An Automatic Classification Web Search Ranking Algorithm (Postprint)

Authors: Liu Mingyu, Liu Xueliang, Hu Jun

Date: 2018-05-20T00:00:00+00:00

Abstract

To address the domain drift phenomenon in the traditional Okapi BM25 web page ranking algorithm, where retrieved pages are often irrelevant to the domain of query keywords, and the issue that improved algorithms require manual construction of domain vectors, we propose a web search ranking algorithm based on BM25 and a Softmax regression classification model. The method first conducts data preprocessing on web page text and employs the bag-of-words model for vector representation, subsequently trains a Softmax regression classification model using a small amount of web page data to predict category scores for test web pages, and combines these with BM25 information retrieval scores to produce the final ranking results. Experimental results show that this retrieval algorithm can achieve excellent web page ranking performance without requiring manual construction of domain vectors.

Full Text

Preamble

Web Page Search Ranking Algorithm Using Automatic Classification

Liu Mingyu, Liu Xueliang, Hu Jun
(School of Computer & Information, Hefei University of Technology, Hefei 230009, China)

Abstract: In the traditional web page ranking algorithm Okapi BM25, there exists a problem where retrieval results are independent of domain keywords, and improved algorithms require manual construction of domain vectors. To address this issue, we propose a web page ranking algorithm based on BM25 and a softmax regression classification model. This method first preprocesses web page text and uses a bag-of-words model for vector representation. Then, it trains a softmax regression classification model using a small amount of web page data to predict category scores for test pages, which are combined with

BM25 information retrieval scores to obtain final ranking results. Experimental results demonstrate that this retrieval algorithm achieves excellent web page ranking performance without requiring manual domain vector construction.

Keywords: domain vector; BM25; softmax regression classification; web page ranking

0 Introduction

With the explosive growth of the Internet, web information has become increasingly important in daily life. However, when faced with vast amounts of information, users' ability to find useful content depends heavily on search engine functionality, making web page ranking algorithms a persistent research focus. User queries are often short and imprecise, leading to top-ranked pages that may be irrelevant to search intent. For example, when searching for "microblog," some users seek the login interface while others want news or stock information about the company. Internet content spans multiple topics, and users typically desire results within a specific domain—this is known as the domain problem.

Information retrieval represents the most critical factor influencing search ranking. Over the years, researchers have proposed many representative models, including the Boolean model, vector space model, and probabilistic model. While Boolean and vector space models treat document terms as independent items, probabilistic models consider the intrinsic relationships between terms and documents, leveraging probabilistic dependencies for information retrieval. The Okapi BM25 algorithm, as a typical ranking algorithm within the probabilistic framework, has been widely applied in search engine page ranking and text weighting in natural language processing.

Recent BM25-based relevance ranking algorithms primarily utilize term frequency information such as TF and IDF. TF refers to the number of times a term appears in a document, while IDF measures term importance by counting documents containing the term. Büttcher et al. incorporated proximity information into BM25 by calculating term distances within documents to improve scoring. Roi-Blanco et al. similarly enhanced BM25 for web retrieval by considering different term sources to compute term importance and defining operators on "virtual zones" to calculate term-document similarity. Although these methods achieved good results, they did not effectively solve domain drift. To address this, reference [13] proposed a topic-sensitive re-ranking (TSRR) algorithm that designs a model independent of page ranking to select domain keywords forming a domain vector, then builds a page information model for re-ranking search results. However, this approach' s effectiveness depends heavily on the quantity and accuracy of selected domain keywords, which are time-consuming to establish and rely on specialized knowledge and intuition.

This paper proposes an automatic classification web page ranking algorithm that differs from previous methods by eliminating manual domain vector construction. Instead, it employs a classifier to automatically obtain page category

probabilities. For information retrieval, we use the BM25 algorithm to compute relevance between search keywords and pages. For domain classification, we train a softmax regression model on a small amount of page data to obtain domain probability scores, which are linearly combined with BM25 scores for final page ranking. This method achieves excellent ranking performance without requiring empirical knowledge or manual techniques.

