IEEE Access
Multidisciplinary ⁝ Rapid Review ⁝ Open Access Journal

Received March 29, 2020, accepted April 24, 2020, date of publication April 28, 2020, date of current version May 11, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2990972

A Study of Optimizing Search Engine Results Through User Interaction

LIN-CHIH CHEN
Department of Information Management, National Dong Hwa University, Hualien 974301, Taiwan
e-mail: lcchen@gms.ndhu.edu.tw

This work was supported by the Ministry of Science and Technology of Taiwan under Grant 108-2410-H-259-048-MY3.

ABSTRACT In this paper, we propose a method for optimizing search engine results based on user interaction. This method generates different search results mainly through the user's operation on different search topics. There are two main differences between our method and the traditional personalized search method: personal privacy and storage space. First, traditional personal search methods need to record search and click records for individual users. However, these records have great personal privacy issues for users; especially in recent years, there have been personal privacy leaks that occurred in many large online companies (such as Facebook and Yahoo). Secondly, because different users need to record their own search records, the size of the storage space is closely related to the number of users and the amount of search records stored. However, the sum of individual users' search and click records is a huge storage space in today's Internet environment. To avoid these two issues for traditional personalized search methods, this study proposes a storage-free approach based on individual users operating on different search topics. In general, in addition to avoiding personal privacy and storage space issues, our method can also achieve optimal linear time in generating personalized search results.

INDEX TERMS Document clustering, machine learning, natural language processing, personalized topic search, user log.

I. INTRODUCTION

With the rapid development of the Internet in recent years, people increasingly rely on the Internet to gain knowledge. The total amount of data on the Internet will grow 10 times from 2013 to 2020, and the total will increase from 4.4 zettabytes to 44 zettabytes [1]. In such a huge amount of information, how to search for relevant information through efficient methods and quickly respond to the user's query needs become an important task. An effective way is to use search engines to help people quickly find the information they want.

A. RESEARCH QUESTIONS FOR THIS STUDY

However, search engines face two problems: (a) **too much data is returned** [2], and (b) **users often enter short queries** [3], [4]. The first problem is the search engine will return hundreds of thousands to millions of results for general user queries [5]. For so many returned results, the user is usually not interested in most of the results. Based on the results of the relevant statistics [6], the percentage of users viewing the top ten search results (or the first page of the search results page) is 58%, and the page views of the first three pages reach 86%. That is, most users are only interested in the search results for the first three pages. The second problem is the queries entered by the user are often very short. According to the relevant statistics [6], the average query length is 2.21 words, and this length also contains the usual stop words. As a result, queries that are enough for the search engine to judge will be shorter. In such a short query, the search engine is difficult to accurately define the user's real search needs.

B. RESEARCH IDEAS FOR THIS STUDY

In this paper, we propose a system to solve these two problems. First, to solve the problem of returning too much data, the system processing data set will focus on the front search results. According to statistics [6], most (about 86%) users only care about the first three search results pages. That is, most users only focus on the first 30 search results. The advantage of focusing on these small datasets with high search rankings is that the system can quickly respond to

The associate editor coordinating the review of this manuscript and approving it for publication was Jenny Mahoney.

user search needs. Second, to solve the short query problem, the system results will be presented in different discussion topics instead of purely flat search results. Compared with the traditional flat list search results, the topic-based search results can not only help users quickly search for topics of interest, but also effectively solve the synonymous and polysemy problems that may occur in the query [7]. In general, there may be parent-child relationship between topics, that is, the parent topic has a more general discussion topic, and the child topic has a more detailed discussion topic. Using the hierarchical tree approach to clearly present the parent-child relationship between query topics is especially useful for poor or ambiguous queries [8], [9]. To present the relationship, the system uses the operations of the sets to calculate possible inclusion relationships between topics. The advantage of using a set is the system can find the relationship at the best time, as described in the experimental section. For the same query, the search needs of different users may not be the same, but how to meet different user needs become a difficult task. This study achieves this task based on topics and binary options selected by individual users. The purpose of binary operations is to quickly respond to individual users' search needs.

C. RESEARCH OBJECTIVES FOR THIS STUDY

Traditionally, personalized search (such as Google History and My Yahoo) generate search results for individual users based on user log files, such as clicking and browsing history. There are two main problems with this approach: (1) personal privacy, and (2) large storage space. For the first problem, since the personalized search requires users to log in to its account, all of the personal browsing records are recorded in the user log file. Such a way is prone to personal privacy issues, especially in today's countries that emphasize privacy. For the second problem, since all users' browsing must be recorded in the log file, this approach needs a lot of storage space. To solve these two problems, the first research objective of this study is to build a personalized search engine that does not use log files. There are two main steps in building our personalized search engine: (1) building hierarchical topic results based on Web page snippets, and (2) generating personalized search results based on topics selected by individual users and personalization option. Since the hierarchical topic presentation method is particularly useful for poor or ambiguous queries, the second research objective of this study is to present all topics in the form of hierarchical trees. Since our personalization options are based on binary operations, the third research objective of this study is to provide users with various possible combinations of topics to achieve the goal of personalized search.

Because the search needs for each user are not exactly the same, general search engines often use log files to record historical search and click records for individual users. However, because this method records the user's search history in detail, it may cause the concern that personal privacy may be violated, especially the recent leakage of personal information of online companies such as Facebook and Yahoo. On the other hand, it needs a large amount of storage space to record the search history of all users in detail. This also requires a significant investment for search engine operators. In this study, we use a method that does not record users' historical search records to achieve individual user search needs. In our method, we only temporarily store the query entered by the user, selected topics and personalization option for subsequent personalized search processing. Once the personalized search process is complete, we delete all data stored in the system by users. Therefore, in fact, we do not store any user search history in our system, so our method can avoid personal privacy and storage space issues. In addition, according to the relevant statistical results [3], the length of the general user input query is short. The main reason is that it is often troublesome for the user to think about the topics or keywords related to the input query. Through our study, we actively recommend the topics or keywords related to the input query, which can save the user's thinking time considerably.

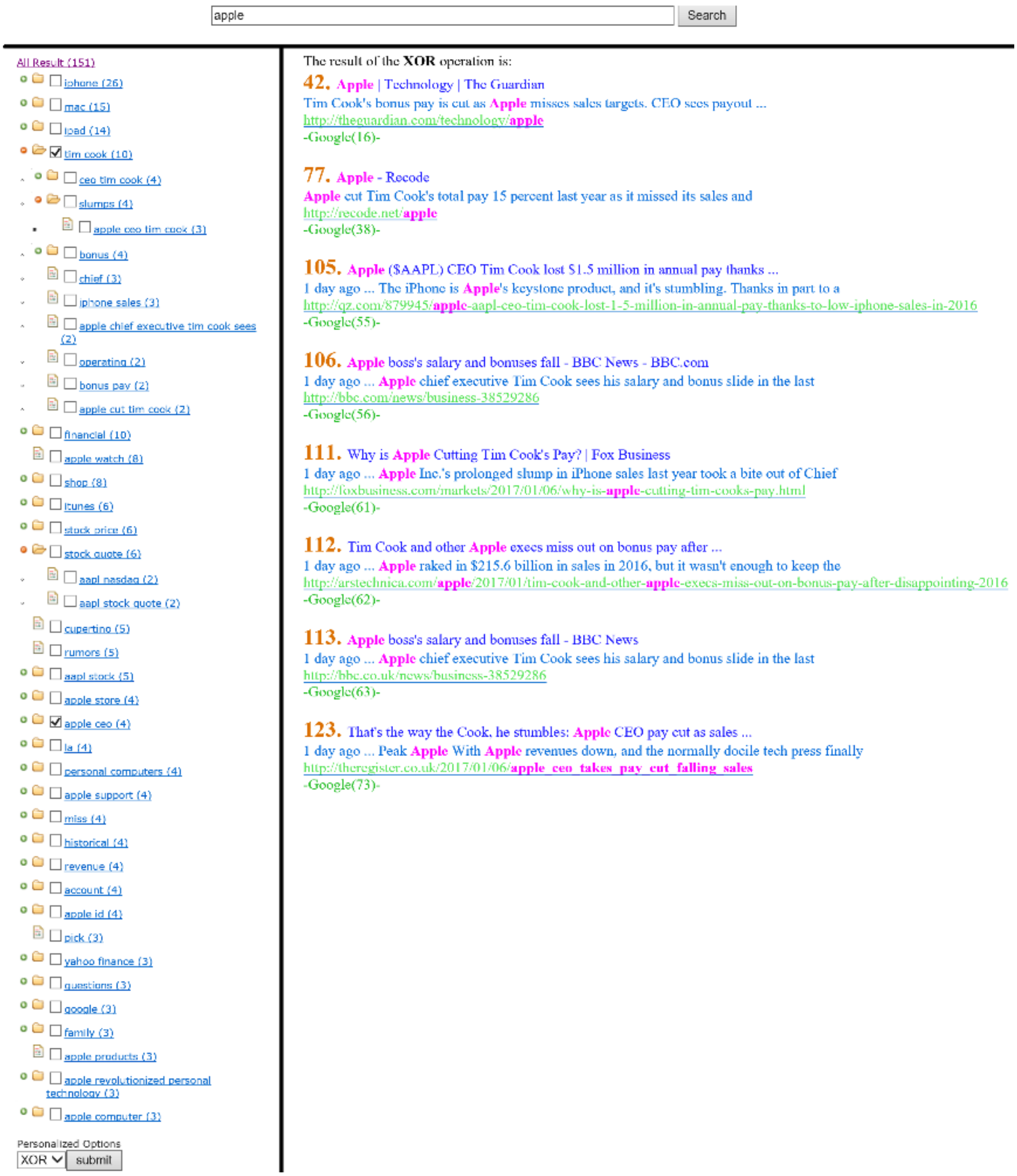
D. A SIMPLE EXAMPLE TO ILLUSTRATE THE RESULTS OF THIS STUDY

Figure 1 is a screenshot of the experimental system in this study when the query is ''apple''. The left-hand side of the figure is a list of topics that are likely to be relevant to the user's query. Here, the system not only generates query-related discussion topics, but also builds suitable parent-child relationships based on subordination between topics. For example, the ''stock quote (6)'' topic in the figure contains two sub-topics related to it, namely ''aapl nasdaq (2)'' and ''aapl stock quote (2)''. In addition, the system also provides personalized search abilities, such as the ''Personalized Options'' option in the lower left corner of the figure. When individual users select different topics they are interested in and select the personalized option they want, the system presents them with personalized results on the right-hand side of the figure. For example, in the figure, when a user chooses the topic of interest to include ''tim cook (10)'' and ''apple ceo (4)'', and through the ''XOR'' personalization option, the system will only produce 8 search results without returning all 151 search results. The system currently offers the ''AND'', ''OR'', ''XOR'' and ''NOT'' personalization options, which are explained in Section III.C.

The following sections of the paper are organized as follows. Section II discusses and compares the literature related to this study. Sections III and IV discuss the idea and experimental results of this study. Section V summarizes this paper and proposes future research directions.

II. LITERATURE REVIEW AND COMPARISON

In this section, we discuss and compare the four types of literature related to this study, namely, personal search, topic acquisition, document clustering, and recommendation system. We provide a table for each type to help readers to read and compare.



apple | Search

All Result (151)

- □ iphone (26)
- □ mac (15)
- □ ipad (14)
- ☑ tim cook (10)
  - □ ceo tim cook (4)
  - □ slumps (4)
    - □ apple ceo tim cook (3)
  - □ bonus (4)
  - □ chief (3)
  - □ iphone sales (3)
  - □ apple chief executive tim cook sees (2)
  - □ operating (2)
  - □ bonus pay (2)
  - □ apple cut tim cook (2)
- □ financial (10)
- □ apple watch (8)
- □ shop (8)
- □ itunes (6)
- □ stock price (6)
- □ stock quote (6)
  - □ aapl nasdaq (2)
  - □ aapl stock quote (2)
- □ cupertino (5)
- □ rumors (5)
- □ aapl stock (5)
- □ apple store (4)
- ☑ apple ceo (4)
- □ la (4)
- □ personal computers (4)
- □ apple support (4)
- □ miss (4)
- □ historical (4)
- □ revenue (4)
- □ account (4)
- □ apple id (4)
- □ pick (3)
- □ yahoo finance (3)
- □ questions (3)
- □ google (3)
- □ family (3)
- □ apple products (3)
- □ apple revolutionized personal technology (3)
- □ apple computer (3)

Personalized Options
XOR | submit

The result of the **XOR** operation is:

42. **Apple** | Technology | The Guardian
Tim Cook's bonus pay is cut as **Apple** misses sales targets. CEO sees payout ...
http://theguardian.com/technology/**apple**
-Google(16)-

77. **Apple** - Recode
Apple cut Tim Cook's total pay 15 percent last year as it missed its sales and
http://recode.net/**apple**
-Google(38)-

105. **Apple** ($AAPL) CEO Tim Cook lost $1.5 million in annual pay thanks ...
1 day ago ... The iPhone is **Apple**'s keystone product, and it's stumbling. Thanks in part to a
http://qz.com/879945/**apple**-aapl-ceo-tim-cook-lost-1-5-million-in-annual-pay-thanks-to-low-iphone-sales-in-2016
-Google(55)-

106. **Apple** boss's salary and bonuses fall - BBC News - BBC.com
1 day ago ... **Apple** chief executive Tim Cook sees his salary and bonus slide in the last
http://bbc.com/news/business-38529286
-Google(56)-

111. Why is **Apple** Cutting Tim Cook's Pay? | Fox Business
1 day ago ... **Apple** Inc.'s prolonged slump in iPhone sales last year took a bite out of Chief
http://foxbusiness.com/markets/2017/01/06/why-is-**apple**-cutting-tim-cooks-pay.html
-Google(61)-

112. Tim Cook and other **Apple** execs miss out on bonus pay after ...
1 day ago ... **Apple** raked in $215.6 billion in sales in 2016, but it wasn't enough to keep the
http://arstechnica.com/**apple**/2017/01/tim-cook-and-other-**apple**-execs-miss-out-on-bonus-pay-after-disappointing-2016
-Google(62)-

113. **Apple** boss's salary and bonuses fall - BBC News
1 day ago ... **Apple** chief executive Tim Cook sees his salary and bonus slide in the last
http://bbc.co.uk/news/business-38529286
-Google(63)-

123. That's the way the Cook, he stumbles: **Apple** CEO pay cut as sales ...
1 day ago ... Peak **Apple** With **Apple** revenues down, and the normally docile tech press finally
http://theregister.co.uk/2017/01/06/**apple_ceo_takes_pay_cut_falling_sales**
-Google(73)-

FIGURE 1. A screenshot of the experimental system.

