Achieving Effective Cloud Search Services: Multi-keyword Ranked Search over Encrypted Cloud Data Supporting Synonym Query

Zhangjie Fu, *Member,* IEEE, Xingming Sun, *Senior Member*, IEEE, Nigel Linge, Lu Zhou

**Abstract — ***In recent years, consumer-centric cloud computing paradigm has emerged as the development of smart electronic devices combined with the emerging cloud computing technologies. A variety of cloud services are delivered to the consumers with the premise that an effective and efficient cloud search service is achieved. For consumers, they want to find the most relevant products or data, which is highly desirable in the "pay-as-you use" cloud computing paradigm. As sensitive data (such as photo albums, emails, personal health records, financial records, etc.) are encrypted before outsourcing to cloud, traditional keyword search techniques are useless. Meanwhile, existing search approaches over encrypted cloud data support only exact or fuzzy keyword search, but not semantics-based multi-keyword ranked search. Therefore, how to enable an effective searchable system with support of ranked search remains a very challenging problem. This paper proposes an effective approach to solve the problem of multi-keyword ranked search over encrypted cloud data supporting synonym queries. The main contribution of this paper is summarized in two aspects: multi-keyword ranked search to achieve more accurate search results and synonym-based search to support synonym queries. Extensive experiments on real-world dataset were performed to validate the approach, showing that the proposed solution is very effective and efficient for multi-keyword ranked searching in a cloud environment [1].*

Index Terms — Cloud computing, consumer-centric cloud, keyword search, ranked search.

I. INTRODUCTION

In recent years, many consumer electronic devices (e.g. Smartphone) with support of high speed computing combined with the emerging cloud computing paradigm provide a variety of service to the consumers. Cabarcos P.A. et al [1] proposed a novel middleware architecture that allows sessions initiated from one device to be seamlessly transferred to a second one under a cloud computing environment. Díaz-Sánchez D. et al [2] presented a cloud computing middleware Media Cloud for set-top boxes for classifying, searching, and delivering media inside home network and across the cloud. Seung G. L. et al [3] proposed a personalized DTV program recommendation system under a cloud computing environment. The system can analyze and use the viewing pattern of consumers to personalize the program recommendations.

However, all these services are likely to be available to consumers only with the premise that an effective and efficient cloud search service is achieved. Consumers want to find the most relevant products or data, which is highly desirable in the "pay-as-you use" cloud computing paradigm.

One hand, consumer-centric cloud computing [4] is a new model of enterprise-level IT infrastructure that provides on-demand high quality applications and services from a shared pool of configuration computing resources for consumers. On the other hand, some problems may be caused in this circumstance since the Cloud Service Provider (CSP) possesses full control of the outsourced data. There may exist unauthorized operation [5] on the outsourced data on account of curiosity or profit. So sensitive data are encrypted before outsourcing to the cloud. However, encrypted data make the traditional data utilization services based on plaintext keyword search useless. The simple and awkward method of downloading all the data and decrypting locally is obviously impractical, because the authorized cloud consumers must hope to search their interested data rather than all the data. Hence, it is an especially important thing to explore an effective search service over encrypted outsourced data.

Existing search approaches cannot accommodate such requirements like ranked search, multi-keywords search, semantics-based search etc. The ranked search enables cloud customers to find the most relevant information quickly. Ranked search can also reduce network traffic as the cloud server sends back only the most relevant data. Multi-keyword search is also very important to improve search result accuracy as single keyword search often return coarse search results. In the real search scenario, it is quite common that cloud customers' searching input might be the synonyms of the predefined keywords, not the exact or fuzzy matching keywords due to the

[1] This work is supported by the NSFC(61232016, 61173141, 61173142, 61173136, 61103215, 61373132, 61373133, 61300238), GYHY201206033, 201301030, 2013DFG12860, BC2013012, Jiangsu Provincial Natural Science Foundation (BK20131004) and PAPD fund.

Z. J. Fu is with the School of Computer and Software & Jiangsu Engineering Centre of Network Monitoring, Nanjing University of Information Science and Technology, Nanjing, 210044, CHINA (e-mail: wwwfzj@126.com).

X. M. Sun is with the Jiangsu Engineering Centre of Network Monitoring, Nanjing University of Information Science and Technology, Nanjing, 210044, CHINA (e-mail: sunnudt@163.com).

N. Linge is with the School of Computing, Science and Engineering, University of Salford, Salford, M5 4WT, UK.(e-mail: n.linge@salford.ac.uk).

L. Zhou is with the School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing, 210044, CHINA (e-mail: zl_0713@163.com).

possible synonym substitution (reproduction of information content), such as commodity and goods, and/or her/his lack of exact knowledge about the data. The existing searchable encryption schemes support only exact or fuzzy keyword search. That is, there is no tolerance of synonym substitution and/or syntactic variation which, on the other hand, are typical user searching behaviours and happen very frequently. Therefore, synonym-based multi-keyword ranked search over encrypted cloud data remains a very challenging problem.

To meet the challenge of effective search system, this paper proposes a practically efficient and flexible searchable scheme which supports both multi-keyword ranked search and synonym-based search. To address multi-keyword search and result ranking, Vector Space Model (VSM) [6] is used to build document index, that is to say, each document is expressed as a vector where each dimension value is the Term Frequency (TF) weight of its corresponding keyword. A new vector is also generated in the query phase. The vector has the same dimension with document index and its each dimension value is the Inverse Document Frequency (IDF) weight. Then cosine measure can be used to compute similarity of one document to the search query. To improve search efficiency, a tree-based index structure which is a balance binary tree is used. The searchable index tree is constructed with the document index vectors. So the related documents can be found by traversing the tree.

