Ranking Method for Domain Specific Search and Building User Preference List

Manoj Chittawar
VITS, Karimnagar,AP,India

Prof. E. Ramakrishna
VITS, Karimnagar,AP,India

Abstract:

With the explosive emergence of vertical search domains, applying the broad-based ranking model directly to different domains is no longer desirable due to domain differences, while building a unique ranking model for each domain is both laborious for labeling data and time consuming for training models. In this paper, we are proposing a regularization-based algorithm called ranking adaptation through which we can adapt an existing ranking model to a new domain, so that the amount of labeled data and the training cost is reduced while the performance is still guaranteed. Our algorithm only requires the prediction from the existing ranking models, rather than their internal representations or the data from auxiliary domains. Also we use folksonomies for building user preference list (UPL) based on user's search history. A UPL is an indispensable source of knowledge which can be exploited by intelligent systems for query recommendation, personalized search, and web search result ranking etc. A UPL consist of list of concepts, and their weights, clustered together using clustering by employing Google Similarity Distance. The experiment reveals that UPL not only captures user interests but also its context and results are very promising.

Keywords: Search Engine, Information retrieval, support vector machines, data adaptation, learning to rank.

1. INTRODUCTION

In learning a ranking model with some documents labeled with their relevancies to some queries, the model is hopefully capable of ranking the documents returned to an arbitrary new query automatically. However, as the emergence of domain-specific search engines, more attentions have moved from the broad-based search to specific verticals, for hunting information constraint to a certain domain. Different vertical search engines deal with different topicalities, document types or domain specific features. Ranking Adaptation desires to adopt the model which is used to predict the ranking for a collection of documents. Though the documents are normally labeled with several relevance levels, it is still difficult to directly use classifier adaption for ranking. The reason lies in two fold: 1) in ranking, the main concerned is about the preference of two documents or the ranking of a collection of documents, which is difficult to be modeled by classification or regression; 2) the relevance levels between different domains are sometimes different and need to be aligned. In this paper, we focus on the adaptation of ranking models. Model adaptation is more desirable than data adaptation, because the learning complexity is now only correlated with the size of the target domain training set, which should be much smaller than the size of auxiliary data set. We are going to investigate three problems of ranking model adaptation: a) whether we can adapt ranking models learned for the existing broad-based search or some verticals, to a new domain, so that the amount of labeled data in the target domain is reduced while the performance requirement is still guaranteed; b) how to adapt the ranking model effectively and efficiently; c) how to utilize domain-specific features to further boost the model adaptation. The first problem is solved by the proposed ranking adaptability measure, which quantitatively estimates whether an existing ranking model can be adapted to the new domain, and predicts the potential performance for the adaptation. We address the second problem from the regularization framework and a ranking adaptation SVM (RA-SVM) algorithm is proposed. Our algorithm is a blackbox ranking model adaptation, which needs only the predictions from the existing ranking model, rather than the internal representation of the model itself or the data from the auxiliary domains. With the black-box adaptation property, we achieved not only the

flexibility but also the efficiency. To resolve the third problem, we assume that documents similar in their domain-specific feature space should have consistent rankings.
Another problem addressed in this paper is building a User Preference List which reflects user interests. User Preference List can be exploited for various purposes such as query recommendation, personalized search, content recommendation etc. The research problem is how to summarize URLs as a list of concepts, How to eliminate noise, and how to cluster concepts to construct UPL? To extract concepts for a given URL, one of the solutions is employing NLP. List of tagged URL is comprehensive and thus enable to disambiguate different meaning associated with a concept (i.e. semantic, disambiguate polysemy, synonyms, and context.). We believe that tags associated with a URL can be modeled as concepts. Moreover, to make these concepts more meaningful i.e. to associate context with them, the related concepts should be grouped together. The final list of grouped concepts with their weights is called as UPL.

