BLOG CLASSIFICATION USING K-MEANS

Ki Jun Lee, Myungjin Lee and Wooju Kim
Yonsei University, seodaemoon-gu seongsan-ro 262, Seoul, Republic of Korea

Keywords: Search, Blog, Classification, Grouping, Clustering, K-means.

Abstract: With the recent exponential growth of blogs, a vast amount of important data has appeared on blogs. However, dynamic, autonomous, and personal features of such blogs make blog pages be quite different from those on general web pages in many aspects. As a result, this also causes many problems which cannot be handled properly by general search engines. One of the problems which we focused in this study is that blog pages are inherently poorly-organized and very much duplicated. This means the blog search engines cannot but provide the poorly-organized and duplicated results. To solve this problem, we propose a blog classification method using K-means and present a blog search result reorganization approach based on this method. In this study, firstly, we review the current status and their performances of blogs and blog search engines. Secondly, we adopt the K-means algorithm as a base algorithm and devise a blog title classification method to reorganize the blog titles resulted by a search engine. Finally, by implementing a prototype system of our algorithm, we evaluate our algorithm's effectiveness, and present a conclusion and the directions for future work. We expect this algorithm can improve the current blog search engines' usability.

1 INTRODUCTION

As the number of blogs – a compound word combining 'web' and 'log' – on the World Wide Web is growing (Technorati Weblog, 2006), they have created a brand new subset of the World Wide Web (Kumar, Novak, Raghavan, and Tomkins, 2003). Because blogs are created based on personal needs and are updated by individuals, blog posts are displayed with personal contents like diaries, one's experiences, observations, discussions, music, movies, and other topics, in essence whatever the bloggers want. This feature of blogs is absolutely different from those of other web pages, and it has fostered new research opportunities like information retrieval, text mining, social studies and blog searching - which we present in this paper. If some researcher offers these kinds of blog information with a more proper result format and more meaningful classification, it can give users useful and special information different from the ones presented by normal web pages.

Since general web search engines do not provide such a special interface for blogs, there are several blog search engines only for blogs (Bloglines, Blogpulse, BLOGRANGER, and BlogWatcher). Each of these search engines provide their own interface format based on their own algorithm, such as blog stats, topic selection, blogger selection, link selection, and so on. However, this area still has a long way to go in terms of effectiveness, since no result from any search engine can completely satisfy the needs of a variety of users.

There are some examples of the failure to get the right information when users try to search with a particular propose in mind. For example, let us suppose a user is trying to find information about a novelist the user knows just the name of. When inputting the novelist's name as a keyword, the user may want to know the ranking of the novelist's novel, which novel he or she should read first, how bloggers appreciate the novelist – the novelist's reputation - and a book review of the novel by the average reader, not by a professional book reviewer. However, it takes a lot of time to satisfy all these user needs through the general search engine classification system because of each particular blog's features (Aixin, Maggy, and Ying, 2007). Although some blog search engines support different categories for blog classification, since blog information reflects tremendous personal interest, such a static category has its limitations.

In this paper, we present a blog classification algorithm using K-means, in an attempt to solve the

Jun Lee K., Lee M. and Kim W. (2009).
BLOG CLASSIFICATION USING K-MEANS.
In *Proceedings of the 11th International Conference on Enterprise Information Systems - Software Agents and Internet Computing*, pages 61-67
DOI: 10.5220/0001949600610067

problems mentioned above. In Section 2, we introduce the papers related to the theme of this paper. In Section 3, we describe the search algorithm, system, and prototype. In Section 4, we wrap up the paper with our conclusions and directions for future work.

2 RELATED WORK

2.1 Blog Search

As blogs have been spotlighted as a new resource to search for information (Fujiki, Nanno, Suzuki, and Okumura, 2004), several search engines perform the search only for blogs, and there are research papers only on the topic of blog searching.