A. PERSONAL SEARCH

When the user enters the query, the traditional search engine (such as the Google search engine) analyzes the page links that match the query and ranks the relevant pages [10], [11]. The ranking of the page is mainly based on the voting behavior between the links [12], [13]. The ranking results of the pages generated by the voting method are sorted according to the majority of user preferences. However, this method does not consider the differences in page preferences between different users [14]. The personal search mainly adjusts the entire search result list according to the preference parameters input by the individual users [15], [16]. Since the parameters entered by different users are not the same, the list generated by the personalized search is not the same. Personal search can mainly provide different search results according to the search habits or needs of different users.

According to the research done by the relevant researchers [17]–[21], personal search can be divided into two ways based on whether the user browsing history is stored. Table 1 shows the comparison of these two ways.

Traditionally, personalized search processing is to store a user's browsing history and use it to recommend, such

TABLE 1. Different perspectives on personal search.

Way Perspective	Stored	Non-stored
Advantages	The results are closer to the user's search needs	● No data storage problem ● No registration required ● The results are produced dynamically ● No privacy issues
Disadvantages	● Data storage problem ● Registration required ● Results cannot be dynamically produced ● Privacy issues	Results may not be of interest to individual users
System	Characteristic	Disadvantage
Google History	User log files	1. Personal privacy 2. Large storage space
My Yahoo		
TagMySearch	1. No personal privacy and large storage space issues 2. Flat topic results 3. Support a personalization option (OR)	1. Hierarchical topic results are not supported 2. Long computing time 3. Poor performance
WSC	1. No personal privacy and large storage space issues 2. Hierarchical topic result	Does not support any personalization options
Carrot2	1. No personal privacy and large storage space issues 2. Flat topic results	1. Hierarchical results are not supported 2. Does not support any personalization options
This Study	1. No personal privacy and large storage space issues 2. Hierarchical topic result 3. Linear time 4. Better performance 5. Support any personalization options	
Researcher	Idea	
Akhlaghian et al. [22]	Ontology and fuzzy concept network	
Jiang and Deng [23]	RSS interest model	
Leung et al. [24]	A set of conceptual preferences	
Mishra et al. [25]	The user's profile and location	
Singh and Alhadidi [26]	Agent-based	
Ramesh and Andrews [27]	Analyze user's Facebook activity	
Tabrizi et al. [28]	Citation network	
Wang et al. [29]	EM algorithm	
Nam et al. [30]	Lifelog	

as Google History [31] and My Yahoo [32]. This way allows users to browse their past searches while saving each search record and its matching search results. This way includes two types of data analysis: non-automated and automated [33], [34]. Non-automated analysis needs the user to enter the relevant information to set up the browsing history. This analysis needs users to enter much extra information, easily lead to user burden. Automated analysis based on user history browsing and clicks to suggest suitable search results. The drawback of this analysis is that when the history and click too much, the analysis time is time-consuming.

The advantage of stored browsing is that it produces search results that are closer to the user's search needs because it stores each person's browsing history and clicks history. However, its disadvantage is that it needs much storage space, and it needs the user to register to identify different users. In addition, it cannot dynamically produce search results because its analysis is time-consuming. Personalized searches based on stored browsing history have a privacy issue in some way [17].

To solve the above-mentioned disadvantages, the relevant researchers [27], [35]–[37] recommend the use of a non-stored way to achieve personal search. Of course, compared with the stored way, this way does not store any user browsing and clicking history, so the search results may not be fully consistent with each user's search needs. This way mainly uses the clustering technique to achieve the personal search. Users can select different topics produced by clustering, and the system displays personalized search results according to the selected topic. The current well-known academic systems are TagMySearch (SnakeT evolution) [17], [38], WSC [39] and Carrot2 [7], which produce a series of topics and checkboxes based on Web page snippets so users can select topics of interest. Compared with other systems, TagMySearch builds a personalized search system with the ability to select multiple topics at the same time. That is, the user can perform an OR operation on different topics.

TABLE 2. A comparison of recent research on topic information acquisition.

Researcher	Method	Object	Focus	Advantage
Chen [42]	Topic time parameters based on LDA	Google blog search	Update timeline for different blog posts	(1) Fast computing (2) Distinguish the relationship between topics at different times
Ahmad et al. [43]	Recursive topic acquisition	YouTube	Expand topic vocabulary	(1) Visual presentation (2) Comparable between topics
Lee et al. [44]	Expert tweet to identify topics	Twitter	Topic nutrition and energy	(1) Topic is highly recognized (2) The timeline is highly readable.
Kim et al. [45]	Dynamic topics based on NMF	NYT & VisPub	Topic expandability	(1) Less computing time (2) Provide an interactive interface
Zhao et al. [46]	An offline pre-training method	Twitter & Yelp	Training model	(1) Processing spatio-temporal reviews on documents (2) Separate multiple topics

However, besides the OR operation between multiple topics, it is possible for the user to perform other binary operations on different topics. With our personalization options (various binary operations), we can automatically help users tailor personalized search results to fit their needs.

Many researchers have tried to use different ideas to develop various personal search. Akhlaghian *et al.* [22] used the idea of ontology to improve the user's browsing history and produce better personal search results. The method is to use the ontology for user browsing history to produce a fuzzy concept network. Search results produced by this network will be more responsive to users' search needs. To set up the interest model of users' browsing history, Jiang and Deng [23] built the model of RSS (Real Simple Syndication) based on user implicit feedback. Based on individual users' automated search feedback, the RSS interest model continues to update and improve personal search accuracy. Leung *et al.* [24] produced a set of conceptual preferences from search results and user click data. These produced preferences are used to adjust the search engine ranking function for individual users. Mishra *et al.* [25] first discovered user profiles based on user activities and interests on the social network. They then produce a location-based personalized search engine based on the profile and user location. Singh and Alhadidi [26] used three agents to create and refine search results for individual users. These agents analyze the user's past usage behavior to produce individual user browsing history. Ramesh and Andrews [27] analyzed and identified information about users' interests based on their activities on Facebook. The user's personalized search results will be re-ranked based on this information. Tabrizi *et al.* [28] used the citation network to assist users in searching for their personally appropriate academic documents and to adjust the results of subsequent personal searches through feedback mechanisms. Wang *et al.* [29] used an EM (Expectation Maximization) algorithm to solve the problem of highly sparse clicks in personal searches. Nam *et al.* [30] built an android-based personal search system based on the lifelog collected by nine smartphone sensors and used it to get new and meaningful information about the user.

B. TOPIC ACQUISITION

In general, when people write relevant documents, they first consider the various topics to be expressed in the document, and then describe the relevant text content in detail according to different topics [40], [41]. Therefore, a good topic acquisition method can effectively help people write good documents. Table 2 shows some new research on how to acquire topic information from the collected documents.

Chen [42] proposed a time topic model based on the LDA (Latent Dirichlet Allocation) semantic analysis model, called LTR (Latent Time Relationship), to improve Google blog search performance. The LTR model adds a series of topic time parameters to the LDA model to improve the impact of time factors on the topic. The LTR model clusters blog posts with similar update timelines based on the topic's time distribution, and sorts the topics according to the discussion heat of the posts. Besides the fast computing time, the advantages of LTR can also distinguish the changes in topics under different timelines.

Ahmad *et al.* [43] proposed a recursive topic acquisition method for the YouTube platform. The method first collects YouTube search pages based on certain seed queries and extracts seed queries and neighboring words to form a topic vocabulary network. Next, the method repeatedly runs the following tasks until enough topics are collected: (1) collecting YouTube search results based on unprocessed topics in the topic vocabulary network and (2) running proximity searches using YouTube results to expand the original topic vocabulary network. The advantages of the method include two: (1) visualizing the topic to help the user understand the topic, and (2) comparing the importance of different topics.

Lee *et al.* [44] proposed a method to identify various emerging topics in Twitter's tweet. The method first identifies emerging topics from tweets written by experts. Next, the method uses the nutrition of the topic (i.e., the significance of the topic) and the energy of the topic (i.e., the time interval of the topic) to calculate the importance of different topics. Finally, the method uses the timeline to distinguish the importance of emerging topics in different time intervals. The advantages of the method are twofold: (1) the topic identified

TABLE 3. A comparison of document clustering.

Clustering Perspective	Data-oriented		Snippet-oriented	
Input	Data vector		Web snippet	
Output	Cluster centroid		Search topic	
Purpose	The average centroid is smaller		Topics can represent user search needs	
Advantage	Interpretable numerical meaning		The topic for the user has a higher reference value	
Disadvantage	The value may be meaningless to the user		Different cluster methods are difficult to compare	
Method	Hierarchical	Partitional	Flat	Hierarchical
Practice	Top-down or bottom-up	the concept of cutting space	In the same hierarchy	A parent-child relationship tree
Ideas	A fuzzy association rule mining algorithm [48] Semantic relation vector [49] A set of patterns based on whether cluster members change [50] Average and complete links [52] An online training workload model [53] Hyperlinks between documents [55] Automatically find the number of clusters [57] FSPSOTC algorithm [59] Hierarchical clustering algorithm based on edge-weighted similarity [60]		Carrot2: Lingo and STC algorithms [7] WSC: Extended STC algorithm [39] Singular value decomposition [51] TagMySearch: The graph of topics [38] Conceptual terms based on Wikipedia [54] PREFCA [56] Cost-effective GA [58]	

by the expert's view is highly recognized, and (2) the use of the timeline representation of different topics is highly readable.

Kim *et al.* [45] proposed a method based on NMF (Non-negative Matrix Factorization) to dynamically generate topics from NYT (New York Times) articles and VisPub conference papers. The method first uses NMF to establish possible preliminary topics in the collected documents. When users want to further expand the initial topic, the method then uses a dynamic NMF to generate detailed topics. The advantages of the method are as follows: (1) less computational time to generate dynamic topics, and (2) providing an interactive interface to generate topics.

Zhao *et al.* [46] proposed an offline pre-training method to find topics of interest to users from spatio-temporal documents such as Twitter tweets and Yelp reviews. The method first uses an offline pre-training method to produce a training model for the collected Twitter and Yelp's spatio-temporal documents. Next, for a new document, the method uses an online algorithm and the training model to predict trends in the topic of time and space for the document. The advantages of this method include the following: (1) it can properly handle spatio-temporal reviews on documents, and (2) it can separate multiple topics in the same document.

C. DOCUMENT CLUSTERING

The main purpose of document clustering is to divide a complex data set into different clusters. Document clustering based on the input source and output data can be divided into data-oriented clustering and snippet-oriented clustering. Table 3 shows the comparison of these two clustering types. The data-oriented clustering first transforms the document into vectors, then clusters the similar vectors through some data clustering methods, and finally calculates and outputs the matching centroids for each cluster. Its purpose is to expect the average centroid of all clusters to be smaller [47]. The advantage of this clustering is the cluster has an interpretable numerical meaning because its output is the centroid of the digital type. However, a clustering method with only a numerical meaning may be meaningless to the user because the user finally cares how the text or graphic is presented after clustering [7]. This clustering typically uses a hierarchical or partitional method to clustering related documents. The hierarchical method uses a top-down technique or a bottom-up technique to split or merge all documents. The partitional method uses the concept of cutting space to cluster some documents with similar distances into a cluster.

Many researchers have used data-oriented clustering to solve different types of information retrieval problems. Chen *et al.* [48] used a fuzzy association rule mining algorithm to find a set of highly relevant fuzzy frequent itemsets that contain candidate clusters. Through these candidate clusters, all documents will be clustered into a hierarchical clustering tree. Jing *et al.* [49] used a set of vectors to represent the semantic relations between terms in a document. They found the clustering effect was better when the distance of vector space in different clusters was greater. Chiang *et al.* [50] produced a series of patterns based on whether cluster members change during each iteration of the clustering process. They quickened the speed of clustering by compressing patterns and finding impossible to change patterns. Nassif and Hruschka [52] studied how to efficiently automatically cluster related documents in computer forensics analysis. They found that when calculating the distance between clusters using average link (average distance between clusters) and complete link (the longest distance between different clusters), this type of document will produce the best clustering effect. Wang *et al.* [53] proposed an incremental clustering algorithm for online training workload mode to detect whether the Web application is in an abnormal state to improve the reliability of the application. Gamare and Patil [55] used the hyperlinks between documents to cluster. They used an agglomerative hierarchical clustering to cluster documents when there is at least one hyperlink between documents. Wang *et al.* [57] proposed an improved online workload clustering method to automatically detect non-spherical clusters and automatically find the number of clusters. Abualigah *et al.* [59] used a FSPSOTC (Feature Selection using Particle Swarm OpTimization Clustering) algorithm to improve the text clustering algorithm by finding a new subset of document features. Chunlin *et al.* [60] used a hierarchical clustering algorithm based on edge-weighted similarity to detect communities in social networks for more accurate information recommendation services.

The snippet-oriented clustering first uses the search engine to gather the relevant snippets, then uses the suitable NLP (Natural Language Processing) techniques to remove the noise in the snippet, and finally produces the search topics based on the noise-free snippet. Its purpose is to expect that all the topics correctly represent the needs of the user's search [39]. The advantage of this clustering is the topic is presented in text or graphics, so it has a high reference value for the users. However, it is difficult to compare the advantages and disadvantages of different clustering methods because their output is a topic rather than a number [61]. This clustering typically uses a flat or hierarchical method to present relevant topics. The flat method presents all the produced topics in the same hierarchy. The hierarchical method sets up the parent-child relationship tree for all the produced topics.

Many researchers have used snippet-oriented clustering to help users find suitable topics. Carpineto, *et al.* [7] used Lingo and STC (Suffix Tree Clustering) algorithms to develop a topic search system. They used a flat form rather than a hierarchical form to show the relationship between topics. Chen [39] developed an extended STC algorithm to express hierarchical relationships between topics. The algorithm can present the topic as a sentence rather than as a word. Prakash and Hanumanthappa [51] used singular value decomposition to produce a variety of possible candidate topics. The way of selecting a topic is determined by whether the candidate topic is greater than the threshold. Scaiella, *et al.* [38] used the graph of topics to show the relationship between different snippets and topics. The nodes in the graph contain related topics and snippets, and the edges and their weights are produced by analyzing the links between Wikipedia entries. Researchers [54] used Wikipedia to capture conceptual terms and decide whether the terms appear in the collected snippets to produce relevant candidate topics. If the clustering similarity between any two topics is larger than the threshold, the two topics will be clustered into a cluster with a common idea. Negm *et al.* [56] used a PREFCA (Portal Retrieval Engine based on Formal Concept Analysis) to analyze the snippets returned by search engines to find conceptual links between different snippets. Chen [58] proposed a cost-effective GA (Genetic Algorithm) to generate page clipping results consisting of snippets corresponding to different topics.

D. RECOMMENDATION SYSTEM

Traditionally, search engines return a list of relevant search results based on user input queries. Since the search engine has returned many results, most users tend to only look at the previous search results [6]. Since users often input short queries [6], it is difficult for the search engine to correctly understand the user's true search intent in such a short query. Some researchers have developed relevant search recommendation systems that attempt to use the results of post-processing of search results to give users appropriate search suggestions. According to the results of relevant researchers, the recommendation system can be divided into the following three types: keyword suggestion, question answering, topic searching. Table 4 is a comparison of these three types.

