Query-biased Learning to Rank for Real-time Twitter Search

Xin Zhang
School of Computer and Control Engineering
Graduate University of Chinese Academy of Sciences
zhangxin510@mails.gucas.ac.cn

Ben He, Tiejian Luo, Baobin Li
School of Computer and Control Engineering
Graduate University of Chinese Academy of Sciences
{benhe,tjluo,libb}@gucas.ac.cn

ABSTRACT

By incorporating diverse sources of evidence of relevance, learning to rank has been widely applied to real-time Twitter search, where users are interested in fresh relevant messages. Such approaches usually rely on a set of training queries to learn a *general* ranking model, which we believe that the benefits brought by learning to rank may not have been fully exploited as the characteristics and aspects unique to the given target queries are ignored. In this paper, we propose to further improve the retrieval performance of learning to rank for real-time Twitter search, by taking the difference between queries into consideration. In particular, we learn a query-biased ranking model with a semi-supervised transductive learning algorithm so that the query-specific features, e.g. the unique expansion terms, are utilized to capture the characteristics of the target query. This query-biased ranking model is combined with the general ranking model to produce the final ranked list of tweets in response to the given target query. Extensive experiments on the standard TREC Tweets11 collection show that our proposed query-biased learning to rank approach outperforms strong baseline, namely the conventional application of the state-of-the-art learning to rank algorithms.

Categories and Subject Descriptors: H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval

General Terms: Experimentation, Performance, Algorithms

Keywords

Real-time Twitter search, query-biased learning to rank, semi-supervised learning

1. INTRODUCTION

Recency is an important dimension of information need in real-time Twitter search, where users tend to be interested in fresh news and events [16]. There are multiple intrinsic features of tweets, such as user authority, mentions, retweets, hashtags, etc. Thus, the application of learning to rank by incorporating diverse sources of evidence of recency and relevance has shown beneficial in previous studies, especially as demonstrated in the TREC 2011 Microblog track [24, 21, 22]. Besides, Duan et al. employ RankSVM to rank the tweets which output the matched tweets based on their relevance to the query in content [5].

The methods mentioned above attempt to learn a "general" ranking model in the sense that the features used are common to different queries, such as the relevance score given by the content-based retrieval model, the user authority, and so on. Therefore, the ranking model learned from the training data is assumed to be able to generalize to the test data. On the other hand, since the user's information need is a query-dependent notion, different queries have their own unique characteristics and aspects, hence the need for the query-biased modeling to capture the differences and boundaries between queries. Indeed, a few previous approaches have been proposed to take query differences into consideration during the learning process to improve the effectiveness of learning to rank, for instance [32, 30, 11, 28].

Inspired by the above mentioned studies on the query differences, in this paper, we construct a combined learning to rank framework by integrating a general ranking model with the query-biased model that takes the query differences into account. In our proposed combined framework, the general ranking model is learned from *training* queries by the conventional learning to rank approach. In addition, we propose to learn a query-biased ranking model by a transductive learning algorithm [27] based on the pseudo relevance information. Finally, the query-biased model is combined with the general model to produce the final tweet ranking for the target queries.

The major contributions of the paper are two-fold. First, we propose a semi-supervised learning algorithm to build a query-biased ranking model. Instead of treating all queries equally, we train a query-biased model based on the queries' intrinsic features through an iterative learning process. Second, we propose a learning to rank framework to combine the query-biased model with a general ranking model learned from the common features. Extensive experiments on the standard TREC Tweets11 collection [24] demonstrate the effectiveness of our proposed query-biased learning to rank approach.

The rest of the paper is organized as follows. In Section 2, we survey the existing learning to rank research on Twitter search. Section 3 gives a detail description about the method to build the learning to rank framework by combining the

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

general learning to rank algorithms and the query-biased algorithm, which is evaluated in Section 4. Finally, we conclude this research and suggest future research directions.