First, one paper entitled "A Study of Blog Search" shows the differences between blogs and general web pages in detail (Gilad, and Maarten, 2006). It focuses on queries to find unique features of blogs that one might apply in a blog search. This paper surely adds to blog searching studies. BLOGRANGER offers many kinds of searches (Fujimura, Toda, Inoue, Hiroshima, Kataoka, and Sugizaki, 2006). It helps users to reach the area that they intend to by presenting topics, blogger, reputation search, and navigational, informational and transactional searches. Its concept comes from the view that blogs are personal and thus different from other web-pages. One paper is on the use of the concept of a keyword map and feedback (Takama, Kajinami, and Matsumura, 2005). It consists of a keyword map as a method of providing feedback. When a user searches with a keyword, he or she gives words related to the keyword as feedback. The user may select the word as the domain that the user intends to look into, and this gives the user more precise search results corresponding to the user's needs. Finally, there is a paper that discusses using tags in blog classification (Aixin, Maggy, and Ying, 2007). In this paper, they analyze tags to find popular tags. They analyze the frequency of these popular tags to give a weight to each blog. This paper's approach is pretty similar to the one we use in this paper. They use tags created by bloggers to avoid a general search engine's categorization, and they try to classify blogs. However, after analyzing the tags, they use statistical standards to evaluate blogs, and such tags cannot include all the various ranges of blog domains. We attempted to make the category right for that keyword only.

2.2 K-means

K-means (MacQueen, 1967) is a clustering algorithm based on distances between instances. It divides data into K number groups based on a certain feature of the data. After dividing the data into K groups, it finds a centre value for each group. Then, it connects each instance of data to the closest centre point to make a new group. After dividing the data into new groups, it repeats the progress of finding new centre values for each group again and again. This regeneration is continued until it satisfies the critical point. Through this process, it tries to find the most suitable classification form of the given data.

In this paper, there are three main issues concerning the use of K-means. One is how to decide the value of K. Another is how to change titles to values in order to make it possible to use K-means. The third is how to decide the end point of K-means. These three issues are very important in determining the effectiveness of the use of K-means, and how we evaluate K-means dealing with these issues will decide the entire set of features of our system. Considering these three issues, we developed our system and algorithm. We introduce the steps in solving these issues in section 3.

3 SYSTEM & ALGORITHM

3.1 Problem Solving

The final object of every search engine is to provide the information intended by the user perfectly. That is why every search engine has its own search algorithm, weight and search classification. For example, the search engine 'Yahoo' (Yahoo) provides its own directory search organization, with categories like image search, job search, product search, audio search, and so on. The search engine 'Google' (Google), like Yahoo, also provides its own categories. However, since current web pages have become more specific and have been created in a wide variety of ways, current general search engines have an evident limitation with static directories, especially concerning blog searches. Because a blog is managed by an individual blogger, the blogger uploads whatever he or she wants, in whatever format. So, every keyword in a blog has its own themes and domains totally different from keywords of other blogs and other types of web pages. Users do not use a blog search to shop or to find a job or a map. If some current engines try to fit

blog information into current static categories, there is a possibility of misclassifying and thus failing to find an appropriate category. In this case, the categories made for users can limit users from finding proper data. Therefore, in order to attempt to satisfy various users' needs, we analyzed the contents from the keywords, without any other data. After analyzing the search results, we let users know all the major issues related to those keywords by ranking them in a clustered form. In addition, we showed users the relative importance of each group. With this system, we expected users to recognize a keyword trend faster than ever.

As we mentioned in section 1, problems remain which current search engines cannot solve. To solve some problems, we present the search algorithm that satisfies the following needs.

1. If searched results have the same keyword, can we cluster this information not with a static category but with a certain keyword's distribution?
2. Can we figure out the most important, the most popular and the most necessary information based on rank?
3. Can we see the information without overlapping and grasp the point at first sight?

We do not think that these suggestions will satisfy all the blog search users' needs. We also are not saying that every search engine should follow our algorithm or change theirs. All we are saying is, at least, this kind of access can be the one big improvement which can add power to current search engines and to the clustering concept discussed in the search engine section.

3.2 System Architecture

The prototype of the program used to test the algorithm based on this paper is similar to the following.

The whole system is managed like the schematic in figure 1, as you see below, and how each part of the prototype works is explained from 3.2.1 to 3.2.5

3.2.1 From System User to Search Engine

A user inputs a keyword just like any user does in existing search engines. Then we send the keyword to the normal search engine. In this case, we used the naver blog search engine (Naver), which is the most famous search engine in Korea. This search engine performs in the way it normally does.

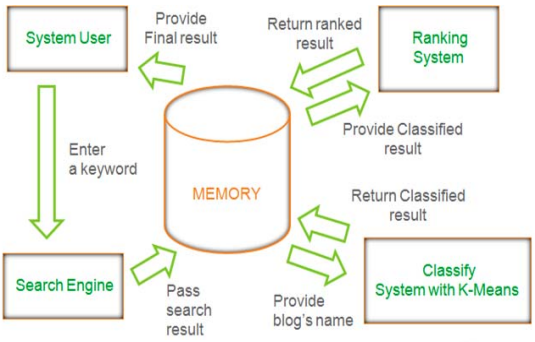

Figure 1: Architecture of Prototype System.

