Investigation on Relevance Ranking of Archives Information Retrieval Results

Xinnan Jiang

Archives, Heilongjiang East College,
No. 331 Xuefu Street, Nangang District, Harbin, Heilongjiang,150086, P.R. China, 18246054006@163.com

Abstract

In this paper the satisfaction of users on information retrieval results was analyzed and the search result was modified and resorted, based on which the relevance ranking algorithm was proposed. The method calculated a value for every returned result and sorted the value in descending order. At last, the value was output and returned it to the users. The realization of relevance ranking improved the retrieval efficiency greatly and it made the returned results more consistent with the users' query intent.

Keywords: information retrieval; archives management; relevance ranking

1. Introduction

Archive searching becomes a difficult problem which affects the using of archives when massive amount of archives scored. It was solved by building the archive information database. Users usually input several keywords to search, as their memories are almost fragmented and probable. What the fuzzy query finds are the intersection of all keywords and it will produce several, dozens, or even hundreds of query results. If the record contained a keyword of the query, it will be returned. The return results are not sorted or distinguished. Its sequence is the order of the system retrieved.

There is a problem with the results returned by system default. Because not all the records that contain keywords are same to the user's query relevance. Some will be very relevant, while some are not relevant. The default sort can't distinguish the difference between these results and query. If a user wants to find out the required records, he must browse all the returned records and do further judgment. It is an extremely time-consuming and tedious work when the quantity of returned results is large. The user would lose patience and give up. Therefore it is necessary to modify the query results. In this paper, the relevance ranking algorithm was proposed to solve this problem.

2. Theory of relevance ranking

The theory of relevance rank is to calculate a value for every return record, which called query value. And then the return records are sort in descending order according to query value.

How to get the query value? The query submitted by users is expressed a vector which is called query vector. Each record is also express as a vector which is called record vector. Then a value is obtained by calculating the dot product of the two vectors. The value is called query value, which is the basis for the record ranking.

How to obtain the query vector? The Analysis of the user's query related to two factors, which are the importance of query keywords and the history query times. The more important the query keywords, and the higher relevance the record containing the keywords are, the greater it influents the ranking. The history query times of keywords reflect its popularity. The query meaning that is expressed by keywords with high popular degree is concerned more by the users. The more a keyword is queried, which proves that it is concerned more by users, and it is more important in users' mind. It will influent the ranking much more.

How to get the record vector? The factors that influent records ranking in database are the frequency that keywords appear in the record, and the position that keywords appear. It shows that the contents of the record relate more with the keywords if the keywords appear more in a record. Because the importance of different fields are not the same, where the keywords appearing will affect the importance of keywords, query relevance and the ranking position directly.

Knowing the factors affecting the relevance, an algorithm is given to calculate the query value to sort the default search results again.

3. Relevance ranking

When a query string is input, the system will divide it into several keywords automatically. In every table, each record containing keyword is searched by the SQL statements in fuzzy query. Query every record containing keywords in every form successively. And then the query results are returned. Here, the returned results are in default order, which are not sorted by relevance ranking. According to the theory of relevance ranking, the value of every returned record is calculated and output to the users after rearranged them in descending rank.

The relevance rank process is shown in figure 1

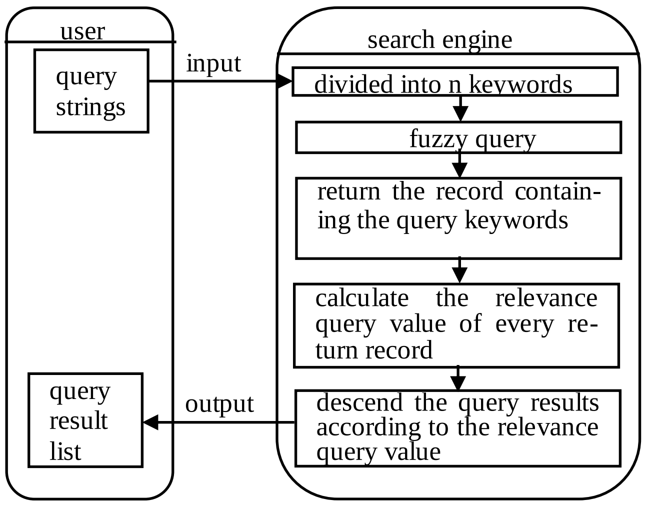
user
query strings
input
search engine
divided into n keywords
fuzzy query
return the record containing the query keywords
calculate the relevance query value of every return record
descend the query results according to the relevance query value
output
query result list

Fig. 1 Relevance rank process

4. Relevance rank algorithm

4.1. The calculation method of the user's query vector

There are two factors that affect the user's query.