2. RELATED WORK

In real-time Twitter search, freshness of the retrieval results is an important aspect of information need. The user experience can be negatively affected when failing to recognize the temporal aspect of the query [4]. Therefore, there have been quite a few previous research on real-time Twitter search to retrieve not only relevant but also fresh tweets [7, 17]. In microblogging services, there are many aspects and characteristics of the social features [16, 2, 5]. All these characteristics and aspects have their potential influence on the relevance of the tweets. By using the learning to rank approach, it is easy to properly integrate the variety of features into the retrieval model [24, 21, 22]. Learning to rank is a family of algorithms that automatically construct a model or function to rank objects. The major advantage of learning to rank is its flexibility in incorporating diverse sources of evidence into the process of retrieval [19].

In addition to the above described approaches, there are other methods proposed in the TREC 2011 Microblog track [13, 18, 20]. Besides, there are research applying semi-supervised learning algorithms to facilitate learning to rank with only limited training data available [33, 6].

However, most of learning to rank approaches treat all the queries equally during the learning and ranking processes. As a result, the unique aspects of different queries are ignored, which may potentially hurt the retrieval effectiveness. Recently, there have been efforts to take query differences into consideration during the learning process to improve the ranking functions. Zheng et al. put forward a minimum effort optimization method by considering all the entire training data within a query during each iteration [32]. Wu et al. propose a listwise query-level regression method, called ListReg, by using the neural network to model the ranking function and gradient descent for optimization [30]. Geng et al. put forward a K-Nearest Neighbour method to learn different ranking functions based on different properties of queries [11]. Besides, Veloso et al. propose a novel method to uncover the patterns or rules in the training data by generating association rules on a demand-driven basis at the query time [28].

We argue that the improvement brought by learning to rank can be further improved if the differences that exist in the diverse queries are considered during the retrieval process. To address this problem, in this paper, we propose a combined framework that makes the use of both the common features and query-specific features for learning to rank, as introduced in the next section.

3. THE COMBINED LEARNING TO RANK FRAMEWORK

We propose a learning to rank framework that utilizes both the common features of Twitter messages, and the query-specific aspects that differentiate between queries. The proposed framework combines a general ranking model and a query-biased ranking model. More specifically, the general ranking model is learned from the training instances, represented by the features common to different queries. The query-biased model is learned from the query-specific features by a semi-supervised learning algorithm. The two models are integrated by a linear combination as follows:

$$Score_{final}(d,Q) = Score_{LTR}(d,Q) + \beta \cdot Score_{QLTR}(d,Q)$$

where $Score_{final}(d,Q)$ is the final score of tweet d for the given query Q; $Score_{LTR}(d,Q)$ is the score given by the general ranking model; $Score_{QLTR}(d,Q)$ is the score given by the query-biased model. The setting of the parameter β is obtained by training on different folds of cross-validation.

3.1 General Ranking Model

3.1.1 Tweet Features Extraction

Our features are organized around the basic entities for each query-tweet tuple to distinguish between the relevant and irrelevant messages, some of which have been widely used in previous work on Twitter search [5, 22, 21, 33]. More specifically, in addition to the five types of features which were used in our previous work [33], we exploit the sentiment features in this paper.

Sentiment refers to those features that indicate the opinions embodied in the given tweet. In our model, we extract the proportion of negative sentiment words and positive sentiment words, as defined in SentiWordNet [9], in the content of the tweet.

3.1.2 Learning to Rank Algorithms

Many learning to rank approaches have been proposed in the literature [19], which can be applied for learning the general ranking model. Among them, in this paper, we choose to use two popular learning to rank algorithms, namely RankSVM [15, 29, 12] and LambdaMART [31, 10], which has shown to be a robust algorithm for solving real world ranking problems [3].

