Inf Retrieval J
DOI 10.1007/s10791-015-9250-6

Active learning for ranking with sample density

Wenbin Cai · Muhan Zhang · Ya Zhang

Received: 23 March 2014 / Accepted: 12 February 2015

Abstract While ranking is widely used in many online domains such as search engines and recommendation systems, it is non-trivial to label enough data examples to build a high performance machine-learned ranking model. To relieve this problem, active learning has been proposed, which selectively labels the most informative examples. However, data density, which has been proven helpful for data sampling in general, is ignored by most of the existing active learning for ranking studies. In this paper, we propose a novel active learning for ranking framework, generalization error minimization (GEM), which incorporates data density in minimizing generalization error. Concentrating on active learning for search ranking, we employ classical kernel density estimation to infer data density. Considering the unique query–document structure in ranking data, we estimate sample density at both query level and document level. Under the GEM framework, we propose new active learning algorithms at both query level and document level. Experimental results on the LETOR 4.0 data set and a real-world Web search ranking data set from a commercial search engine have demonstrated the effectiveness of the proposed active learning algorithms.

Keywords Active learning · Density · Learning to rank

1 Introduction

Learning to rank, which is to automatically construct ranking functions through supervised learning, has been widely adopted by many Information Retrieval (IR) applications such as

W. Cai · M. Zhang · Y. Zhang (✉)
Shanghai Key Laboratory of Multimedia Processing and Transmissions, Shanghai Jiao Tong University, 800 Dongchuan Road, Shanghai, China
e-mail: ya_zhang@sjtu.edu.cn

W. Cai
e-mail: cai-wenbin@sjtu.edu.cn

M. Zhang
e-mail: ufo000@sjtu.edu.cn

Published online: 24 February 2015

web search and recommendation. A typical learning-to-rank process can be described as follows: (1) A set of queries, their retrieved documents, and the corresponding relevance judgments are given as the training set. (2) A ranking function is learned by minimizing a given loss function defined on the training data. (3) For a new query, the ranking function is used to sort its related documents according to their predicted ranking scores. Existing learning-to-rank approaches fall into three major categories: pointwise approaches (Cossock and Zhang 2006), pairwise approaches (Cao et al. 2006), and listwise approaches (Xia et al. 2008). Like many other supervised learning tasks, the performance of a learned ranking function is highly correlated with the amount of training data available. However, in many cases, the high cost associated with data annotation makes it expensive or even prohibitive to obtain a large number of labeled data.

To reduce the labeling cost, active learning is proposed to selectively choose informative data for labeling. A typical active learning process, as shown in Fig. 1, iterates through the following steps: (1) Generate a base model from a small initial training set; (2) use a certain sampling function to sample from a large volume of unlabeled data and query their labels; (3) add the newly labeled examples to the training set and update the model. This data sampling process is iterated until a given performance threshold is reached or the labeling budget is exhausted. The key idea behind active learning is that if a learner is allowed to choose the data from which it learns, it can achieve better performance with fewer labeled instances.

In recent years, active learning has been well studied for classification tasks. Existing active learning for classification strategies can be categorized into two major classes: uncertainty sampling (Lewis and Gale 1994), which selects examples whose labels the current model is least certain about, and query by committee (QBC) (Freund et al. 1997), which selects points having the highest disagreement among a group of models. However, the major shortcoming is that they cannot differentiate outliers from informative points, and thus often fail by selecting outliers (Settles 2012). To solve this so-called outlier problem, several density-weighted active learning approaches have been proposed by modeling the input distribution explicitly during data sampling (Xu et al. 2003; Zhu et al. 2010; McCallum and Nigam 1998; Nguyen and Smeulders 2004. The central idea of using prior data density in active learning is that it considers the whole input space rather than individual data points. Figure 2 shows an illustrative example for explaining the importance of prior data density in active learning. Because the data instance A is close to the decision boundary, it would be chosen as the most uncertain. However, choosing B (density-weighted sampling) is more helpful to improve the model's performance as it is representative of other instances in the data distribution. Thus, active learning with the representative example B is better.

Compared to active learning for classification problems, there are relatively fewer researches on active learning for ranking. Earlier efforts of active learning for ranking mainly focus on sample selection at either query level (Yilmaz and Robertson 2009; Cai et al. 2011; Qian et al. 2013) or document level (Aslam et al. 2009; Ailon 2011; Yu 2005; Donmez and Carbonell 2008; Silva et al. 2011), which ignored one important characteristic of the ranking data: its query–document structure. That is, two examples may share the same query or the same document. In fact, the above property makes data examples for ranking dependent of each other. Considering this data dependence, Long et al. (2010) investigated a new two stage active learning framework for ranking. Despite of these above efforts, data density, which has been proven helpful for data sampling in general, is ignored by most existing active learning for ranking studies. Furthermore, due to the fact that the data structure in ranking is totally different from classification data, existing density-

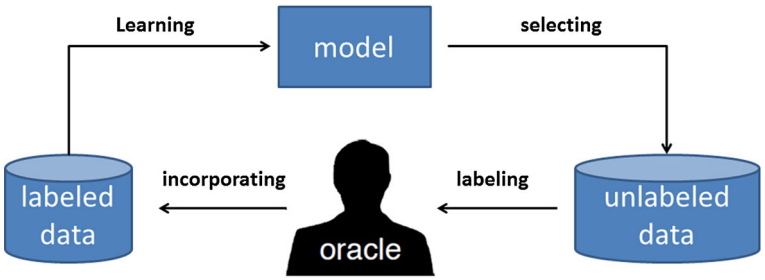

Fig. 1 A general active learning process

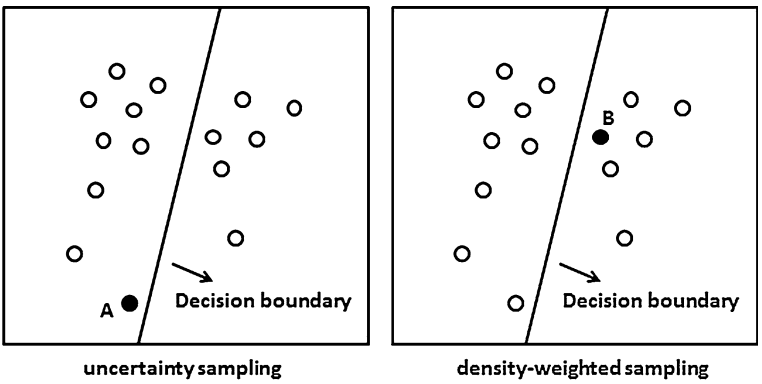

Fig. 2 An illustrative example for explaining the importance of sample density in active learning. Since the instance A is near the decision boundary, it would be chosen as the most uncertain one. However, choosing the example B is more helpful to improve the model's quality as it is representative of other instances in the data distribution

weighted algorithms are not directly applicable to ranking. Hence, active learning for ranking with data density is greatly needed.

In this paper, we are interested in applying data density into active learning with applications to search ranking. More specifically, we attempt to solve the following two challenges: (1) how to accurately estimate data density by taking particular consideration of the unique query–document structure in the ranking data, and (2) how to incorporate sample density in active learning to guarantee the generalization performance of the ranking function learned. We propose a novel active learning framework called generalization error minimization (GEM), which theoretically incorporates the data density in minimizing the generalization error, and further connect GEM to density-weighted active learning for classification in order to provide a uniform view. The former challenge is addressed by employing the classical Kernel Density Estimation (KDE) approach to estimate the ranking data density. For the latter challenge, we apply the proposed GEM framework to derive new algorithms at both query level and document level, and a two stage active learning algorithm is further derived. To our best knowledge, this is the first time that sample density is incorporated into active learning for ranking. Extensive experimental results on the LETOR 4.0 data set and a real-world search ranking data set from

a commercial search engine have demonstrated the effectiveness of the proposed active learning algorithms.

The main contributions of this paper can be summarized as follows.

- We propose a novel active learning framework called GEM, which explicitly incorporate data density in minimizing generalization error.
- We further link our GEM algorithm with previous work on density-weighted method for classification in order to provide a uniform view.
- We bring the idea of density-based active learning into the domain of learning to rank, and propose an effective method to estimate sample density by considering the query–document structure.
- Under the GEM framework, we accordingly derive new active learning for ranking algorithms at both query level and document level.