The contributions of this paper are summarized as follows:

(1) For the first time, a semantics-based multi-keyword ranked search technology over encrypted cloud data which supports synonym queries is proposed. The search results can be achieved when authorized cloud customers input the synonyms of the predefined keywords, not the exact or fuzzy matching keywords, due to the possible synonym substitution and/or her lack of exact knowledge about the data.

(2) By incorporating the state-of-art text feature extraction technique TFIDF (term frequency-inverse document frequency), an enhanced semantic feature extraction method E-TFIDF is proposed. The E-TFIDF algorithm, which can extract the most representative keywords from outsourced text documents, improves the accuracy of search results.

(3) Extensive experiments on the real-world dataset further show the effectiveness and efficiency of proposed solution.

In the remainder of this paper, the following information is presented: In Section II, related research is discussed. Then, problem formulation is described in Section III. In Section IV, the proposed method for building keyword set extended by synonym in cloud is presented in detail. Section V presents the proposed search schemes. Performance analysis is presented in Section VI. Finally, in Section VII, the paper concludes with some suggestions for future work.

II. Related Work

A. Consumer-centric Cloud Services

Cabarcos P.A. et al [1] in 2012 proposed a novel middleware architecture that allows sessions initiated from one device to be seamlessly transferred to a second one under a cloud computing environment. Díaz-Sánchez D. et al [2] presented a cloud computing middleware Media Cloud for Set-top boxes for classifying, searching, and delivering media inside home network and across the cloud. Seung Gwan Lee et al [3] proposed a personalized DTV Program Recommendation system under a cloud computing environment. The system can analyze and use the viewing pattern of consumers to personalize the program recommendations. Grzonkowski S. et al [7] proposed a user centric approach to authentication for home networks. This approach enables the sharing of personalized content and more sophisticated network-based services over a conventional TCP/IP infrastructure. Sanchez R. et al [8] proposed an IdM architecture based on privacy and reputation extensions to enable the global scalability and usability for consumer cloud computing paradigm. However, all these services are likely to be available to consumers only with the premise that an effective and efficient cloud search service is achieved.

B. Searchable Encryption in Cloud

To apply the searchable encryption to cloud computing, some researchers have been studying further on how to search over encrypted cloud data efficiently. Li et al. [9] firstly proposed a fuzzy keyword search scheme over encrypted cloud data, which combines edit distance with wildcard-based technique to construct fuzzy keyword sets, to address problems of minor typos and format inconsistence. Wang et al. [10] proposed a ranked search scheme, in which by giving each keyword a weight TF-IDF, the cloud server can rank relevant data files with no knowledge of a specific keyword weight. But this scheme supports only single keyword search. Then Cao et al. [11] proposed a ranked scheme supporting multi-keyword, which uses a vector space model and characteristics of matrix to realize trapdoor unlinkablility and thereby preserves data privacy. Chai et al. [12] proposed a verifiable symmetric search encryption scheme, which can prove the correctness and completeness of results. Sun et al. [13] also proposed a multi-keyword ranked search scheme based on vector space model (VSM). The VSM can measure the similarity between document index vector and query vector and hence support more accurate ranked search results. But this scheme cannot support semantics-based search.

III. Problem formulation

A. The System Model

The system model considered in the paper involves three different entities: the data owner, the data user and the cloud server, as illustrated in Fig.1. The data owner, individual or enterprise, has a document collection *DC* which will be outsourced into the cloud. The data owner encrypts *DC* in the form of *C* before outsourcing to the cloud. And for the purpose of searching interested data, the data owner will also generate a searchable index *I* based on a set of distinct keywords *W* extracted from *DC*. Then, the encrypted file collection *C* and searchable index *I* will be outsourced to the cloud together by the data owner. In the search stage, the system will generate an encrypted search trapdoor based on the keywords or the synonyms of the predefined keywords entered by the user (has been authorized by data owner). Given the trapdoor, the cloud server will search the index I and then return search results to the user. The search result is a set of encrypted documents containing the entered keywords, and they are well-ranked according to similarity measures. An additional feature provided by the system is that it can return a certain number of documents instead of all relevant documents. By sending a parameter k together with the search query, the user can get top-k most relevant documents.

