Turkish Online Journal of Qualitative Inquiry (TOJQI)
Volume 12, Issue 7, July, 2021:1828 – 1839

A Fine-Grained Top K Multikeyword Search Over Large Scale Distributed Data Processing Paradigm

A. Sumathi [*1], S. Nandhinidevi [*2], K. Yasotha [*3]

Abstract

With rapid advance of data mining technologies, large scale data processing is largely considered to be improving the quality of e commerce industries. Despite of popularity, data processing and management has become primary importance to handle the streaming data in e commerce applications. It is one of the large scale and potential application growing around the world. In order to provide an efficient solution to handle large scale distributed application data, Top K Multikeyword search provision has been modelled to provide fine-grained keyword data search over the large scale distributed data at the particular time using searchable models. Nevertheless, Top K Multikeyword scheme already developed is unable to support fine-grained data search in an effective manner and fails to provide accurate data retrieval over streaming data. In order to manage those issues, in this paper, a new fine-grained ranked multi-keyword distributed data search scheme over streaming data has been proposed by developing a data intensive –distributed processing paradigm using deep learning architectures has been proposed for along leveraging efficient ranked multi-keyword search methodologies. In Specific, popular data representation model named as vector space model and the term representation model named as TF*IDF model are combined to form the data index construction, data query vector construction and automatic trap door generation. In this distributed multi-keyword top- k data search mechanism has been constructed in order to improve the data query efficiency for data retrieval by simultaneously supporting dynamic update operations on the distributed servers. A deep learning algorithm has been applied to process the data records, index, query vector. Meanwhile the proposed model ensures accurate data relevance score computation between data index and data query vectors. Finally it enables users to gain a broad scope of access to their retrieved data based on their criteria's, in that high level criteria never requires dynamic strategies for retrieving from the data servers. The proposed scheme for data retrieval has been validated extensively in terms of accuracy and efficiency and its performance of the model has found to be good and excellent model in comparison against other state of art schemes and it offers high level of data retrieval with shortened execution time.

Keywords: *Data Processing, Multikeyword search, Automated Trap Door Generation, Health Records, Data Integrity, Deep learning*

[*1]Associate Professor, Department of Computer Science, Dr. N.G.P Arts and Science College, Coimbatore, Tamilnadu. Email: sumathi@drngpasc.ac.in
[*2] Assistant Professor (SRG), Department of Computer Technology - UG, Kongu Engineering College, Perundurai, Tamilnadu. Email: nandhinidevi.ctug@kongu.edu
[*3] Assistant Professor, School of Computer Science, PPG College of Arts and Science, Coimbatore, Tamilnadu. Email: yasothak.cas@ppg.edu.in

1. Introduction

The data mining is an important data processing domain which is extremely required in e commerce applications. The next generation e commerce infrastructure is expected to manage a category of data recommendation applications with their ability to communicate with each application aspects within and across the organizational boundaries. Many supervised and unsupervised model has been discussed to generate the potential solution but deep learning architecture in the e commerce industry indicating advantage of data processing based technology with faster pace [1]. The fast advent along with evolution of deep learning, both small scale and large scale enterprises are upgraded to process large volume of the data in the external distributed servers. Despite of the various benefits of distributed servers, data management and retrieval brings major concerns. A common approach to manage the data integrity is to generate the index for the data before distributing to server. However, it produces major implication in terms of huge cost on data usability. In order to address the above problem, keyword based search schemes [2] have made specific contributions which enable the provision to store the data to the external servers and execute keyword based data search over data domain.

In this paper, fine-grained Top K Multikeyword search and data management over the distributed data using tree based data search has been proposed by supporting multi-keyword ranked search and dynamic data operation on the distributed data partitioning [3]. It develops deep learning architectures by leveraging the ranked multi-keyword search through inclusion of vector space model and Term Frequency determination conditions. Due to the distributed data structure, the data index construction, query vector construction for data retrieval and automatic trap door generation can be achieved to improve the retrieval efficiency on dynamic update operations [4]. The data retrieving strategies has been set based on conceptual change of the distributed data and its flexible retrieval strategies to the large scale indexed data.