3.2.2 From Search Engine to Memory

After the blog search is finished, we extract the search data served by the naver blog search engine and send it to our prototype's memory. Through this process we simply extract the naver blog search engine's results without performing any editing.

3.2.3 Memory and Classification System

When the memory provides the search results to the classification system, it edits the search results to perform our classification algorithm. It detaches only titles from the search results to follow our classification system, which depends on the similarity between titles, and saves the titles in our data structure which we describe in the next section. Through this process, the search results provided by the general search engine is ready for our system. When this process is completed, the memory sends the newly created data structure to the classification system.

After the classification system clusters the search results, it sends its results to the memory. The detailed process of clustering the search results in the data structure is described in the next section. The classified results are also returned along with the data structure.

3.2.4 Memory with Ranking System

The memory sends the classified results provided by the classification system to the ranking system. The ranking system consists simply of a method which ranks the result groups based on their size.

The ranking system returns its ranked result to the memory when the ranking process is completed.

3.2.5 From Memory to User

After receiving the ranked results from the ranking system, the memory sends the final results to the user.

3.3 Definition of Key Algorithm

At first, we would like to present the basic definitions of our system which are used in this paper. These definitions indicate a form of data on the memory from the time of extracting the title from the search engine to the time the ranking system ranks the result.

Definition 1. Let $T = \{ t_1, t_2, \ldots, t_n \}$ be the title set. When we receive the titles, we name each of them as $t_1, t_2, \ldots, t_n$. For instance, if we receive 100 titles, then $|T| = 100$.

Definition 2. To earn a measurement from a title, we divide each title into morphemes. So, each title has its own morpheme group. We define this group as a keyword set described like $KS = \{ ks_1, ks_2, \ldots, ks_n \}$. The number n is the same as the n in Definition 1, since *KS* is a morpheme set of *T*. In addition to this, since each title consists of words, we define each title *KS* as $KS = \{ w_1, w_2, \ldots, w_m \}$. The number m depends on the morpheme number of each *T*.

Definition 3. From *KS*, we create a keyword pool as *KP*, described as $KP = \{ w_1, w_2, \ldots, w_x \}$. Basically, it collects the keywords from *KS* to make the entire keyword set without any keyword repetition. Therefore, *KP* has every keyword included in the search results. The number x means the number of all the morphemes the search engines have.

Definition 4. $KV = \{ kv_{ij} \mid i = 1, 2, \ldots, n, j = 1,2, \ldots, x \}$ is the keyword vector which tells us that a certain title has a certain morpheme. Every single value of KV_{ij} is 0 or 1. For example, when comparing *KS* with *KP*, if KS_{ab} is matched with KP_c we set the KV_{ac} vector value as 1. The number *a* could be any number which is in the range from 1 to n. The number *b* or *c* could be any number which is in the range from 1 to x. The number x means the number of all the morphemes the search engines have.

3.3.1 Title Analyzing

Since we chose the titles of the analyzed blog as the classification measure, we detached titles from the search results as the first step. This was not that hard to do if we could see the structure of the pages by using an HTML parser. The other parts of the pages are linked to the title of each page and are added to the final result.

3.3.2 Morpheme Analyzing

Before we adjusted the K-means to the search result served by the general search engine, we had to fit the form correctly for K-means. K-means requires a certain heuristic to divide each instance into groups. We have selected this form as vectors, as you can see in 3.3.1

To give some values to each title, we decided to divide the titles into morphemes and give them values of 1 or 0. We used a morpheme analyzer to extract morphemes from each title. In this process we erased each blog's unique characters, such as an emoticon.

3.3.3 Keyword Pool Making

In order to make a standard keyword, we created a keyword pool which had all keywords included in the search results. This was an essential task in our algorithm because we had to perform a vector product with each title in the search results, and this task made that possible. The process of making a keyword pool is quite straightforward. Comparing all the elements in KS, we can simply delete those that overlapped.

3.3.4 Keyword Vector Making

Finally, we made a keyword vector in order to give a specific measurement for performing K-means. Through this progress, we could calculate the distance between two tiles which reflected the similarity between them. And this similarity, the result of the vector product, plays an important role in classifying titles.