The rest of this paper is organized as follows. We first briefly review related work in Sect. 2. Section 3 proposes the GEM framework and explore the linkage to density-weighted active learning for classification. We detail probability density estimation and apply GEM to learning-to-rank in Sect. 4. The experimental results are presented in Sect. 5. Finally, Sect. 6 concludes the paper and discusses possible directions for future work.

2 Related work

Active learning has been well motivated in many machine learning domains. In this section, we first briefly review existing active learning methods in general, and then summarize active learning for ranking algorithms.

2.1 General active learning

So far, various active learning strategies have been proposed. A comprehensive active learning survey can be found in Settles (2012).

Generally, among various types of strategies for active learning, uncertainty sampling (Lewis and Gale 1994) and query by committee (QBC) (Freund et al. 1997) are the two major active learning schemes. Uncertainty sampling selects unlabeled examples whose labels the current classifier is most uncertain about, and is usually straightforward to implement for probabilistic models using entropy as the uncertainty measure. The QBC framework generates a committee of member models and selects unlabeled data instances about which the models disagree the most. A popular function to quantify the disagreement is vote entropy.

One common weakness of the above active learning methods is that it cannot differentiate outliers from informative examples. To address this so-called outlier problem, several density-weighted active learning heuristics have been proposed to balance informativeness and data density for active sample selection. Xu et al. (2003) proposed a representative sampling method, which first cluster the unlabeled examples located in the margin of an SVM classifier, and then queries the labels of the examples that are close to each cluster centroid. Zhu et al. (2010) presented a method called K-Nearest-Neighbor-based density measure that quantifies density by the average similarity between an unlabeled example and its K nearest neighbors, and weighted the entropy-based uncertainty by the KNN density. McCallum et al. (1998) proposed a density-weighted QBC algorithm, which chooses examples with the highest committee disagreement in predicted labels weighted by sample

density. Nguyen et al. (2004) suggested a probabilistic framework that incorporates clustering information into active learning, and the clustering structure is used to estimate the data density based on a Gaussian mixture model. They argued that examples lying close to the decision boundary with higher density are more likely to be informative.

As listed above, density-weighted methods have been extensively studied in classification by incorporating data density into sampling function explicitly. Most of them apply a clustering-based technique (e.g., K-means clustering) in density estimation and heuristically incorporate estimated density in sample selection (e.g., choosing examples that are close to the centroid of each cluster located in the uncertain region). However, they are not readily applicable to ranking problems because ranking data, with unique query–document structure, is very different from classification data. In addition, they usually suffer from the following two main drawbacks. First, it is difficult to determine how many clusters are appropriate for a specific active learning problem. Secondly, the clustering methods perform poorly when the data have no clear cluster structure, leading to inaccurate density estimation. Figure 3 illustrates the importance of cluster structure. The clustering algorithm works well when there exists clear cluster structure (left panel). On the contrary, it performs poorly when the data do not have clear cluster structures (right panel), making the clustering-based active learning inapplicable. Furthermore, the clustering-based density weighted sampling methods are quite heuristics and lack of solid theoretical justification w.r.t. the generalization performance of the model learned.

In contrast, the work presented in this paper utilizes kernel-based density estimation to estimate data density. The main advantage over the clustering-based approaches is that it estimates sample density from the data directly, and does not rely on the strong assumption of having a certain cluster structures. We then incorporate the estimated data density into sampling function under the statistical learning theory to directly optimize the generalization error.

2.2 Active learning for ranking

Compared to other supervised learning applications, a unique query–document structure exists in ranking tasks, which results in a unique data dependence relationship. Considering the structure and dependence relationship, existing active learning for ranking algorithms may be categorized into two types: query level active learning and document level active learning.

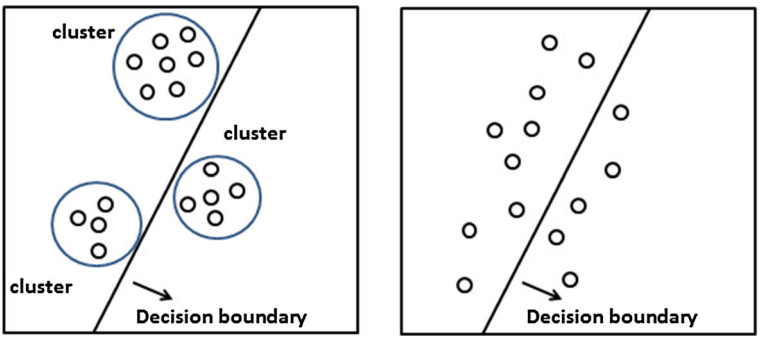

Fig. 3 An illustration of the importance of cluster structure for clustering. The clustering algorithm works well when there exists clear cluster structure (*left panel*). On the contrary, clustering methods perform poorly when the data have no clear cluster structure (*right panel*), making clustering-based active learning inapplicable

Query level active sampling chooses all documents related to a query. Yilmaz et al. (2009) empirically showed that having more queries but shallow documents performed better than having less queries but deep documents. Cai et al. (2011) proposed a query selection strategy by combining domain adaptation and QBC-based active learning. In recent research, Qian et al. (2013) introduced a pairwise query selection method under a layered hashing framework. A drawback of query level data sampling is that it may include some non-informative documents when there are a large number of documents associated with the selected query.

Document level active sampling selects documents independently. Aslam et al. (2009) empirically compared several document selection strategies for learning to rank, e.g., depth-k pooling and uniform random sampling. Yu (2005) proposed a document level active sampling algorithm, which treats the document pairs with similar predicted relevance scores as the most informative one. Document sampling algorithm is applied to RankSVM (Herbrich et al. 2000), a classical pairwise learning-to-rank approach. Donmez et al. (2008) proposed a theoretical document selection method to choose the query–document pairs that are expected to maximally change the current ranking model once labeled and added into the training set. The base ranking functions are RankSVM and RankBoost (Freund et al. 2003). Based on statistical learning theory, Ailon (2011) analyzed the query complexity for pairwise ranking. Silva et al. (2011) proposed a novel document sampling algorithm based on association rules, which does not rely on any initial training seed. However, document level sampling ignores the conditional dependence relationship between query–document pairs given a query, and thus may lead to undesirable results.

Taking particular consideration of the unique query–document structure in ranking data, Long et al. (2010) proposed a two stage active learning framework to integrate query level active learning and document level active learning. Under the Bayesian framework, the expected loss optimization (ELO) principle is introduced for active learning. Recently, following this two stage active learning strategy, Cai et al. (2012) introduced a novel active learning for ranking method, which is to choose the examples with the highest variance in terms of ranking through noise perturbation.

As summarized above, one shortcoming of the existing active learning for ranking algorithms lies in the neglect of data density. In this study, we attempt to incorporate sample density into active learning for ranking to achieve better prediction performance. Of the existing studies, the one most related to ours is Long et al. (2010). The major extension is that we incorporate sample density to weight the expected loss. Besides, we follow the two stage framework (Long et al. 2010; Cai and Zhang 2012) in designing our active learning algorithms.

3 The framework of generalization error minimization

In this section, we first propose the GEM framework for active learning, which theoretically incorporates data density in optimizing the generalization error. We further explore the connection between GEM and general density-weighted active learning for classification to provide a uniform view behind these two different strategies.

3.1 Generalization error minimization for active learning

In supervised learning, the objective of the training process is to learn a model $\mathcal{H}$ from a labeled data set $\mathcal{D}$ that minimizes the generalization error on future unseen data:

$$\epsilon_{\mathcal{D}} = \int_{\mathcal{X}\times\mathcal{Y}} \mathcal{L}[\mathcal{H}(x), y(x)] dP(x, y), \tag{1}$$

where $y(x)$ is the true label of x, $\mathcal{H}(x)$ is the predicted label, and $\mathcal{L}[\mathcal{H}(x), y(x)]$ is a given loss function. As derived in Roy and McCallum (2001), the optimal active learning principle is to choose the example x^* from a large set of unlabeled data (denoted as pool set hereafter) such that when added to the training set with a given label y^*, the classifier trained on the enlarged set would have the least error:

$$\epsilon_{\mathcal{D}+(x^*,y^*)} < \epsilon_{\mathcal{D}+(x,y)}, \quad \forall (x, y) \in \text{pool}. \tag{2}$$

However, the naive implementation of this approach would be quite inefficient and almost intractable due to the necessity of retraining models.