The reminder of this article is organized into various segments as. Discussion of Related work in Section 2, and Section 3 gives a architecture of the proposed model in detail. Section 4 provides the experiments outcomes of the model on various performance analyses. Finally Section 5 provides the conclusion of the work with its future suggestions.

2. Related works

In this section, various Data Searchable schemes enabling the provision to place the data into the distributed server and employ the keyword search over data domain has been summarized on different deep learning primitives below

2.1. Rank-ordered search

In this paradigm, term frequency computation for each data has been carried out to build data indices as traditional retrieval systems for data retrieval has been computed using Euclidean distance computation. Rank-ordered search has been considered a framework towards incorporate the data relevance scoring methods on distributed data and deep learning techniques on providing the efficient and accurate data search utilities to rank-order files in server to its various response to a data query [5].

2.2. Enabling ranked keyword search over distributed data

In this paradigm, distributed data have to be transformed to servers using data placement techniques under searchable retrievals aspects to allow searching the file over distributed data through suggested keywords. In this ranked keyword search, data retrieval through deep learning greatly improves the data usability by enabling search result relevance ranking instead of providing undifferentiated results. Further this model, proposed model provides the effective file retrieval accuracy on exploration of relevance score measure to build a file searchable index [6].

3. Proposed Model

In this part, we briefly introduce the preliminaries which are utilized by the architecture of the particular model defined in depth on several aspects.

3.1. Preliminaries

W-{w1, w2, w3...} denotes set of keywords

m- Number of file keywords

W_q – Cluster Subset of keywords to the data query

f – Collection of plaintext file

C –documents Collection

T – Index for plain text

I – Searchable Index tree for file

Q- Query Vector for keyword set of the particular file

TD – Trap Door for search query

- **Vector space model**

Vector space model is largely used in data representation model to compute the classification and clustering. In this work, Vector space model implemented to generate the keywords on rank it for data search in distributed server. In this process, each file is categorized into a terms. Terms have been represented in frequency table and further it is treated as a vector. In Retrieval Operation, queries have been processed as terms and it is represented as vectors. File Relevance score calculation [7] and cosine similarity has been used as similarity measure for the file retrieval. Cosine similarity measure computes the distance between the query vectors and file vector

Cosine Similarity is given by $\frac{d \times d'}{|d||d'|}$

Where d mentioned as file vector

d' Mentioned as query vector

- **Term frequency Computation**

Term frequency is the count of repeating terms appearing in the processing file. Term Frequency is given by

$$\mathrm{Tf}_{u,wi} = \frac{TF'}{\sum wi \xi W(TF')}$$

Where TF' is the TF value of wi in document f.

W is the set of keywords

- **Inverse Document Frequency Computation**

The inverse document frequency is computed by partitioning the term cardinality of file collection to the number of file containing the similar keyword [8]. It is computed to identify infrequent words more important than frequent words.

Inverse Document frequency is given by

$$idf_i = \log \frac{N}{n_i}$$

Where N is the number of file

n_i is the number that files contain infrequent word i

3.2. System Model

In this section, each entity of the model has been detailed with their functional process carried to achieve some specific goal.

3.2.1. Data Owner

This particular entity to large scale data server contains a collection of file which has to be distributed to the server in index form to enable effective data retrieval on various search to it for effective utilization. The data owner first builds a data Index T from file collection F, and then produces an inverted file collection C for F. Data owner distributes the file collection C and the T data index to the distributed server along the dynamically produced trap door using Term Frequency and Inverse Document Frequency computation. Further, the data owner is responsible for the data update operation of file to the server and file stored in the server. While file updating, the data owner produces the term update information locally and sends it to the distributed server.

3.2.2. Data User

The data user entity is to enable data to the file of data owner based on several functionalities and strategies. Objective function of the data search through keyword is build to data access in the server with broad scope of file access protocol to retrieve data based on their keyword similarity, in that similarity measures never requires any complicated processing as it process using deep learning architectures on the distributed file using map reduce paradigm[10]. The process initiates with providing file query keywords. The query keyword will be calculated against the query vector constructed using VSM and appropriate file will be retrieved based on cosine similarity on utilizing the trap door. It has been calculated using file Relevance score.

