in algorithm 1, and hence they are omitted in algorithm 2).

Algorithm 2: Document Level Active Learning

**Given:** the small labeled set $L$, the large unlabeled pool set

**Document Sampling:**

1: For $q = 1 \cdots |Q|$ in pool

   6: $p(q) \leftarrow \text{Eq.7}$

7: End For

8: For $d = 1 \cdots |D|$ in pool

9: $p(d|q) \leftarrow \text{Eq.9}$

10: $p(d) \leftarrow p(d|q) \times p(q)$

11: $\text{EL}[|d|] \leftarrow \text{Eq.17}$

12: current error of $d \leftarrow \text{EL}[|d|] \times p(d)$

13: End For

**Output:** The document $d^*$ with the largest current error

Two-Stage Active Learning Both the query level active learning and the document level active learning have their own disadvantages. The query level sampling selects all documents associated with a query. It may include some non-informative documents because there are usually a large number of documents associated with a selected query, especially in the Web search ranking application. Since the quality of a ranking model is mainly measured by a small number of top ranked documents, most of them are non-informative. On the other hand, the document level sampling selects documents individually. This sampling strategy ignores the query-document structure and the data dependency relationship, and hence may not be optimal. To address the problem, Long et al [10] 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 related to the selected queries. The two-stage algorithm captures the query-document structure, and the corresponding advantage is that it can discard the non-informative documents related to the informative queries. In this work, we still follow this two-stage active learning strategy in designing the proposed algorithm.

5 Experiment

5.1 Data Set

We use a real-world Web search data set on learning to rank from a commercial search engine (denoted as WEB-SEARCH). Each query-document pair is represented by 36 features, including the query features, document features and query-document features. The relevance score is labeled with five-level relevance scheme: \{Bad, Fair, Good, Excellent, Perfect\}. All the features have been normalized as:

$$f_{N}^{(i,j)} = \frac{f_{(i,j)} - \min_{i \in n}\{f_{(i,j)}\}}{\max_{i \in n}\{f_{(i,j)}\} - \min_{i \in n}\{f_{(i,j)}\}}$$  \hspace{1cm} (18)

Where $n$ denotes the number of documents in the data set, and $f_{(i,j)}$ represents the $j$-th feature from the $i$-th document.

We randomly split the dataset into three parts at the query level: base training set, pool set, and test set. We use the base training set (denoted as base set) as the small labeled data set $L$ 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 dataset is listed in Table 1. In practice, the initial base training set is often collected by depth-$k$ retrieval with randomly selected queries from the underlying distribution.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Active Learning set</th>
<th>#queries</th>
<th>#documents</th>
<th>#documents per query</th>
</tr>
</thead>
<tbody>
<tr>
<td>WEB-SEARCH</td>
<td>base set</td>
<td>200</td>
<td>4,102</td>
<td>~20 docs per query</td>
</tr>
<tr>
<td></td>
<td>pool set</td>
<td>3,000</td>
<td>60,609</td>
<td></td>
</tr>
<tr>
<td></td>
<td>test set</td>
<td>564</td>
<td>11,363</td>
<td></td>
</tr>
</tbody>
</table>

5.2 Experimental Setting

For the base learner, we use Gradient Boosting Decision Tree (GBDT) to train our ranking models. GBDT is a well-known non-linear regression model which belongs to pointwise approach. More details about GBDT can be found in [7].

To validate the effectiveness of our active learning algorithm, we first compare the proposed algorithm with two other algorithms. One is ELO-DCG algorithm, representing one of state-of-art. The other one is random selection, representing a baseline. The algorithms select top $n$ informative examples and adds it to base set to train a new rank function. In this study, the active learning process iterates 20
Incorporating Density in Active Learning with Application to Ranking

In each round of active selection, 100 queries were selected at the query level and 2000 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 [22]. The performance of the new ranking model is evaluated on the test set, and we use DCG [9] to measure the performance. Because users of a retrieval system are only interested in the top \( k \) retrieved documents, we select to use DCG@5 and DCG@10 as the performance measurement. Then, we experiment the reduction in labeling cost by comparing the proposed algorithm to the supervised learning algorithm. We repeat each experiment for 10 times and report the average results.

5.3 Comparison Results and Discussion

**Query Level Active Learning** We first compare the proposed query level active learning algorithm with query level ELO-DCG (denoted by ELO-DCG-Q) algorithm and random query selection (denoted by Random-Q).

Figure 1 shows the learning curves of the three query level active learning algorithms on the WEB-SEARCH data set. The x-axis denotes the number of iteration of active learning process. For all three methods, the DCG increases with the number of iteration. This agrees with the intuition that the quality of a ranking function is positively correlated with the number of examples in the training set. We observe that both the proposed algorithm and ELO-DCG-Q perform better than Random-Q. A possible explanation is that both the proposed algorithm and ELO-DCG-Q optimize the loss based on DCG that is directly related to the objective function used to evaluate the ranking function. Furthermore, the proposed algorithm consistently performs better than ELO-DCG-Q. The results may be based on the following explanation. Compared to the ELO-DCG-Q that treats the example with highest expected DCG loss as the most informative example, the proposed algorithm selects the example with the largest expected DCG loss weighted by density. The information about the data density could
bring useful knowledge to the active learning process and result in a better performance. T-test indicates that the proposed algorithm performs significantly better (p < 0.05) than ELO-DCG-Q and RANDOM-Q in most of the cases.