The keyword suggestion type produces a series of related suggested keywords based on the query entered by the user. The main practice of this type uses the proximity search method to find the keywords related to the query in the search log and present them based on the popularity of the keywords. [62]–[65]. This type has the following two advantages. The first one is that it is easy to implement because it uses the proximity search to simply compare the search logs [62], [66]. The second is that it can effectively find suggested keywords with hot trends [67], [68]. However, this type has the following two disadvantages. The first one is that the majority of the keyword suggestion systems can only find syntactically related rather than semantically related suggested keywords because it uses proximity search to search the search log [69], [70]. The second is that it needs to store a huge amount of user search logs [68], [71]. Some researchers have tried to develop different keyword suggestion systems to provide relevant recommendations to users. Jiang *et al.* [72] proposed a new keyword suggestion paradigm to effectively integrate diversity and personalization into a unified framework. In this framework, the suggested keywords are effectively diversified to cover different aspects of the input query while personalizing the ranking of suggested keywords to ensure that the highest ranked keywords are consistent with the user's personal preferences. Renjie *et al.* [73] proposed a method to utilize video clusters on a referrer video graph to obtain relevant suggested keywords and rank the keywords based on their relevance and the potential to attract video viewing. Zhou *et al.* [74] proposed a method based on generative neural network to establish relevant suggested keywords. In addition to generating diversified and domain-consistent keywords, the method can automatically generate keywords in different fields according to user needs.

The question answering type returns a highly accurate Web page result or a clear answer string based on a complete and colloquial question query. The main practice of this type uses a series of NLP techniques to understand the user's questions and to use the question classifier module to determine the type of question. Finally, it answers the appropriate answers from the knowledge base based on the type of question [75]–[77]. This type has the following two advantages. The first one is that it answers the user's question with a precise answer [76], [78]. The second is that it has a fairly high accuracy rate for specific domains (such as medicine or automotive maintenance) [76], [79]. However, this type has the following two disadvantages. The first one is that human questions are

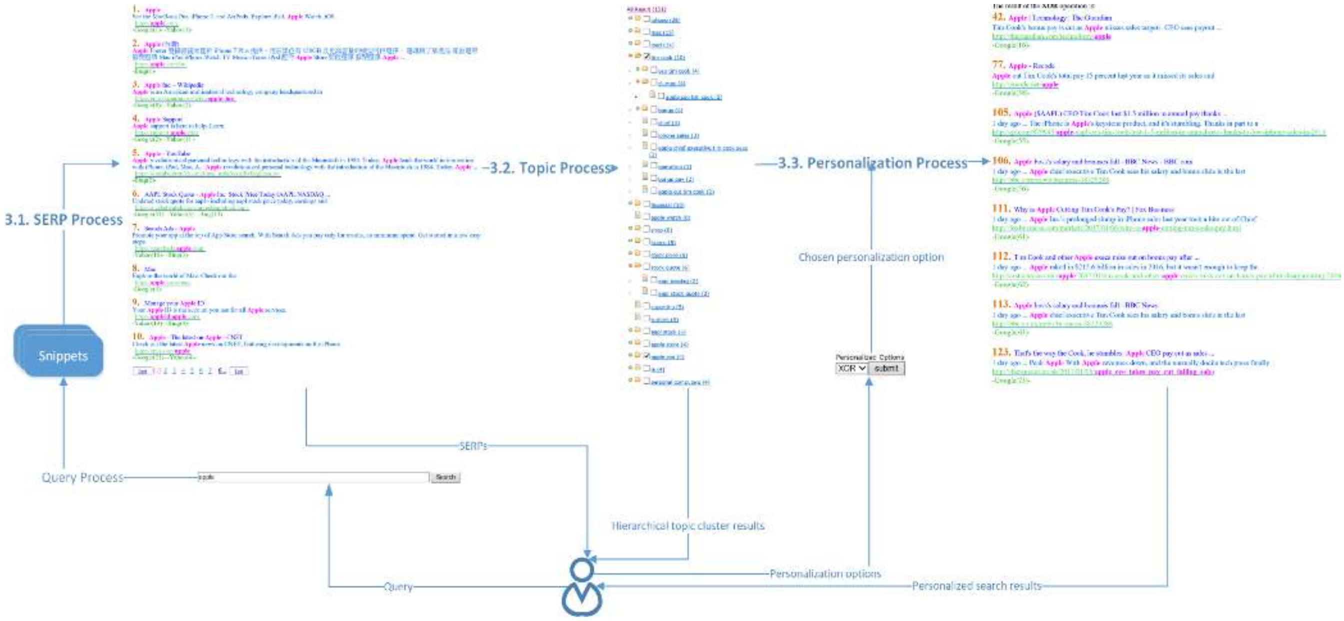


FIGURE 2. The research flow for this study.

complex and diverse, so this type is more difficult to apply to open domain questions. [80], [81]. The second is that the accuracy of open domain questions tends to be low [82], [83]. Some researchers have used question answering techniques to build relevant recommendation systems. Brill *et al.* [84] proposed an AskMSR question answering system that uses N-gram techniques to resolve user questions. The system differs from most question answering systems in its dependence on data redundancy rather than on sophisticated language analysis of questions or candidate answers. Abacha and Zweigenbaum [76] proposed a MEANS question answering system that uses NLP and Semantic Web technologies to solve medical domain related questions. The system is based on Semantic Web technologies which provide more expressiveness, standard formalized language and makes corpus and question annotations shareable via the Web. Bhandwaldar and Zadrozny [85] proposed a UNCC question answering system that uses extractive summarization techniques to solve biomedical domain related questions. The system answers facts and lists questions based on the method of named entities and the summary of the constructed paragraph size based on the lexical chain.

The topic searching type produces different search topics based on clustering results between documents. The main approach of this type uses different document clustering methods to generate different search topics [7], [17], [86]. This type has the following two advantages. The first one is that the specific topic it generates can bring together related documents so that it can effectively reduce the time for user judgment [55], [87]. The second is that it is based on a short document (a fragment of a Web page) to cluster related documents so it can quickly produce relevant topic results [58], [88]. However, this type has the following two disadvantages. The first one is that the topics it generates are closely related to the source document. Since its source document is a fragment of the Web page rather than the entire Web page to generate the topics, it cannot cover all the topics of the Web page [7], [89]. The second is that it produces a series of topic names rather than numerical results, so it is more difficult to evaluate different methods [58], [61]. Some researchers have used different topic searching methods to build relevant recommendation systems. Scaiella *et al.* [38] established a TagMySearch topic search system that uses a topic graph to represent the relationships between documents and topics. The nodes in the graph represent related topics and documents, while the edges and their weights are generated by analyzing Wikipedia entries. Gamare and Patil [55] proposed a system that uses HAC (Hierarchical Agglomerative Clustering) and link-based algorithms to cluster the documents based on the contents of the page and the links in the pages. Chen [58] proposed a hybrid system that uses an N-gram language model and a hashing method to generate different topics. To facilitate the user to browse the topic results, it also uses the concept of mathematical set to place different topics into a topic tree.

III. RESEARCH METHOD

In this section, we describe and discuss the design methods and mathematical models used in our experimental system. Figure 2 is the research flow for this study. First, the user enters a query into our system. The ‘‘Query Process’’ collects relevant snippets from the search engines based on the user query. Next, the ‘‘SERP Process’’ uses a series of NLP techniques to remove the noise from the collected snippets. It also uses an evaluation function based on the user's browsing behavior to reorder all the collected snippets. Then, the ‘‘Topic Process’’ first uses the N-gram statistical language model to generate topics related to the query. The process

TABLE 4. A comparison of different types of recommendation systems.

Characteristic Type	Result	Practice	Advantage	Disadvantage
Keyword suggestion	Related keywords	Proximity search	1. Easy to implement 2. Hot trend keywords	1. Suggested keywords lack semantics 2. A huge storage space
	Researcher	Idea		
	Jiang et al. [72]	They considered the suggested keywords to cover different aspects at the same time.		
	Renjie et al. [73]	They proposed a video clustering method based on the referrer video graph to obtain relevant suggested keywords.		
	Zhou et al. [74]	They proposed a method based on generative neural network to generate diversified and domain-consistent suggested keywords.		
Question answering	Result	Practice	Advantage	Disadvantage
	A clear answer	NLP	1. A precise answer 2. High accuracy for some areas	1. Difficult to apply to open domain questions 2. Open domain questions are less accurate
	Researcher	Idea		
	Brill et al. [84]	They proposed a system that uses different N-gram techniques to check the dependency of data redundancy.		
	Abacha and Zweigenbaum [76]	They provided more expressive, standard formal language and make corpus and question annotations shareable		
	Bhandwaldar and Zadrozny [85]	They answered facts and lists questions based on the method of named entities and the summary of the constructed paragraph size based on the lexical chain.		
Topic searching	Result	Practice	Advantage	Disadvantage
	Related search topics	Document clustering	1. It reduces user judgment time 2. Generate topic quickly	1. It cannot cover all the topics of the page 2. Different methods are difficult to compare
	Researcher	Idea		
	Scaiella et al. [38]	They used a graphical concept to analyze the relationship between topics and documents.		
	Gamare and Patil [55]	They used HAC and link-based algorithms to cluster the documents		

then uses the concept of mathematical sets to organize related topics into a topic tree for user selection. Finally, the ''Personalization Process'' generates personalized search results based on topics selected by the user from the topic tree and different binary operations.

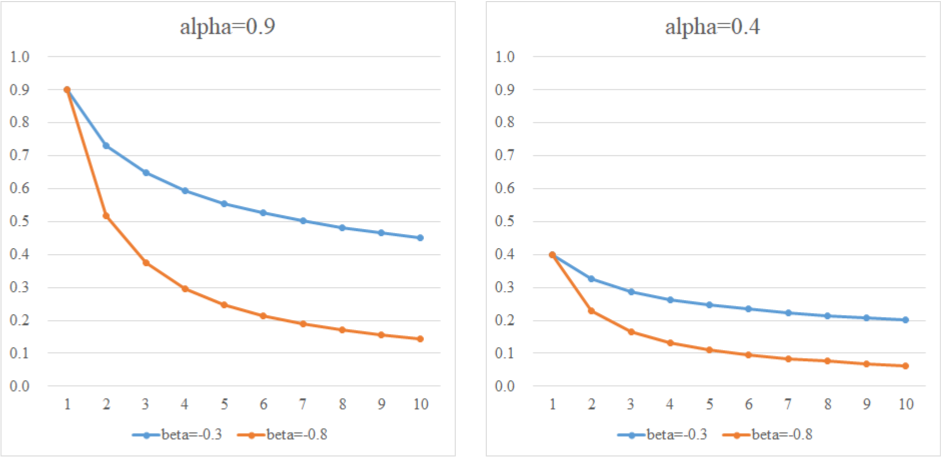
This study is based on the metasearch mechanism [90] for post-processing of general search engines. The mechanism mainly sends user queries to the general search engines and aggregates the results returned by the general search engines [91]. The advantages of this mechanism are: (a) increasing the scope of the search [92], (b) saving on enterprise build costs [93], and (c) balancing the views of multiple search engine results [94].

A. SERP PROCESS

The snippet is the search result generated by the search engine. It mainly includes the title, URL and fragment text related to the query processed by the search engine [17], [58], [95], [96]. Snippets generated by the search engine have the following three advantages [17], [95]: (a) the essence of the Web page, (b) saving the user time to browse the whole page, and (c) the processing time of the snippet is often shorter than the whole page. Since the returned search snippets are documents in an unstructured format [97], we need to convert these unstructured documents into structured documents for later processing. In this study, we use the following NLP techniques to perform this conversion: PCRE (Perl Compatible Regular Expressions), stemming, stop words, and non-words. PCRE [98] is a set of functions that help us to find the rank, title, URL, and brief description of each search result in the snippets according to our custom grammar rules. The goal of finding each search result is to allow our search ranking function to calculate the relevant weights. Since each word in a snippet can be represented in different forms, it is necessary to aggregate the same word with different forms into the root form. In this study, we use the Porter stemming algorithm [99] to perform this aggregation. Porter stemming algorithm is a well-known stemming algorithm, which can convert those words with different parts of speech (noun, verb, adjective, adverb) to its root form. Since the stop word does not make any sense and will slow the performance of information retrieval, we must remove all possible stop words. In this study, we use the 421 stop words proposed by Fox [100] as the basis and expand the relevant stop words on the Internet to remove all stop words in the snippet. For the same reason, all non-words (such as HTML tags, numbers, punctuation) must also be removed.

For all the returned and NLP processed snippets, we must re-rank all the snippets from different sources. Re-ranking is based on our search ranking function to complete, as shown in Equation 1.

$$w_s = \frac{\sum_e \alpha_e r_{s,e}^{\beta_e}}{\sum_e \alpha_e} (where\ \alpha_e > 0 \quad and\ \beta_e < 0) \tag{1}$$



(a) PE distribution when alpha = 0.9

(b) PE distribution when alpha = 0.4

FIGURE 3. Corresponding PE distribution at different α_e.

where w_s is the weight of the snippet s, e is a search engine from a variety of sources, α_e is the user's preference for the first search result returned by the engine e, $r_{s,e}$ is the rank of the snippet s in the engine e, and β_e is the user's preference for the engine e. The two parameters α_e and β_e in equation (1) are mainly based on the PE (Primary Effect) in psychology [101]–[103]. Next, we describe the basis for establishing these two parameters.

In general, search results are presented as a list and the list is sorted with decreasing relevance. That is, the user prefers the results of the previous search list and the preference continues to decrease. Figure 3 shows the trend distributions of user browsing by PE. PE mainly determines its distribution based on two parameters: α_e and β_e. The parameter α_e indicates the user's first impression in the object list e, and the parameter β_e indicates the user's preference decline for the object list e. The main difference between the two subfigures in Figure 3 is that subfigure (a) has a relatively large α_e value. This means that compared with the subfigure (b), the user in the subfigure (a) has a stronger first impression of the object list e; that is, a stronger click preference. In addition, the two curves in each subfigure represent the degree of decline of the user's preference for the object list e. When the degree of decline is greater, that is, the slope of the curve is larger (or the β_e value is smaller), it represents that the degree of user preference in this curve will decrease faster. PE has two main characteristics: (a) the user's browsing preferences continue to decline, and (b) the user's browsing behavior is random. In this study, we mainly use PE to simulate the browsing behavior of users. Our simulation method uses a mathematical model rather than actually storing the user's search history. According to the literature, human behavior is a random process [104], [105]. Moreover, human preference is determined between a lower limit and an upper limit [106]. Therefore, we use random numbers to simulate user behavior in the α_e and β_e parameters of user preferences in this study. The initial ranges of α_e and β_e parameters are obtained from our previous experimental results [93], [107].