In the learning process, after the positive and negative examples are appended to the labeled set by making using of the relevance assessments information, we empirically assign preference values according to the temporal distance between the timestamps of the tweet and the query. The larger the preference value is, the higher the tweet is relevant to the given query. This labeling strategy is mainly due to the fact that recency is a crucial factor of relevance in real-time Twitter search. The fresh tweets are more likely to be relevant than those outdated.

The target values of RankSVM define the order of the examples of each query. We arbitrarily reassign the target values of the relevant tweets with an interval of 0.5, ranging from 0.7 to 3.2, according to the temporal distance in days between the timestamps of the tweet and the query [33]. Since the target values in LambdaMART is of the integer type, we round the double target values used in RankSVM for the listwise LambdaMART approach.

3.2 Query-biased Ranking Model

3.2.1 Query-specific Tweet Representation

Since the purpose of the query-biased modeling is to utilize the query-specific characteristics to boost the retrieval performance, it is a challenging issue to select the appropriate features that are unique to the given queries to represent the tweets. In this paper, we choose to represent the tweets by the most informative terms in the pseudo relevance set, namely the top-ranked tweets in the initial retrieval. As queries are different to each other in their topical concepts, it is a natural choice to represent the query-specific aspects by the most weighted terms in the pseudo relevance set,

Input
D: initial retrieved tweets returned by a content-based retrieval model for a batch of query topics
N: the maximum number of terms to be selected to represent a document

Output
W: the selected term set

Method
Do the following for each query:
(1) Get R, the top-k ranked tweets from D
(2) Compute the KL divergence weight for each unique term in R
(3) Extract the N terms with the highest weights as the term features
(4) Represent each tweet retrieved for the given query with the selected terms and their KL divergence weights

Figure 1: The tweet representation algorithm.

which are usually assumed to be highly related to the query topics.

Figure 1 provides the algorithm used for extracting the term features for the query-specific tweet representation. In particular, all the unique terms in the top-30 tweets are taken as candidate terms, and the 10 terms with highest KL divergence weights are chosen as the query-specific features. Thus, the selected words and their corresponding KL divergence weights are used as attributes and values to represent the given tweets. Our arbitrary choice of selecting the top-10 terms from the top-30 tweets is mainly due to the fact that this setting was found to provide the best query expansion effectiveness in the TREC 2011 Microblog track, as reported in [1]. The KL divergence weight of a candidate term t in the top-k ranked tweets in the initial retrieval is computed as follows:

$$w(t, R_k) = P(t|R_k) \log_2 \frac{P(t|R_k)}{P(t|C)}$$

where $P(t|R_k)$ is the probability of generating the candidate term t from the set of top-k ranked tweets R_k, and $P(t|C)$ is the probability of generating t from the entire collection C.

Note that the relevance scores produced by query expansion have already been used in Section 3.1.1 as features to learn a general ranking model. In the query-biased model, instead of being wrapped up as an expanded query into the content-based model to produce a relevance score, the most weighted terms are treated as unique terms to represent the query-specific aspects of the tweets.

3.2.2 Transduction for Query-biased Modeling

In this paper, a query-biased transductive learning algorithm is devised to boost the retrieval performance. Transductive learning [27] is a semi-supervised method for classification by utilizing limited data using transduction, it is not necessary to generate a model to predict the label of any unobserved point during the process of learning. The idea of learning from limited training data to improve retrieval performance has been studied in a number of previous publications [14, 26]. Different from the previous work, in this paper, the query-biased transduction learning algorithm and retrieval are integrated into a learning to rank paradigm. A general description of the proposed method is given in Figure 2. The underlying idea of our proposed method is similar to that of pseudo relevance feedback [25], which assumes a high degree of relevance of the top-ranked documents. However, we determine to adopt the transduction algorithm to select the appropriate tweets, instead of just the top-ranked ones which are used in the conventional content-based query expansion models, due to the fact that the top documents do not necessarily represent an optimal feedback set. Before training, it is only required to predict the labels of a given test set examples [8]. After the initial retrieval using the content-based model, we label the top-ranked and bottom-ranked tweets are selected as the positive and negative examples, respectively, on the hypothesis that the top-ranked tweets are highly relevant to the given query topic. In contrast, the bottom-ranked are believed to some extent off-topic in their contents. A model is learned to predict the ranking of the remaining unlabeled tweets, from which the very top-ranked and bottom-ranked tweets are appended to the labeled data set. Such a process repeats until the number of iterations exceed a predefined threshold. Finally, a ranking of all the retrieved tweets are produced by the query-biased model learning by the above transduction process. All the parameters in the proposed query-biased transductive learning algorithm, for example the number of iterations (i.e. in Figure 2), are obtained by tuning on training queries.