The importance of query keywords, represented as W_{zy}.

When query keywords are submitted, the keywords considered as most important are always put at first, followed by some keywords which can possible illustrate the query intention or limit the query scope further. Therefore the keywords in front can describe the user's real query intention better, and they are more important than the ones behind them.

Supposing the query string inputted by the users was divided into n keywords, the positions of keywords show their importance. Then give them different weights according to their importance.

The weight of the first keyword is shown in Eq. 1.

$$W_{zy}(1)=\frac{n}{n+(n-1)+(n-2)+\cdots+[n-(n-1)]} \quad (1)$$

The weight of the second keyword is shown in Eq. 2.

$$W_{zy}(2)=\frac{n-1}{n+(n-1)+(n-2)+\cdots+[n-(n-1)]} \quad (2)$$

$$\vdots$$

The weight of the nth keyword is shown in Eq. 3.

$$W_{zy}(n)=\frac{1}{n+(n-1)+(n-2)+\cdots+[n-(n-1)]} \quad (3)$$

Where, $n \in Z$ and $n \geq 1$

Therefore, the weight of the lth keyword is shown in Eq. 4.

$$W_{zy}(l)=\frac{n-(i-1)}{\sum_{i=1}^{n}[n-(i-1)]} \quad i,l \in [1,n] \quad (4)$$

The weight of history query times of keywords, represented as W_{ls}.

Assume the number of query times in the database is m. The number of keywords is n. The first keyword is queried f_1 times. The second keyword is queried f_2 times. Then the nth keyword is queried f_n times. The weight of history query times of the first keyword is shown in Eq. 5.

$$W_{ls}(1)=\frac{f_1}{m} \quad (5)$$

The weight of the history query times of the second keyword is shown in Eq. 6.

$$W_{ls}(2)=\frac{f_2}{m} \quad (6)$$

$$\vdots$$

The weight of the history query times of the nth keyword is shown in Eq. 7.

$$W_{ls}(n)=\frac{f_n}{m} \quad (7)$$

Therefore the weight of the history query times of the lth keyword is shown in Eq. 8.

$$W_{ls}(l)=\frac{f_l}{m} \quad l \in [1,n] \quad (8)$$

How to calculate the query vector with the weight values W_{zy} and W_{ls}? The query vector is represented as Q, which is shown in Eq. 9.

$$Q_l=C_1W_{zy}(l)+C_2\omega W_{ls}(l)=C_1\frac{n-(l-1)}{\sum_{i=1}^{n}[n-(i-1)]}+C_2\frac{\omega f_l}{m}$$
$$i,l \in [1,n]\ C_1+C_2=1 \quad (9)$$

In Eq. 9, ω and m are constants. ω is a value that adjusts the weight of history query times according to the test result. If the value of $\frac{f_l}{m}$ is too small, it can be negligible for the little influence in the equation that calculates query vector. Then the influence of history query times on relevance is not embodied. So the weight value of it should be adjusted according to test results. m is the total number of queries in the database. C_1 and C_2 are constants and their sum is 1. It shows the important difference between the important degree of keywords and history query times in user's mind. Usually the important degree of keywords is more important than history query times, therefore there is $C_1 > C_2$. n is the number of query keywords. l represents query keyword that from 1 to n. f_i represents the number of query times of the lth keyword.

4.2. The calculation method of the record vector

There are two factors that influent the relevance of every record.

The weight of times that keywords occur in every attribute, represented as W_{cs}.

Keywords can appear in the different fields also called attributes in the database. When other factors are the same, the more keywords appear in a record, the higher it relevant with the user query.