2. RELATED WORK

We present some work that closely related to concept of ranking model adaptation and construction of user profile. To create ranking model that can rank the document according to the relevance to or given query, various types of models have been proposed, some of which have been successfully applied to web search engine. Classical BM25 [2] and Language Models for Information Retrieval , [1] work quite stable for the broad-based search with few parameters needing adjusted. Some rank algorithm based on machine learning technique transform the ranking problem into a pairwise classification problem, which takes a pair of document samples, with the binary label taken as the sign of the relevance difference between the two documents. e.g., Ranking SVM [3], RankBoost [4], RankNet [5]. For natural language processing, Blitzer et al. [R6] introduced a structural correspondence learning method which can mine the correspondences of features from different domains.
The Automatic construction of user profile usually deals with the observation of user browsing behavior. Kelly and Teevan [7] reviewed several possible approaches for inferring user preferences. In another research work [8], Agichtein et al., authors have organized the user interests as a set of features. Teevan et al. [9] and Chirita et al. [10] uses user's desktop to estimate their interests and used that to construct his/her profile. A major limitation of these approaches is that there can be a lot of terms, on users' desktop, which can make the user profile noisy or misleading. In previous work[11], user profile was build for personalization using click history [12] and anchor text [13] – build a wrapper system [14] on top of existing search engines. The average AR improvement is reported to be 30%. The proposed approach had significant improvement over nonpersonalized search engine except for 5% of the queries where personalization had a negative impact. The reasons behind discrepancies observed that the system was able to detect user query intent however context of user query intent was missing. Following, the recent work that uses Folksonomy [15, 16, and 17] in IR domain, their limitation and how system improvises over them is discussed.
Limitation 1: A resource like URL is tagged by many users. But since, users don't tag resources religiously; it may be possible that a particular URL receive higher weights while others don't. The current work doesn't take into account the biasness of user's tagging behavior. To alleviate such biasness, we propose to normalize the tag weights.
Limitation 2: The current published work assumes that a user has an account in some kind of Social Networking Service. We don't make such assumption. We observe and analyze user search behavior to construct his profile. Thus the proposed system in this work is applicable to all the users.
Limitation 3: After collecting group of terms, such a group is termed as a user profile; User profile is further used to re-rank search results by calculating cosine similarity of a resource profile with all the terms in the user profile. This approach is a good solution to reranking search results; however, there is still some scope of improvement, as later, we will show in our experiment section that clustering the terms in a user profile indeed makes the UPL more meaningful.

3. RANKING ADAPTATION

Ranking adaptation can be formally defined for the target domain as follows: a query set $Q= \{q_1, q_2 ..... q_m\}$ and a document set $D=\{d_1,d_2....d_n\}$ are given. For each query q_i € Q , a list of documents $d_i=\{d_{i1},d_{i2} ....d_i,n(q_i)\}$ returned and labeled with the relevance degrees $Y_i=\{y_{i1},y_{i2},.....y_i,n(q_i)\}$ by human judges. The relevance degree is generally a real value, i.e., y_{ij} € IR, so that different returned documents can be compared for sorting an ordered list. For each query document pair $<q_i ,d_{ij}>$, an s-dimensional query dependent feature vector $Ǿ(q_i, d_{ij})$ € IR^s is obtained, e.g., the query keyword's term frequency q_i in the title, body, URL of the document dij. $n(q_i)$ denotes the number of returned documents for query q_i. The objective of the learning to rank is to calculate approximately a ranking function f € IR^s –> IR so that the documents d can be ranked for a given query q according to the value of the prediction *f(Ǿ(q,d))* [19].

3.1 RANKING ADAPTATION WITH DOMAIN-SPECIFIC FEATURE

Data from different domains are characterized by certain domain specific features, e.g., when we adopt the ranking model that is used in a webpage search domain to an image search domain [20], the information surrounding the image can also provide additional information to support text based ranking model adaptation. Here the domain specific features are utilized. These domain specific features are difficult to translate into textual forms. This boosts the performance of RA-SVM. The rule is that documents with similar domain specific features have to be ranked with similar rankings in the domain. This assumption is called as consistency assumption which implies that a robust textual ranking function should perform relevance prediction that is consistent to the domain-specific features.