File Relevance score is given by

$$\text{File Relevancy score} = \sum \text{Tf}_{u,wi} * idf$$

3.2.3. Distributed Server

The Distributed server stores the file collection C which placed in the dynamic server, Trap door TD formation for query key work and file keyword and index I of the data distributed to the servert. The distributed server computes the retrieval operation upon receiving the query vector from the data user by executing the search over the data index and trap door to return the corresponding file collection on computation of top-k ranked mechanism on it [11]. The figure 1 represents the architecture of the proposed mechanism.

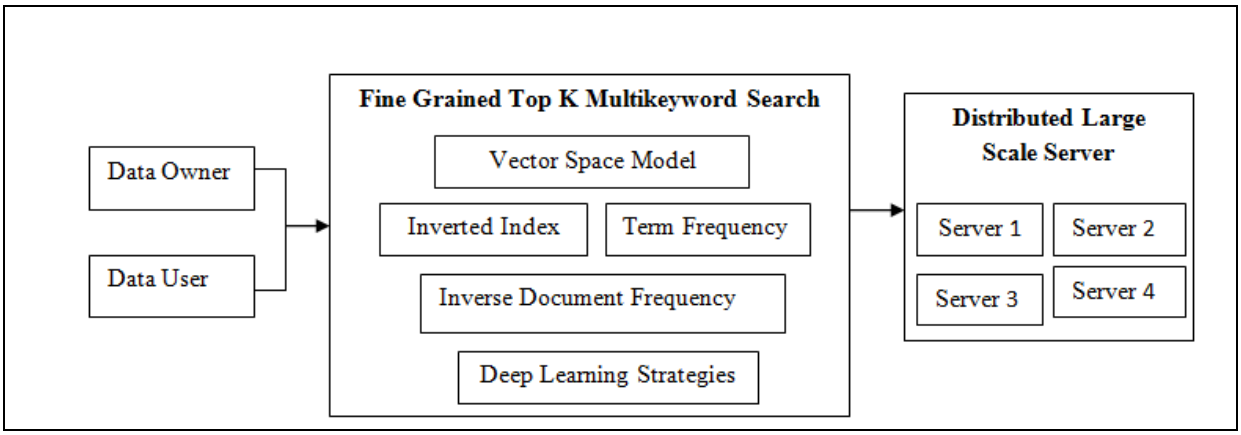


Figure 1: Architecture diagram of the proposed mechanism

Besides, upon receiving the file update information from the distributed server for the data owner content, the server needs to update the data index I and file collection C according to the received information of the request.

3.3. Deep Learning Strategies for Data Retrieval Policy

It is to enable efficient and accurate mechanism to the dynamic multi-keyword file ranked search over distributed data along following features

- Dynamic updates on distributed data: It provides dynamic update on document collections in addition to multi-keyword query and accurate result ranking.
- Data Processing: it is to classify the data in the server through learning architecture with multiple dynamic strategies information about the file collection, the data index tree, and the user query. In addition, index and query confidentiality has to be carried out.

3.3.1. Index Tree Construction

Initially, data index tree T is built on file collection F by using data index tree build function. Secondly the data owner produces two unique vectors Du and $D_u^{|}$ document index vector according to the keyword vector S from the unique words extracted from C.

Data Index Tree generate function is given by

T = generateindextree (F)

generateindextree ()

for each file content F

do
construct subset s for file collection f
Include subset to current set C
end for
while (no of user file subset > server file current set)
do
if (no of file keywords in current set is even or odd)
then
Increase new file keyword set to the insert the file in index server
End if
End for
End while

3.3.2. Automatic Trap Door Generation

The trap door for the data files is produced automatically by employing producing trap door function for server files. It leverages the Term frequency * Inverse Document Frequency file computation functionalities which is detailed in the section 3.1. The function is as follows