Equation 1 is based on a new UBF (User Behavior Function) [93]. The idea of UBF includes two main points: (1) the user's preference for the object will be continuously decreasing from the front to the back, and (2) the user preferences for different groups of objects will vary. Our ranking function is obtained by running the vector inner product of the UBF results of different engines and normalizing the weights. After watching the ranking function, we find that it has two potential characteristics. One is that if the more engines vote for the snippet s, it will get the greater weight. The other is that if the snippet s gets a higher ranking in engine e, it will also get the bigger weight.

B. TOPIC PROCESS

The topic process first uses our topic generation method to generate the relevant discussion topics from the SERP process. Next, the process uses our hierarchical topic representation method to find the parent-child relationship between topics to produce hierarchical topic cluster search results, as shown on the left side of Figure 1.

1) TOPIC GENERATION METHOD

Our topic generation method is based on the N-gram statistical language model [108] and TF-ISF (Topic Frequency-Inverse Snippet Frequency) statistical approach. In this method, we first produce a sequence of consecutive N (from 1 to N) words for each word in the snippet, called the NS (N-gram Sequence). That is, for the snippet s = "XYZ" (where X, Y, and Z are words), we produce the following NS: unigram (X, Y, Z), bigram (XY, YZ), trigram (XYZ).

Next, we use the N-gram statistical language model to calculate the probability of occurrence for sequence τ =

TABLE 5. A comparison of the advantage and disadvantage of the two parameters.

	$N < 3$ small (or threshold > 0.45)	$N > 3$ (or threshold < 0.45)
Advantage	Less cost	Better performance
Disadvantage	Worse performance	More cost

$(\tau_1\tau_2\ldots\tau_n)$ in *NS*, as shown in Equation 2.

$$p(\tau) = p(\tau_1)\prod\nolimits_{x=2}^{n} \quad p(\tau_x|\tau_1, \ldots, \tau_{x-1}) \qquad (2)$$

where $p(\tau)$ is the probability of sequence τ; $p(\tau_1)$ is the probability of word τ_1, and $p(\tau_x|\tau_1, \ldots, \tau_{x-1})$ is the conditional probability that τ_x occurs when consecutive words τ_1 to $\tau_{\mathrm{x}-1}$ occur. We use this model to calculate the probability of consecutive words occurring via Bayes' theorem.

We then use the TF-ISF statistical approach to calculate the TF-ISF weights for each sequence τ, as shown in Equation 3.

$$tfisf(\tau) = \frac{n_{\tau,s}}{\sum_k n_{k,s}} \times \log(\frac{TN}{|TN_\tau|}) \qquad (3)$$

where $tfisf(\tau)$ is the TF-ISF weight for sequence τ, $n_{\tau,s}$ is the number of occurrences of sequence τ in snippet s, $\sum_k n_{k,s}$ is the total number of all sequences in snippet s, TN is the total number of search snippets returned by the search engine, and $|TN_\tau|$ is the number of occurrences of sequence τ in the number of snippets.

TF-ISF is based on the idea of TF-IDF (Term Frequency-Inverse Document Frequency) [109]. That is, the topic in TF-ISF is a term in TF-IDF, and the snippet in TF-ISF is a document in TF-IDF. The main idea of TF-ISF is that if sequence τ is high in snippet s and rarely occurs in other snippets, sequence τ will have a good discriminative ability.

Finally, we use the idea of expected value to calculate the expected weight of sequence τ, as shown in Equation 4; where $exp(\tau)$ is the expected weight for sequence τ. After watching this equation, we find that $exp(\tau)$ is larger in the following two cases. One is that when $p(\tau)$ is bigger, the chance of consecutive sequence τ is bigger. The other is that when $tfisf(\tau)$ is larger, this represents the sequence τ is an important sequence for the snippet s. Both cases indicate that sequence τ is an important sequence, so it must set a larger expected weight.

$$exp(\tau) = p(\tau) \times tfisf(\tau) \qquad (4)$$

A sequence τ is chosen to be a discussion topic when its expected weight is greater than the threshold. There are two parameters that affect the performance of the discussion topic: (1) the maximum N in N-gram, and (2) the threshold.

Table 5 shows the advantage and disadvantage of these two parameters for different sizes. When $N < 3$ small (or threshold > 0.45),[1] the total number of produced sequences will be less. This means the execution time (cost) needed to generate the discussion topic will be less. Conversely, when $N > 3$ (or threshold < 0.45), the total number of sequences produced will be greater. This means the more discussion topics are available, there is a greater chance of generating good topics (performance). In Section IV.A.3, we will discuss how to set the parameters N and threshold.

2) HIERARCHICAL TOPIC REPRESENTATION METHOD

Our hierarchical topic representation method is based on the concept of sorting, binary encoding, and mathematical set. In this method, we first sort all the topics that satisfy the threshold condition in descending order, based on the number of snippets contained in the topic.

Next, we use the concept of binary encoding to encode the topic, as shown in Equation 5; where $encode(t)$ is the binary encoding of the topic t, $b_{t,i}$ is a binary bit representing whether or not the topic t contains the snippet i, $join(b_{t,1}, \ldots, b_{t,i}, \ldots, b_{t,n})$ is a function that concatenates binary bits from $b_{t,1}$ to $b_{t,n}$, and n is the total number of snippets returned by the query.

$$encode(t) = join(b_{t,1}, \ldots, b_{t,i}, \ldots, b_{t,n}),$$
$$where\ b_{t,i} = \begin{cases} 1, & if\,(the\ topic\ t\ has\ a\ snippeti) \\ 0, & otherwise \end{cases} \qquad (5)$$

In this study, binary encoding is used to achieve the following two purposes: (1) to quickly set up the hierarchical tree, and (2) to achieve different binary operations. The first purpose is that we need to find the parent-child relationship between topics in the process of building a hierarchical tree. In this study, we use the concept of the mathematical set to find the relationship, as described below. When we use the binary encoding and mathematical set to build hierarchical tree, it can produce hierarchical topic cluster search results in the fastest time, as described in the follow-up experiments. The second purpose is that we can use various binary operations for different users to produce their personal search results. To perform possible binary operations on different topics, we also need to encode the topic in binary format.

Finally, we use the concept of mathematical set to find the parent-child relationship between topics, as shown in Equation 6; where t_α is a child node that can be in the topic t, t_β is one of the known child nodes in topic t, $\subseteq$ is a subset operation for two sets, and $\not\subset$ is a non-subset operation for two sets.

$$t \leftarrow t_\alpha$$
$$\Leftrightarrow (encode(t_\alpha) \subseteq encode(t)\ \&encode(t_\alpha) \not\subset encode(t_\beta)) \qquad (6)$$

[1] The results of 3 and 0.45 are based on the results of Section IV.A.3.

The meaning of this equation is that t_α can be a child topic of t if and only if t_α is a subset of t and t_α is not a subset of any child topics t_β in t. The equation has two conditions: one is that t_α is a subset of t, and the other is that t_α is not a subset of t_β. The first condition is intuitive, that is, the snippet contained in t_α must be part of t, it can become a descendant of t. The second condition is the key to building a multi-layer tree instead of a two-layer tree. We know that when t_α is a child topic of t_β and t_β is a child topic of t, we know the relationship of these three topics should be $t \leftarrow t_\beta \leftarrow t_\alpha$ according to the transitive law. If we do not add the second condition, the relationship between the three topics becomes $t \leftarrow t_\alpha$ and $t \leftarrow t_\beta$. That is, the hierarchical tree only sets up the two-layer relation.

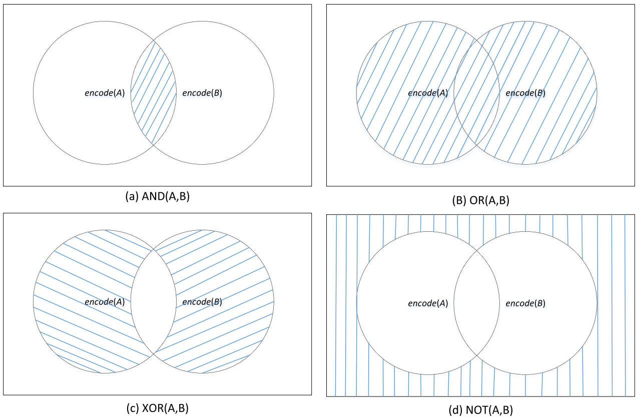
Next, we use a proof to verify that the second condition is a key condition for building a multi-layer tree.

Proof: Assume that the second condition is not true. That is, t_α is a subset of any child topics t_β in t. In this hypothesis, the topics t_α and t_β are descendants of the topic t and the topic t_α is the descendant topic of t_β. For the topics t, t_α and t_β, our method first looks for all subtopics contained in t by sorting. We know that topic t_β is a subtopic of topic t and topic t_α is a subtopic of topic t_β. Therefore, when topic t is looking for all subtopics, it will first join t_β and then join t_α. At this point, the condition establishes the architecture of two-layer tree instead of multi-layer tree, that is, the topic t contains the topics t_β and t_α.

The most important operation of the whole building hierarchical tree is the subset operation. Therefore, the key point about the speed of tree building is how to quickly perform the subset operation. A subset operation can easily run a binary AND operation. That is, t_α is a subset of t when $encode(t_\alpha)$ AND $encode(t)$ is equal to $encode(t_\alpha)$.

C. PERSONALIZATION PROCESS

The personalization process produces personalized search results based on user-selected discussion topics and personalization option, as shown on the right side of Figure 1. Personalized searches in this study were done using a non-stored user's browsing history. We add a checkbox to each discussion topic so the user can select the topic he or she wants. This is similar to TagMySearch; however, it can only perform a union (OR) operation on the selected topics. That is, when the user selects A and B topics, it displays all the search results contained in both topics. However, in reality, the user may need to run other operations, such as the user wants to know that those search results contain both A and B topics. If the user only uses a personalization option provided by TagMySearch for personal search, more time may be needed to analyze the search results. With our wide range of personalization options, we can help users to filter the personal search results they want. Theoretically, we can provide any possible binary operations, since our personalization runs on a binary-encoded basis. Currently, we offer the four most common binary operations for the user to choose: AND, OR, XOR, and NOT.



(a) AND(A,B)

(B) OR(A,B)

(c) XOR(A,B)

(d) NOT(A,B)

FIGURE 4. The different binary operations for topics A and B.

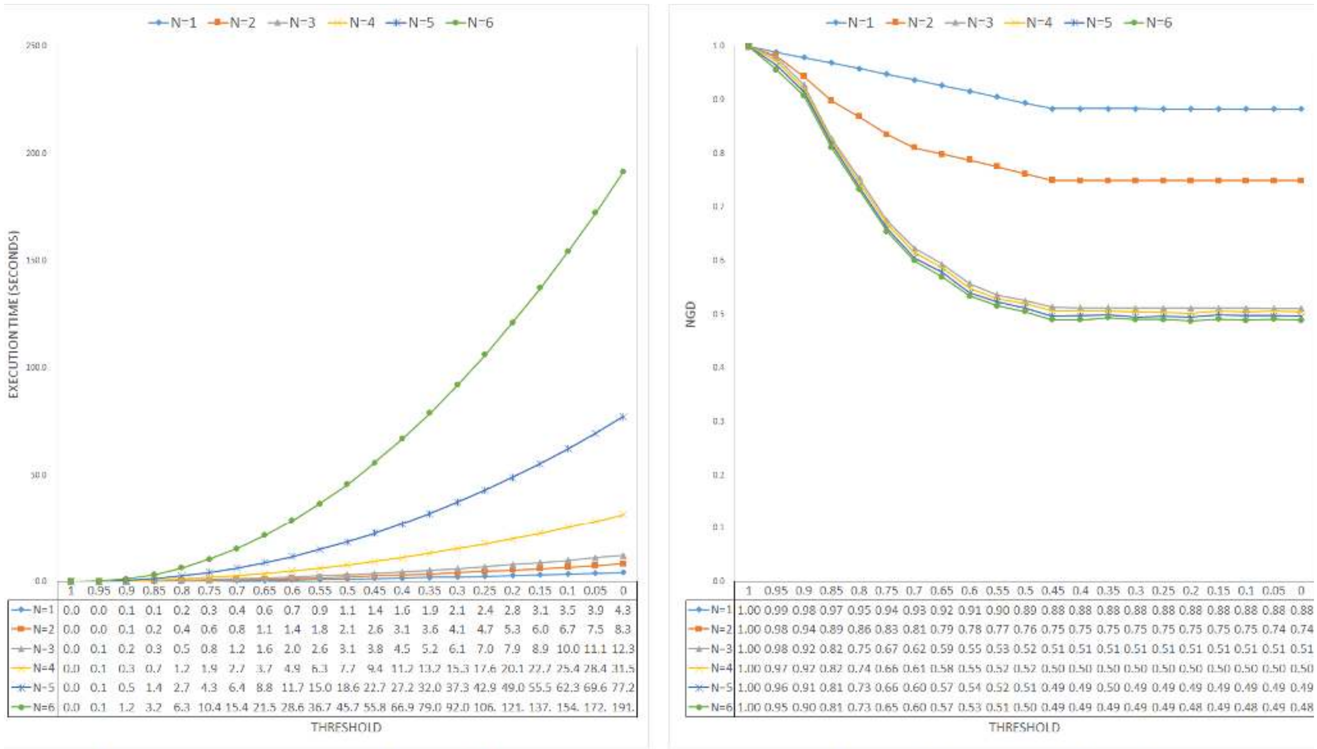
The following equation represents the mathematical model for these four operations, respectively; where $OP(A, B)$ is the bit string returned by the $OP \in \{AND, OR, XOR, NOT\}$ operation for topics A and B, $\cap$ is the intersection of the two sets, $\cup$ is the union of the two sets, C is the complement of a set.

$$\begin{aligned} AND\ (A, B) &= encode\ (A) \cap encode\ (B) \\ OR\ (A, B) &= encode(A) \cup encode(B) \\ XOR\ (A, B) &= \left(encode\ (A) \cap encode\ (B)^C\right) \\ &\quad \cup \left(encode\ (B) \cap encode\ (A)^C\right) \\ NOT\ (A, B) &= encode(A)^C \cap encode(B)^C \end{aligned} \tag{7}$$

- AND(A, B): For the user-selected topics A and B, the operation can produce those snippets that appear simultaneously in A and B. In practice, we allow the user to select k topics at the same time, where $k \geq 1$.
- OR(A, B): For the user-selected topics A and B, the operation can produce those snippets that appear in A or B.
- XOR(A, B): For the user-selected topics A and B, the operation can produce those snippets that appear only in A or B. That is, the operation is to remove the snippets shared by these topics.
- NOT(A, B): For the user-selected topics A and B, the operation can produce other snippets that are not contained in A and B.

To facilitate the reader to read, we use a figure to illustrate Equation 7. Figure 4 is a diagram of the different binary operations for topics A and B.