4. BUILDING USER PREFERENCE LIST

The purpose of using folksonomy in this work is to extract or retrieve tags from a folksonomy system given a URL. Using delicious API, input a URL, the output received is a list of tags and tag frequency. Since a URL is tagged by various users; each tag represents a shared understanding about that URL. Moreover, since different users assign different tags for the same URL, the collection of tags aids in disambiguating polysemy or to capture varying concepts that represent the same URL. Also, each tag has a weight associated with it which represents its importance or relevance to the given URL. The higher the weight, more important the tag is to the given URL. In this work, tags, associated with a URL are included in the UPL, are referred to as terms or concepts. Our system takes the top three terms and as explained above, to increase fairness, term weights are normalized using equation 1.

$$ntw_i = \frac{tw_i}{\sum_k tw_k} \qquad (1)$$

Here, *twi* represents term weight and *ntwi* represents normalized term weight of term *i*. The summation of denominator normalizes the term weight of each term thus reducing the final value within 0 and 1. For repeated terms, its corresponding weight is accumulated; terms that have higher weights are at the top, and those that have lower weights are at the bottom. Periodically, the terms with lower weights are simply discarded. We observed during computation of term similarity matrix that the terms with lower weight have very low similarity with other terms. This implies, if the system doesn't discard terms at an early stage, they will later be classified as outliers by clustering algorithm. Therefore, we simply delete terms with low weight in the beginning stage to reduce the time complexity of calculating the similarity matrix, cluster algorithm, and visualization algorithm. In order to determine cluster of terms, we first compute the term-term similarity matrix using Normalized Google Distance (NGD) [18]. NGD computes similarity distance between two terms based on information distance and Kolmogorov complexity. Equation 2 shows how to calculate the similarity value between two terms t1 and t2.

$$NGD(t1,t2) = \frac{Max[logf(t1), log(t2)) \;\; logf(t1,t2)}{Log\ N- Min\ (log\ f(t1),log\ f(t2))} \qquad (2)$$

The variables f(t1), f(t2), f(t1,t2) are number of search results for term t1, t2, and t1 and t2 together respectively. The value, NGD(t1,t2), lies between 0 and infinity. If the value equals 0, it signifies high relatedness and greater the values lesser the relatedness. We set 1.5 as threshold value in this work, i.e. any NGD(t1,t2) value greater than or equal to 1.5 were replaced to -1. The rationale for using NGD is its richness and correctness of result. There are many algorithms for calculating similarity distance between two terms, for instance, cosine similarity, hamming distance, correlation, jaccard index, Latent Semantic Analysis (LSA). In the paper [18], authors have shown that NGD, when used for classification using Support Vector Machine, has a mean accuracy of 87% which surpasses any other semantic distance algorithm to date.

5. SYSTEM ARCHITECTURE

The system outline is as follows: a User types a query, search engine return a ranked list of URLs- user then click URLs that he likes. The list of clicked URLs is an indication of user interests- using folksonomies extract tags and weights for all the clicked URLs. Compute similarity between tags and represent it as a similarity matrix using NGD [7] (Normalized Google Distance). Employ agglomerative clustering to group the tags into more meaningful groups, such that, each group represents a set of terms and its related context.

We start with the basic premise that, a user issue query terms to a search engine and in turn he/she receives a result set of URLs. Furthermore, a user clicks on URLs that he likes or finds interesting which in other words mean the clicked URLs are relevant to the query. We want to use these clicked URLs as an indication of user interest. By employing folksonomies on clicked URLs, we construct User Preference List (UPL).

A natural language parser (NLP) is a program that works out the grammatical **structure of sentences**, for instance, which groups of words go together (as "phrases") and which words are the **subject** or **object** of a verb.