By factoring the joint probability $P(x,y)$, we can rewrite the above generalization error in Eq. (1) as:

$$\begin{aligned} \epsilon_{\mathcal{D}} &= \int_{\mathcal{X}\times\mathcal{Y}} \mathcal{L}[\mathcal{H}(x), y(x)] dP(x, y) \\ &= \int_{\mathcal{X}} \int_{\mathcal{Y}} \mathcal{L}[\mathcal{H}(x), y(x)] p(y|x) p(x) dy dx \\ &= \int_{\mathcal{X}} \mathrm{EL}[\mathcal{H}(\mathrm{x}), \mathrm{y}(\mathrm{x})|\mathrm{x}] \mathrm{p}(\mathrm{x}) \mathrm{dx}, \end{aligned} \tag{3}$$

where $\mathrm{EL}[\ldots|\mathrm{x}]$ is called expected loss, and $p(x)$ is the marginal probability of the instance x.

It formally shows that the overall generalization error is the average of the expected error over the input distribution. Therefore, in order for minimizing the generalization error, we derive the active learning criterion, GEM, to select the example with the largest error:

$$x^* = \arg\max_{\mathrm{x}} \mathrm{EL}[\ldots|\mathrm{x}] \mathrm{p}(\mathrm{x}). \tag{4}$$

As shown above, this sample selection criteria balances two types of examples: examples with the largest expected loss and examples with highest probability. Moreover, if x is uniformly distributed, the example having the largest expected loss can be treated as the most informative one. If $p(x)$ is non-uniform, then data density could be helpful for selecting examples to minimize generalization error. Previous work with similar idea has been studied in classification (Nguyen and Smeulders 2004).

3.2 Link to density-weighted active learning for classification

In this subsection, we focus on the binary discrimination problem, i.e., the class labels are represented as $\{1,-1\}$. It can be generalized to multi-class problem. A typical density-weighted active sampling framework for classification can be formulated as Settles (2012):

$$x^* = \arg\max_{\mathrm{x}} \phi(\mathrm{x}) \left(\frac{1}{\mathrm{N}} \sum_{\mathrm{i}=1}^{\mathrm{N}} \mathrm{sim}(\mathrm{x}, \mathrm{x_i}) \right)^{\beta}, \tag{5}$$

where $\phi(x)$ stands for the informativeness of x (e.g., the uncertainty), and the second term sim(.) approximates the density of x by its average similarity to other data points in the input space (usually approximated with the large pool set). A variant of this method first

clusters the data, and then computes the average similarity to instances within the same cluster, which tends to choose the centroid examples for labeling.

Here, we show that there is a strong connection between the above density-weighted sampling strategy and GEM for classification with the following two conditions:

- The loss function is the 0/1 loss:

$$\mathcal{L}[\mathcal{H}(x), y(x)] = \mathbf{1}(\mathcal{H}(\mathbf{x}) \neq \mathbf{y}(\mathbf{x})), \tag{6}$$

  where $\mathbf{1}(.)$ is the indicator function.
- The Gaussian kernel is used as the similarity measure:

$$\mathrm{sim}(\mathrm{x}, \mathrm{x_i}) = \exp(-\frac{||\mathrm{x} - \mathrm{x_i}||^2}{2\lambda^2}). \tag{7}$$

Under the 0/1 loss, the optimal prediction is given as: $\hat{y}(x) = \arg\max p(y|x; \mathcal{D})$ and the expected 0/1 loss turns out to be: $\min\{\mathrm{p(y} = 1|\mathrm{x}; \mathcal{D}), \mathrm{p(y} = -1|\mathrm{x}; \mathcal{D})\}$, which is maximum when $p(y = 1|x; \mathcal{D}) = p(y = -1|x; \mathcal{D}) = 0.5$, i.e., the model is most uncertain about the class label (Long et al. 2010). Clearly, the probability density of x estimated by KDE is proportional to its similarity computed with Gaussian kernel function. Thus, we obtain

$$\mathrm{EL}[\ldots|\mathrm{x}] = \phi(\mathrm{x}), \tag{8}$$

$$p(x) \propto \frac{1}{N}\sum_{i=1}^{N} \mathrm{sim}(\mathrm{x}, \mathrm{x_i}). \tag{9}$$

Putting together Eqs. (8) and (9), we have

$$\arg\max_{\mathrm{x}} \mathrm{EL}[\ldots|\mathrm{x}]\mathrm{p}(\mathrm{x}) = \arg\max_{\mathrm{x}} \phi(\mathrm{x}) \left(\frac{1}{\mathrm{N}}\sum_{\mathrm{i}=1}^{\mathrm{N}} \mathrm{sim}(\mathrm{x}, \mathrm{x_i})\right)^{\beta}. \tag{10}$$

As a result,the proposed GEM framework is equivalent to the widely adopted density-weighted active learning strategy in classification problems with 0/1 loss and Gaussian kernel based similarity.

3.3 Link to expected loss optimization

Assuming x is uniformly distributed, then $p(x)$ is identical for all x. We have

$$x^* = \arg\max_{\mathrm{x}} \mathrm{EL}[\ldots|\mathrm{x}]\mathrm{p}(\mathrm{x}) = \arg\max_{\mathrm{x}} \mathrm{EL}[\ldots|\mathrm{x}], \tag{11}$$

which is equivalent to the ELO framework proposed in Long et al. (2010). Consequently, the ELO framework can be viewed as a special case of GEM. Furthermore, If $p(x)$ is non-uniform, the only difference between GEM and ELO is data density.

4 Active learning for ranking with density

Based on the above derivation, we can see that *expected loss* and *probability density* are two most essential components under the GEM framework. In this section, we first provide a brief introduction to the expected loss in ranking (Long et al. 2010). Then, we focus on the probability density estimation for ranking data taking particular consideration of data

dependence. Finally, we apply the proposed GEM framework to ranking in designing our active learning algorithms at both the query level and the document level, and a two stage algorithm is derived.

4.1 Expected loss for ranking

For ranking, we are only interested in the ranking order of data samples. Given a query q and its associated document lists, we denote an ordered permutation of the document lists as π. The expected loss of q can be formally expressed as:

$$\mathrm{EL}[\ldots|\mathrm{q}] = \min_{\pi} \int_{\mathcal{Y}} \mathcal{L}(\pi, \mathrm{y}) \mathrm{p}(\mathrm{y}|\mathrm{q}; \mathcal{D}) \mathrm{dy}, \tag{12}$$

where $\mathcal{L}(\pi, y)$ quantifies the loss in ranking according to π if the true relevance scores are given by y.

For document level expected loss, by considering the prediction distribution for the i-th query–document pair, the expected loss of the i-th pair ($\mathbf{d_i}$, $\mathbf{q}$) can be written as:

$$\mathrm{EL}[\ldots|(\mathbf{d_i}, \mathbf{q})] = \int_{\mathcal{Y}^{\mathbf{i}}} \min_{\pi} \int_{\mathcal{Y}_{\mathbf{i}}} \mathcal{L}(\pi, \mathbf{y}) \mathbf{p}(\mathbf{y}|\mathbf{q}; \mathcal{D}) \mathbf{dy_i dy^i}, \tag{13}$$

where y_i denotes the relevance judgement of i-th document $\mathbf{d_i}$ associated with the query q, and y_i is the vector y after removing y_i.

If the ranking metric is discounted cumulative gain (DCG) (Jarvelin and Kekalainen 2000), the associated loss is the difference between the DCG for that ranking and the largest DCG:

$$\mathcal{L}(\pi, \mathrm{y}) = \max_{\pi^*} \mathrm{DCG}(\pi^*, \mathrm{y}) - \mathrm{DCG}(\pi, \mathrm{y}), \tag{14}$$

where

$$\mathrm{DCG}(\pi, \mathrm{y}) = \sum_{\mathrm{i}} \frac{2^{\mathrm{y_i}} - 1}{\log(1 + \pi(\mathrm{i}))}. \tag{15}$$

More practical details about the expected loss computation can be found in Long et al. (2010)

4.2 Probability density estimation for ranking data

We estimate the data density based on the classical KDE (Silverman 1986), also known as Parzen Windows. The kernel-based density estimation has been successfully applied in the information retrieval (IR) domain. For example, Wang et al. (2008) use the kernel-based approach for acquiring the probabilities of user-item pairs using cosine similarity, which has been shown to produce highly accurate predictions for collaborative filtering. In this subsection, we first estimate the probability density at query level, and then use the data dependence to infer query–document pair's density.