In our personalization, we only store the following information: the current user query, the search topics selected by the user, and the personalized option selected. Based on the above stored information, we actually use JavaScript's client-side scripting language to process user personalized results, that is, all personalized results are stored on the user's computer, not in our system. At the end of the personalization process, our system will delete the relevant stored information. In addition to avoiding the related problems that may be



	1	0.95	0.9	0.85	0.8	0.75	0.7	0.65	0.6	0.55	0.5	0.45	0.4	0.35	0.3	0.25	0.2	0.15	0.1	0.05	0
N=1	0.0	0.0	0.1	0.1	0.2	0.3	0.4	0.6	0.7	0.9	1.1	1.4	1.6	1.9	2.1	2.4	2.8	3.1	3.5	3.9	4.3
N=2	0.0	0.0	0.1	0.2	0.4	0.6	0.8	1.1	1.4	1.8	2.1	2.6	3.1	3.6	4.1	4.7	5.3	6.0	6.7	7.5	8.3
N=3	0.0	0.1	0.2	0.3	0.5	0.8	1.2	1.6	2.0	2.6	3.1	3.8	4.5	5.2	6.1	7.0	7.9	8.9	10.0	11.1	12.3
N=4	0.0	0.1	0.3	0.7	1.2	1.9	2.7	3.7	4.9	6.3	7.7	9.4	11.2	13.2	15.3	17.6	20.1	22.7	25.4	28.4	31.5
N=5	0.0	0.1	0.5	1.4	2.7	4.3	6.4	8.8	11.7	15.0	18.6	22.7	27.2	32.0	37.3	42.9	49.0	55.5	62.3	69.6	77.2
N=6	0.0	0.1	1.2	3.2	6.3	10.4	15.4	21.5	28.6	36.7	45.7	55.8	66.9	79.0	92.0	106.	121.	137.	154.	172.	191.

(a) Execution time for different N and threshold values

	1	0.95	0.9	0.85	0.8	0.75	0.7	0.65	0.6	0.55	0.5	0.45	0.4	0.35	0.3	0.25	0.2	0.15	0.1	0.05	0
N=1	1.00	0.99	0.98	0.97	0.95	0.94	0.93	0.92	0.91	0.90	0.89	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88
N=2	1.00	0.98	0.94	0.89	0.86	0.83	0.81	0.79	0.78	0.77	0.76	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.74	0.74
N=3	1.00	0.98	0.92	0.82	0.75	0.67	0.62	0.59	0.55	0.53	0.52	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51
N=4	1.00	0.97	0.92	0.82	0.74	0.66	0.61	0.58	0.55	0.52	0.52	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50
N=5	1.00	0.96	0.91	0.81	0.73	0.66	0.60	0.57	0.54	0.52	0.51	0.49	0.49	0.50	0.49	0.49	0.49	0.49	0.49	0.49	0.49
N=6	1.00	0.95	0.90	0.81	0.73	0.65	0.60	0.57	0.53	0.51	0.50	0.49	0.49	0.49	0.49	0.49	0.48	0.49	0.48	0.49	0.48

(b) NGD for different N and threshold values

FIGURE 5. Execution time and NGD for different N and threshold values.

caused by storing personal search records, our practice can also quickly complete the relevant personalized processing results.

IV. EXPERIMENTAL ANALYSIS AND DISCUSSION

In this section, we first discuss how to set the relevant parameters in the topic process. Next, we analyze the performance and cost of the clustering results generated from the topic process. We then use common metrics to measure the performance difference between personalized search and traditional search. Finally, we discuss the advantages of this study.

A. DISCUSSION ON RELATED PARAMETERS

Here we discuss the two parameters of the topic process: N and the threshold. Refer to Section III.B.1 for the description of the parameters. Based on the results in Table 5, we understand that N and the threshold must be set suitably to achieve a balanced solution between cost and performance. In this experiment, we first describe the test data set and training corpus used in this study. Next, we describe the performance metric used in this experiment. Finally, we compare the cost and performance of N and the threshold.

1) THE TEST DATA SET AND TRAINING CORPUS

The test data set used in this study is a set of 1000 test queries, which are real queries for people searching on Google's search engine from 2013-3-13 to 2016-11-8. Since the number of test data sets is large, we encourage interested readers to refer to it at http://hlcs.sytes.net/pwsc/1000.pdf.

In general, the data set selected will affect the performance of the system. The evaluation data set of 1000 queries selected in this study is the top 1000 real user query in Google trends during 1336 days. The purpose of the data set selected for this study is that we want to assess and understand the real search needs of most users over a long evaluation period.

This study uses the metasearch mechanism based on the general search engines. That is, the Google search engine is one of the source search engines for our metasearch mechanism. Because the test data set and performance metric used in this study are all based on the Google search engine, we selected it as our training corpus. The way to access this corpus is to count the number of hits returned by the query in the Google search engine.

2) THE PERFORMANCE METRIC

This experiment uses NGD (Normalized Google Distance) [110] as our performance measure, as shown in Equation 8; Q is a user query, t is any topic, where CS is the size of the Google search engine corpus, $f(\lambda)$ is the number of hits returned by the query $\lambda \in \{Q, t, Q\ and\ t\}$ in Google search engine.

$$NGD\left(Q,t\right)=\frac{\max\left\{\log f\ (Q)\ ,\log f\ (t)\right\}-\log f\ (Q\ and\ t)}{\log CS-\min\left\{\log f\ (Q)\ ,\log f\ (t)\right\}} \tag{8}$$

NGD uses the Google search engine to evaluate the semantic distance between Q and t. When NGD is smaller, it means that Q and t are closer in the semantic level. This is suitable for use in this study to evaluate the performance of the generated topic results. Interested readers can test our NGD simulation program at http://hlcs.sytes.net/ngd.

3) PERFORMANCE AND COST COMPARISON FOR N AND THRESHOLD

Figures 5-(a) and 5-(b) show the execution time and the NGD distribution at different N and threshold values, respectively. Each dot in the figure shows the average execution time and average NGD for 1000 test queries. First, for each query, we calculate the average execution time and average NGD for the query Q and its top 10 topics ts. Next, for 1000 queries, we average the average execution time and average NGD of all queries to obtain the final average execution time and average NGD (i.e., each dot in Figures 5-(a) and 5-(b)). Looking at Figure 5-(a), we found that when N is less than 3, the execution time is much less (the average execution time is from 1.5 to 4.2 seconds). This time is an acceptable wait time for the user to perform the query. Conversely, when N is greater than 3, the execution time is much more (the average execution time is from 10.7 to 64.1 seconds). Users cannot wait for such a long time to produce results.

Looking at Figure 5-(b), we found that when N is less than 3, NGD performance is significantly lower (the average NGD is from 0.92 to 0.79). Conversely, when N is greater than or equal to 3, NGD is significantly better (the average NGD is from 0.59 to 0.57). The figure also shows that when N is greater than 3, there is no significant increase in NGD performance. That is, when we use N greater than 3 to run the cluster, the total number of sequences produced will be extremely large. This will not only significantly increase the cost (execution time), but also improve the overall performance (NGD) is not significant. Therefore, we set N to 3 in the study.

Looking again at Figure 5-(b), no matter which N, we found that when the threshold is less than 0.45, the performance of NGD hardly increases. Therefore, we set the threshold to 0.45 in this study. For an N equal to 3 and a threshold equal to 0.45, we can respond to the user's query demand at an average of 3.8 seconds and produce an average NGD of 0.46, as shown in Figures 5-(a) and 5-(b). This means that we can generate good clustering topics within a timeframe that is acceptable for users.

B. COMPARISON OF PERFORMANCE AND COST OF CLUSTERING RESULTS

Table 6 is a comparison of different systems for snippet-oriented clustering. The concept of different systems is described in Section II.C. The different methods to generate a topic are as follows: Carrot2 and WSC are based on the STC algorithm, TagMySearch is the topic graph, this study is based on the N-gram language model and the different operations on the set. Based on the presentation of the clustering results, Carrot2 and TagMySearch are presented in a flat structure, while the WSC and this study are presented in a hierarchical structure. For personal search comparison, Carrot2 and WSC do not provide any binary operation, and TagMySearch can only run the OR operation on different topics. This study can run different binary operations for different topics to meet the different search needs of users.

Based on the results in Table 6, we found that TagMySearch has the lowest performance (highest NGD). The main reason is that it uses the concept of frequent itemsets to generate topics, the concept is to select several of the most frequent term patterns as candidate topics. However, it is easy to choose topics that do not have differential effects or ignore potentially important topics. The reason this study is superior to STC-based systems is that we use the concept of mathematical set to combine different topics with the same meaning but different permutations into a single topic. That is, when the sequences $\tau_1 = (X, Y)$ and $\tau_2 = (Y, X)$ meet the conditions in Section III.B.1, the two sequences will be merged into a sequence τ after passing through the operations of the mathematical set. At this point, the snippet of the merged sequence τ will contain all the snippets of τ_1 and τ_2. In addition, this study is better than other systems at execution time, because we use a multithreaded way to collect the snippets returned by the search engine. Theoretically, through our multithreaded way, the collection time of multiple snippets is equal to one snippet.

Next, we discuss the time we need to build the parent-child hierarchy tree. In this experiment, we use a computer with Intel Core 2 Duo T9600 and 2GB memory to build hierarchical tree. Table 7 is a comparison of the execution time required for different numbers of snippets when using our hierarchical tree generation method. According to the results in the table, we found that when the number of snippets is 10000 and 20000, the total time required to build the hierarchical trees is 2.73 and 5.24 seconds, respectively. Similarly, we also found that when the number of snippets is 100000, the total time required is 26.73 seconds. That is, the average time for each snippet to build the hierarchical tree is about 270 microseconds. The other snippet numbers in the table also show this trend. This means that there is a linear relationship between the number of snippets and the execution time. According to some literatures [111], [112], it is shown that the best time for hierarchical clustering is linear time. Therefore, based on the results in Table 7, the time we need to build the hierarchical tree is the best linear time.

According to the experimental results, it requires more computation time to perform binary encoding when the system selects more snippets. To enable the system to respond to the user's query needs in real time, we generate a suitable topic tree by selecting the results of the search engine snippets that most users may prefer. According to relevant statistical results [6], 86% of users browse the search engine results in no more than 30 snippets. However, if the system only selects search results with 30 or fewer snippets, it may generate fewer topics because fewer snippets are selected. In practice,

TABLE 6. Comparison of snippet-oriented systems.

System Characteristic	Carrot2	WSC	TagMySearch	This study
Method used	STC-based	STC-based	Topic graph	N-gram and set
Flat or Hierarchical	Flat	Hierarchical	Flat	Hierarchical
Provide personalization	No	No	Only OR	Any binary operation
Average NGD	0.63	0.58	0.73	0.46
Average execution time (second)	5.746	5.124	7.432	3.873

TABLE 7. The execution time for different numbers of snippets.

#Snippets	10000*	20000	30000	40000	50000	60000	70000	80000	90000	100000
Time	2.73	5.24	8.10	10.92	13.74	15.72	19.33	22.21	23.63	26.73

* For each query, we use the parameter num in Google search to set the maximum number of returned snippets to 100, that is, every 10000 snippets are the search results returned by 100 queries.

TABLE 8. Performance evaluation results of different systems.

Measure	TN_s	Google	Bing	This study
P_s	3	0.88	0.86	0.94
	5	0.73	0.71	0.91
	10	0.70	0.68	0.91
R_s	3	0.35	0.34	0.38
	5	0.31	0.30	0.37
	10	0.24	0.24	0.36
F_s	3	0.50	0.49	0.54
	5	0.44	0.42	0.53
	10	0.36	0.35	0.52

in order to achieve a balance between the computation time and the number of topics generated, the number of snippets processed by our system is between about 100 and 150 for each user query. The number of snippets returned by each query is not fixed because the metasearch mechanism is used in the Query Process to collect the number of snippets from different search engines (100 for Google, 50 for Yahoo, and 50 for Bing) at the same time. Because some URLs in search results returned by different engines may be repeated, the number of snippets we actually process will not be fixed, it is between about 100 and 150 snippets. That is, the average time taken by the system to build a hierarchical tree is between about 0.0270 and 0.0405 seconds for each query.

C. PERFORMANCE COMPARISON BETWEEN PERSONAL SEARCH AND TRADITIONAL SEARCH

In this experiment, we want to compare the performance difference between the topic search results generated by this study and the snippets generated by the traditional search engines. We choose Google and Bing search engines as our rating targets because they are the top two search engines in the world. In this study, we use Precision (P_s), Recall (R_s), F-measure (F_s) [17], [113], [114] to evaluate the performance of different systems, as shown below.

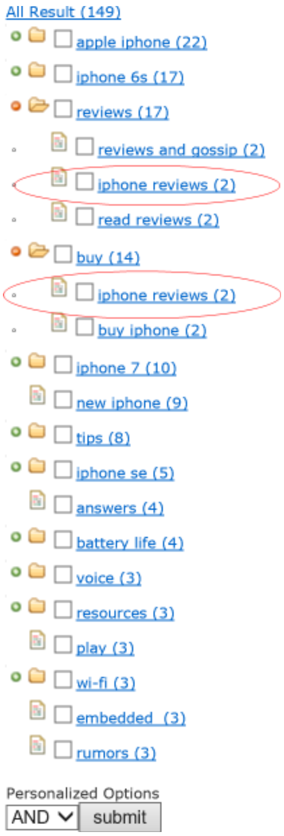
$$P_s = \frac{|M_s|}{TN_s} \tag{9}$$

$$R_s = \frac{|M_s|}{|\bigcup_s M_s|} \tag{10}$$

$$F_s = 2 \times \frac{P_s \times R_s}{P_s + R_s} \tag{11}$$

where $s \in \{Google, Bing, Oursystem\}$ is a comparison system, $|M_s|$ is the number of search results that have been manually marked as relevant among the top ten search results produced by system s, TN_s is the total number of SERPs returned by the query for system s, $|\cup_s M_s|$ is the number of search results such that the union of all M_ss contains all search results for all s. In this experiment, we also use the same test data set to evaluate different systems.

Table 8 shows the performance evaluation results for different systems. Based on the results in the table, we found that whether it is evaluated by Precision, Recall or F-measure, the results of the personalized search generated by this study are better than the snippet results produced by traditional search engines. The search results generated by the traditional search engine are based on the queries entered by the user. In general, the number of results generated by this method is often quite impressive. The main reason why this study is better than the traditional search engine is that the personalized search results generated by this study are mainly based on the search results generated by the topics of interest to the users and not just based on the queries entered by the users. That is, we filter out the results of most of the topics that are not of interest to the user for subsequent processing. Therefore, the results generated by our method can not only be closer to the user's needs, but also the number of results is controlled within a certain range without causing user information overload. The scores for Google and Bing in the table will decrease as the number of TN_s increases and there is a clear downward



All Result (149)
- □ apple iphone (22)
- □ iphone 6s (17)
- □ reviews (17)
  - □ reviews and gossip (2)
  - □ iphone reviews (2)
  - □ read reviews (2)
- □ buy (14)
  - □ iphone reviews (2)
  - □ buy iphone (2)
- □ iphone 7 (10)
- □ new iphone (9)
- □ tips (8)
- □ iphone se (5)
- □ answers (4)
- □ battery life (4)
- □ voice (3)
- □ resources (3)
- □ play (3)
- □ wi-fi (3)
- □ embedded (3)
- □ rumors (3)

Personalized Options

FIGURE 6. **The hierarchical topic search results when the query is "iphone".**

trend. This means that the user is more interested in the previous search results. However, the scores of this study do not have a significant downward trend with the number of TN_s increasing. The reason is that the source of this study is the relevant pages matching to the topic selected by the user. When the user selects a topic, it can filter out pages that are not of interest to the user. This study only deals with those pages that are of interest to the user, so its performance does not decrease significantly as the number of TN_s increases.