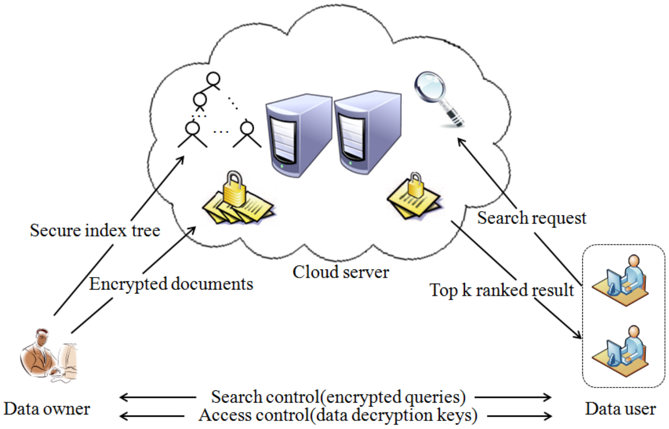


Fig. 1. Framework of the search over encrypted cloud data

B. Notation

- DC – the plaintext document collection, expressed as a set of m documents $DC = \{d \mid d_1, d_2, ..., d_m\}$.
- C – the encrypted form of DC stored in the cloud server, expressed as $C = \{c \mid c_1, c_2, ...c_m\}$.
- W – the keyword dictionary, including n keywords, expressed as $W = \{w \mid w_1, w_2, ...w_n\}$.
- I – the searchable index tree generated from the whole document set DC. (Each leaf node in the index tree is associated with a document in DC.)
- D_d – the index vector of document d for all the keywords in W.
- Q – the query vector for the keyword set $\widetilde{W}$.
- $\widetilde{D}_d$ – the encrypted form of D_d.
- $\widetilde{Q}$ – the encrypted form of Q.
- $f(sk,\cdot)$ —pseudorandom function (PRF), defined as: $\{0,1\}^* \times sk -> \{0,1\}^k$, sk is a secret key.
- $g(\cdot,\cdot)$ —pseudorandom function (PRF), defined as: $\{0,1\}^k \times \{0,1\} -> \{0,1\}^l$.
- $\widetilde{W}$ – a subset of W, which represent the keywords in a search query, expressed as $\widetilde{W} = \{w_{i1}, w_{i2}, ..., w_{it}\}$.
- $T_{\widetilde{W}}$ – the encrypted form of $\widetilde{W}$, in the form of $T_{\widetilde{W}} = \{f(sk, w_{i1}), f(sk, w_{i2}), ..., f(sk, w_{it})\}$.
- λ – a static hash table for all keywords in W. There are n entries in λ and each entity is a tuple(key, value), in which the key is from a domain of exponential size, i.e., from $\{0,1\}^k$ representing a keyword in W, and value is a ciphertext of a boolean value. For a key x in λ, $\lambda[x]$ is the value associated with key x.

C. Preliminaries

Synonym expansion: Synonyms are words with the same or similar meanings. In order to improve the accuracy of search results, the keywords extracted from outsourced text documents need to be extended by common synonyms, as cloud customers' searching input might be the synonyms of the predefined keywords, not the exact or fuzzy matching keywords due to the possible synonym substitution and/or her lack of exact knowledge about the data. A common synonym thesaurus is built on the foundation of the New American Roget's College Thesaurus (NARCT) [14]. Then the keyword set is extended by using the constructed synonym thesaurus.

Rank function: In information retrieval, a ranking function is usually used to evaluate relevant scores of matching files to a request. Among lots of ranking functions, the "*TF*×*IDF*" rule [6] is most widely used, where *TF* (term frequency) denotes the occurrence of the term appearing in the document, and *IDF* (inverse document frequency) is often obtained by dividing the total number of documents by the number of files containing the term. That means, *TF* represents the importance of the term in the document and *IDF* indicates the importance or degree of distinction in the whole document collection. Each document is corresponding to an index vector D_d that stores normalized *TF* weight, and the query vector Q stores normalized *IDF* weight. Each dimension of D_d or Q is related to a keyword in *W*, and the order is same with that in *W*, that is, $D_d[i]$ is corresponding to keyword w_i in *W*. The similarity evaluation function [15] is employed for cosine measure. The notations used in similarity evaluation function are showed as follows:

- $f_{d,j}$, the TF of keyword w_j within the document *d*;
- f_j, the number of documents containing the keyword w_j;
- M, the total number of documents in the document collection;
- N, the total number of keywords in the keyword dictionary;
- $w_{d,j}$, the TF weight computed from $f_{d,j}$;
- $w_{q,j}$, the IDF weight computed from N and f_j;

The definition of the similarity function is as follows:

$$SC(Q, D_d) = \frac{\sum_{j=1}^{N} w_{q,j} \cdot w_{d,j}}{\sqrt{\sum_{j=1}^{N} (w_{q,j})^2} \cdot \sqrt{\sum_{j=1}^{N} (w_{d,j})^2}} \quad (1)$$

Where $w_{q,j} = 1 + \ln f_{d,j}$, $w_{q,j} = \ln(1 + \frac{N}{fj})$.The normalized *TF* and *IDF* weight are $\frac{w_{d,j}}{\sqrt{\sum_{j=1}^{N} (w_{d,j})^2}}$ and $\frac{w_{q,j}}{\sqrt{\sum_{j=1}^{N} (w_{q,j})^2}}$ respectively, and hence, the vector Q and D_d are both unit vectors.

Searchable Index Tree: Searchable index is a balance binary tree, a dynamic data structure, showed in Fig.2. Given the document collection $DC = \{d \mid d_1, d_2, ..., d_m\}$ (each document d_i is corresponding to an identifier i and an index vector D_{di}), the index tree I can be built. The data structure is built using the procedure, which is expressed as buildIndex (DC), showed as follows:

(1)For each document d_i in DC , a leaf node is generated, where stores identifier i and index vector D_{di} (the value of each dimension of D_{di} is a normalized TF weight).

(2)Then the tree is built following a postorder traversal with all leaf nodes generated in step (1). Each internal node u of the index tree stores an n-bit vector D (each dimension of D is corresponding to a keyword in W with the same order in D_d, i.e. $D[i]$ is corresponding to w_i). If there is at least one path from u to a leaf node storing identifier i and document d_i contains keyword w_j (that is to say, $D_{di}[j]\neq 0$), $D_u[i]=1$, otherwise $D_u[i]=0$.

(3)This step introduces how to generate vector D in each internal node. Let v and w be the left child and right child of internal node u respectively, then $D_u[i]=1$ if $D_v[i]=1$ when v is an internal node ($D_{dj}[i]\neq 0$ when v is a leaf node and stores identifier j) or $D_w[i]=1$ when w is an internal node ($D_{dj}[i]\neq 0$, when w is a leaf node and stores identifier j), otherwise $D_u[i]=0$.