**Document Level Active Learning** In this section, we show that the proposed document level algorithm effectively selects the most informative documents to improve the rank performance. We compare the proposed active learning algorithm with the document level ELO-DCG (denoted by ELO-DCG-D) algorithm and random document selection (denoted by Random-D).

The results of the three document level active learning algorithms on the WEB-SEARCH data set are plotted in Figure 2. We observe that the proposed active learning algorithm converges much faster than the other two methods. Furthermore, the proposed algorithm converges much faster than the two other algorithms, i.e. the proposed algorithm achieves the best DCG score with much less examples added to the training set, which the entire labeled data can attain. The comparison results demonstrate that the proposed algorithm is effec-

![Fig. 2. Document level comparison results in terms of DCG@5 and DCG@10.](image)

![Fig. 3. Two-Stage comparison results in terms of DCG@5 and DCG@10.](image)
tive in selecting the most informative documents for labeling. T-test shows that the proposed algorithm performs statistically better ($p < 0.05$) than ELO-DCG-D and Random-D most of the times.

**Two-Stage Active Learning** In this section, we compare the proposed two-stage active learning algorithm with two other two-stage 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. As mentioned above, we simply fix the number of documents selected for each selected query to be 10 for all three two-stage algorithms.

Figure 3 shows the comparison results for the three two-stage algorithms on the WEB-SEARCH data set. We observe that among the three methods, the proposed two-stage algorithm achieves the highest DCG scores, and ELO-DCG-QD performs the second. The results indicates that both the proposed algorithm and ELO-DCG-QD can select more informative queries and more informative documents than Random-QD. T-test demonstrates that the proposed two-stage algorithm still significantly outperforms ($p < 0.05$) ELO-DCG-QD and Random-QD in most of the cases.

### 5.4 Labeling Cost Reduction

In this section, we show the reduction in labeling cost by comparing our active learning algorithms with supervised learning. The performance of the ranking function trained with the whole labeled data set, i.e. the base set plus the pool set, is called optimal performance. When the performance of the active learning model obtained with our algorithm is comparable to the optimal performance, we call the size of training data as the saturated size. The saturated size and the percentage of labeling cost reduction are summarized in Table 2.

<table>
<thead>
<tr>
<th>The Proposed Algorithms</th>
<th>Saturated Size</th>
<th>Labeling Cost Reduction (%)</th>
<th>Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query Level</td>
<td>~32k</td>
<td>~50.3%</td>
<td>DCG@5</td>
</tr>
<tr>
<td>Document Level</td>
<td>~30k</td>
<td>~53.6%</td>
<td></td>
</tr>
<tr>
<td>Two-Stage</td>
<td>~30k</td>
<td>~53.6%</td>
<td></td>
</tr>
<tr>
<td>Query Level</td>
<td>~34k</td>
<td>~47.5%</td>
<td>DCG@10</td>
</tr>
<tr>
<td>Document Level</td>
<td>~32k</td>
<td>~50.5%</td>
<td></td>
</tr>
<tr>
<td>Two-Stage</td>
<td>~32k</td>
<td>~50.5%</td>
<td></td>
</tr>
</tbody>
</table>

We observe that the proposed algorithms reduce the labeling cost very effectively. Furthermore, the proposed query level active learning algorithm performs the worst among the three level active learning in terms of labeling cost reduction. As mentioned above, the query level sampling selects all documents associated with a query, and may include some non-informative documents. Here we are particularly interested in the comparison results between the document level active learning and the two-stage active learning. The results show that the two-stage algorithm performs as well as the document level algorithm, rather than
outperforms it. An explanation of the results may be as follows. Compared to the document level sampling, the two-stage active learning algorithm captures the query-document structure, and the corresponding advantage is that it can discard the non-informative documents related to the informative queries. However, the disadvantage is that it may ignore some informative documents which belong to the non-informative queries, and such drawback may result in inferior performance. Therefore, how to overcome the disadvantage should be further considered.

6 Conclusion and Future Work

In this paper, we explore the information about data density using the Kernel Density Estimation (KDE) method. In the case of Web search ranking, we estimate the probability density at the query level and the document level respectively. Then, under the Generalization Error Reduction (GER) framework, we propose a novel active learning strategy for ranking, which aims to select the most informative example that minimizes the generalization error. The proposed strategy is applied at both the query level and the document level, and extends to a two-stage active learning algorithm. Experimental results on a real-world Web search ranking dataset have demonstrated the effectiveness of the proposed active learning algorithms.

In this study, the information about data density is combined with expected loss. Actually, it can be weighted with many other active learning criterion such as uncertainty, which has been widely studied in the classification task. The proposed algorithm performs batch mode active learning. The correlation or similarity among the selected examples at each batch is not considered. Possible extension of the work is to consider the diversity of the selected data set to further minimize the labeling cost.

7 Acknowledgements

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).

References