Proposed Algorithm

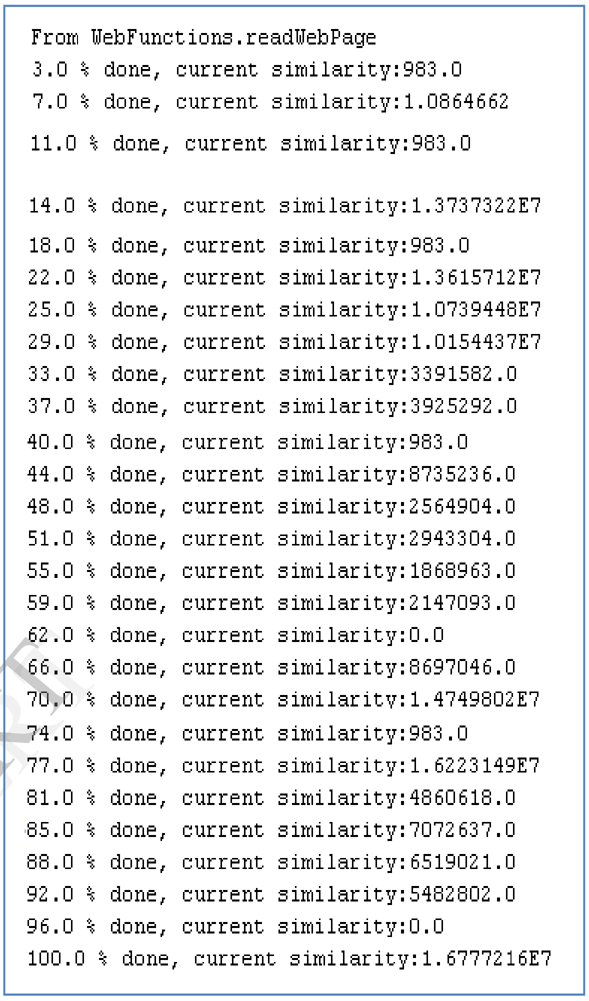
```
1.  Input Username
     If exist then Login else  Signup
2.  Enter search query
3.  Get Google Search results  which contains
    URL
     It list contents and scores
4.  read user Profile
5.  Goto each search result,
6.  Fetch the URL & read content of URL
7.  Remove Tags from URL content
 8.  Match your profile with the URL
    content – Semantic score  (Find Similarity)
9.  Links with higher similarity go at the top &
    remaining link go to the bottom [sort in
    reverse order]
10. Ask user to pick a link
    url read- unwanted content
    NLP (Natural language processor)  is used to
    find Action words (noun, pronoun, Adjective,
    Adverb, verb)
11. profile update
12.  Goto  step 2
```

6. IMPLEMENTATION

In this paper we presented the working and design of the system for ranking the user query terms. User will provide the keyword to search engine and in turn receives a result set of URLs. When we press search button it will search all URL from Google website and list contents and score. Then it removes the tags from URL contents and Match user profile with the URL content and find similarity from semantic score. Then displays the links in order from high similarity value to low similarity value and asks user to click the link.

7. EXPERIMENTAL RESULTS

Search String: Indian Rail



```
From WebFunctions.readWebPage
3.0 % done, current similarity:983.0
7.0 % done, current similarity:1.0864662
11.0 % done, current similarity:983.0

14.0 % done, current similarity:1.3737322E7
18.0 % done, current similarity:983.0
22.0 % done, current similarity:1.3615712E7
25.0 % done, current similarity:1.0739448E7
29.0 % done, current similarity:1.0154437E7
33.0 % done, current similarity:3391582.0
37.0 % done, current similarity:3925292.0
40.0 % done, current similarity:983.0
44.0 % done, current similarity:8735236.0
48.0 % done, current similarity:2564904.0
51.0 % done, current similarity:2943304.0
55.0 % done, current similarity:1868963.0
59.0 % done, current similarity:2147093.0
62.0 % done, current similarity:0.0
66.0 % done, current similarity:8697046.0
70.0 % done, current similarity:1.4749802E7
74.0 % done, current similarity:983.0
77.0 % done, current similarity:1.6223149E7
81.0 % done, current similarity:4860618.0
85.0 % done, current similarity:7072637.0
88.0 % done, current similarity:6519021.0
92.0 % done, current similarity:5482802.0
96.0 % done, current similarity:0.0
100.0 % done, current similarity:1.6777216E7
```

1.Score: 1.6777216E7	Result:cached__::__http://webcache.googleusercontent.com/search?q=cache:q_ykAAxJd3kJ:www.mapsofindia.com/maps/india/india-railway-map.htm+&cd=15&hl=en&ct=clnk
2.Score: 1.6223149E7	Result:indian railways time table, indian railway trains timing & schedule__::__http://www.mapsofindia.com/railway-timetable/
3.Score: 1.4749802E7	Result:indian train fire kills more than 20__::__http://www.theguardian.com/world/2013/dec/28/indian-train-fire-kills-bangalore-nanded
4.Score: 1.3737322E7	Result:train berth availability__::__http://www.indianrail.gov.in/dont_Know_Station_Code.html
5.Score: 1.3615712E7	Result:train between important stations__::__http://www.indianrail.gov.in/between_Imp_Stati