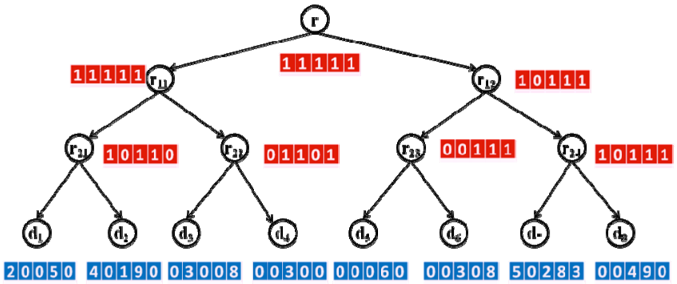


Fig. 2. The index tree for a document collection of m=8 documents and with n=5 keywords

Tree-based Search Algorithm: The sequential search process for keywords in a search request $\widetilde{W}$ conducts as follows: the procedure starts from the root node and when arrives at an internal node u , if at least a keyword w in $\widetilde{W}$ leads to $D_u[k]=1$ (k is the order number of w in W), it continues to search both subtrees of u , otherwise stops searching in the subtree T_u (T_u denotes the tree who's root is u) because none of leaf node in T_u contains keyword in search query. When arrive at a leaf node, the process computes the cosine value between the index vector stored in the leaf node and the query vector as the similarity score. The number of documents that contain the keyword in the search query are denoted as r . In the sequential search, the procedure will traverse as many paths as r . So, the search complexity is $O(r\log m)$ as the height of a balance binary tree with m leaf nodes is $\log m+1$.

IV. Construction of Keyword Set Extended by Synonym in Cloud

A. Improved Keyword Extraction Method

In order to search the interested data rather than all the data efficiently, keywords need to be extracted firstly from cloud data before outsourcing. This section proposes a new text feature weighting method that adds a new weighting factor to reflect the distinguishability of the keyword on the base of the original TFIDF (term frequency-inverse document frequency) method. The proposed method can extract keywords which can more accurately represent the features of text. TFIDF is calculated as:

$$W_{ik}=TF\times IDF=TF\times\frac{1}{DF}=f_{ik}\times\log\frac{N}{n_k} \qquad (2)$$

Where f_{ik} is the frequency of term i in a text, N is the total number of texts, n_k is the number of the texts which contain term i. However, the TFIDF method also has some problems. It could not effectively reflect the distinguishability of feature term between categories. Therefore, a new weighting factor needs to be added to the calculation of TFIDF to represent the categories distinguishability of feature term between categories.

Let N be the total number of texts in corpus, let n be the number of texts containing the term i in corpus, let E_1 be the number of texts in the largest category containing the term i, let E_2 be the number of texts in the second largest category containing the term i. The new weighting factor C_d is added to the formula of TFIDF, the improved formula is as follows:

$$\begin{aligned}W'_{ik}&=TF\times IDF\times C_d=TF\times\frac{1}{DF}\times C_d\\&=f_{ik}\times\log\frac{N}{n_k}\times\frac{E_1-E_2}{n}\end{aligned} \qquad (3)$$

So the keywords are extracted from each outsourced text document by using the improved method. All keywords extracted from the same one text form one keyword subset, and all subsets form the keyword set at last. All the outsourced text documents can be expressed as follows:

$$\begin{cases}\text{file } 1: k^1_{f_1},k^2_{f_1},\dots,k^{n-1}_{f_1},k^n_{f_1};\\ \text{file } 2: k^1_{f_2},k^2_{f_2},\dots,k^{n-1}_{f_2},k^n_{f_2};\\ \dots\dots\\ \text{file } (m\text{-}1): k^1_{f_{m-1}},k^2_{f_{m-1}},\dots,k^{n-1}_{f_{m-1}},k^n_{f_{m-1}};\\ \text{file } m: k^1_{f_m},k^2_{f_m},\dots,k^{n-1}_{f_m},k^n_{f_m}.\end{cases} \qquad (4)$$

B. Building Keyword Set Extended by Synonym

In order to achieve a better semantics-based search algorithm for outsourced data, the keyword set need to be extended by common synonym.

Firstly, a common synonym thesaurus is built on the foundation of the New American Roget's College Thesaurus (NARCT) [14]. NARCT is decreased in quantity according to the following two principles: (1) Selecting the common words; (2) Selecting the words which can be semantically substituted completely. The constructed synonym set contains a total of 6353 synonym groups after the reduction.

Secondly, the keyword set is extended by using the constructed synonym thesaurus. The new keyword set containing synonym is shown as follows:

$$\begin{cases} \text{file } 1: k_{f_1}^1 \textit{ or } s_1,\ k_{f_1}^2 \textit{ or } s_2,\ ...,\ k_{f_1}^{n-1} \textit{ or } s_{n-1},\ k_{f_1}^n \textit{ or } s_n; \\ \text{file } 2: k_{f_2}^1 \textit{ or } s_1,\ k_{f_2}^2 \textit{ or } s_2,\ ...,\ k_{f_2}^{n-1} \textit{ or } s_{n-1},\ k_{f_2}^n \textit{ or } s_n; \\ ...... \\ \text{file } (m\text{-}1): k_{f_{m-1}}^1 \textit{ or } s_1,\ k_{f_{m-1}}^2 \textit{ or } s_2,\ ...,\ k_{f_{m-1}}^{n-1} \textit{ or } s_{n-1},\ k_{f_{m-1}}^n \textit{ or } s_n; \\ \text{file } m: k_{f_m}^1 \textit{ or } s_1,\ k_{f_m}^2 \textit{ or } s_2,\ ...,\ k_{f_m}^{n-1} \textit{ or } s_{n-1},\ k_{f_m}^n \textit{ or } s_n. \end{cases} \tag{5}$$

Where s_1 represents the synonym of $k_{f_i}^1$. If a keyword has two or more synonyms, then all synonyms are added into the keyword set. The repetitive keywords are deleted to reduce the burden of storage. At last, a simplified keyword set and corresponding keyword scoring table are constructed.

V. Efficient Ranked Search Scheme

This section will firstly describe a basic scheme in detail, and then present an enhanced scheme to further protect the sensitive frequency information from leakage. To design efficient multi-keyword ranked search schemes based on synonym, a four step procedure including {**Setup, GenIndex** (DC, SK, sk) , **GenQuery** $(\widetilde{W}, SK)$, **Search** $(I, \widetilde{Q}, T_{\widetilde{W}}, k)$ } is performed.