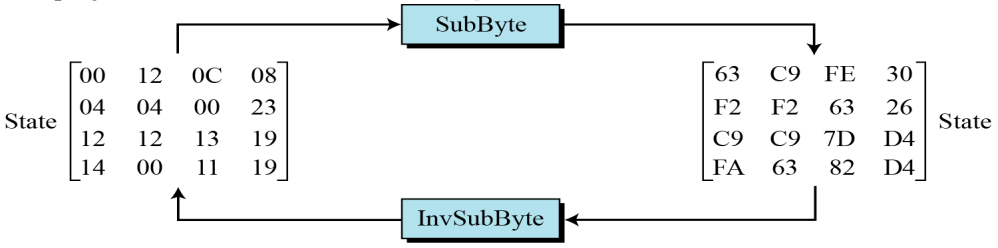
Generatetrapdoor(W)
With keyword set from generateindextree ()
File Keyword set = U
FileKeyword set employs
Compute Term F (U) & Compute Inverse file frequency (U)
Trap door TD = (top K (TF) & Top k (IDF))

3.3.3. Data Distribution using Deep learning

The data distribution using map reduce algorithm [12] named as Distributed Large scale processing paradigm implements following procedure to produce the index for the document collection F.

- Setup. The data has been initializes with distributed properties of the file by generating fileGen, and pre-processes the data file operation C by using generateIndex to produce the Index from the unique words gathered from file collection C
- D Process ()

Data transformation ()
File Word = Collection of terms in file
File Block = Collection of Bytes in the file
File State = Data Matrix representation of Blocks and bytes
Employ Block to State transformation ()



Subbyte
File Byte is converted into hexadecimal digits

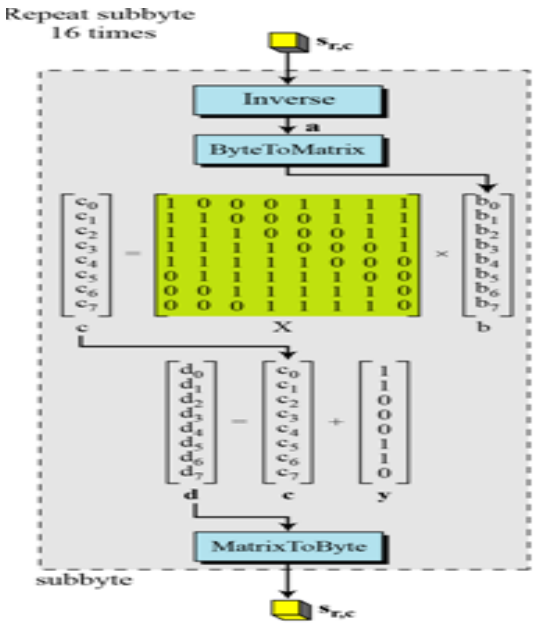
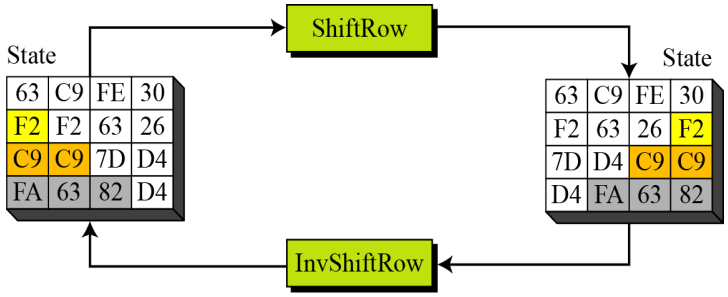


Figure 2: Byte transformation

Shiftrows in matrix ()

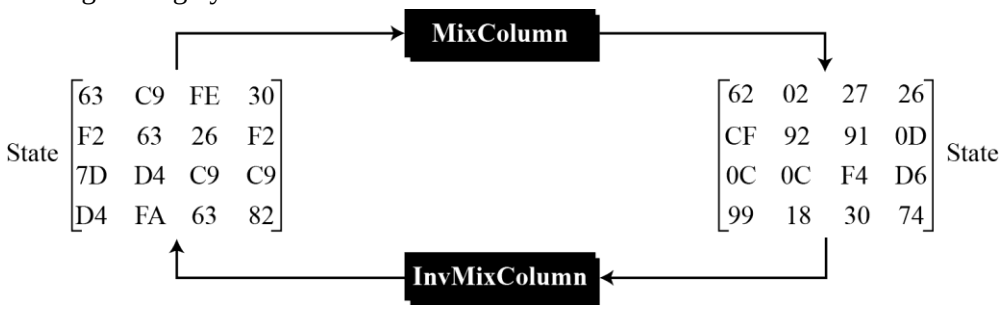
It implements the 1 byte shift |2 byte shift |3 byte

Data Shift in the file based on no of rows in the data matrix generated on right side of the data



Mix column of the data matrix ()

It is interbyte converted that changes the bits inside a byte, based on the bits residing the neighboring bytes of the matrix.