	ons.html
6.Score: 1.0864662E7	Result:cached__::__http://webcache.googleusercontent.com/search?q=cache:2HkmgycqmQkJ:www.indianrail.gov.in/+&cd=1&hl=en&ct=clnk
7.Score: 1.0739448E7	Result:seat availability__::__http://www.indianrail.gov.in/seat_Avail.html
8.Score: 1.0154437E7	Result:train schedule__::__http://www.indianrail.gov.in/train_Schedule.html
9.Score: 8735236.0	Result:cached__::__http://webcache.googleusercontent.com/search?q=cache:1tb5oFGqbLQJ:indiarailinfo.com/+&cd=9&hl=en&ct=clnk
10.Score: 8697046.0	Result:__::__http://www.theguardian.com/world/2013/dec/28/indian-train-fire-kills-bangalore-nanded
11.Score: 7072637.0	Result:india railway map__::__http://www.mapsofindia.com/maps/india/india-railway-map.htm
12.Score: 6519021.0	Result:maps__::__http://www.mapsofindia.com/maps
13.Score: 5482802.0	Result:india__::__http://www.mapsofindia.com/maps/india/
14.Score: 4860618.0	Result:cached__::__http://webcache.googleusercontent.com/search?q=cache:5UZ5rkULB-4J:www.mapsofindia.com/railway-timetable/+&cd=14&hl=en&ct=clnk
15.Score: 3925292.0	Result:cached__::__http://webcache.googleusercontent.com/search?q=cache:mhL2yXQipYUJ:www.irctc.co.in/+&cd=8&hl=en&ct=clnk
16.Score: 3391582.0	Score:3391582.0, Result:irctc online passenger reservation system__::__http://www.irctc.co.in/
17.Score: 2943304.0	Result:cached__::__http://webcache.googleusercontent.com/search?q=cache:_aV1aCW1r2QJ:www.indianrailways.gov.in/+&cd=10&hl=en&ct=clnk
18.Score: 2564904.0	Result:indian railway__::__http://www.indianrailways.gov.in/
19.Score: 2147093.0	Result:cached__::__http://webcache.googleusercontent.com/search?q=cache:0wrtvb4y070J:erail.in/+&cd=11&hl=en&ct=clnk
20.Score: 1868963.0	Result:indian railways irctc timetable pnr status fare live status ...__::__http://erail.in/
21.Score: 983.0	Result:welcome to indian railway passenger reservation enquiry__::__http://www.indianrail.gov.in/
22.Score: 983.0	Result:welcome to indian railway passenger reservation enquiry__::__http://www.indianrail.gov.in/
23.Score: 983.0	Result:welcome to indian railway passenger reservation enquiry__::__http://www.indianrail.gov.in/
24.Score: 983.0	Result:welcome to indian railway passenger reservation enquiry__::__http://www.indianrail.gov.in/
25.Score: 983.0	Result:welcome to indian railway passenger reservation enquiry__::__http://www.indianrail.gov.in/
Which Link Number to Open:	

8. CONCLUSION

As various vertical search engines emerge and the amount of verticals increases dramatically, a global ranking model, which is trained over a data set sourced from multiple domains, cannot give a sound performance for each specific domain with special topicalities, document formats, and domain-specific features. Building one model for each vertical domain is both laborious for labeling the data and time consuming for learning the model. In this paper, we propose the ranking model adaptation, to adapt the well learned models from the broad-based search or any other auxiliary domains to a new target domain. By model adaptation, only a small number of samples need to be labeled, and the computational cost for the training process is greatly reduced.

Based on the regularization framework, the Ranking Adaptation SVM algorithm is proposed, which performs adaptation in a black-box way, i.e., only the relevance predication of the auxiliary ranking models is needed for the adaptation. In RA-SVM, it is proposed to utilize the domain specific features to further facilitate the adaptation, by assuming that similar documents should have consistent rankings, and constraining the margin. Furthermore, we propose ranking adaptability, to quantitatively measure whether an auxiliary model can be adapted to a specific target domain and how much assistance it can provide. The proposed RA-SVM can better utilize both the auxiliary models and target domain labeled queries to learn a more robust ranking model for the target domain data.