4.2.1 Estimating the probability density of queries

In active learning settings, the volume of the pool set is much larger than that of the labeled data. Hence the sample density can be approximately estimated with the pool set. In the

context of ranking, focusing on the unlabeled pool set, we are given a set of unlabeled queries $Q = \{q_1, q_2, \ldots, q_m\}$. Each query is associated with a list of documents $\{\mathbf{d_1}, \mathbf{d_2}, \ldots, \mathbf{d_n}\}$, and each query–document pair is represented as a p-dimension feature vector $< f_1, f_2, \ldots, f_p >$.

The KDE method, which has been extensively employed for nonparametric density estimation, constructs the probability density by locating kernels at each of the observed data point. However, the problem of applying KDE to estimate the query density lies in the fact that KDE is derived to estimate the probability of a single data point whereas a query is a set represented by its retrieved documents. Thus we cannot estimate the probability of queries with KDE directly.

Motivated by recent work on local ranking (Banerjee et al. 2009), which introduced an effective representation of queries by feature aggregation, we compress each query into a query feature vector by aggregating all of the retrieved documents, and then utilize KDE to estimate the probability of query. In the literature, the feature aggregation technique to represent a query has been extensively adopted in many ranking applications, e.g., local ranking (Banerjee et al. 2009), ranking adaptation (Cai et al. 2011), and query ranking (Geng et al. 2008).

Similar to local ranking (Banerjee et al. 2009), we define three types of feature aggregations as follows:

$$\textbf{Mean}: \quad \mu_i = \frac{1}{n}\sum_{\mathbf{d}\in\mathbf{q}} f_i, \quad \boldsymbol{\mu} \in \mathbb{R}^p, \tag{16}$$

$$\textbf{Variance}: \quad \sigma_i^2 = \frac{1}{n}\sum_{\mathbf{d}\in\mathbf{q}} (f_i - \mu_i)^2, \quad \boldsymbol{\sigma^2} \in \mathbb{R}^p, \tag{17}$$

$$\textbf{Skewness}: \quad \varphi_i = \frac{1}{n\sigma_i^3}\sum_{\mathbf{d}\in\mathbf{q}} (f_i - \mu_i)^3, \quad \boldsymbol{\varphi} \in \mathbb{R}^p, \tag{18}$$

where f_i stands for the i-th feature from the document $\mathbf{d}$ related to the query q. With aggregated features, we construct the query feature vector $\mathbf{q}$ as:

$$\mathbf{q} = < \underbrace{\mu_1, \ldots, \mu_p}_{\boldsymbol{\mu}}, \underbrace{\sigma_1^2, \ldots, \sigma_p^2}_{\boldsymbol{\sigma^2}}, \underbrace{\varphi_1, \ldots, \varphi_p}_{\boldsymbol{\varphi}} >, \tag{19}$$

where $\mathbf{q}$ is a p^*-dimension vector, and $p^* = 3p$.

After we create the query vectors with feature aggregation, we utilize KDE to estimate the probability density of queries:

$$p(\mathbf{q}) = \frac{\mathbf{1}}{|\mathbf{Q}|\lambda^{\mathbf{p^*}}} \sum_{\mathbf{i=1}}^{\mathbf{|Q|}} \mathbf{K}\left(\frac{\mathbf{1}}{\lambda}(\mathbf{q} - \mathbf{q_i})\right), \tag{20}$$

where $|Q|$ represents the number of queries in the pool set, and λ is the kernel width (the smoothing parameter). In this study, the Gaussian kernel is chosen as the kernel function $K(.)$:

$$K(x) = (2\pi)^{-p_x/2}\exp\left(-\frac{1}{2}\mathrm{xTx}\right), \tag{21}$$

which is a symmetric kernel with its value smoothly decaying away from the kernel center. Combining (20) and (21), the probability density for the query $\mathbf{q}$ can be estimated with:

$$p(\mathbf{q}) = \frac{\mathbf{1}}{|\mathbf{Q}|(\mathbf{2}\pi\lambda^{\mathbf{2}})^{\mathbf{p}^*/\mathbf{2}}} \sum_{\mathbf{i}=\mathbf{1}}^{|\mathbf{Q}|} \exp\left(-\frac{||\mathbf{q}-\mathbf{q_i}||^{\mathbf{2}}}{2\lambda^2}\right). \tag{22}$$

For KDE, the choice of the kernel width λ has a big impact on probability estimation. Previous studies have revealed that the optimal kernel width λ^* can be determined through minimizing the mean square integrated error (MSIE). For the Gaussian kernel function, we choose its kernel width λ based on the result proposed in Silverman (1986).

4.2.2 Estimating the probability density of query–document pairs

Taking particular consideration of the dependency relationship in ranking data, i.e., the query–document pairs are conditionally independent given a query, we infer the probability density of a query–document pair ($\mathbf{d}$, $\mathbf{q}$) using the chain rule:

$$p(\mathbf{d}, \mathbf{q}) = \mathbf{p}(\mathbf{d}|\mathbf{q})\mathbf{p}(\mathbf{q}), \tag{23}$$

where $p(\mathbf{d}|\mathbf{q})$ is estimated in a similar manner with KDE as before:

$$p(\mathbf{d}|\mathbf{q}) = \frac{\mathbf{1}}{|\mathbf{D_q}|(\mathbf{2}\pi\lambda^{\mathbf{2}})^{\mathbf{p}/\mathbf{2}}} \sum_{\mathbf{i}=\mathbf{1}}^{|\mathbf{D_q}|} \exp\left(-\frac{||\mathbf{d}-\mathbf{d_i}||^{\mathbf{2}}}{2\lambda^2}\right), \tag{24}$$

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

4.2.3 Computational complexity for density estimation

Here, we analyze the computational complexity of density estimation for query level and document level, respectively.

For query level probability estimation, there are two main operations: query vector construction, and query density estimation. The time is $O(p|D_q|)$ to construct a single query feature vector. Given query vectors, the time to estimate the probability of a single query is $O(p^*|Q|)$. Therefore, the total computation time required for the query density estimation is $O(p|D_q||Q| + p^*|Q|^2)$.

Given the queries' estimated probability, the computation time to estimate each single query–document pair's probability is $O(p|D_q|)$, and thus the time cost for all the unlabeled query–document pairs is $O(p|D_q|^2|Q|)$. Consequently, the total computation time needed for the document level density estimation is $O(p|D_q||Q| + p^*|Q|^2 + p|D_q|^2|Q|)$.

As the ranking data is usually collected with top-k retrieval in practice, the number of documents per query $|D_q|$ is actually very small. Thus the proposed density estimation method is acceptable for medium-scale ranking applications since the time cost quadratically increases with the number of queries $|Q|$.

4.3 GEM for active learning in ranking

In ranking problems, due to the query–document structure, the generalization error can be categorized into two levels: the query level error and the document level error. Hence, GEM is derived at different levels to minimize corresponding errors. Here, we apply the proposed GEM framework to ranking in designing our algorithms at both the query level and the document level, and a two stage algorithm is further derived.

4.3.1 GEM at query level

Query level active learning chooses all documents associated with the selected query. By applying the proposed GEM principle, our query level active learning algorithm can be expressed as:

$$q^* = \arg\max_{\mathrm{q}} \mathrm{EL}[\ldots|\mathrm{q}]\mathrm{p}(\mathbf{q}), \tag{25}$$

where q^* denotes the selected query. The query level active learning algorithm can be derived by combining (12) and (22). The corresponding pseudo-code for query sampling is shown in Algorithm 1.

Algorithm 1 GEM at the query level

Input: the small labeled data set, the unlabeled pool set, the ranking model $\mathcal{H}$ trained with the labeled data set.

1: **for** each q in pool set **do**
2: Calculate the expected loss of the query via Eq. (12).
3: Construct the query feature vector $\mathbf{q}$ via Eq. (19).
4: Estimate the probability of the query via Eq. (22).
5: **end for**
6: Select the q^* having the largest expected loss weighted by density.

Output: the query q^*.

4.3.2 GEM at document level

Document level active learning selects the query–document pair independently. Similar to the query level sampling strategy, our document level data selection function based on the GEM framework may be formulated as:

$$(\mathbf{d}^*, \mathbf{q}) = \arg\max_{(\mathbf{d},\mathbf{q})} \mathrm{EL}[\ldots|(\mathbf{d}, \mathbf{q})]\mathbf{p}(\mathbf{d}, \mathbf{q}), \tag{26}$$

where $\mathbf{d}^*$, $\mathbf{q}$ denotes the selected query–document pair. Putting together (13) and (23), the document level active sampling algorithm is derived. The pseudo-code for document sampling is given in Algorithm 2.