Making a keyword vector is not that complex a process. Adding to the explanation in 3.3's definition 4, here is a more concrete example. Let's suppose that there are only five morphemes in *KP* which can be described as *KP* = {a, b, c, d, e} and there are also two *KSs* which can be described as KS_1 = {a, b, e} and KS_2 = {a, c, d}. Then, the *KV* of KS_1 and KS_2 will be {1, 1, 0, 0, 1} and {1, 0, 1, 1, 0}.

3.4 K-means

3.4.1 First Classifying and Deciding K Value

When using K-means as a classification method, deciding the value of K is one of the most important issues. Although most cases using K-means set K as a static value, we set K as a dynamic value because we wanted to reflect the fact that each keyword has absolutely different kinds of domains and quantities. It makes every searched word have its own K. The process of deciding the K value is easy to understand. During the process of classification, we checked every *KS* in detail. The first one, KS_1, should consist of the first result group since there are no prior result groups. Following that, one performs vector product with the existing result group. Then, the following *KS* is added to the result group, thus making the biggest vector product value with it. If the vector product values performed with all the existing result groups are all zero, the following *KS* creates its own result group since the following *KS* has no similarity with the others. Finally, the number of result groups is the number of this searched word's K.

3.4.2 Centre Value and Classification

K-means is based on a reputation, so we should decide a centre value for each result group and reclassify all *KS*s into the new result groups. The centre value of each group is easy to find since we have each *KS*s vector value *KV*. Just add up all the vectors in each result group. Then divide each sum by the number of each group's *KV*s. This gives a perfect centre value for each result group. So, we added all *KV*s to the new group, which made the biggest vector product value with this group's centre value. After completing the classification process, we simply repeated this process from the beginning.

3.4.3 Deciding the End Point

Deciding the critical value to end K-means is also an important issue concerning K-means. Through 3.4.2's process, each search result may find its perfect position. Therefore, when there were no more changes in every result group, although K-means did its reputational classifying work, we stopped doing K-means. This offered the best result groups to the users.

3.5 Ranking System

Presenting search results with rank and distribution is also an important aspect of our system. The reason that we provide rank with distribution is that with rank only users cannot figure out the importance of each group completely. Among the classified groups, we calculate each group's size first. The largest one is the first one to be presented since the largest group can be considered as the most spotlighted one. The distribution is served in the form of a percentage. The percentage provides an additional relative importance of that group. Therefore, when users receive the final results of our system, users can understand the degree to which the subject and how many subjects are the most important in this keyword's area.

4 EVALUATION

In order to evaluate section 3's algorithm and system, we made a prototype which can perform real–time blog searches and estimate the effectiveness of the prototype. The prototype system's range is on Korean blogs and the estimation method is based on the CSIM method (Chung and Lee, 2001).

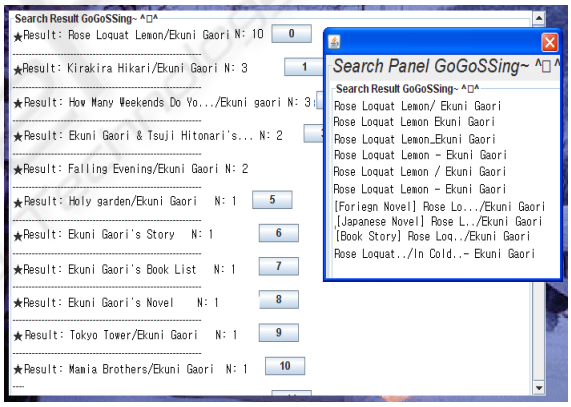

Figure 2: Translated Screen Image of Prototype System.

The prototype system demonstrates its result, as shown in Fig. 2. We performed this test in Korean but we translated Fig 2 into English to help you to see how the process works. First, it shows its result only by the group's title which can represent that group, and when a user clicks on the button following the title, a new panel shows the user the entire data set of the group. If the user clicks on a certain title in the result panel, the system links the user to that page. With such an interface a user can figure out the distribution of the data immediately.

Cluster SIMilarity (CSIM) is derived from Rand's method (Rand, 1971) which is often used in estimating clustered group similarity. It is a method

which applies Dice's coefficient to Rand's method in order to overcome some defects of Rand's method.