Supposing the number of keywords is n, then every keyword is represented as: $x_1, x_2, \cdots, x_n$

Supposing the number of records found is δ, then every record is represented as: $r_1, r_2, \cdots, r_\delta$

Supposing the number of fields is μ, then every field is represented as: $f_1, f_2, \cdots, f_\mu$

The appearing times of first keyword in the first record from the first to the μth attribute are represented as: $\lambda_{x_1 r_1 f_1}, \lambda_{x_1 r_1 f_2}, \cdots, \lambda_{x_1 r_1 f_\mu}$

The appearing times of second keyword in the first record from the first to the μth attribute are represented as: $\lambda_{x_2 r_1 f_1}, \lambda_{x_2 r_1 f_2}, \cdots, \lambda_{x_2 r_1 f_\mu}$

$$\vdots$$

The appearing times of nth keyword in the first record from the first to the μth attribute are represented as: $\lambda_{x_n r_1 f_1}, \lambda_{x_n r_1 f_2}, \cdots, \lambda_{x_n r_1 f_\mu}$

The appearing times of every keyword in the first record is represented as:

$$\begin{bmatrix} \lambda_{x_1 r_1 f_1} & \lambda_{x_2 r_1 f_1} & \cdots & \lambda_{x_n r_1 f_1} \\ \lambda_{x_1 r_1 f_2} & \lambda_{x_2 r_1 f_2} & & \lambda_{x_n r_1 f_2} \\ \vdots & \vdots & & \vdots \\ \lambda_{x_1 r_1 f_\mu} & \lambda_{x_2 r_1 f_\mu} & \cdots & \lambda_{x_n r_1 f_\mu} \end{bmatrix}$$

The appearing times of every keyword in the second record is represented as:

$$\begin{bmatrix} \lambda_{x_1 r_2 f_1} & \lambda_{x_2 r_2 f_1} & \cdots & \lambda_{x_n r_2 f_1} \\ \lambda_{x_1 r_2 f_2} & \lambda_{x_2 r_2 f_2} & & \lambda_{x_n r_2 f_2} \\ \vdots & \vdots & & \vdots \\ \lambda_{x_1 r_2 f_\mu} & \lambda_{x_2 r_2 f_\mu} & \cdots & \lambda_{x_n r_2 f_\mu} \end{bmatrix}$$

$$\vdots$$

The appearing times of every keyword in the δth record is represented as:

$$\begin{bmatrix} \lambda_{x_1 r_\delta f_1} & \lambda_{x_2 r_\delta f_1} & \cdots & \lambda_{x_n r_\delta f_1} \\ \lambda_{x_1 r_\delta f_2} & \lambda_{x_2 r_\delta f_2} & & \lambda_{x_n r_\delta f_2} \\ \vdots & \vdots & & \vdots \\ \lambda_{x_1 r_\delta f_\mu} & \lambda_{x_2 r_\delta f_\mu} & \cdots & \lambda_{x_n r_\delta f_\mu} \end{bmatrix}$$

The total times that the first keyword occurring in the first record is: $\lambda_{x_1 r_1 f_1} + \lambda_{x_1 r_1 f_2} + \cdots + \lambda_{x_1 r_1 f_\mu}$

The occurring weight of the first keyword in the first field in the first record is shown in Eq. 10.

$$W_{cs}(1,1,1) = \frac{\lambda_{x_1 r_1 f_1}}{\sum_{j=1}^{\mu} \lambda_{x_1} r_1 f_j} \quad j \in [1, \mu] \tag{10}$$

The occurring weight of the ith keyword in the jth field of the γth record is shown in Eq. 11.

$$W_{cs}(i, \gamma, j) = \frac{\lambda_{x_i r_\gamma f_j}}{\sum_{j=1}^{\mu} \lambda_{x_i} r_\gamma f_j}$$

$$i \in [1, n] \quad j \in [1, \mu] \quad \gamma \in [1, \delta] \tag{11}$$

The position weight of keywords, represented as W_{wz}.

Usually there are several tables to store data in the database and in each of the tables there are many fields. When a keyword appears in different fields, its importance degree is different. For example, appearing in the title is more important than in the text. Appearing in the subject is more important than in subtitle or annotations. So it is necessary to consider further which field the keywords appear in.

Supposing the number of fields is μ. The position weight of the first field is θ_1. The position weight of the second field is θ_2. The position weight of the μth field is θ_μ.