Algorithm 2 GEM at the document level

Input: the small labeled data set, the unlabeled pool set, the ranking model $\mathcal{H}$ trained with the labeled data set.

1: **for** each $(\mathbf{d}, q)$ in pool set **do**
2: Calculate the expected loss of the query-document pair via Eq. (13).
3: Estimate the probability of the query-document pair via Eq. (23).
4: **end for**
5: Select the $(\mathbf{d}^*, q)$ having the largest expected loss weighted by density.

Output: the query-document pair $(\mathbf{d}^*, q)$.

4.3.3 GEM at two stage

Both query level active learning and document level active learning have their own disadvantages. The query level data sampling selects all documents associated with a query. It may include some non-informative documents because there are usually a large number of

documents related to a selected query, especially in Web search ranking applications. Since the quality of a ranking model is determined mainly by the top ranked documents, most of them are non-informative. The document level sampling ignores the query–document structure and selects documents independently. This sampling strategy simply ignores the query–document structure and the data dependency relationship, and hence the result may not be optimal. For example, only one document is selected for a query, which is not a good example in ranking learning.

To address this problem, Long et al. (2010) 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 for each selected queries. This two stage strategy matches with the realistic assumption for ranking that queries are independent and query–document pairs are conditionally independent given a query. In this study, we follow their two stage strategy in designing our algorithm.

5 Experiments

5.1 Data sets

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[1], a benchmark data set on learning-to-rank. Each query–document pair is represented by 46 features, including both the low-level features such as term frequency, inverse document frequency and their combinations, and the high-level features such as BM25 and PageRank. The query–document pairs are labeled with a three-level relevance judgment: {Bad, Fair, Good}. The second data set is the Web search data set from a commercial search engine (denoted as WEB-SEARCH), and each query–document pair is represented with 36 features. The relevance is judged with five-level relevance scheme: {Bad, Fair, Good, Excellent, Perfect}. The features from both of the two data sets have been normalized with the function below:

$$f^N_{(i,j)} = \frac{f_{(i,j)} - min_{i\in n}\{f_{(i,j)}\}}{max_{i\in n}\{f_{(i,j)}\} - min_{i\in n}\{f_{(i,j)}\}} \tag{27}$$

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.

Both of the two data sets are randomly split into three disjoint parts at the query level to simulate the active learning scenario. The statistics of the two learning-to-rank data sets are listed in Table 1. In practice, the initial training set is often collected by depth-k retrieval with randomly selected queries from the underlying distribution.

5.2 Experimental settings

Note that in expected loss calculation it needs to estimate the relevance score. Since pairwise and listwise approaches would not provide such relevance estimates, we focus in this work on pointwise methods. Similar to ELO (Long et al. 2010), we use Gradient Boosting Decision Tree (GBDT), a classical pointwise learning-to-ranking approach which has been recently employed as the state-of-art method in many ranking tasks (Chapelle

[1] http://research.microsoft.com/en-us/um/beijing/projects/letor/.

Table 1 The statistics of the two learning to rank data sets

Data set	AL data set	#Queries	#Documents	#Features
LETOR 4.0	Base set	60	2000	46
	Pool set	1940	66,383	
	Test set	297	10,262	
WEB-SEARCH	Base set	200	4102	36
	Pool set	3000	60,609	
	Test set	564	11,363	

et al. 2011), to train our ranking functions. More details about GBDT can be found in Friedman (2001).

We first experiment on the density estimation, the added computation step for sample selection, to test the efficiency of the KDE-based method in ranking tasks. Then, the proposed GEM algorithms are compared against the following three competitors to validate its effectiveness at different sampling levels:

- ELO (Long et al. 2010): The ELO algorithm chooses examples with the largest expected DCG loss, representing the state-of-the-art.
- CLUSTER: The CLUSTER method first clusters the data, and then selects the centroid examples for labeling. This is the central idea in previous work on density-based active learning for classification (Xu et al. 2003). In this study, we use the classical K-means as the cluster algorithm. For the query level sampling, the cluster algorithm is performed with the aggregated feature vector.
- Random (denoted as RAND): The random selection, which is widely used in practice, represents a baseline.

In this work, 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 empirically fix the number of documents selected for each query to be 10 based on the results from Yilmaz and Robertson (2009). After these examples have been added to the training set, the ranking models are re-trained and evaluated on the separate test set. To avoid random fluctuation, we repeat each experiment for 10 runs and report the average results.

5.3 Evaluation metrics

For evaluation, we use Discounted Cumulative Gain (DCG) (Jarvelin and Kekalainen 2000), the dominant performance metric of ranking, to evaluate the performance of ranking models on the test set. DCG at rank n for a given query is computed as:

$$\mathrm{DCG@n} = \sum_{\mathrm{r}=1}^{\mathrm{n}} \frac{2^{\mathrm{l(r)}} - 1}{\log(1 + \mathrm{r})}, \tag{28}$$

where $l(r)$ stands for the relevance score of the document associated with the query q at the rank r. Since users of a search engine are only interested in the top-k retrieved documents of a query, we use DCG@3, DCG@5, and DCG@10 as the performance measurements.

5.4 Computation time for density estimation

We report the CPU run time of the proposed KDE-based method for density estimation in ranking on a standard desktop computer with 2.27 GHz CPU (Intel Xeon) and 4 GB of memory. Table 2 reports the running time, together with the information of the pool set.

We observe that, for the query level density estimation, it takes 15 s on the LETOR 4.0 data set, and 30 s on the WEB-SEARCH data set respectively using an un-optimized C++ implementation. For the document level density estimation, the run time is 192 s on the LETOR 4.0 data set, and 244 s on the WEB-SEARCH data set. These above run times are fairly reasonable in practical ranking applications.

5.5 Comparison results and interpretations

In this subsection, we compare our GEM algorithms with several other active learning algorithms and present the experimental results. We denote the query level, the document level and the two stage GEM algorithm as GEM-Q, GEM-D and GEM-QD, respectively.

5.5.1 Query level active learning

We first compare the GEM-Q algorithm with the following three active learning algorithms: query-level ELO (denoted as ELO-Q), query-level CLUSTER (denoted as CLUSTER-Q), and query-level RAND (denoted as RAND-Q).

Figures 4, 5 and 6 plot the learning curves of the four query level active learning algorithms on the two data sets measured by DCG@3, DCG@5 and DCG@10. The X-axis represents the number of iterations for active learning, and the Y-axis stands for the DCG scores. For all four active learning algorithms, the DCG increases with the number of iterations. This observation matches the intuition that the quality of a ranking function is positively correlated with the number of examples in the training set. We observe that GEM-Q and ELO-Q perform better than CLUSTER-Q and RAND-Q. A possible explanation is that GEM-Q and ELO-Q optimize the DCG loss that is directly related to the ranking metrics (i.e., DCG@3, DCG@5, and DCG@10) used to evaluate the ranking functions. Furthermore, the proposed GEM-Q algorithm consistently outperforms ELO-Q during the data selection process except for one check point on LETOR 4.0 in terms of DCG@5, demonstrating that data density can provide useful information for active learning. In addition, we are interested in the comparison between GEM-Q and CLUSTER-Q because both of them aim to consider the data distribution in active learning. We observe that GEM-Q significantly outperforms CLUSTER-Q on these two data sets. The poor performance of CLUSTER-Q may be due to the following two reasons: (1) the ranking data may have no clear cluster structures, and (2) the number of clusters is not appropriately determined since the ground truth is unknown.