For the three types of web pages (Border, 2002), we decided to choose queries from informational classes which are estimated as the most proper ones for blog searches (Gliad and Maarten, 2006). We chose one query each from movie, music, and book categories. First, 'X-Japan,' the name of one of the most famous rock bands in Japan, is the query from music. Second, we chose 'Cha T.H,' who is one of the most famous actors in Korea, as the query word from movies. Lastly, 'Ekuni Gaori,' who is one of the most famous Japanese novelists, is the query from books.

We tested 50 sets of blog data from the Naver search engine, without performing any editing. Although our prototype system could test its algorithm with all blog pages on the web, we decided to test just 50 pages this time. We will test more pages for more accuracy later. Before our prototype system perofrmed its task, we made an ideal clustered set by hand. After the prototype system created its result set, we compared this with the ideal set by CSIM.

Table 1: The evaluation result.

Query word	X-Japan	Cha T.H.	Ekuni Gaori
CSLM value	0.857	0.711	0.805

Since the CSIM value is quite close to 1, we can conclude that the prototype system is successful in clustering blog information. Although the test was performed on only 50 sets of blog data, it certainly clustered data which should be clustered, so we think that the larger the example set is, the more exact the results will be. We expect that this system will be able to offer useful and special information to users and companies that want to know the public's response to their products or image.

5 CONCLUSIONS

In this paper, we discuss a blog search algorithm that considers the characteristics of blog content based on the assumption that the resultant blog classification can provide more valuable information to users. We also made a simple prototype to evaluate our algorithm. In order to test this system, we tried to find features of a blog and the problems of general search engines, and then find a solution which could solve those problems to an extent. We decided to use the concept of K-means as the classification method. We developed our own algorithm to adjust K-means to blog information. As shown in section 4, our algorithm and system provides certain benefits to users with clustered groups. It may not satisfy all the users, but it can give additional useful data to users and suggest a new approach to the blog search engine field.

For future research, there is something else to consider. There were three important issues in making an algorithm with K-means, as you can see in section 2.2, and we do not think that our solution suggested in this paper is the only possible one. So we will try to find the best solution which can extract a better weight from the blog and choose a better K and critical point. In addition, we will study more classification methods which can be matched more closely with blog searches. Finally, nowadays a variety of search algorithms and methods used in search engines exist. Since our final goal is to present the best blog algorithm, we will study other search mechanisms, including classification.

ACKNOWLEDGEMENTS

This research was financially supported by the Ministry of Knowledge Economy(MKE) and Korea Industrial Technology (KOTEF) through the Human Resource Training Project for Strategic Technology.

REFERENCES

Aixin, S., Maggy, S., Ying, L. 2007. Blog Classification Using Tags: An Empirical Study. In *ICADL 2007.*

Bloglines: http://www.bloglines.com/.

Blogpulse: http://www.blogpulse.com/.

BLOGRANGER: http://ranger.labs.goo.ne.jp/.

BlogWatcher: http://blogwatcher.pi.titech.ac.jp/.

Broder A. 2002. A Taxonomy of Web Search. In *SIGIR Forum.*

Chung, Y.M., Lee, J.Y. 2001. A corpus-based approach to comparative evaluation of statistical term association measures. In *J. of the American Society for Information Science and Technology.*

Fujiki, T., Nanno, T., Suzuki, Y., Okumura, M. 2004. Identification of Bursts in a Document Stream. In *First International Workshop on Knowledge Discovery 2004.*

Fujimura, K.,Toda, H., Inoue, T., Hiroshima, N., Kataoka, R., Sugizaki M. 2006. BLOGRANGER – A multi-faceted Blog Search Engine. In *WWW 2006.*

Gilad, M., Maarten, R. 2006. A Study of Blog Search. In *ECIR 2006.* LNCS 3936.

Google, http://www.google.com/.

Kumar, R., Novak, J., Raghavan, P., Tomkins, A. 2003. On the bursty evolution of blogspace. In *WWW'03:*

Proceedings of the 12th international conference on world wide web. ACM Press.

Macqueen J. 1967. Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics. University of California Press.

Mukul, J., Nikhil, B. 2006. BlogHarvest: Blog Mining and Search Framework. In *COMAD 2006*.

Naver: http://www.naver.com

Rand, W.M. 1971. Objective Criteria for The Evaluation of clustering Methods. In *J. of the American Statistical Association.*

Takama, Y., Kajinami, T., Matsumura, A. Application of Keyword Map-based Relevance Feedback to Interactive Blog Search. In *IEEE 2005*.

Technorati Weblog: State of the Blogsphere, http://technorati.com/weblog/2006/02/83.html

Yahoo, http://www.yahoo.com/.