The position weight that keywords appear in the first field is shown in Eq. 12.

$$W_{wz}(1) = \frac{\theta_1}{\theta_1 + \theta_2 + \cdots + \theta_\mu} \tag{12}$$

The position weight that keywords appear in the second field is shown in Eq. 13.

$$W_{wz}(2) = \frac{\theta_2}{\theta_1 + \theta_2 + \cdots + \theta_\mu} \tag{13}$$

$$\vdots$$

The position weight that keywords appear in the μth field is shown in Eq. 14.

$$W_{wz}(\mu)=\frac{\theta_\mu}{\theta_1+\theta_2+\cdots+\theta_\mu} \quad (14)$$

Then the position weight that keywords appear in the jth field is shown in Eq. 15.

$$W_{wz}(j)=\frac{\theta_j}{\sum_{i=1}^{\mu}\theta_i} \quad i,j\in[1,\mu] \quad (15)$$

Where, θ_j represents a constant

With the weight values W_{cs} and W_{wz}, how to calculate the record vector? The record vector T is shown in Eq. 16.

$$T_l=\sum_{p=1}^{\mu}[W_{cs}(l,p)\cdot W_{wz}(p)]=\sum_{p=1}^{\mu}\left(\frac{\lambda_{x_l f_p}}{\sum_{j=1}^{\mu}\lambda_{x_l f_j}}\cdot\frac{\theta_p}{\sum_{j=1}^{\mu}\theta_j}\right)$$

$$i,l\in[1,n]\ j,p\in[1,\mu] \quad (16)$$

While n is the number of query keywords. μ is the number of fields. x_l represents the lth keyword. f_p represents the pth field. $\lambda_{x_l f_p}$ represents the time that the lth keyword appears in the pth field. θ_p represents the weight of the pth field.

4.3. The calculation method of the query value

The query vector and the record vector are obtained, then the query value D_k can be calculated with Eq. 17. k is the number of records that found.

$$D_k=\sum_{l=1}^{n}Q_l\cdot T_l=\begin{pmatrix}Q_1\\Q_2\\\vdots\\Q_n\end{pmatrix}^T\cdot\begin{pmatrix}T_1\\T_2\\\vdots\\T_n\end{pmatrix} \quad l\in[1,n] \quad (17)$$

The query value of every record is calculated according to the formula. The records are sorted in descending order by the query value, and then the query results are obtained.

5. Conclusion

Archive information retrieval is important for archives management, and is a shortcut to find the archives for the users. Since the default results is the order of data entered and not the desired results. Users have to browse them one by one to find useful information. It makes them feel bother, time-consuming, labor-intensive. In order to solve this problem and improve the efficiency of query, relevance rank was researched in this paper.

The theory of relevance rank is to calculate a value for every return result record, and then sort the results in descending order according to this value. The calculation method is to express the query submitted and each record as a vector respectively, and then obtain a value by calculating the dot product of the two vectors. This value is the basis of the record rank. According to the magnitude of the query values, output the results in descending order. In this way, the records that with maximum relevance and concerned most by users are in the front, and it is easy for the user to see first.

The relevance rank method improves the original rank. It makes the return results more consistent with the users' query intent and the retrieval efficiency is improved.

6. References

[1] S. Guan, "Search Engine Sorting Algorithms Based on the Relation Degree of the Word," Master dissertation, China: LanZhou University, 2010.

[2] Y. Zheng and R. Qian, "A correlation ranking algorithm based on link analysis and application in topic-specific intelligent search engine," Computer Applications and Software, Vol. 24, No. 7, pp. 54-55, 65, July 2007.

[3] Q. Wen, "Research and development of search engine based on focus relevance ranking," Master dissertation, China: DongHua University, 2010.

[4] T. Zhang, "Research on the aggregation sorting technology in meta search engine," Master dissertation, China: Beijing University of Technology, 2012.

[5] W. Wei, K. Liang, R. Li, X. Gu and Z. Lu, "A page ranking algorithm based on topic similarity," Microelectronic & Computer, Vol. 25, No. 9, pp. 221-224, September 2008.

[6] H. Zan, Y. Su, B. Sun and S. Yu, "The WebPages Relevance Research Based on the Shallow Parsing," JSCL. Harbin China, pp. 501-506. August 2003.