D. THE ADVANTAGES OF THIS STUDY

For the discussion of the advantages of this study, we first explain the contribution of this study, then we explain the difference between this study and the existing research. Finally, we discuss the difference between the binary operations of this study and the binary operations of the traditional search engine.

1) DISCUSSION ON THE CONTRIBUTION OF THIS STUDY

This study has three main contributions: building a multi-cluster relationship, quickly building a hierarchical tree and quickly running different binary operations, as described below.

- Building a multi-cluster relationship: Figure 6 is the hierarchical topic search results we produced when the query is "iphone". Looking at the topic "iphone reviews (2)", we found that it is also a child topic of the topics "reviews (17)" and "buy (14)". This means that no matter what the user wants to know about the "reviews" or "buy" on the iPhone, he or she can learn about "iphone review" to decide.
- Quickly build a hierarchical tree: According to the analysis in Section IV.B, we learned there is a linear relationship between the number of snippets and the execution time. Again, according to the discussion in Section IV.B, we found that time is the best.
- Quickly run different binary operations: We provide various binary operations based on the concept of binary encoding. Through the binary operations between different topics, we can easily and quickly produce individual user search results. Theoretically, using our binary encoding concept can quickly run all possible binary operations.

2) DISCUSSION ON THE DIFFERENCE BETWEEN THIS STUDY AND EXISTING RESEARCH IN PERSONALIZED SEARCH

Table 1 shows the difference between this study and the existing research in personalized search. Since both Google History and My Yahoo use user log files to handle personalized searches, they all have personal privacy and large storage issues. Conversely, since TagMySearch, WSC, Carrot2, and this study all achieve personalized search without storing user log files, they can avoid personal privacy and large storage space issues. For the presentation of personalized topics, TagMySearch and Carrot2 only support flat topic presentation without support for hierarchical topic presentation. Conversely, WSC and this study support hierarchical topic presentation. Relevant literature shows that hierarchical topic presentation is particularly useful for poor or ambiguous queries. According to the experimental results, we found that TagMySearch has the worst computing time and the worst performance in the non-storage method. Conversely, the computing time of this study only needs linear time and the best performance. For personalization options, the options do not support other systems except for the personalized search options of TagMySearch and this study. When comparing the personalization options provided by these two systems, TagMySearch only supports the OR option and this study supports any binary operation options.

3) DISCUSSION ON THE DIFFERENCE BETWEEN THIS STUDY AND THE TRADITIONAL SEARCH ENGINE IN BINARY OPERATIONS

The differences between this study and the general search engine, such as Google or Bing, for binary operations are as follows:

- Processing data set: The general search engine focuses on the relevant pages of user queries or their binary operations. This approach requires the user to define query keywords for the relevant binary operations. Such work is difficult for the user because the average length of the query entered by the user is short [6]. This study

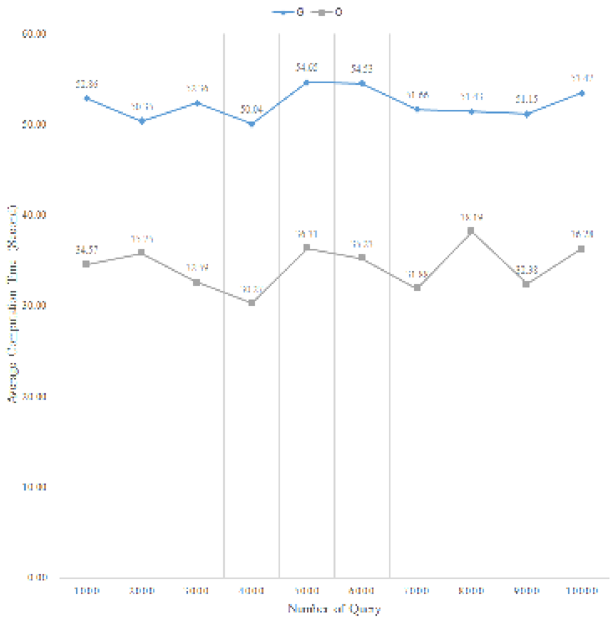


FIGURE 7. Comparison of the computation time between the general search engine (G) and this study (O).

focuses on the binary operations of related topics in the query. The advantage of our approach is that our system automatically generates topics that are highly relevant to queries. This can greatly reduce the time it takes for users to think about related queries or topics.

- Processing speed: The main time taken by the general search engine is that it needs to re-index the search engine database to generate suitable search results when the query contains different binary operations. The re-indexing time is closely related to the computing power of the machine. The main time spent in this study is that it needs to rebuild the hierarchical topic tree when using binary operations. Based on the experimental results in Section IV.B, we found that the time required to build the hierarchical tree in this study is between about 0.0270 and 0.0405 seconds for each query. Next, we set up an experiment to analyze the time difference between re-indexing the database for general search engine and rebuilding the hierarchical topic tree for this study. In this experiment, two different search mechanisms were run on the same machine (Intel Core 2 Duo T9600 and 2GB memory). The general search engine uses the most famous PageRank algorithm [115] in the search engine field as the main re-indexing algorithm. Figure 7 is a comparison of the computation time for the general search engine (the line of G in the figure) and this study (the line of O in the figure). Each dot in the figure represents the total computation time required for 1000 queries. For the sake of comparison, we averaged 10000 queries for the G and O methods. The average computation time for each query under the G and O methods is 0.0523 and 0.0343 seconds, respectively.

  According to the results of this experiment, the average computation time of our study is better than the general search engine. This experiment shows that the hierarchical topic tree method proposed in this study can respond to the user's query requirements faster than the general search engine.
- Saving space: For the collected Web pages, the general search engine needs to store the cached pages and related index files. The size of the index file is related to the following two parameters: the number of cached pages and the number of index fields. However, this study does not need to store any index files because there is no indexing action.

In view of the differences between points 2 and 3 above, the general company does not have sufficient financial resources to build a corresponding search system. However, through the design steps of this study, we only need a general-purpose personal computer to establish a suitable search system.

4) DISCUSSION ON THE DIFFERENCE BETWEEN THIS STUDY AND THE COST-EFFECTIVE GA METHOD

There are three main differences between this study and the cost-effective GA method [58]: solve object, implementation detail and computation time, as shown in Table 9.

- Solve object: The purpose of this study is to design a personalized search system to generate search results that meet the search needs of individual users. These personalized search results are generated by performing the binary operation according to the search engine snippets corresponding to the topics selected by the user. Therefore, in essence, personalized search results are mainly re-arranged search engine snippets without changing any snippets. The purpose of the cost-effective GA method is to design a page clipping search result that combines related paragraphs in multiple pages. These clipping results are generated by running GA on all the whole pages corresponding to the topics selected by the user. Therefore, in essence, the clipping results are mainly produced by clipping and synthesizing some paragraphs of the whole pages without including any snippets.
- Implementation detail: There are four differences in implementation details between this study and the cost-effective GA method. The first difference is that the two methods use different topic generation methods. Because the number of topics generated by the cost-effective GA method is much lower than this study, the method runs fewer merge operations on topics with similar names to keep the number of topics presented by the method not too small. In fact, the cost-effective GA method uses a hash method to merge only topics with the same topic name but different permutations. This study uses the operations of mathematical set to generate topics. In addition to merging topics with the

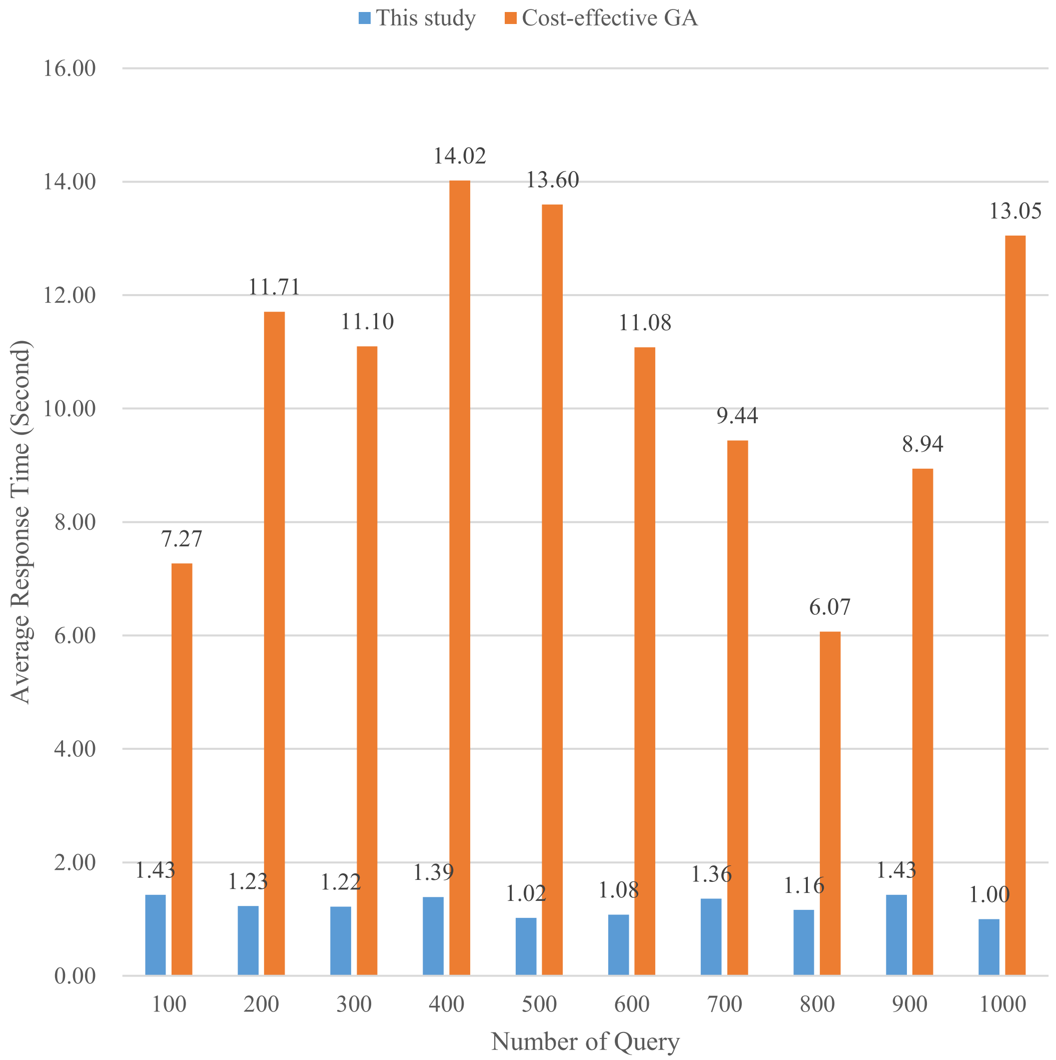


FIGURE 8. Comparison of response time between this study and cost-effective GA method.

TABLE 9. A comparison between this study and the cost-effective GA method.

Point of difference		This study	Cost-effective GA
Solve object		Personalized search	Page clippings
Implementation detail	Topic generation	Mathematical set	Hash
	Collected data	More snippets and no whole page	Fewer snippets and some whole pages
	Ranking function	UBF	MRR
	Noise processing	More	Less
Computation time		Faster	Slower

same topic name but different permutations, this study also merges two topics with nearly similar snippets into a longer topic name. For example: if the topics *XY* and *XYZ* contain nearly similar snippets, we merge these two topics and present only the longer topic name *XYZ*. That is, we use a longer name to describe the topic because a longer name has a richer description than a shorter name. The second difference is that the data collected and processed by the two methods are different. In this study, the main data we collect and process are the snippets returned by the search engines. The advantages of using snippets instead of whole pages as post-processing objects are that we can greatly save subsequent processing time and space because the returned snippets are the relevant text of the page summarized by the search engine. The cost-effective GA method first generates topics with a small number of snippets. Next, it runs a GA on the whole page content of related pages to synthesize different paragraphs on different pages to achieve the results of clipping synthesis. Because GA requires a lot of computation time, it also needs a lot of storage space because the computation results of each generation must be saved for the next generation [116]. Therefore, this method requires longer computation time and storage space. The third difference is that the two methods use different ranking functions to arrange search results. The cost-effective GA method uses MRR (Mean Reciprocal Rank) to rearrange snippets returned by different search engines. MRR only considers the order of the ranking of snippets in different search engines [94]. The advantage of MRR is that its calculation method is simple, so it is widely used in the index of evaluation in the field of information retrieval. However, its disadvantage is that it does not consider the differences in user preferences for different search engines [93]. In this study, we use UBF as a ranking function to rearrange the snippets returned by different search engines. Because each user's preferences for different search engines are not the same in personalized search, we use UBF as a ranking function to simulate the differences in viewing and clicking behavior of different users. The reason why UBF is a suitable ranking function in this study is that the system can adjust the parameters in the function according to the behavior preferences of different users to meet the user's search needs. The fourth difference is that the noise processing ranges used by the two methods are different. Noise (such as stop words, punctuation, HTML tags) is meaningless characters and often appears in documents. Therefore, the depth of noise processing often affects the performance of information retrieval. Because stop words are the biggest noise in the document, using a large stop word lexicon can help the system to filter out unnecessary noise. The cost-effective GA method mainly uses 421 stop words suggested by Fox [100] as a stop word lexicon. However, in reality, meaningless stop words continue to increase as new Internet slang words continue to appear. In this study, we use the 421 stop words as a basis and expand the common stop words on the Internet as our stop word lexicon. In fact, the main sources of our expanded stop words include Google [117] and MySQL [118], and we have removed the recurring stop words from these sources to get the final stop word lexicon. At present, our stop word lexicon has 1085 stop words.