Table 2 The CPU running time of the proposed KDE-based method for density estimation (s)

Data set	Size of pool set (#Queries \ #Docs \ #Features)	Query level	Document level
LETOR 4.0	1940 \ 66,383 \ 46	15	192
WEB-SEARCH	3000 \ 60,609 \ 36	30	244

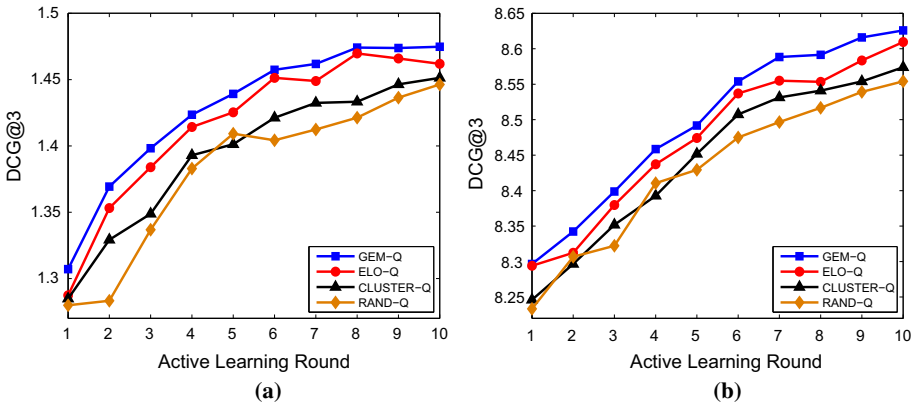

Fig. 4 Comparison results of GEM-Q, ELO-Q, CLUSTER-Q and RAND-Q on the LETOR 4.0 (**a**) and WEB-SEARCH (**b**) data set in terms of DCG@3

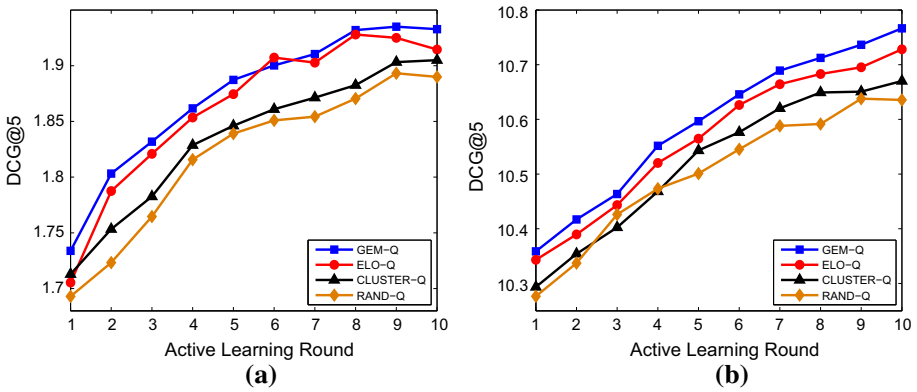

Fig. 5 Comparison results of GEM-Q, ELO-Q, CLUSTER-Q and RAND-Q on the LETOR 4.0 (**a**) and WEB-SEARCH (**b**) data set in terms of DCG@5

5.5.2 Document level active learning

In this subsection, we show that the GEM-D algorithm effectively selects the most informative documents to improve the ranking performance. We compare the proposed GEM-D algorithm with the following three active learning algorithms: document-level ELO (denoted as ELO-D), document-level CLUSTER (denoted as CLUSTER-D), and document level-RAND (denoted as RAND-D). The document level active learning is similar to the traditional active learning framework, which ignores the query–document structure in ranking data and chooses the examples independently.

The comparison results of the four document level sampling algorithms on the LETOR 4.0 data set are presented in Figs. 7a, 8a and 9a. As shown in the figures, GEM-D performs the best among the four data selection methods, and ELO-D performs better than the other two methods with a huge performance gap, especially at the latter of the sampling process. We explain this phenomenon as follows. As both GEM and ELO require the

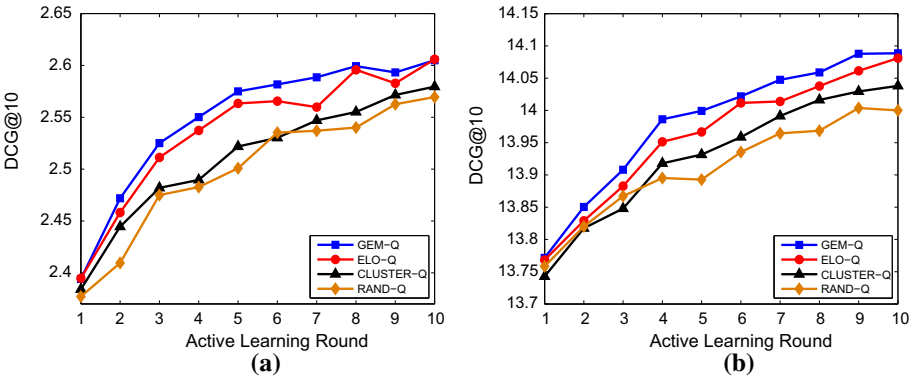

Fig. 6 Comparison results of GEM-Q, ELO-Q, CLUSTER-Q and RAND-Q on the LETOR 4.0 (**a**) and WEB-SEARCH (**b**) data set in terms of DCG@10

function ensemble learned from the bootstrap examples to compute the prediction distribution, their performance are highly correlated with the amount of training set available. With the size of training set increasing, they may have very high prediction accuracy in loss computation, leading to superior performance. Again, GEM-D is observed to consistently outperform CLUSTER-D during the entire data selection process. Similar results are obtained when comparing the four algorithms on the real WEB-SEARCH data set (as shown in Figs. 7b, 8b, 9b).

5.5.3 Two stage active learning

Here, we compare the proposed GEM-QD algorithm with two-stage ELO (denoted as ELO-QD), two-stage CLUSTER (denoted as CLUSTER-QD), and two-stage RAND (denoted as RAND-QD). As mentioned previously, we fix the number of documents selected for each query to be 10 for all four two-stage algorithms based on the conclusions of Yilmaz and Robertson (2009).

Figures 10, 11 and 12 present the comparison results for the four two-stage algorithms on the LETOR 4.0 and WEB-SEARCH data set. We observe that among the four methods, the proposed GEM-QD algorithm achieves the highest DCG scores, and ELO-QD ranks the second. The results indicate that the proposed GEM-QD algorithm can select more informative queries and more informative documents than the other three sampling algorithms.

5.5.4 Significance test

To better test the effectiveness of the proposed GEM algorithms, we conduct the significance test on the comparisons.

Tables 3 and 4 present the results of one tailed paired T-test of GEM versus the competitors at different sampling levels on LETOR 4.0 and WEB-SEARCH, respectively. We compare the DCG over 10 runs at each evaluation point and present the percentage of evaluation points at which GEM statistically outperforms the competitors, denoted as Win%. The results show that GEM performs significantly better than compared algorithms in most cases.

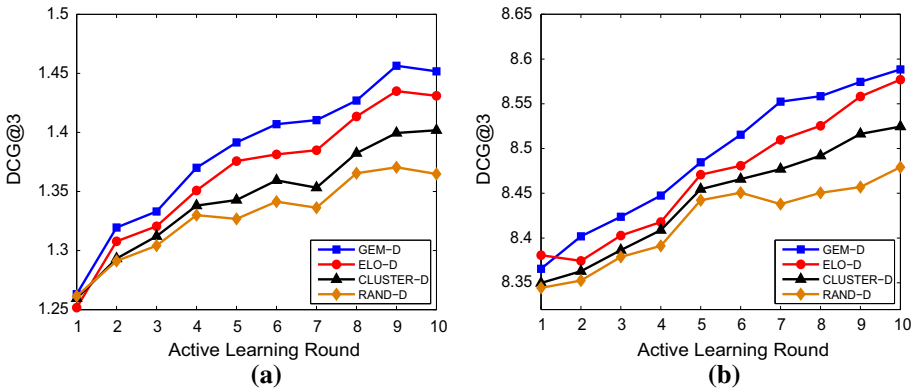

Fig. 7 Comparison results of GEM-D, ELO-D, CLUSTER-D and RAND-D on the LETOR 4.0 (**a**) and WEB-SEARCH (**b**) data set in terms of DCG@3

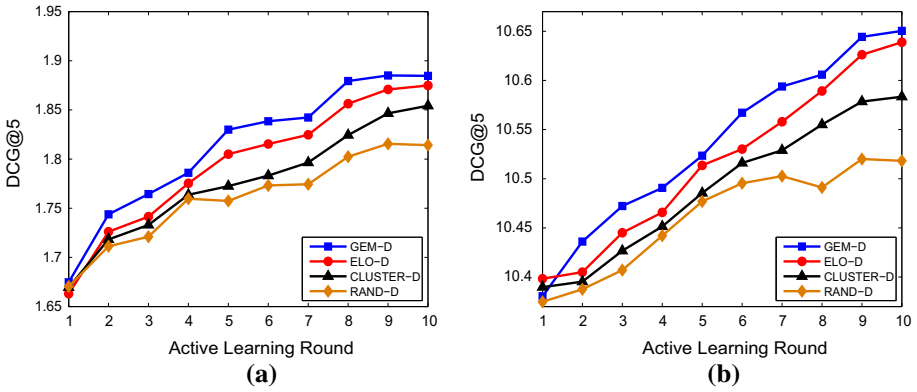

Fig. 8 Comparison results of GEM-D, ELO-D, CLUSTER-D and RAND-D on the LETOR 4.0 (**a**) and WEB-SEARCH (**b**) data set in terms of DCG@5

5.5.5 Discussion on practical problems

In this subsection, we discuss several potentially important problems faced by practical industry.