Add round key ()

AddRoundKey adds a round key of the word in the file with each state column matrix

Retrieval process

The Retrieval process implements file State to file block conversion on the following form from data matrix to file block conversion from which file to file byte conversion occurs as presented below

retrieval process ()

Inv Subbyte ()

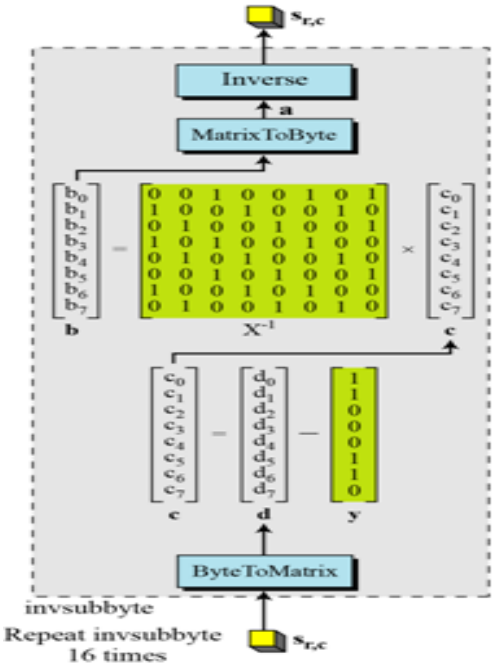


Figure 3: State transformation to block

Inv Shiftrows ()

InvMixcolumn ()

The retrieval form will be obtained on above mention process.

3.3.4. Data access strategies

The data access strategies consist of two strategies which high level constraints and low level constraints. In this higher level data constraints undergoes retrieval with the data index key whereas lower level strategies undergoes retrieval on request of key from the data owner[13]. On obtaining data index key, process undergoes several data conversion stages as mentioned in the above section.

3.3.5. Flexible Revocation policies

The data owner has provided with access permission providing strategies to data user such as hold, accept and allow. In addition, the flexible revocation provision has been provided to deny the access of the user at any time [14].

4. Experimental Results

In this section , we analyse the efficient data processing architecture of the proposed architecture against the various file size is been calculated and detailed in terms of performance measures in tables and charts for various data processing performance measure like processing time, keyword matching ,memory utilization and accuracy[15] .

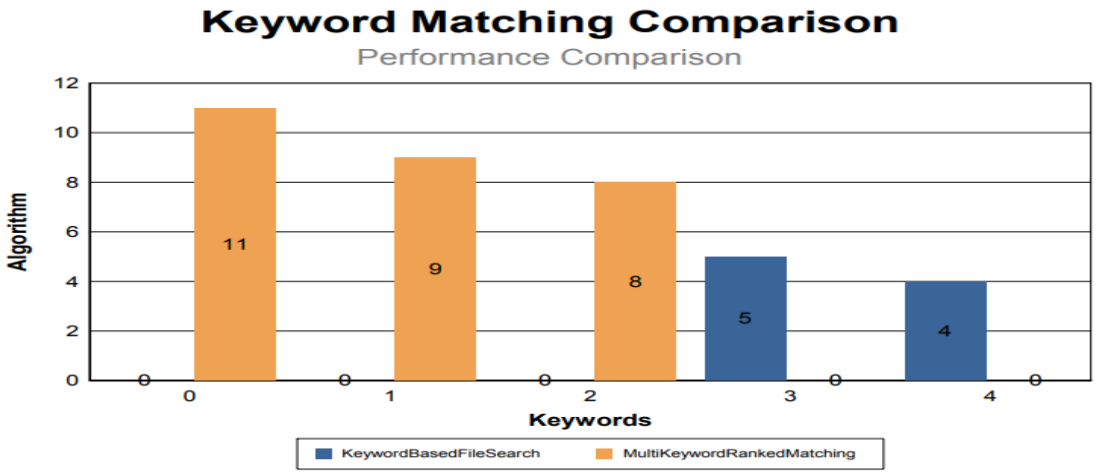


Figure 4: Performance evaluation of keyword matching between keywords generated

The keyword matching utilizes the time in terms of index vector analysis and vector splitting process analysis. The figure 4 and figure 5 represents the performance of the keyword matching on file and trap door in this work.