Moreover, we define two variants of the query-biased learning as follows:

Per-query learning: for each query, a query-biased model is learned with the unique term features. Thus, it is able to distinguish the special aspects of the individual target queries. This is equivalent to the algorithm in Figure 2 when there is only one test query. However, this method may suffer from the problem of sparseness since the learning is based only on the pseudo relevance set of single queries.

Batch learning: Instead of building the query-biased model for each target query, this method treats the selected highly weighted terms of different queries as the common features for a batch of target queries. As this method takes the pseudo relevance sets of a batch of queries as the training data, it is expected to be able to deal with the sparseness problem.

We only apply RankSVM for learning a query-biased model in this paper. This is mainly due to the small amount of pseudo training data available, where the advantage of the listwise approaches such as LambdaMART is negated. The query-biased model is an on-line learning process, which takes about 17 seconds for each query.

4. EXPERIMENTS

4.1 Experimental Settings

4.1.1 Dataset and Indexing

We experiment on the Tweets11 collection, which is used for evaluating the participating real-time Twitter search systems over 50 official topics in the TREC 2011 Microblog track. Our crawl of the Tweets11 collection consists of 13,401, 964 successfully downloaded unique tweets. Standard stopword removal and Porter's stemmer are applied during indexing and retrieval. The official measure in the TREC 2011 Microblog Track, namely precision at 30 (P30), is used as the evaluation metric in our experiments. All of the indexing and retrieval experiments are conducted on an in-house version of Terrier [23].

Input
D: initial retrieved tweets returned by content-based retrieval models for a batch of target queries
U: a set of unlabeled tweets
I: a set of labeled tweets, if any
F: a set of term features representing the tweets
k: the number of top-ranked and bottom-ranked tweets to be initially added to I, if I is empty
IN: the number of iterations

Output
L: a ranked list for each query in the target queries

Method
If I is empty, add the top and bottom-k tweets in D into I as positive and negative examples, respectively
1. Do the following for IN iterations
   (1) Learn a ranking model M from I using F to represent the tweets
   (2) Use M to rank tweets in U represented by F
   (3) For all the topics in the training dataset, select the very top-ranked and bottom-ranked tweets T from U
   (4) Add T to I and remove T from U
2. Get the best effectiveness and get the number of iterations for the batch of queries or per-query in the test set
3. Gradually add the labeled examples for the query in the test set
4. Use the generated data to train a model M
5. Return a ranked list for each of the target queries by applying M

Figure 2: The query-biased transductive learning algorithm.

Table 1: Values used in grid search for LambdaMART parameter tuning on the training queries.

Parameter.	Values
Max Number of Leaves	2, 3, 5, 7, 10
Min Percentage of Obs. per Leaf	0.05, 0.12, 0.15, 0.75
Learning rate	0.01, 0.04, 0.045, 0.05, 0.07, 0.2
Sub-sampling rate	0.05, 0.1, 0.3, 0.5, 0.7, 1
Feature Sampling rate	0.1, 0.3, 0.5, 0.7, 1

4.1.2 Parameter Tuning

The choice of the hyperparameter C, the regularization parameter for RankSVM, plays an important role in the retrieval performance. C can be selected by carrying out cross-validation experiments through a grid search from 1 to 40 on the training queries.