In practice, there are some infrequent queries which are not located in the high density region of the data distribution. Due to the reason that rare queries will seriously affect the user's satisfaction, they are indeed critical for practical search engines. In this case, ordinary density-based active learning might not be appropriate for them as it aims to select representative examples in order for capturing the input distribution. However, our GEM strategy, which balances loss and data density, is expected to have the ability to handle this problem. A possible reason is as follows. Because the current ranking function learned from small initial training set less likely captures the information about rare queries, the expected loss for them may be very large, making them more likely to be selected. In principle, defining an appropriate sampling function for rare queries is cost-sensitive active learning, and we consider it as our future work.

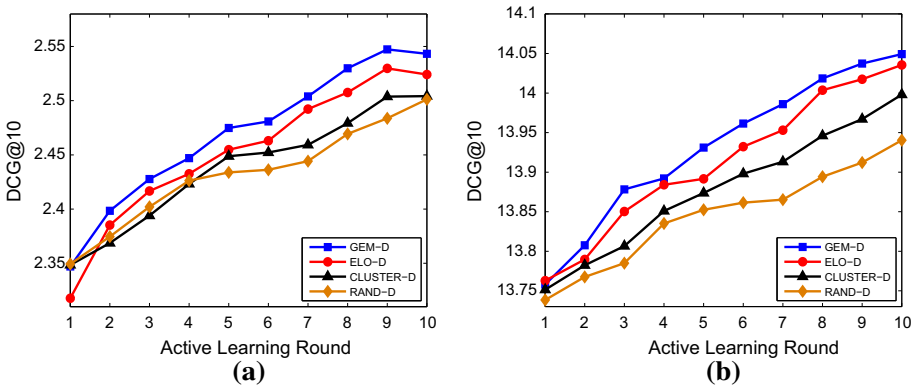

Fig. 9 Comparison results of GEM-D, ELO-D, CLUSTER-D and RAND-D on the LETOR 4.0 (**a**) and WEB-SEARCH (**b**) data set in terms of DCG@10

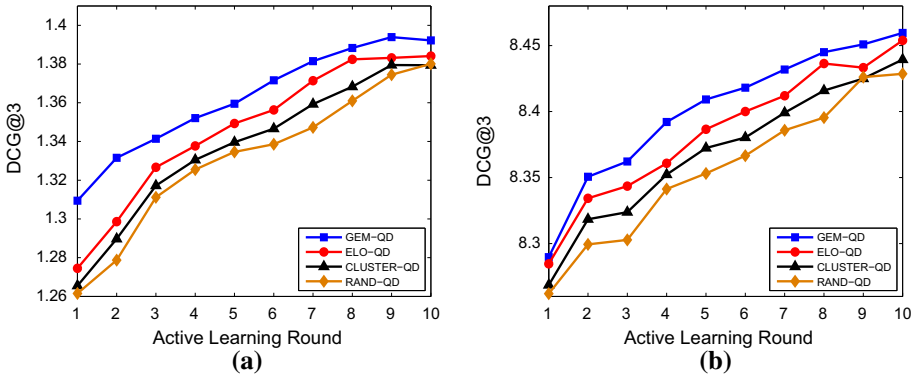

Fig. 10 Comparisons of GEM-QD, ELO-QD, CLUSTER-QD and RAND-QD on the LETOR 4.0 (**a**) and WEB-SEARCH (**b**) data set in terms of DCG@3

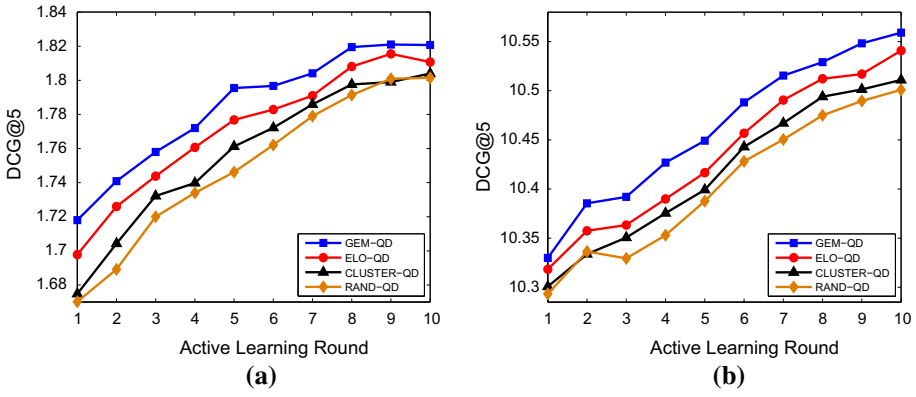

Fig. 11 Comparisons of GEM-QD, ELO-QD, CLUSTER-QD and RAND-QD on the LETOR 4.0 (**a**) and WEB-SEARCH (**b**) data set in terms of DCG@5

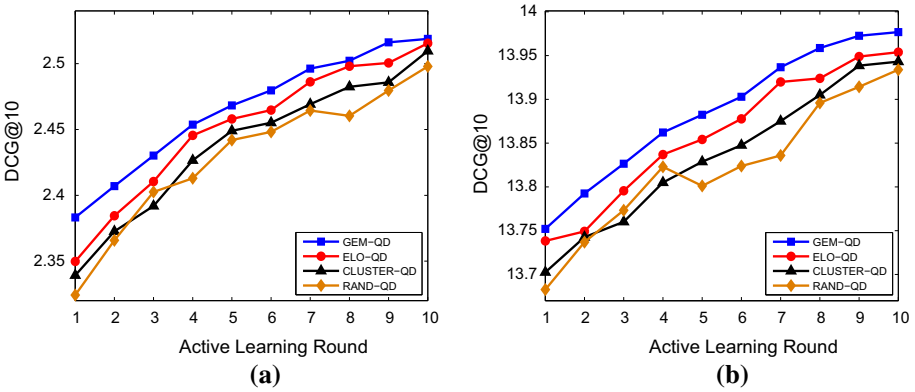

Fig. 12 Comparisons of GEM-QD, ELO-QD, CLUSTER-QD and RAND-QD on the LETOR 4.0 (**a**) and WEB-SEARCH (**b**) data set in terms of DCG@10

Table 3 Win% of GEM versus the other strategies in one-tailed paired T-test at 95 % significance level ($p < 0.05$) on LETOR 4.0 data set

Paired T-test	LETOR 4.0			Sampling level
	DCG@3 (%)	DCG@5 (%)	DCG@10 (%)	
GEM-D versus ELO-D	40	30	30	Document level
GEM-D versus CLUSTER-D	90	90	80	
GEM-D versus RAND-D	90	100	90	
GEM-Q versus ELO-Q	50	60	40	Query level
GEM-Q versus CLUSTER-Q	90	90	80	
GEM-Q versus RAND-Q	90	90	80	
GEM-QD versus ELO-QD	60	70	50	Two stage
GEM-QD versus CLUSTER-QD	100	100	90	
GEM-QD versus RAND-QD	100	100	100	

Table 4 Win% of GEM versus the other strategies in one-tailed paired T-test at 95 % significance level ($p < 0.05$) on WEB-SEARCH data set

Paired T-test	WEB-SEARCH			Sampling level
	DCG@3 (%)	DCG@5 (%)	DCG@10 (%)	
GEM-D versus ELO-D	40	20	30	Document level
GEM-D versus CLUSTER-D	80	90	80	
GEM-D versus RAND-D	80	90	70	
GEM-Q versus ELO-Q	40	50	30	Query level
GEM-Q versus CLUSTER-Q	90	90	80	
GEM-Q versus RAND-Q	100	90	90	
GEM-QD versus ELO-QD	50	40	40	Two stage
GEM-QD versus CLUSTER-QD	90	100	100	
GEM-QD versus RAND-QD	100	100	100	

Another potentially practical problem lies in data annotation. In practice, we may have multiple editors to judge the selected query–document pairs, and therefore each pair might receive different relevance judgements. To tackle this problem, Metrikov et al. (2013) introduced an effective methodology to optimize IR metric (e.g., NDCG) gains by minimizing effect of label inconsistency. For the topic of active learning, many recent studies on active learning with noisy labelers have been proposed in the literature (Zheng et al. 2010).