A. The Basic Scheme

Setup

In this phase, the system is initialized. The data owner generates the secret key SK and picks a random key sk . The SK includes: (1) A n-bit randomly generated vector S ; (2) Two n*n invertible matrices $\{M_1, M_2\}$. Hence, SK is in the form of a 3-tuple as $\{S, M_1, M_2\}$.

GenIndex (DC, SK, sk)

The data owner calls procedure buildindex(*DC*). Then, every document index vector D_d is splitted into two random vectors as $\{D'_d, D''_d\}$. The splitting procedure is expressed as follow: take S as the splitting indicator, if the *j*-th bit of S is 0, $D'_d[j]$ and $D''_d[j]$ are set as the same as $D_d[j]$; if the j-th bit of S is 1, $D'_d[j]$ and $D''_d[j]$ are set randomly so long as their sum is equal to $D_d[j]$. So, the encrypted index vector $\widetilde{D}_d$ is denoted as $\widetilde{D}_d = \{M_1{}^T D'_d, M_2{}^T D''_d\}$. Store $\widetilde{D}_d$ at the leaf node that stores correspondent D_d and delete D_d. For each internal node *u* in the index tree, a hash table λ is generated. There are *n* tuples (key, value) in λ , and for every i=1,2…n, set $\lambda_u[f(sk, w_i)] = g(f(sk, w_i), D_u[i])$. Store λ_u in internal node *u* and delete D_u. Finally, the encrypted searchable index tree I is generated.

GenQuery $(\widetilde{W}, SK)$

With *t* keywords of interest in $\widetilde{W}$, the query vector Q is generated where each dimension is a normalized IDF weight $w_{q,j}$ .Specifically, if w_i is in $\widetilde{W}$, set $Q[i] = w_{q,i}$, otherwise set $Q[i] = 0$. Next, Q is split into two random vectors as $\{Q', Q''\}$ with the similar splitting procedure used for document index vector. The difference is that if the *j*-th bit of *S* is 0, $Q'[j]$ and $Q''[j]$ are set randomly so long as their sum is equal to $Q[j]$; if the *j*-th bit of S is 1, $Q'[j]$ and $Q''[j]$ are set as the same as $Q[j]$. Then, the encrypted query vector $\widetilde{Q}$ is in the form of $\{M_1{}^{-1}Q', M_2{}^{-1}Q''\}$. Next, $T_{\widetilde{W}} = \{f(sk, w_{i1}), f(sk, w_{i2}), ..., f(sk, w_{it})\}$ is produced by encrypting each item in $\widetilde{W}$. Finally, the $\{T_{\widetilde{W}}, \widetilde{Q}\}$ is sent to the cloud server.

Search $(I, \widetilde{Q}, T_{\widetilde{W}}, k)$

The cloud server follows the search algorithm expressed in section III. Let *u* be an internal node in I , and let $a_t = D_u[f(sk, w_t)]$ for each item in $T_{\widetilde{W}}$. If exist at least one a_t satisfies $Dec(f(sk, w_t), a_t) = 1$, the procedure continue to search *u*'s all children. When arrive at a leaf node, the procedure obtains the encrypted document vector $\widetilde{D}_d$ and compute the similarity of $\widetilde{D}_d$ and $\widetilde{Q}$ using the following formula.

$$\begin{aligned} & SC(\widetilde{Q}, \widetilde{D}_d) \\ & = \{M_1{}^{-1}Q', M_2{}^{-1}Q''\} \cdot \{M_1{}^T D'_d, M_2{}^T D''_d\} \\ & = Q' \cdot D'_d + Q'' \cdot D''_d \\ & = Q \cdot D_d \end{aligned} \tag{6}$$

B. The Enhanced Scheme

In the basic scheme, the keyword privacy leakage is possible in the known background model because the cosine value calculated from encrypted vector $\widetilde{D}_d$ and $\widetilde{Q}$ is equal to the one form vector D_d and Q. For the purpose of eliminating such equality property, some dummy keywords can be used. To be specific, all vectors (including document index vectors and query vectors) are extended to (*n*+*U*)-dimensions, where *U* is the number of dummy keywords and each extended dimension is corresponding to a dummy keyword. The only difference between the basic scheme and enhanced scheme is that several dummy keywords are introduced in enhanced scheme to protect similarity scores. The details in the enhanced scheme are not repeated here.

C. Privacy-Preserving

(1) Index confidentiality and Query confidentiality: Note that the only difference between the basic scheme and enhanced scheme is that several dummy keywords are introduced in enhanced scheme to protect similarity scores. Though dummy keywords result in extended dimension of related vectors and matrices, the main cryptographic method is same in two schemes. Hence, the enhanced scheme can protect index confidentiality and query confidentiality in both two threat models.

(2) Query unlinkability: As some dummy keywords are introduced, the randomly selected number ε_i will allow the enhanced scheme produce different similarity scores even for the same search keywords. The value of ε_i can be adjusted to control the level of variance thus the level of unlinkability. Hence, query unlinkability is much enhanced compared with the basic scheme to the extent that it is hard for the attacker to link the queries. However, since access pattern is not actively protected accounting for efficiency, the returned results from the same request will always bear some similarity which could be exploited with powerful statistical analysis by the very motivated cloud server. This is a trade-off that one has to make between efficiency and privacy.

VI. Performance Analysis

The overall performance of the proposed schemes is estimated by implementing the search system on a cloud server. The document set is built from the real data set: Reuters News stories. This dataset is a collection of 18, 821 newsgroup documents including 11, 293 train documents and 7, 528 test documents. The performance of the scheme is evaluated regarding the accuracy of the proposed keyword extraction method, as well as the performance of the proposed search approach.