1 Our Method

The proposed method is illustrated in [Figure 1: see original paper]. First, a crawler program extracts web page text, which undergoes preprocessing including tokenization, stop word removal, and bag-of-words vectorization to form our dataset. Next, we train a softmax regression classification model to obtain category probabilities for each page. Based on user-provided search keywords, relevant pages are retrieved via the BM25 algorithm. The BM25 scores and softmax category scores are then fused to produce final page scores for ranking and returning results to users.

1.1 Data Preprocessing

We first crawl web pages from different domain websites and extract text content based on page tags. Chinese word segmentation is performed using Jieba' s precise mode, followed by stop word removal. After preprocessing, we obtain a tokenized web page text dataset. We compute term weights in each page using the bag-of-words model: if a term appears n times in a page text, its corresponding weight in the page vector is n; otherwise, it is 0. The page text vector size is m × |v|, where m is the number of pages and |v| represents the vector length, specifically the number of terms in the dictionary.

To improve query efficiency and reduce response time, we employ an inverted index mechanism. Under global search conditions, inverted indexes establish and store mapping relationships between terms and documents, enabling rapid retrieval of document lists containing specific terms. After preprocessing, each document is converted into a term list <term, doc>, and we build inverted indexes for all documents based on their terms, including a dictionary and comprehensive inverted record tables, as shown in [Figure 2: see original paper].

1.2 Okapi BM25 Algorithm

Okapi BM25 is a classic probabilistic model that ranks documents based on relevance between search queries and matching documents. Given a query vector Q containing keywords q , q , ···, q , the BM25 score for a document D is:

$$\mathrm{BM25}(D,Q) = \sum_{i=1}^{n} \mathrm{IDF}(q_i) \cdot \frac{f(q_i,D)\cdot(k+1)}{f(q_i,D) + k\cdot\left(1-b+b\cdot\frac{|D|}{\mathrm{avgdl}}\right)}$$

where f(q , D) is the term frequency of keyword q in document D, |D| is document length, avgdl is average document length, and k and b are adjustable parameters controlling term frequency weighting and document length normalization, respectively. Experiments verify that k is typically set between [1.2, 2.0]; we use 1.2, and b is set to 0.75.

The inverse document frequency IDF(q) is calculated as:

$$\mathrm{IDF}(q_i) = \log \frac{N - n(q_i) + 0.5}{n(q_i) + 0.5}$$

where N is the total number of collected documents and n(q) is the number of documents containing the keyword. Observing this formula, if a term q appears in more than half of the documents, IDF(q) becomes negative; therefore, we remove all stop words during preprocessing.

1.3 Web Page Text Classification

Web pages contain multi-domain data, making page classification a multi-class problem. We adopt the softmax regression classification model, which offers several advantages: it directly models classification probabilities without assuming data distribution, avoiding problems from inaccurate assumptions, and it provides approximate probability predictions essential for our task.

The model is mathematically defined as:

$$p(y_i = j|x_i) = \frac{e^{W_j^T x_i}}{\sum_{k=1}^{K} e^{W_k^T x_i}}$$

where W represents model parameters, x is the i-th page text vector, and p(y = j | x) denotes the probability of softmax regression classifying x into category j. For a given input instance x, we compute probabilities for all categories and select the category with maximum probability.

For the five crawled page categories, we first label all pages. To approximate online model effects, we randomly select 1,500 pages for training and use the remaining 13,500 pages as test data without overlap. After parameter tuning for optimal accuracy, we fix network parameters and input the 13,500 test pages to obtain category probability predictions from the softmax regression model.

For domain classification, we use cosine similarity to compute similarity between the softmax regression model' s category probability output p and category vector l to obtain category scores. The category probability output p and category vector l are represented as:

$$\mathbf{p} = (p_1, p_2, \dots, p_n)\mathbf{l} = (l_1, l_2, \dots, l_n)$$

where n is the number of categories, and only one element l (i = 1, 2, ···, n) in category vector l has value 1 while others are 0. Cosine similarity is calculated as:

$$\text{score}_p = \cos(\mathbf{p}, \mathbf{l}) = \frac{\mathbf{p} \cdot \mathbf{l}}{\|\mathbf{p}\| \cdot \|\mathbf{l}\|}$$

The automatic classification page ranking algorithm proceeds as follows: (a) compute similarity between user search keywords and each page using equations (1) and (2) to obtain score_BM25; (b) predict page probabilities with the softmax regression model, then compute cosine similarity between softmax output p and category vector l using equation (4) to obtain score_p; (c) weighted sum of both scores yields the final ranking score:

$$\text{score} = \alpha \cdot \text{score}_{BM25} + \beta \cdot \text{score}_p$$

where + = 1 and , [0, 1]. Pages are ranked according to this final score.