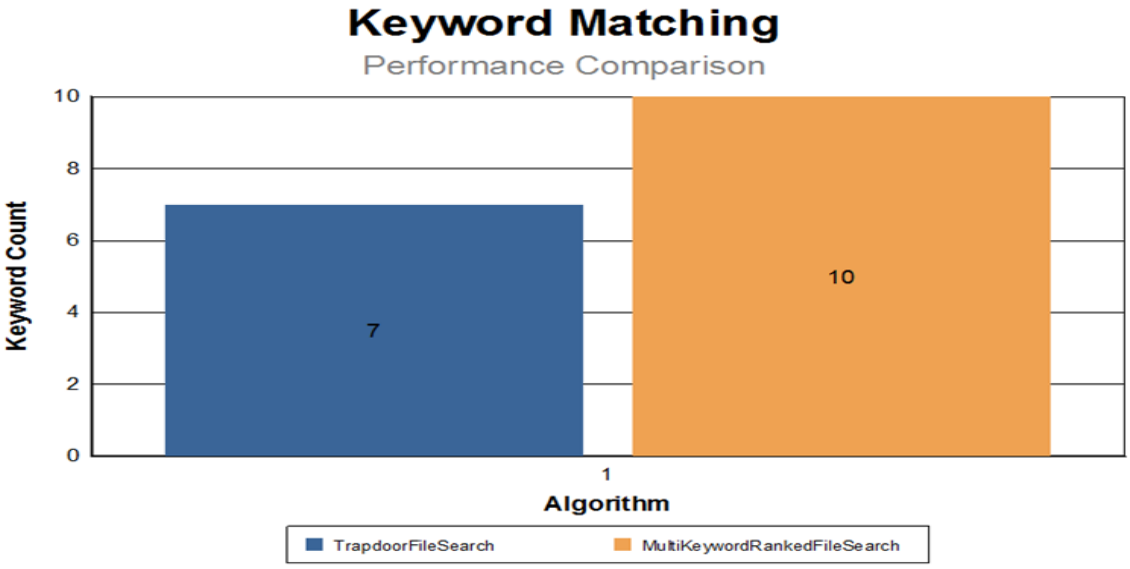


Figure 5: performance evaluation on Keyword matching between proposed and existing approaches

The memory utilization is referred in term of the index tree construction consumption as it is relatively much time at the data owner side, it is noteworthy that this is a one-time operation. The memory utilization for decryption on proposed model is less compared to state of art approaches. The figure 6 represents the performance of the memory utilized for retrieval process.