In this paper, we presented a methodology to build a User Preference List for a user based on his clicked URL history. To test the proposed approach, AOL query log data set is used. Given a set of click URLs for a particular user, the system uses delicious APIs to extract tags associated with each URL clicked by user. Further, it builds a term-term similarity matrix where values represent similarity between them. To determine the similarity between terms we used Google Similarity Distance. Similarity matrix in input to clustering algorithm.

REFERENCES :

[1] J.M. Ponte and W.B. Croft,"A Language Modeling Approach to Information Retrieval," Proc. 21st Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval, pp. 275-281, 1998
[2] S. Robertson and D.A. Hull, "The Trec-9 Filtering Track Final Report," Proc. Ninth Text Retrieval Conf., pp. 25-40, 2000.
[3] T. Joachims, "Optimizing Search Engines Using Clickthrough Data," Proc. Eighth ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD '02), pp. 133-142, 2002.
[4] Y. Freund, R. Iyer, R.E. Schapire, Y. Singer, and G. Dietterich, "An Efficient Boosting Algorithm for Combining Preferences," J. Machine Learning Research, vol. 4, pp. 933-969, 2003.
[5] C.J.C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender, "Learning to Rank Using Gradient Descent," Proc. 22th Int'l Conf. Machine Learning (ICML '05), 2005.
[6] J. Blitzer, R. Mcdonald, and F. Pereira, "Domain Adaptation with Structural Correspondence Learning," Proc. Conf. Empirical Methods in Natural Language Processing (EMNLP '06), pp. 120-128, July 2006.
[7] Kelly, D. and Teevan, J., "Implicit feedback for inferring user preference: A bibliography", In SIGIR, 2003, pp. 18-28.
[8] Agichtein, E., Brill, E., Dumais, S., and Robert Ragno, "Learning user interaction models for predicting web search result preferences", In SIGIR, 2006, pp. 3-10.
[9] Teevan, J., Dumais, S.T., and Horvitz, E., "Personalizing search via automated analysis of interests and activities", In SIGIR, 2005, pp. 449-456.
[10] Chirita, P.A., Firan, C., and Nejdl, W., "Summarizing local context to personalize global web search", In CIKM, 2006, pp. 287-296.
[11] Harshit Kumar, Pil-Seong Park, Hong-Gee Kim "Using Folksonomy for Building User Preference List" 9th IEEE International Symposium on Parallel and Distributed Processing with Application Workshops
[12] Kumar H., Park S., and Kang, S., A Personalized URL Re-ranking Methodology Using User's Browsing Behavior, KES-AMSTA 2008, LNAI, pp.212-221.
[13] Kumar, H. and Kang, S., Another Face of Search Engine: Web Search API's., IEA/AIE 2008, pp. 311-320.
[14] Kumar, H. and Kang, S., Exclusively Your's: Dynamic Individuate Search by Extending User Profile, Journal of New Generation Computing, Vol. 28, No. 1 , pp.73-94, February, 2010.
[15] Michael G. Noll and Meinel, C., "Web Search Personalization Via Social bookmarking and Tagging", ISWC 2007, LNCS, vol. 4825, pp. 365-378.
[16] Vallet, D., Cantador, I., and Jose, J.M., "Personalizing Web Search with Folksonomy based User and Document Profiles", ECIR 2010, vol. 5993, pp. 420-431, Springer LNCS.
[17] Xu, S., Bao, S., Fei, B., Su, Z., and Yu, Y., Exploring folksonomy for personalized search, SIGIR 08, pp. 155-162.
[18] Cilibrasi, R.L., Vitanyi, P.M.B., "The Google Similarity Distance", IEEE Transactions on Knowledge and Data Engineering, vol. 19, pp. 370-383, 2007.
[19] M.S.Gayatri, S.Leela, "Ranking Adaptation SVM for Target Domain Search."
[20] Bo Geng, Linjun Yang, Chao Xu, and Xian-Sheng Hua, (April 2012) "Ranking Model Adaptation for Domain-Specific Search", IEEE Transaction on Knowledge and Data Engineering, vol 24, No. 4.