The LambdaMART algorithm has five parameters that need to be tuned to achieve the best results. We apply a greedy boosting algorithm for the parameter tuning. Firstly, we optimize each of the five parameters, while keep the other four parameters to the default values. Then, we set the parameter that has the highest optimized retrieval performance to its optimized value. This process repeats until all the parameters are optimized. We apply grid search to optimize each parameter. Values we scan for each of the parameters are listed in Table 1.

4.1.3 Evaluation Design

The aim of our experiments is to evaluate the effectiveness of the proposed learning to rank framework which combines the general learning to rank approach and transductive query-biased method for real-time Twitter search. In particular, two state-of-the-art learning to rank approaches, namely Ranking SVM (RankSVM) [15], a classical pairwise learning to rank algorithm and LambdaMART [31], a tree-based listwise learning to rank algorithm which has the best performance in the 2010 Yahoo! Learning to Rank challenge [3], are applied. We compare our proposed combined learning to rank framework, denoted as **CombLTR**, to the general ranking model **(LTR)**, which represents the conventional application of the state-of-the-art learning to rank algorithms. Extensive experiments are conducted using RankSVM and LambdaMART, where the general ranking models are learned from the official relevance assessments from the Microblog track 2011. Using the LTR approach, we conduct experiments in different granularities of the partitioning of the train-test topics, namely the 2-fold (**LTR_2**) 5-fold (**LTR_5**),10-fold (**LTR_10**) and leave-one-out (**LTR_L1**) cross-validations, respectively. Cross-validation is a technique to estimate the performance of a predictive model. In K-fold cross-validation, the 50 topics is partitioned equally into K subsets. Among them, one subset is choosen to test the model, while the others are used for training.

4.2 Results and Discussions

The combined framework CombLTR is compared to the state-of-the-art LTR approaches, which learn the ranking models from the official relevance assessments. We examine to which extent our proposed combined framework is able to improve the retrieval effectiveness by taking query differences into consideration. 2, 5, 10-fold, and leave-one-out cross-validations are conducted for the evaluation. Our proposed approaches, using QLTR_Batch and QLTR_PerQ respectively, are compared to the general LTR approaches which utilizing the relevance assessment information in Table 2. We can see that CombLTR_Batch significantly outperforms RankSVM on 5 and 2-fold cross-validations, and outperform LambdaMART on the 10-fold cross-validation; while CombLTR_PerQ outperforms LambdaMART on 10-fold and leave-one-out cross-validations. Comparing to RankSVM, CombLTR_PerQ shows no notable improvement. In addition, a surprising observation is that, using RankSVM, CombLTR has better performance with fewer training data available, i.e. during the 2-fold and 5-fold cross-validations. The opposite is observed when using LambdaMART. We suggest that this is possibly due to the fact that RankSVM has a better ability in dealing with the sparseness problem than LambdaMART in the learning process. Besides, the granularity of cross-validation is believed as a factor affecting the retrieval effectiveness, just as Table 2 shows that CombLTR_Batchoutperform RankSVM on 5 and 2-fold cross-validations, while CombLTR_PerQ outperforms LambdaMART on 10-fold and leave-one-out cross-validations.

In summary, our proposed method, namely the combined learning to rank framework, is able to significantly improve the retrieval effectiveness when comparing with the content-based retrieval models and the popular learing to rank approaches. In addition, despite the noise during the learning of the query-biased ranking model, a series of experiments on Tweets11 demonstrate the benefit brought by the method we put forward in this paper, especially when there is only limited amount of training queries available.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed a combined learning to rank framework that utilizes both the general and query-specific evidence of relevance for the real-time Twitter search.

Table 2: Comparison of the combined framework (CombLTR) with the general ranking model (LTR).