A. Performance of Keyword Extraction Method

In order to verify the effectiveness of the improved E-TFIDF, KNN (K Nearest Neighbor) classifier is used to train and predict the category labels of documents. Two steps are performed as follows:

(1) Train documents are firstly trained using KNN classifier, and then a classification result is achieved; (2) Test document is classified according to the similarity of the test document and classifier.

The similarity of text vectors can be measured by cosine of vectorial angle. The greater cosine is, the higher similarity is. The cosine of vectorial angle can be expressed as:

$$\cos(d_i, d_j) = \frac{\sum_{k=1}^{n}(a_k^j * a_k^i)}{\sqrt{\sum_{k=1}^{n}(a_k^j)^2} \times \sqrt{\sum_{k=1}^{n}(a_k^j)^2}} \qquad (7)$$

Where $d_i(a_1^i, a_2^i, \cdots a_n^i)$ and $d_j(a_1^j, a_2^j, \cdots a_n^j)$ represent two text vectors. If $\cos(d_i, d_j) > \partial$ (∂ is a threshold), there is a high similarity between d_i and d_j.

The Macro *F1* is used to measure the classification accuracy, and it is calculated as follows:

$$Macro\_F1 = \sum_{i=1}^{m}\frac{N_i}{N} \times F1 = \sum_{i=1}^{m}\frac{N_i}{N} \times P(C_i \mid t)\frac{2 \times precision_i \times recall_i}{precision_i + recall_i} \quad (8)$$

Where N represents the number of test documents, m represents the number of categories, N_i represents the number of test documents which belong to i-th category, $precision_i$ represents the accuracy rate of i-th category, and $recall_i$ represents the recall rate of i-th category.

The classification accuracy of the improved method and the original method combining with KNN algorithm are shown in the following Fig. 3. It shows that the Macro *F1* of the improved method combining with KNN algorithm is higher than the original method combining with KNN algorithm. The maximum Macro *F1* value of original method is 90.382 which is lower than the proposed method. In addition, the standard deviation of the proposed method is lower than original method, which shows that the proposed method reduces dependence on the number of features. Therefore, it shows that the improved method $TF \times IDF \times C_d$ could accurately extract keywords from texts.

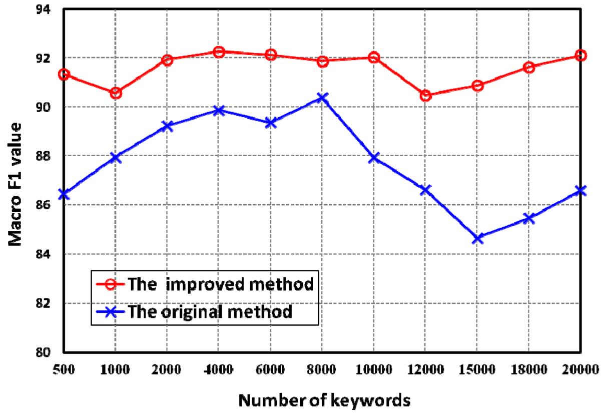


Fig. 3. The classification accuracy of two methods

Using the improved E-TFIDF method, keywords are extracted from the Reuters News stories of 18, 821 newsgroup documents. Total number of the extracted keywords is 106,715, and the final number of distinct keywords in keyword set is 46,153 with the average word length 5.63 after removing the repeated keywords.

B. Efficiency of Index Tree Construction

Given the keyword set constructed using synonym-based method, the time cost of index tree construction for the basic scheme and enhanced scheme is measured. It is obvious that the time cost of the index tree construction is mainly affected by the number of documents in the dataset and keywords in dictionary. For each internal node in the searchable index tree, the major computation is the encryption of the hash table, the time cost of which is proportional to the number of keywords in the dictionary. And for each leaf node in the index tree, the main computation is the encryption of the document index vector, which mainly depends on the time cost for two multiplications of a $H \times H$ matrix and an H -dimension vector where H is n in basic scheme and H is $n+U$ in enhanced scheme. And the whole number of nodes in the index tree (or the layers of the tree) is related to the number of documents in the dataset.

Fig. 4 indicates that the time cost for constructing index tree is proportional to the number of keywords in the dictionary with the same size of dataset. It can be shown that the index tree construction time of the enhanced scheme is a little more than the basic scheme account for the dimension extension. Although the time cost for constructing index tree is not an ignorable overhead for the data owner, it is a one-time operation before data outsourcing. In addition, the storage overhead of the index tree for the different sizes of dictionary with the fixed size of dataset m=18, 821 is shown in Fig. 5, which indicates that the sizes of index tree are very close in two schemes. However, the storage space is not a main problem in the cloud computing environment, because the index data only consume a small amount of storage space.