2 Experiments

2.1 Experimental Setup

Our experimental machine uses an Intel(R) Xeon(R) CPU E5-2620@2.10GHz with 64 GB RAM, Ubuntu 14.04 OS, and algorithms implemented in Python 2.7. The softmax regression classifier uses 50 iterations with step size 10 . When classification accuracy reaches its optimum of approximately 94.6%, we obtain optimal classifier parameters and compute category probabilities for test data.

2.2 Retrieval Keywords and Corpus

We crawled news corpora from popular websites including Tencent, Sina, and IT Times Network, covering five major categories: IT, entrepreneurship, academia, current affairs, and entertainment. The dataset contains 15,000 pages total, with 3,000 pages per category. Retrieval keywords were selected from 2016 Internet hot terms across domains, with three keywords per domain. Detailed data selection is shown in .

2.3 Evaluation Metrics

To validate our ranking method' s effectiveness, we employ two evaluation metrics:

User Satisfaction: Users score each of the top ten pages using a five-point scale (1 = very dissatisfied, 5 = very satisfied), and we compute the average score:

$$\text{Satisfaction} = \frac{1}{n}\sum_{i=1}^{n} S_i$$

where n is the number of users and S represents the average score for the top ten pages.

Precision at K (P@K): P@K is an intuitive information retrieval metric reflecting the proportion of relevant documents among the top K retrieved results. Following evaluation standards from reference [16], we consider pages scoring 3 or above (satisfied and very satisfied) as relevant results. The formula is:

$$P@K = \frac{K_s}{K}$$

where K is the number of relevant pages among the top K results. Since successful retrieval systems should return desired results as early as possible, we evaluate P@2, P@4, P@6, P@8, and P@10 to incorporate ranking position information.

2.4 Parameter Tuning

Equation (5) determines final page ranking scores, requiring us to balance the importance of each component through parameters and . We recruited five volunteers, selected one keyword per domain, and used the user satisfaction formula to score ranking results from different parameter combinations to determine optimal values. Each keyword underwent nine experiments with parameters ranging from 0.1, 0.9 to 0.9, 0.1 . Results are shown in [Figure 4: see original paper].

Based on experimental results, IT, academic, and entertainment domains achieved highest user satisfaction at 0.4, 0.6 , while entrepreneurship and current affairs reached their second-highest scores. The average across all five categories also peaked at 0.4, 0.6 . Therefore, we select = 0.4 and = 0.6 as optimal parameters.

2.5 Comparative Experimental Results

We compare our method against the TSRR algorithm, which also addresses domain drift. Using our dataset, TSRR's optimal parameters were determined as = 0.3, = 0.7 through the same tuning method. Without informing volunteers which results belonged to which algorithm, five volunteers scored retrieval results for each domain' s keywords based on expected results for that domain.

Results are shown in [Figure 5: see original paper]. Keywords inherently containing domain information (e.g., "virtual reality," "bitcoin" in IT; "cloud computing," "live streaming," "bike sharing" in entrepreneurship; "genes," "biology" in academia; "One Belt One Road," "cultural confidence" in current affairs;

"Wang Baoqiang," "Yang Yang," "fans" in entertainment) yielded TSRR user satisfaction of 3.53 versus our algorithm' s 3.87–9.6% higher. However, ambiguous keywords like "drone" (IT or entrepreneurship), "artificial intelligence" (IT, entrepreneurship, or academia), and "Internet+" suffer severe domain drift. For these three terms, TSRR averaged 2.83 while our algorithm achieved 3.73–31.8% higher, effectively solving domain drift.