- Computation time: Next, we run an experiment to compare the response times of the two methods. In this experiment, we use 1000 queries in the test data set as evaluation data and run two methods separately to obtain the true response time. Both methods run on the same machine (Intel Core 2 Duo T9600 and 2GB memory). We use Apache ab [119], a tool to evaluate the performance of a Web server, to run a script that contains all test queries to generate response times for different methods. Figure 8 is a comparison of the two methods for response time. According to the results in the figure, the average response time of this study and cost-effective GA method for every 100 queries is 1.23 and 10.63 seconds, respectively. The cost-effective GA method needs to run a certain number of generational GAs to achieve a cost-effective solution for clipping. Each generation of GA needs to perform selection, crossover, and mutation operations on all selected page content. The execution time of each generation of GA is related to the number of selected pages and the size of each page. Therefore, the response time of the cost-effective GA method is related to the following three factors: the number of generations actually run, the number of pages selected, and the size of each page. Although the number of snippets selected in this study is larger than the number of pages selected by the cost-effective GA method, the number of executions and the size of each snippet in this study are significantly smaller than the number of generations and the page size in the cost-effective GA method. Based on the above reasons of the number of executions and the size of each snippet, the response time of this study is significantly better than the cost-effective GA method. In addition, the standard deviations of the response time of this study and the cost-effective GA method for every 100 queries are 0.166 and 2.669 seconds, respectively. Compared with this study, the cost-effective GA method is very unstable in response time for each query. The main reason is that GA is a random search process and the number of generations it performs is not fixed [120]. This means that users are less likely to predict when search results will be generated.

V. CONCLUSIONS AND FUTURE RESEARCH DIRECTIONS

In this paper, we have proposed a personalized topic search system. The system has the following advantages. First, it presents query related topics in the form of a hierarchical tree. Presenting query related topics in this form is particularly useful for users who often enter shorter ambiguous queries. Second, because it does not store any user browsing and clicking history, it can avoid personal privacy and large storage space issues that often occur in personalized searches. This advantage is that it not only avoids possible privacy leaks that may occur with online companies, but also reduces the company's investment costs. Third, it can establish multi-cluster relationships, that is, the child topics it generates can appear in multiple parent topics. Multi-cluster relationships can help users understand the meaning of different topics at the same time because a topic may have multiple meanings. Fourth, it can quickly build hierarchical trees and generate personalized search results. This ensures that our system can respond to user search needs as quickly as other online search engines.

Future research directions include the following two tasks. First of all, the related NLP processing we currently use is mainly for English documents, but the NLP processing for other language documents is different. Therefore, for other language documents, we will look for NLP processing suitable for the language to analyze the document content. Secondly, although we used a large number of data sets for related experiments in the experimental section, our test data set could not contain popular keywords generated at any time due to the booming Internet. Therefore, for the test data set, we will continue to collect popular keywords and perform related experiments, and adjust the relevant parameters of the system based on the experimental results.

REFERENCES

[1] IDC. (2014). *The Digital Universe of Opportunities: Rich Data and the Increasing Value of the Internet of Things*. Accessed: Jan. 10, 2019. [Online]. Available: https://goo.gl/GbmFKN

[2] Y. Li, C. Luo, and S. M. Chung, ''A parallel text document clustering algorithm based on neighbors,'' *Cluster Comput.*, vol. 18, no. 2, pp. 933–948, Jun. 2015.

[3] D. E. Rose and C. Stevens, ''V-Twin: A lightweight engine for interactive use,'' in *Proc. 5th Text Retr. Conf. (TREC)*, Gaithersburg, MD, USA, 1996, pp. 1–12.

[4] F. Wiesman, H. J. van den Herik, and A. Hasman, ''Information retrieval by metabrowsing,'' *J. Amer. Soc. Inf. Sci. Technol.*, vol. 55, no. 7, pp. 565–578, May 2004.

[5] A. K. Sharma, N. Aggarwal, N. Duhan, and R. Gupta, ''Web search result optimization by mining the search engine query logs,'' in *Proc. Int. Conf. Methods Models Comput. Sci. (ICM2CS)*, Dec. 2010, pp. 39–45.

[6] B. J. Jansen, A. Spink, and T. Saracevic, ''Real life, real users, and real needs: A study and analysis of user queries on the Web,'' *Inf. Process. Manage.*, vol. 36, no. 2, pp. 207–227, Mar. 2000.

[7] C. Carpineto, S. Osinski, G. Romano, and D. Weiss, ''A survey of Web clustering engines,'' *ACM Comput. Surveys*, vol. 41, no. 3, pp. 1–38, Jul. 2009.

[8] W. K. T. M. Gunarathne, C. Chootong, W. Sommool, A. Ochirbat, Y.-C. Chen, S. Reisman, and T. K. Shih, ''Web-based learning object search engine solution together with data visualization: The case of MERLOT II,'' in *Proc. IEEE 42nd Annu. Comput. Softw. Appl. Conf. (COMPSAC)*, Jul. 2018, pp. 1026–1031.

[9] J. Wang, J. Z. Huang, J. Guo, and Y. Lan, ''Query ranking model for search engine query recommendation,'' *Int. J. Mach. Learn. Cybern.*, vol. 8, no. 3, pp. 1019–1038, Jun. 2017.

[10] M. R. Henzinger, ''Link analysis in Web information retrieval,'' *IEEE Data Eng. Bull.*, vol. 23, no. 3, pp. 3–8, Sep. 2000.

[11] C. Palihawadana and G. Poravi, ''A comparative study of link analysis algorithms,'' in *Proc. 8th Int. Conf. Intell. Syst., Model. Simul. (ISMS)*, May 2018, pp. 100–104.

[12] Y. Li, ''Toward a qualitative search engine,'' *IEEE Internet Comput.*, vol. 2, no. 4, pp. 24–29, 4th Quart., 1998.

[13] K. Sugihara and J. Nanzan, ''Using complex numbers in website ranking calculations: A non-ad hoc alternative to Google's PageRank,'' *J. Softw.*, vol. 14, pp. 58–64, Feb. 2019.

[14] M. Speretta and S. Gauch, ''Personalized search based on user search histories,'' in *Proc. IEEE/WIC/ACM Int. Conf. Web Intell. (WI)*, Compiègne, France, 2005, pp. 622–628.

[15] S. Dey and S. Abraham, ''User interface for a search engine: A customized and multi-domain approach,'' *Int. J. Comput. Inf. Syst. Ind. Manage. Appl.*, vol. 4, pp. 169–179, Dec. 2012.

[16] H. Zamani, M. Bendersky, X. Wang, and M. Zhang, ''Situational context for ranking in personal search,'' in *Proc. 26th Int. Conf. World Wide Web*, Apr. 2017, pp. 1531–1540.

[17] P. Ferragina and A. Gulli, ''A personalized search engine based on Web-snippet hierarchical clustering,'' *Softw., Pract. Exper.*, vol. 38, no. 2, pp. 189–225, Feb. 2008.

[18] R. Brenneke, T. Mandl, and C. Womser-hacker, ''The development and application of an evaluation methodology for person search engines,'' in *Proc. 1st Eur. Workshop Hum.-Comput. Interact. Inf. Retr.*, Newcastle, U.K., 2011, pp. 42–45.

[19] C. N. Pushpa, N. K. V. Kumar, T. Shivaprakash, J. Thriveni, S. H. Manjula, K. R. Venugopal, and L. M. Patnaik, ''Improving the precision and recall of Web people search using hash table clustering,'' in *Proc. 5th Int. Conf. Inf. Process.*, Bengaluru, India, 2011, pp. 155–160.

[20] D. Verma, K. Minocha, and B. Kochar, ''A multi-agent based personalized search engine with topical crawling capabilities,'' *IUP J. Comput. Sci.*, vol. 8, pp. 20–33, 2014.

[21] N. Saxena, S. Agarwal, and V. Katiyar, ''Personalized Web search using user identity,'' *Int. J. Comput. Appl.*, vol. 147, no. 12, pp. 14–17, 2016.

[22] F. Akhlaghian, B. Arzanian, and P. Moradi, ''A personalized search engine using ontology-based fuzzy concept networks,'' in *Proc. Int. Conf. Data Storage Data Eng.*, Feb. 2010, pp. 137–141.

[23] Z. Jiang and X. Deng, ''A personalized search engine model based on RSS user's interest,'' in *Proc. 2nd Int. Conf. Future Comput. Commun.*, 2010, pp. V2-196–V2-199.

[24] K. W.-T. Leung, D. L. Lee, W. Ng, and H. Y. Fung, ''A framework for personalizing Web search with concept-based user profiles,'' *ACM Trans. Internet Technol.*, vol. 11, no. 4, pp. 1–29, Mar. 2012.

[25] V. Mishra, P. Arya, and M. Dixit, ''Improving mobile search through location based context and personalization,'' in *Proc. Int. Conf. Commun. Syst. Netw. Technol.*, May 2012, pp. 392–396.

[26] A. Singh and B. Alhadidi, ''Knowledge oriented personalized search engine: A step towards wisdom Web,'' *Int. J. Comput. Appl.*, vol. 76, no. 8, pp. 1–9, 2013.

[27] N. Ramesh and J. Andrews, ''Personalized search engine using social networking activity,'' *Indian J. Sci. Technol.*, vol. 8, no. 4, pp. 301–306, 2015.

[28] S. A. Tabrizi, A. Shakery, H. Zamani, and M. A. Tavallaei, ''PERSON: Personalized information retrieval evaluation based on citation networks,'' *Inf. Process. Manage.*, vol. 54, no. 4, pp. 630–656, Jul. 2018.

[29] X. Wang, N. Golbandi, M. Bendersky, D. Metzler, and M. Najork, ''Position bias estimation for unbiased learning to rank in personal search,'' in *Proc. 11th ACM Int. Conf. Web Search Data Mining (WSDM)*, Marina Del Rey, CA, USA, 2018, pp. 610–618.

[30] Y. Nam, D. Shin, and D. Shin, ''Personal search system based on android using lifelog and machine learning,'' *Pers. Ubiquitous Comput.*, vol. 22, no. 1, pp. 201–218, Feb. 2018.

[31] Google. (2017). *Google Search History*. Accessed: Jan. 10, 2019. [Online]. Available: https://google.com/history

[32] Yahoo. (2017). *My Yahoo*. Accessed: Jan. 10, 2019. [Online]. Available: http://my.yahoo.com

[33] R. Baraglia, P. Dazzi, M. Mordacchini, and L. Ricci, ''A peer-to-peer recommender system for self-emerging user communities based on gossip overlays,'' *J. Comput. Syst. Sci.*, vol. 79, no. 2, pp. 291–308, Mar. 2013.

[34] R. Divya and C. R. R. Robin, ''Onto-search: An ontology based personalized mobile search engine,'' in *Proc. Int. Conf. Green Comput. Commun. Electr. Eng. (ICGCCEE)*, Mar. 2014, pp. 1–4.

[35] C. Cobos, H. Muñoz-Collazos, R. Urbano-Muñoz, M. Mendoza, E. León, and E. Herrera-Viedma, ''Clustering of Web search results based on the cuckoo search algorithm and balanced Bayesian information criterion,'' *Inf. Sci.*, vol. 281, pp. 248–264, Oct. 2014.

[36] B. Alattar and N. M. Norwawi, ''A personalized search engine based on correlation clustering method,'' *J. Theor. Appl. Inf. Technol.*, vol. 93, pp. 345–352, Nov. 2016.

[37] S. Chawla, ''A novel approach of cluster based optimal ranking of clicked URLs using genetic algorithm for effective personalized Web search,'' *Appl. Soft Comput.*, vol. 46, pp. 90–103, Sep. 2016.

[38] U. Scaiella, P. Ferragina, A. Marino, and M. Ciaramita, ''Topical clustering of search results,'' in *Proc. 5th ACM Int. Conf. Web Search Data Mining (WSDM)*, 2012, pp. 223–232.

[39] L. Chen, ''Building a Web-snippet clustering system based on a mixed clustering method,'' *Online Inf. Rev.*, vol. 35, no. 4, pp. 611–635, Aug. 2011.

[40] F. Laylavi, A. Rajabifard, and M. Kalantari, ''Event relatedness assessment of Twitter messages for emergency response,'' *Inf. Process. Manage.*, vol. 53, no. 1, pp. 266–280, Jan. 2017.

[41] N. Aletras, T. Baldwin, J. H. Lau, and M. Stevenson, ''Evaluating topic representations for exploring document collections,'' *J. Assoc. Inf. Sci. Technol.*, vol. 68, no. 1, pp. 154–167, Jan. 2017.

[42] L.-C. Chen, ''An effective LDA-based time topic model to improve blog search performance,'' *Inf. Process. Manage.*, vol. 53, no. 6, pp. 1299–1319, Nov. 2017.

[43] U. Ahmad, A. Zahid, M. Shoaib, and A. Alamri, "HarVis: An integrated social media content analysis framework for YouTube platform," *Inf. Syst.*, vol. 69, pp. 25–39, Sep. 2017.

[44] K.-C. Lee, C.-H. Hsieh, L.-J. Wei, C.-H. Mao, J.-H. Dai, and Y.-T. Kuang, "Sec-buzzer: Cyber security emerging topic mining with open threat intelligence retrieval and timeline event annotation," *Soft Comput.*, vol. 21, no. 11, pp. 2883–2896, Jun. 2017.

[45] M. Kim, K. Kang, D. Park, J. Choo, and N. Elmqvist, "TopicLens: Efficient multi-level visual topic exploration of large-scale document collections," *IEEE Trans. Vis. Comput. Graph.*, vol. 23, no. 1, pp. 151–160, Jan. 2017.

[46] K. Zhao, G. Cong, J.-Y. Chin, and R. Wen, "Exploring market competition over topics in spatio-temporal document collections," *VLDB J.*, vol. 28, no. 1, pp. 123–145, Feb. 2019.

[47] T. Zhang, Y. Y. Tang, B. Fang, and Y. Xiang, "Document clustering in correlation similarity measure space," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 6, pp. 1002–1013, Jun. 2012.

[48] C.-L. Chen, F. S. C. Tseng, and T. Liang, "Mining fuzzy frequent itemsets for hierarchical document clustering," *Inf. Process. Manage.*, vol. 46, no. 2, pp. 193–211, Mar. 2010.

[49] L. Jing, M. K. Ng, and J. Z. Huang, "Knowledge-based vector space model for text clustering," *Knowl. Inf. Syst.*, vol. 25, no. 1, pp. 35–55, Oct. 2010.

[50] M.-C. Chiang, C.-W. Tsai, and C.-S. Yang, "A time-efficient pattern reduction algorithm for k-means clustering," *Inf. Sci.*, vol. 181, no. 4, pp. 716–731, Feb. 2011.

[51] B. R. Prakash and M. Hanumanthappa, "Web snippet clustering and labeling using lingo algorithm," *Int. J. Adv. Res. Comput. Sci.*, vol. 3, pp. 262–265, Mar. 2012.

[52] L. F. D. C. Nassif and E. R. Hruschka, "Document clustering for forensic analysis: An approach for improving computer inspection," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 1, pp. 46–54, Jan. 2013.

[53] T. Wang, J. Wei, W. Zhang, H. Zhong, and T. Huang, "Workload-aware anomaly detection for Web applications," *J. Syst. Softw.*, vol. 89, pp. 19–32, Mar. 2014.