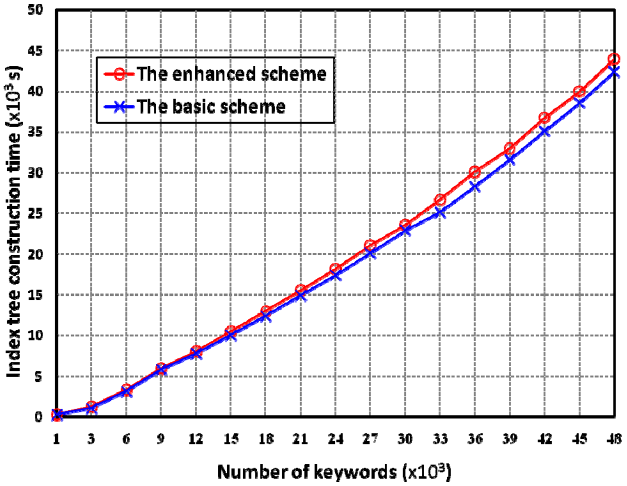


Fig. 4. Index tree construction time for the basic scheme and enhanced scheme

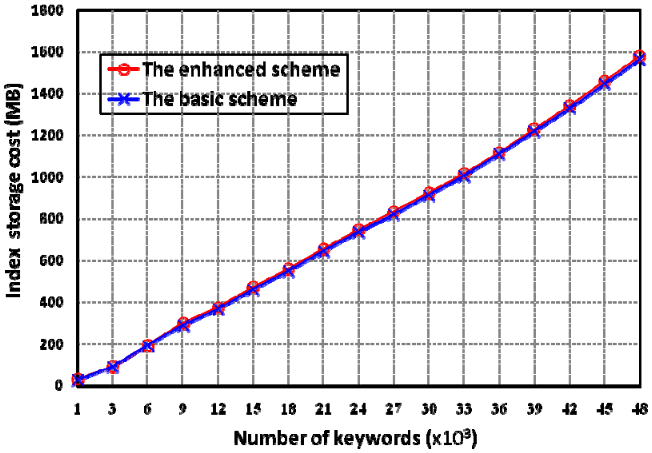


Fig. 5. Index storage cost of the basic scheme and enhanced scheme

C. Search Efficiency

In this section, the performance of search scheme is evaluated as the number of documents increases. The search process, which is implemented by the cloud server, is composed by computing the similarity scores of relevant documents and result ranking based on these scores. Fig. 6 shows the search time for the basic scheme and enhanced scheme. Let r represent the number of documents including the search keywords. Fig. 6 shows that the search time is mainly depends on the number of documents in the dataset when r is fixed and the time cost of two schemes is similar.

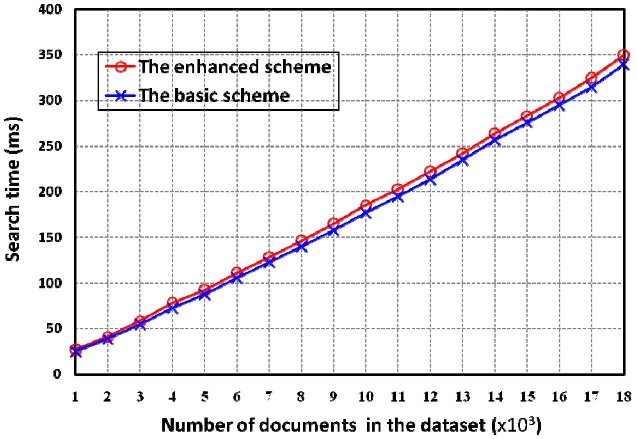
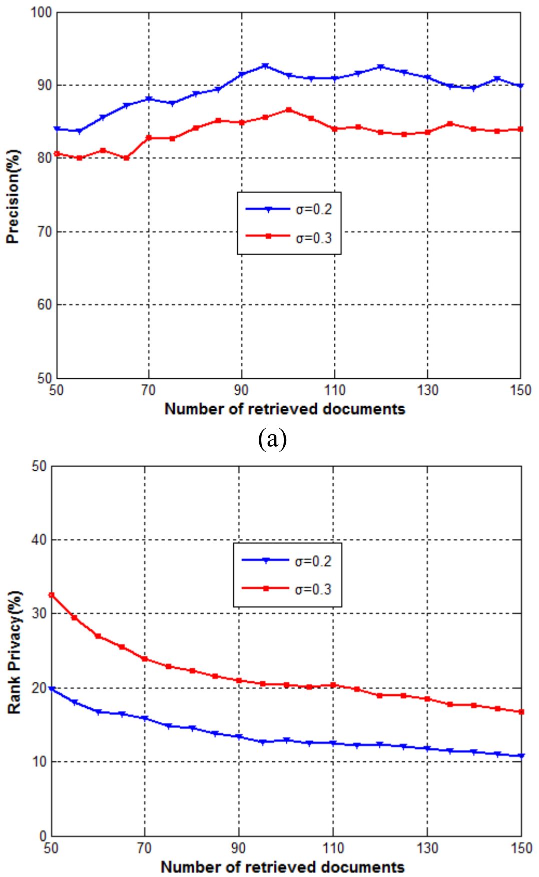


Fig. 6. Search time for the basic scheme and enhanced scheme (For the different size of dataset with the same number of documents including search keywords, r=100)

D. Precision and Privacy

In the enhanced scheme, similarity scores of related documents will be not exactly accurate since dummy keywords are introduced to protect keyword privacy. The definition of "precision" proposed by N. Cao et al [11] is employed to evaluate the accuracy of search result in enhanced scheme. Specifically, "precision" is defined as $P_k = k'/k$, where k is the number of ranked documents returned by the cloud server and k' is the number of real top-k documents in k returned documents. Fig. 7(a) shows that the precision is affected by the standard deviation σ of the random variable ε and the effectiveness of the enhanced scheme is not affected much with a small σ . That is to say, cloud customer can enjoy nearly the same search result as basic scheme with a smaller σ . The "rank privacy" is also affected by the value of σ , where "rank privacy" is also proposed by N. Cao et al [11]. Namely, the "rank privacy" at point k is defined as the average rank perturbation $\tilde{p}_k$ for each document d in the returned documents, expressed as $\tilde{P}_k = \sum \tilde{p}_d / k^2$. $\tilde{p}_d$ is denoted as $|r_d - \tilde{r}_d|$, where r_d is the rank number of document d in the returned Top-k documents and $\tilde{r}_d$ is its rank number in the real ranked documents. Fig. 7(b) indicates that with a large σ , the enhanced scheme will enjoy better capacity of protect rank information.



(a)

(b)

Fig.7. With different standard deviation σ selected for the random variable ε , there exists tradeoff between (a) Precision and (b) Rank privacy in the enhanced scheme.