RankSVM			LambdaMART		
Leave-one-out					
LTR_L1	CombLTR_L1_Batch	CombLTR_L1_PerQ	LTR_L1	CombLTR_L1_Batch	CombLTR_L1_PerQ
0.4016	0.4020, 0.10%	0.4020, 0.10%	0.3878	**0.4116, 6.14%**∗	**0.4116, 6.14%**∗
10-fold					
LTR_10	CombLTR_10_Batch	CombLTR_10_PerQ	LTR_10	CombLTR_10_Batch	CombLTR_10_PerQ
0.4061	0.4218, 3.87%	0.4116, 1.35%	0.3986	0.4143, 3.94%	**0.4211, 5.64%**∗
5-fold					
LTR_5	CombLTR_5_Batch	CombLTR_5_PerQ	LTR_5	CombLTR_5_Batch	CombLTR_5_PerQ
0.3980	**0.4313, 8.37%**∗	0.4109, 3.24%	0.3980	0.4102, 3.07%	0.4027, 1.81
2-fold					
LTR_2	CombLTR_2_Batch	CombLTR_2_PerQ	LTR_2	CombLTR_2_Batch	CombLTR_2_PerQ
0.3816	**0.4150, 8.75%**∗	0.3912, 2.52%	0.3946	0.3932, -0.35%	0.4007, 1.55%

In particular, a query-biased ranking model is learned by a semi-supervised transductive learning algorithm in order to better capture the characteristics of the given queries. Such a query-biased ranking model is combined with a general ranking model given by the conventional learning to rank approach to produce the final ranking of the Twitter messages, namely the tweets, in response to the user information need. Extensive experiments have been conducted on the standard Tweets11 dataset to evaluate the effectiveness of our proposed approach. Results show that the our proposed combined learning to rank approach is able to outperform strong baseline, namely the state-of-the-art learning to rank algorithms. Moreover, our study in this paper suggests the possibility of simulating a training data without involving human labels on given new queries, while achieving an effective retrieval performance by a combination with the conventional learning to rank approaches.

According to the experimental results, the number of iterations in the transductive learning process is a major factor that affects the effectiveness of the proposed approach. In the future, we plan to improve the robustness of the proposed approach by introducing an adaptive halting criterion of the iterative process.

Acknowledgements

This work is supported in part by the National Natural Science Foundation of China (61103131/F020511), the President Fund of GUCAS (Y15101FY00/Y25102HN00), and the CAS R&E Project (110700EA12).

6. REFERENCES

[1] G. Amati, G. Amodeo, M. Bianchi, A. Celi, C. D. Nicola, M. Flammini, C. Gaibisso, G. Gambosi, and G. Marcone. Fub, iasi-cnr, univaq at trec 2011. In *TREC*, 2011.

[2] M. Cha, H. Haddadi, F. Benevenuto, and K. P. Gummadi. Measuring user influence in twitter: The million follower fallacy. In *ICWSM*, 2010.

[3] O. Chapelle and Y. Yang. Yahoo! Learning to Rank Challenge Overview. In *JMLR*, 2011.

[4] A. Dong, Y. Chang, Z. Zheng, G. Mishne, J. Bai, R. Zhang, K. Buchner, C. Liao, and F. Diaz. Towards recency ranking in web search. In *WSDM*, 2010.

[5] Y. Duan, L. Jiang, T. Qin, M. Zhou, and H.-Y. Shum. An empirical study on learning to rank of tweets. In *COLING*, pages 295–303, 2010.

[6] K. Duh and K. Kirchhoff. Learning to rank with partially-labeled data. In *SIGIR*, pages 251–258, 2008.

[7] M. Efron and G. Golovchinksy. Estimation methods for ranking recent information. In *SIGIR*, pages 495–504, 2011.

[8] R. El-yaniv and D. Pechyony. Stable transductive learning. In *COLT*, pages 35–49, 2006.

[9] A. Esuli and F. Sebastiani. Sentiwordnet: A publicly available lexical resource for opinion mining. In *LREC*, pages 417–422, 2006.