6 Conclusions and future work

In this article, we propose a novel active learning framework, called GEM, which theoretically incorporates the data density to minimize the generalization error. We also connect GEM to density-weighted active learning for classification to provide a uniform view.

Focusing on learning-to-rank problems, we explore information about data density using KDE, and estimate the data density at both query level and document level. Under the proposed GEM framework, we derive new algorithms at both query level and document level, and a two stage active learning algorithm is further derived. Experimental results on the LETOR 4.0 data set and a real-world Web search ranking data set from a commercial search engine have demonstrated the effectiveness of the proposed active learning algorithms.

In this study, the proposed algorithms perform batch mode active learning, i.e., selecting a batch of k informative examples in each active learning iteration. The correlation or similarity among the selected examples at each batch is not considered. Possible extension of this work is to consider the diversity of the selected data set to further minimize the labeling cost.

References

Ailon, N. (2011). Active learning ranking from pairwise preferences with almost optimal query complexity. In *Advances in neural information processing systems (NIPS' 11)* (pp. 810–818).

Aslam, J. A., Kanoulas, E., Pavlu, V., Savev, S., & Yilmaz, E. (2009). Document selection methodologies for efficient and effective learning-to-rank. In *Proceedings of the 32nd annual international ACM SIGIR conference on research and development in information retrieval (SIGIR' 09)* (pp. 468–475).

Banerjee, S., Dubey, A., Machchhar, J., & Chakrabarti, S. (2009). Efficient and accurate local learning for ranking. In *Proceedings of SIGIR' 09 workshop on learning to rank for information retrieval* (pp. 1–8).

Cai, W., & Zhang, Y. (2012). Variance maximization via noise injection for active sampling in learning to rank. In *Proceedings of the 21st conference on information and knowledge management (CIKM' 12)* (pp. 1809–1813).

Cai, P., Gao, W., Zhou, A. Y., & Wong, K. F. (2011). Query weighting for ranking model adaptation. In *Proceedings of the 49th annual meeting of the association for computational linguistics (ACL' 11)* (pp. 112–122).

Cai, P., Gao, W., Zhou, A. Y., & Wong, K. F. (2011). Relevant knowledge helps in choosing right teacher: Active query selection for ranking adaptation. In *Proceedings of the 34th annual international ACM SIGIR conference on research and development in information retrieval (SIGIR' 11)* (pp. 115–124).

Cao, Y. B., Xu, J., Liu, T. Y., & Li, H. (2006). Adapting ranking SVM to document retrieval. In *Proceedings of the 29th annual international ACM SIGIR conference on research and development in information retrieval (SIGIR' 06)* (pp. 186–193).

Chapelle, O., Shivaswamy, P., Vadrevu, S., Weinberger, K., Zhang, Y., & Tseng, B. (2011). Boosted multi-task learning. *Machine Learning*, *85*, 149–173.

Cossock, D., & Zhang, T. (2006). Subset ranking using regression. In *Proceedings of the 16th international conference on learning theory (COLT' 06)* (pp. 605–619).

Donmez, P., & Carbonell, J. G. (2008). Optimizing estimated loss reduction for active sampling in rank learning. In *Proceedings of the 25th international conference on machine learning (ICML' 08)* (pp. 248–255).

Freund, Y., Seung, H. S., Shamir, E., & Tishby, N. (1997). Selective sampling using the query by committee algorithm. *Machine Learning*, *28*, 133–168.

Freund, Y., Iyer, R., Schapire, R. E., & Singer, Y. (2003). An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, *4*, 933–969.

Friedman, J. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, *29*(5), 1189–1232.

Geng, X., Liu, T. Y., Qin, T., Arnold, A., Li, H., & Shum, H. Y. (2008). Query dependent ranking using k-nearest neighbor. In *Proceedings of the 31st annual international ACM SIGIR conference on research and development in information retrieval (SIGIR' 08)* (pp. 115–122).

Herbrich, R., Graepel, T., & Obermayer, K. (2000). *Large margin rank boundaries for ordinal regression. Advances in large margin classifiers*. Cambridge: MIT Press.

Jarvelin, K., & Kekalainen, J. (2000). IR evaluation methods for retrieving highly relevant documents. In *Proceedings of the 23th annual international ACM SIGIR conference on research and development in information retrieval (SIGIR' 00)* (pp. 41–48).

Lewis, D. D., & Gale, W. A. (1994). A sequential algorithm for training text classifiers. In *Proceedings of the 17th annual international ACM SIGIR conference on research and development in information retrieval (SIGIR' 94)* (pp. 3–12).

Long, B., Chapelle, O., Zhang, Y., Chang, Y., Zheng, Z., & Tseng, B. (2010). Active learning for ranking through expected loss optimization. In *Proceedings of the 33th annual international ACM SIGIR conference on research and development in information retrieval (SIGIR' 10)* (pp. 267–274).

McCallum, A., & Nigam, K. (1998). Employing EM and pool-based active learning for text classification. In *Proceedings of the 15th international conference on machine learning (ICML' 98)* (pp. 359–367).

Metrikov, P., Pavlu, V., & Aslam, J. (2013). Optimizing nDCG gains by minimizing effect of label inconsistency. In *Proceedings of the 35th European conference on information retrieval (ECIR' 13)* (pp. 760–763).

Nguyen, H. T., & Smeulders, A. (2004). Active learning using pre-clustering. In *Proceedings of the 21st international conference on machine learning (ICML' 04)* (pp. 623–630).

Qian, B., Wang, X., Wang, J., Li, H., Cao, N., Zhi, W., & Davisdon, I. (2013). Fast pairwise query selection for large-scale active learning to rank. In *Proceedings of the 13th international conference on data mining (ICDM' 13)* (pp. 607–616).

Roy, N., & McCallum, A. (2001). Toward optimal active learning through sampling estimation of error reduction. In *Proceedings of the 18th international conference on machine learning (ICML' 01)* (pp. 441–448).

Settles, B. (2012). *Active learning, synthesis lectures on artificial intelligence and machine learning* (Vol. 6, pp. 1–114). Morgan & Claypool.

Silva, R., Gonçalves, M. A., & Veloso, A. (2011). Rule-based active sampling for learning to rank. In *Proceedings of Eurpeon conference on machine learning and principles and practise of knowleged discovery in databases (ECML-PKDD' 11)* (pp. 240–255).

Silverman, B. W. (1986). *Density estimation for statistics and data analysis*. New York: Chapman and Hall.

Wang, J., Varies, A., & Reinders, M. (2008). Unified relevance models for rating prediction in collaborative filtering. *ACM Transactions on Information Systems (TOIS)*, *26*(3), 1–42.

Xia, F., Liu, T. Y., Wang, J., Zhang, W., & Li, H. (2008). Listwise approach to learning to rank: Theory and algorithm. In *Proceedings of the 25th international conference on machine learning (ICML' 08)* (pp. 1192–1199).

Xu, Z., Yu, K., Tresp, V., Xu, X. W., & Wang, J. Z. (2003). Representative sampling for text classification using support vector machines. In *Proceedings of ECIR' 03* (pp. 393–407).

Yilmaz, E., & Robertson, S. (2009). Deep versus shallow judgments in learning to rank. In *Proceedings of the 32nd annual international ACM SIGIR conference on research and development in information retrieval (SIGIR' 09)* (pp. 662–663).

Yu, H. (2005). SVM selective sampling for ranking with application to data retrieval. In *Proceedings of the 11st ACM SIGKDD international conference on knowledge discovery and data mining (KDD' 05)* (pp. 354–363).

Zheng, Y., Scott, S., & Deng, K. (2010). Active learning from multiple noisy labelers with varied costs. In *Proceedings of the 10th international conference on data mining (ICDM' 10)* (pp. 639–648).

Zhu, J. B., Wang, H. Z., & Tsou, B. K. (2010). Active learning with sampling by uncertainty and density for data annotations. *IEEE Transactions on Audio, Speech, and Language Processing*, *18*, 1323–1331.