VII. Conclusion

This paper, for the first time, proposes an effective approach to solve the problem of synonym-based multi-keyword ranked search over encrypted cloud data. The main contributions are summarized in two aspects: synonym-based search and similarity ranked search. The search results can be achieved when authorized cloud customers input the synonyms of the predefined keywords, not the exact or fuzzy matching keywords, due to the possible synonym substitution and/or her lack of exact knowledge about the data. The vector space model is adopted combined with cosine measure, which is popular in information retrieval field, to evaluate the similarity between search request and document. Finally, the performance of the proposed schemes is analyzed in detail, including search efficiency and search accuracy, by the experiment on real-world dataset. The results show that the proposed solution is very efficient and effective in supporting synonym-based searching.

The next work is to research semantics-based search approaches over encrypted cloud data that support syntactic transformation, anaphora resolution and other natural language processing technology. The aim is that cloud consumers can search the most relevant products or data by using the designed system.

References

[1] P.A. Cabarcos, F.A. Mendoza, R.S. Guerrero, A.M. Lopez, and D. Diaz-Sanchez, "SuSSo: seamless and ubiquitous single sign-on for cloud service continuity across devices," *IEEE Trans. Consumer Electron.,*vol. 58, no. 4, pp. 1425-1433, 2012.

[2] D. Diaz-Sanchez, F. Almenarez, A. Marin, D. Proserpio, and P.A. Cabarcos, "Media cloud: an open cloud computing middleware for content management," *IEEE Trans. Consumer Electron.,* vol. 57, no. 2, pp. 970-978, 2011.

[3] S. G. Lee, D. Lee, and S. Lee, "Personalized DTV program recommendation system under a cloud computing environment," *IEEE Trans. Consumer Electron.,* vol. 56, no. 2, pp. 1034-1042, 2010.

[4] L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner, "A break in the clouds: towards a cloud definition," *ACM SIGCOMM Comput. Commun. Rev.,* vol. 39, no. 1, pp. 50-55, 2009.

[5] S. Kamara, and K. Lauter, "Cryptographic cloud storage," *FC 2010 Workshops, LNCS 6054,* PP. 136-149, Jan. 2010.

[6] I. H. Witten, A. Moffat, and T. C. Bell, *Managing gigabytes: Compressing and indexing documents and images,* Morgan Kaufmann Publishing: San Francisco, May 1999, PP. 36-56.

[7] S. Grzonkowski, and P. M. Corcoran, "Sharing cloud services: user authentication for social enhancement of home networking," *IEEE Trans. Consumer Electron.,* vol. 57, no. 3, pp. 1424-1432, 2011.

[8] R. Sanchez, F. Almenares, P. Arias, D. Diaz-Sanchez, and A. Marin, "Enhancing privacy and dynamic federation in IdM for consumer cloud computing," *IEEE Trans. Consumer Electron.,* vol. 58, no. 1, pp. 95-103, 2012.

[9] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou, "Fuzzy keyword search over encrypted data in cloud computing," *Proceedings of IEEE INFOCOM'10 Mini-Conference*, San Diego, CA, USA, pp. 1-5, Mar. 2010.

[10] C. Wang, N. Cao, J. Li, K. Ren, and W. Lou, "Secure ranked keyword search over encrypted cloud data," *Proceedings of IEEE 30th International Conference on Distributed Computing Systems (ICDCS)*, pp. 253-262, 2010.

[11] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-preserving multi-keyword ranked search over encrypted cloud data," *Proceedings of IEEE INFOCOM 2011*, pp. 829-837, 2011.

[12] Q. Chai, and G. Gong, "Verifiable symmetric searchable encryption for semi-honest-but-curious cloud servers," *Proceedings of IEEE International Conference on Communications (ICC'12)*, pp. 917-922, 2012.

[13] W. Sun, B. Wang, N. Cao, M. Li, W. Lou, and Y. T. Hou, "Privacy-preserving multi-keyword text search in the cloud supporting similarity-based ranking," *ASIACCS 2013*, Hangzhou, China, May 2013, pp. 71-82, 2013.

[14] P. D. Morehead, *The New American Roget's College Thesaurus in Dictionary Form*, 3rd ed., Signet Press: Seattle, 2002, pp. 10-900.

[15] D. A. Grossman, and O. Frieder, *Information retrieval: algorithms and heuristics*, 2nd ed., Springer Publisher: Berlin, 2004, pp. 18-20.

Biographies

Zhangjie Fu received his BS in education technology from Xinyang Normal University, China, in 2006; received his MS in education technology from the College of Physics and Microelectronics Science, Hunan University, China, in 2008; obtained his PhD in computer science from the College of Computer, Hunan University, China, in 2012. Currently, he works as an assistant professor in College of Computer and Software, Nanjing University of Information Science and Technology, China. His research interests include information systems, protocols, mobile systems and cloud computing.

Xing Ming Sun is a professor in the School of Computer and Software, Nanjing University of Information Science and Technology, China from 2011. He received the B.S.degree in Mathematical Science from Hunan Normal University and M.S. degree in Mathematical Science from Dalian University of Technology in 1984 and 1988, respectively. Then, he received the Ph.D degree in Computer Engineering from Fudan University in 2001. His research interests include mobile systems, applications of networking technology, information systems, cryptography and ubiquitous computing.

Nigel Linge received his BSc degree in Electronics from the University of Salford, UK in 1983, and his PhD in Computer Networks from the University of Salford, UK, in 1987. He was promoted to Professor of Telecommunications at the University of Salford, UK in 1997. His research interests include location based and context aware information systems, protocols, mobile systems and applications of networking technology in areas such as energy and building monitoring.

Lu Zhou received her BE in Software Engineering from Nanjing University of Information Science and Technology, China, in 2012. She is currently pursuing her MS in computer science and technology at the School Of Computer and Software, Nanjing University of Information Science and Technology, China. Her research interests include information systems, protocols, mobile systems and cloud computing.