[54] S. Jinarat, C. Haruechaiyasak, and A. Rungsawang, "Graph-based concept clustering for Web search results," *Int. J. Electr. Comput. Eng.*, vol. 5, no. 6, pp. 1536–1544, 2015.

[55] P. S. Gamare and G. A. Patil, "Web document clustering using hybrid approach in data mining," *Int. J. Res. Advent Technol.*, vol. 3, pp. 92–97, Jul. 2015.

[56] E. Negm, S. AbdelRahman, and R. Bahgat, "PREFCA: A portal retrieval engine based on formal concept analysis," *Inf. Process. Manage.*, vol. 53, no. 1, pp. 203–222, Jan. 2017.

[57] T. Wang, W. Zhang, C. Ye, J. Wei, H. Zhong, and T. Huang, "FD4C: Automatic fault diagnosis framework for Web applications in cloud computing," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 46, no. 1, pp. 61–75, Jan. 2016.

[58] L.-C. Chen, "A novel page clipping search engine based on page discussion topics," *Knowl. Inf. Syst.*, vol. 58, no. 3, pp. 525–550, Mar. 2019.

[59] L. M. Abualigah, A. T. Khader, and E. S. Hanandeh, "A new feature selection method to improve the document clustering using particle swarm optimization algorithm," *J. Comput. Sci.*, vol. 25, pp. 456–466, Mar. 2018.

[60] C. Li, J. Bai, Z. Wenjun, and Y. Xihao, "Community detection using hierarchical clustering based on edge-weighted similarity in cloud environment," *Inf. Process. Manage.*, vol. 56, no. 1, pp. 91–109, Jan. 2019.

[61] K. Sadaf and M. Alam, "Web search result clustering-a review," *Int. J. Comput. Sci. Eng. Surv.*, vol. 3, pp. 85–92, Aug. 2012.

[62] Y. Chen, G.-R. Xue, and Y. Yu, "Advertising keyword suggestion based on concept hierarchy," in *Proc. Int. Conf. Web Search Web Data Mining (WSDM)*, 2008, pp. 251–260.

[63] L.-C. Chen, "Using a two-stage technique to design a keyword suggestion system," *Inf. Res., Int. Electron. J.*, vol. 15, pp. 1–15, Mar. 2010.

[64] S. Gerani, M. Keikha, M. Carman, and F. Crestani, "Personal blog retrieval using opinion features," in *Proc. 33rd Eur. Conf. Adv. Inf. Retr.*, Dublin, Republic of Ireland, 2011, pp. 747–750.

[65] D. Qiao, J. Zhang, Q. Wei, and G. Chen, "Finding competitive keywords from query logs to enhance search engine advertising," *Inf. Manage.*, vol. 54, no. 4, pp. 531–543, Jun. 2017.

[66] D. Jiang, K. W.-T. Leung, and W. Ng, "Query intent mining with multiple dimensions of Web search data," *World Wide Web*, vol. 19, no. 3, pp. 475–497, May 2016.

[67] F. Cai and M. de Rijke, "Learning from homologous queries and semantically related terms for query auto completion," *Inf. Process. Manage.*, vol. 52, no. 4, pp. 628–643, Jul. 2016.

[68] S. Tahery and S. Farzi, "Customized query auto-completion and suggestion—A review," *Inf. Syst.*, vol. 87, Jan. 2020, Art. no. 101415.

[69] L.-C. Chen, "Using a new relational concept to improve the clustering performance of search engines," *Inf. Process. Manage.*, vol. 47, no. 2, pp. 287–299, Mar. 2011.

[70] L. Chen, "Term suggestion with similarity measure based on semantic analysis techniques in query logs," *Online Inf. Rev.*, vol. 35, no. 1, pp. 9–33, Feb. 2011.

[71] M. P. Kato, T. Sakai, and K. Tanaka, "When do people use query suggestion? A query suggestion log analysis," *Inf. Retr.*, vol. 16, no. 6, pp. 725–746, Dec. 2013.

[72] D. Jiang, K. W.-T. Leung, L. Yang, and W. Ng, "Query suggestion with diversification and personalization," *Knowl.-Based Syst.*, vol. 89, pp. 553–568, Nov. 2015.

[73] R. Zhou, S. Khemmarat, L. Gao, J. Wan, J. Zhang, Y. Yin, and J. Yu, "Boosting video popularity through keyword suggestion and recommendation systems," *Neurocomputing*, vol. 205, pp. 529–541, Sep. 2016.

[74] H. Zhou, M. Huang, Y. Mao, C. Zhu, P. Shu, and X. Zhu, "Domain-constrained advertising keyword generation," in *Proc. World Wide Web Conf. (WWW)*, 2019, pp. 2448–2459.

[75] O. Ferret, B. Grau, M. Hurault-Plantet, G. Illouz, C. Jacquemin, and L. Monceaux, "How NLP can improve question answering," *Knowl. Org.*, vol. 29, pp. 135–155, Sep. 2002.

[76] A. Ben Abacha and P. Zweigenbaum, "MEANS: A medical question-answering system combining NLP techniques and semantic Web technologies," *Inf. Process. Manage.*, vol. 51, no. 5, pp. 570–594, Sep. 2015.

[77] A. Abdi, N. Idris, and Z. Ahmad, "QAPD: An ontology-based question answering system in the physics domain," *Soft Comput.*, vol. 22, no. 1, pp. 213–230, Jan. 2018.

[78] L. Hirschman and R. Gaizauskas, "Natural language question answering: The view from here," *Natural Lang. Eng.*, vol. 7, no. 4, pp. 275–300, Dec. 2001.

[79] S. P. Lende and M. M. Raghuwanshi, "Question answering system on education acts using NLP techniques," in *Proc. World Conf. Futuristic Trends Res. Innov. Social Welfare (Startup Conclave)*, Feb. 2016, pp. 1–6.

[80] K. C. Raghavi, M. K. Chinnakotla, and M. Shrivastava, "'Answer ka type kya he?': Learning to classify questions in code-mixed language," in *Proc. 24th Int. Conf. World Wide Web-WWW Companion*, 2015, pp. 853–858.

[81] Z. Yin, C. Zhang, D. W. Goldberg, and S. Prasad, "An NLP-based question answering framework for spatio-temporal analysis and visualization," in *Proc. 2nd Int. Conf. Geoinformatics Data Anal.*, Mar. 2019, pp. 61–65.

[82] P. Gupta and V. Gupta, "A survey of text question answering techniques," *Int. J. Comput. Appl.*, vol. 53, no. 4, pp. 1–8, 2012.

[83] Y. Yang, W.-T. Yih, and C. Meek, "WikiQA: A challenge dataset for open-domain question answering," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2015, pp. 2013–2018.

[84] E. Brill, S. Dumais, and M. Banko, "An analysis of the AskMSR question-answering system," in *Proc. ACL Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2002, pp. 257–264.

[85] A. Bhandwaldar and W. Zadrozny, "UNCC QA: Biomedical question answering system," in *Proc. 6th BioASQ Workshop Challenge Large-Scale Biomed. Semantic Indexing Question Answering*, 2018, pp. 66–71.

[86] Z. Salah, A. Aloqaily, M. Al-Hassan, and A.-R. Al-Ghuwairi, "A methodology to refine labels in Web search results clustering," *Int. J. Comput. Intell. Syst.*, vol. 12, no. 1, pp. 299–310, 2018.

[87] O. E. Zamir, "Clustering Web documents: A phrase-based method for grouping search engine results," Ph.D. dissertation, Dept. Comput. Sci., Univ. Washington, Seattle, WA, USA, 1999.

[88] F. Omara, M. Amoon, N. El-Fishawy, and S. Elkazaz, "Analysing Anchor Links to Enhance the Web Snippet Clustering Technique," in *Proc. Int. Conf. Inf. Syst.*, 2012, pp. SE-7–SE-11.

[89] R. Qumsiyeh and Y.-K. Ng, "Enhancing Web search by using query-based clusters and multi-document summaries," *Knowl. Inf. Syst.*, vol. 47, no. 2, pp. 355–380, May 2016.

[90] M. Manoj and J. Elizabeth, "Information retrieval on Internet using meta-search engines: A review," *J. Sci. Ind. Res.*, vol. 67, pp. 739–746, Oct. 2008.

[91] A. E. Howe and D. Dreilinger, ''SAVVYSEARCH: A metasearch engine that learns which search engines to query,'' *AI Mag.*, vol. 18, p. 19, Jun. 1997.

[92] E. J. Glover, S. Lawrence, W. P. Birmingham, and C. L. Giles, ''Architecture of a metasearch engine that supports user information needs,'' in *Proc. 8th Int. Conf. Inf. Knowl. Manage. (CIKM)*, 1999, pp. 210–216.

[93] L. Chen and C. Luh, ''Web page prediction from metasearch results,'' *Internet Res.*, vol. 15, no. 4, pp. 421–446, Sep. 2005.

[94] M. S. Desarkar, S. Sarkar, and P. Mitra, ''Preference relations based unsupervised rank aggregation for metasearch,'' *Expert Syst. Appl.*, vol. 49, pp. 86–98, May 2016.

[95] O. Zamir and O. Etzioni, ''Web document clustering: A feasibility demonstration,'' in *Proc. 21st Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Melbourne, VIC, Australia, 1998, pp. 46–54.

[96] T. Trinh, D. Wu, and J. Z. Huang, ''C3C: A new static content-based three-level Web cache,'' *IEEE Access*, vol. 7, pp. 11796–11808, 2019.

[97] X. Hong, T. Shen, L. Shen, Z. Yu, and J. Guo, ''Unstructured data extraction of chinese expert Web page,'' *Int. J. Wireless Mobile Comput.*, vol. 7, no. 2, pp. 132–136, 2014.

[98] P. Hazel. (2018). *PCRE—Perl Compatible Regular Expressions*. Accessed: Dec. 10, 2019. [Online]. Available: http://www.pcre.org/

[99] M. Porter and R. Boulton. (2017). *Snowball: A Language for Stemming Algorithms*. Accessed: Dec. 10, 2019. [Online]. Available: http://snowball.tartarus.org/

[100] C. Fox, ''A stop list for general text,'' *ACM SIGIR Forum*, vol. 24, nos. 1–2, pp. 19–21, Sep. 1989.

[101] J. Murphy, C. Hofacker, and R. Mizerski, ''Primacy and recency effects on clicking behavior,'' *J. Comput.-Mediated Commun.*, vol. 11, no. 2, pp. 522–535, Jan. 2006.

[102] Y. Xu and D. Wang, ''Order effect in relevance judgment,'' *J. Amer. Soc. Inf. Sci. Technol.*, vol. 59, no. 8, pp. 1264–1275, Jun. 2008.

[103] J. Unkel and A. Haas, ''The effects of credibility cues on the selection of search engine results,'' *J. Assoc. Inf. Sci. Technol.*, vol. 68, no. 8, pp. 1850–1862, Aug. 2017.

[104] A. C. Yang and S.-J. Tsai, ''Is mental illness complex? From behavior to brain,'' *Prog. Neuro-Psychopharmacol. Biol. Psychiatry*, vol. 45, pp. 253–257, Aug. 2013.

[105] P. G. Nestor and R. K. Schutt, *Research Methods in Psychology: Investigating Human Behavior*. Newbury Park, CA, USA: Sage, 2018.

[106] B. Plank, H. Martínez Alonso, . Agić, D. Merkler, and A. Søgaard, ''Do dependency parsing metrics correlate with human judgments?'' in *Proc. 19th Conf. Comput. Natural Lang. Learn.*, 2015, pp. 315–320.

[107] L.-C. Chen, C.-J. Luh, and C. Jou, ''Generating page clippings from Web search results using a dynamically terminated genetic algorithm,'' *Inf. Syst.*, vol. 30, no. 4, pp. 299–316, Jun. 2005.

[108] B. Croft and J. Lafferty, *Language Modeling for Information Retrieval*. Springer, 2013.

[109] J. Leskovec, A. Rajaraman, and J. D. Ullman, *Mining of Massive Datasets*. Cambridge, U.K.: Cambridge Univ. Press, 2014.

[110] R. L. Cilibrasi and P. M. B. Vitanyi, ''The Google similarity distance,'' *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 3, pp. 370–383, Mar. 2007.

[111] H. Zhao and Z. Qi, ''Hierarchical agglomerative clustering with ordering constraints,'' in *Proc. 3rd Int. Conf. Knowl. Discovery Data Mining*, Phuket, Thailand, Jan. 2010, pp. 195–199.

[112] O. Zamir and O. Etzioni, ''Grouper: A dynamic clustering interface to Web search results,'' *Comput. Netw.*, vol. 31, nos. 11–16, pp. 1361–1374, May 1999.

[113] E. Decencière, A. Belhedi, S. Koudoro, F. Flament, G. François, V. Rubert, I. Pécile, and J. Pierre, ''A 2.5 D approach to skin wrinkles segmentation,'' *Image Anal. Stereol.*, vol. 38, no. 1, pp. 75–81, 2019.

[114] H. Zhang, F. Boons, and R. Batista-Navarro, ''Whose story is it anyway? Automatic extraction of accounts from news articles,'' *Inf. Process. Manage.*, vol. 56, no. 5, pp. 1837–1848, Sep. 2019.

[115] S. Brin and L. Page, ''The anatomy of a large-scale hypertextual Web search engine,'' *Comput. Netw. ISDN Syst.*, vol. 30, nos. 1–7, pp. 107–117, Apr. 1998.

[116] T. Back, *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. New York, NY, USA: Oxford Univ. Press, 1996.

[117] Google. (2014). *Google Stop-Words*. Accessed: Dec. 28, 2018. [Online]. Available: https://code.google.com/archive/p/stop-words/

[118] MySQL. (2017). *MySQL Full-Text Stopwords*. Accessed: Dec. 28, 2018. [Online]. Available: https://dev.mysql.com/doc/refman/5.5/en/fulltext-stopwords.html

[119] Apache Foundation. (2020). *Ab—Apache HTTP Server Benchmarking Tool*. Accessed: Mar. 24, 2020. [Online]. Available: https://httpd.apache.org/docs/2.4/programs/ab.html

[120] X.-Y. Liu, Y. Liang, S. Wang, Z.-Y. Yang, and H.-S. Ye, ''A hybrid genetic algorithm with wrapper-embedded approaches for feature selection,'' *IEEE Access*, vol. 6, pp. 22863–22874, 2018.

LIN-CHIH CHEN received the B.S., M.S., and Ph.D. degrees in information management from the National Taiwan University of Science and Technology, Taipei, Taiwan, in 1996, 1998, and 2005, respectively. He is currently a Professor with the Department of Information Management, National Dong Hwa University, Hualien, Taiwan. He is the Head of the Cayley Team and has developed dozens of web-based smart systems. He has published more than 60 academic articles and most of them are sole authors. His current research interests include Web mining, Web intelligence, semantic analysis, big data analysis, cloud computing, and AI.