[10] Y. Ganjisaffar, R. Caruana, and C. Lope. Bagging gradient-boosted trees for high precision, low variance ranking models. In *SIGIR*, pages 85–94. ACM, 2011.

[11] X. Geng, T.-Y. Liu, T. Qin, A. Arnold, H. Li, and H.-Y. Shum. Query dependent ranking using k-nearest neighbor. In *SIGIR*, 2008. ACM.

[12] R. Herbrich. Large margin rank boundaries for ordinal regression. *Advances in Large Margin Classifiers.*, pages 115–132, 2000.

[13] D. Hong, Q. Wang, D. Zhang, and L. Si. Query expansion and message-passing algorithms for trec microblog track. In *TREC*, 2011.

[14] X. Huang, Y. R. Huang, M. Wen, A. An, Y. Liu, and J. Poon. Applying data mining to pseudo-relevance feedback for high performance text retrieval. In *ICDM*, 2006.

[15] T. Joachims. Optimizing search engines using clickthrough data. In *KDD*, pages 133–142. ACM, 2002.

[16] H. Kwak, C. Lee, H. Park, and S. Moon. What is twitter, a social network or a news media? In *WWW*, pages 591–600, 2010.

[17] X. Li and W. B. Croft. Time-based language models. In *CIKM*, pages 469–475. ACM, 2003.

[18] Y. Li, Z. Zhang, W. Lv, Q. Xie, Y. Lin, R. Xu, W. Xu, G. Chen, and J. Guo. Pris at trec2011 micro-blog track. In *TREC*, 2011.

[19] T. Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, (3):225–331, 2009.

[20] S. Louvan, M. Ibrahim, M. Adriani, C. Vania, B. Distiawan, , and M. Z. Wanagiri. University of indonesia at trec 2011 microblog track. In *TREC*, 2011.

[21] D. Metzler and C. Cai. Usc/isi at trec 2011: Microblog track. In *TREC*, 2011.

[22] T. Miyanishi, N. Okamura, X. Liu, K. Seki, and K. Uehara. Trec 2011 microblog track experiments at kobe university. In *TREC*, 2011.

[23] I. Ounis, G. Amati, V. Plachouras, B. He, C. Macdonald, and C. Lioma. Terrier: A high performance and scalable information retrieval platform. In *SIGIR OSIR*, 2006.

[24] I. Ounis, C. Macdonald, J. Lin, and I. Soboroff. Overview of the TREC 2011 microblog track. In *TREC*, 2011.

[25] J. Rocchio. Relevance feedback in information retrieval. In *Prentice-Hall Englewood Cliffs*, 1971.

[26] S. Sellamanickam, P. Garg, and S. K. Selvaraj. A pairwise ranking based approach to learning with positive and unlabeled examples. In *CIKM*, pages 663–672, 2011.

[27] V. N. Vapnik. *Statistical learning theory*. New York: Wiley, 1998.

[28] A. A. Veloso, H. M. Almeida, M. A. Gonçalves, and W. Meira Jr. Learning to rank at query-time using association rules. In *SIGIR*, pages 267–274, 2008.

[29] W. W. Cohen, R. E. Schapire, and Y. Singer. Learning to order things. *Journal of Artificial Intelligence Research*, 10:243–270, 1998.

[30] J. Wu, Z. Yang, Y. Lin, H. Lin, Z. Ye, and K. Xu. Learning to rank using query-level regression. In *SIGIR*, pages 1091–1092, 2011.

[31] Q. Wu, C. Burges, K. Svore, and J. Cao. Ranking boosting and model adaptation. Technical report, Microsoft, 2008.

[32] Z. Zheng, H. Zha, and G. Sun. Query-level learning to rank using isotonic regression. In *AnnualAllerton Conference on Communication, Computing*, 2008.

[33] X. Zhang, B. He, and T. Luo. Transductive learning for real-time Twitter search. In *ICWSM*, 2012.