P@K results in show our algorithm improves P@2 by 81.3%, P@4 by 58.8%, P@6 by 54.8%, P@8 by 50.5%, and P@10 by 45.5%. These substantial improvements ensure desired results appear earlier. In summary, combining softmax regression classification with BM25 effectively addresses domain drift while ranking relevant pages higher.

3 Conclusion

This paper proposes a web page ranking algorithm combining BM25 and softmax regression classification. The method trains a classifier on small amounts of page data to obtain category scores, which are combined with BM25 retrieval scores for final ranking. Without requiring manual domain vector construction, it effectively solves domain drift while ensuring relevant pages rank higher. Future research will incorporate user search history and behavioral tendencies to further improve ranking effectiveness.

References

[1] Fonseca B M, Golgher P B, Moura E S D, et al. Using association rules to discover search engines related queries [C]// Proc of LA-WEB. 2003: 66-71.

[2] Zhuang Ziming, Cucerzan S. Re-ranking search results using query logs [C]// Proc of the 15th ACM International Conference on Information and Knowledge Management. 2006: 860-861.

[3] Cooper W S. Getting beyond boole [J]. Information Processing and Management, 1998, 24 (3): 243-248.

[4] Salton G, Yang C S, Yu C T. A theory of term importance in automatic text analysis [J]. Journal of the American Society for Information Science and Technology, 2010, 26 (1): 33-44.

[5] Robertson, Stephen E, Jones S, et al. Relevance weighting of search terms [J]. Journal of the Association for Information Science and Technology, 2014, 27 (3): 129-146.

[6] Robertson S, Zaragoza H. The probabilistic relevance framework: BM25 and beyond [J]. Foundations and Trends® in Information Retrieval, 2009, 3 (4): 333-389.

[7] Niu Jianwei, Zhao Qingjuan, Wang Lei, et al. OnSeS: a novel online short text summarization based on bm25 and neural network [C]// Proc of IEEE

Global Communications Conference. 2016: 1-6.

[8] Li Ying, Sha Fei, Wang Shujuan, et al. The improvement of page sorting algorithm for music users in Nutch [C]// Proc of the 15th IEEE//ACIS International Conference on Computer and Information Science. 2016: 1-4.

[9] Bestgen Y. Improving the character n-gram model for the DSL task with BM25 weighting and less frequently used feature sets [C]// Proc of the 4th Workshop on NLP for Similar Languages, Varieties and Dialects. 2017: 115-123.

[10] Kazemian S, Zhao Shunan, Penn G. Evaluating sentiment analysis in the context of securities trading [C]// Proc of the 54th Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics. 2016: 2094–2103.

[11] Büttcher S, Clarke C L, Lushman B. Term proximity scoring for Ad hoc retrieval on very large text collections [C]// Proc of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. 2006: 621-622.

[12] Blanco R, Boldi P. Extending BM25 with multiple query operators [C]// Proc of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2012: 921-930.

[13] Pan Cheng, Wu Gongqing, Li Lei, et al. Topic-sensitive page ranking algorithm based on domain model [J]. Computer Systems and Applications, 2015, 24 (11): 107-114.

[14] Jones K S, Walker S, Robertson S E. A probabilistic model of information retrieval: development and comparative experiments [J]. Information processing and management, 2000, 36 (6): 809-840.

[15] Manning C D, Raghavan P, Schütze H. An Introduction to Information Retrieval [M]. Cambridge: Cambridge University Press, 2008: 1-18.

[16] Agichtein E, Brill E, Dumais S. Improving Web search ranking by incorporating user behavior information [C]// Proc of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. 2006: 19-26.

[17] Wu Yao, DuBois C, Zheng A X, et al. Collaborative denoising auto-encoders for top-n recommender systems [C]// Proc of the 9th ACM International Conference on Web Search and Data Mining. 2016: 153-162.

[18] Wu Chaoyuan, Ahmed A, Beutel A, et al. Recurrent recommender networks [C]// Proc of the 10th ACM International Conference on Web Search and Data Mining. 2017: 495-503.

[19] Zhuang Fuzhen, Luo Dan, Yuan N J, et al. Representation Learning with Pair-wise Constraints for Collaborative Ranking [C]// Proc of the 10th ACM International Conference on Web Search and Data Mining. 2017: 567-575.