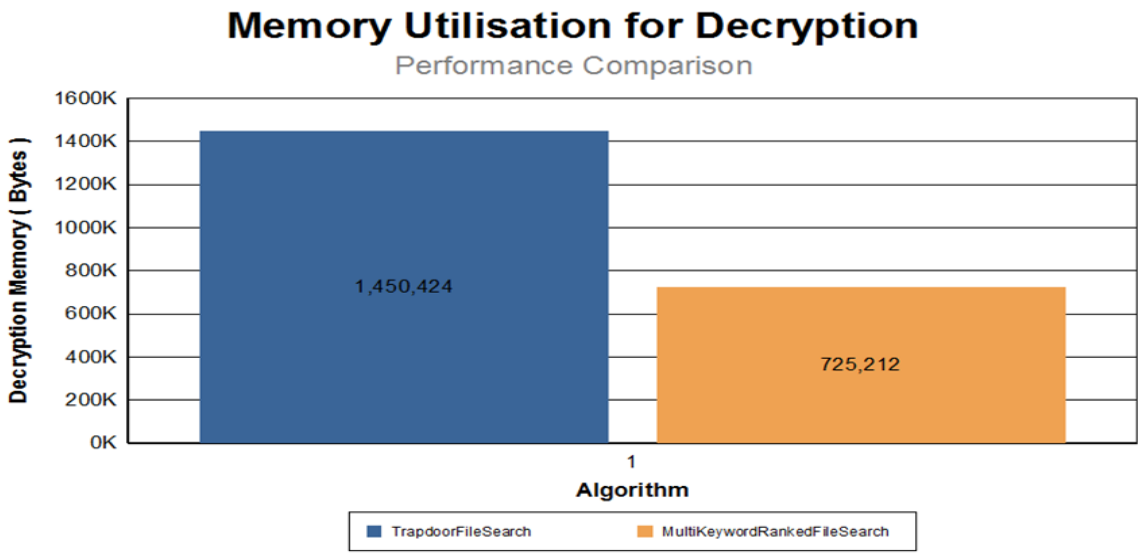


Figure 6: Performance Evaluation of the Memory utilization on retrieval process on proposed model and existing models to various file sizes

The search precision of scheme is affected by the dummy Keywords. If keyword standard deviation is used set random variable, it is supposed to obtain higher precision. The figure 6 provides the precision, recall and f measures values for the proposed and existing models on various data sizes of the distributed data records.

Precision = number of real top-k documents/ retrieved k documents

Recall = rank number of document in the retrieved top-k documents/ real rank number in the whole

Ranked results

F measure = harmonic Mean of precision and Recall

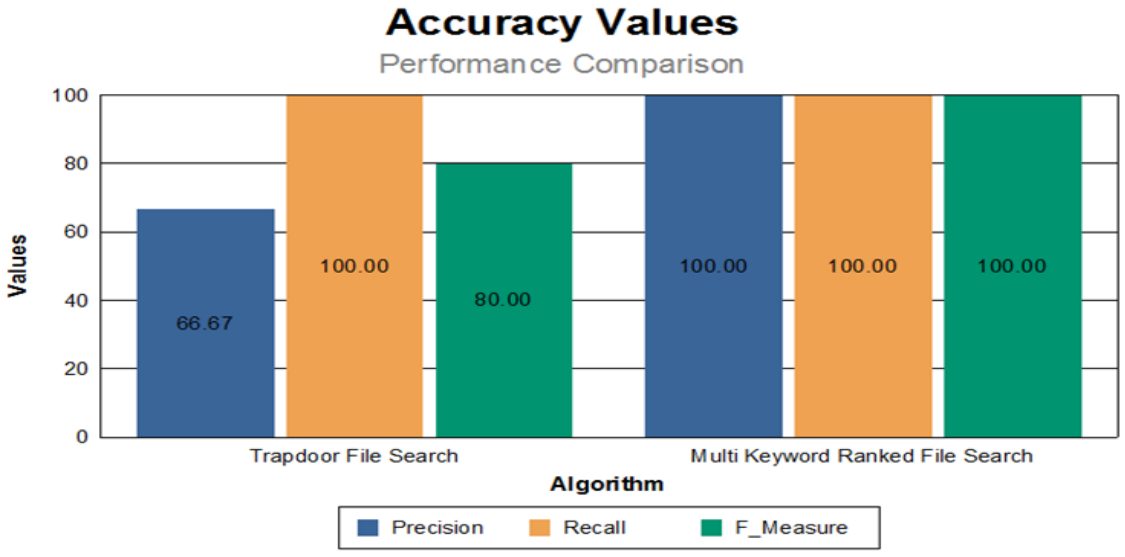


Figure 7: Performance Evaluation of the Accuracy measures on proposed model and existing models to various file sizes

Figure 7 represents the performance evaluation of the proposed model in terms of accuracy. Table 1 represents the performance evaluation of the proposed model against the existing model.

Table 1: Performance Evaluation of the Security Model

S. No	**Technique**	**Memory Utilization in mb**	**Decryption time in ms**	**Accuracy**
1	Fine grained Top K ranked multikeyword search	90mb	21600	99.25
2	Ensemble Scheme	120mb	24000	98.56

Efficiency is proportional to the size of dictionary when the document collection is fixed as deletion of a document takes nearly logarithmic time with the size of document collection [16, 17].

5. Conclusion

We have designed and implemented a fine grained top k ranked multikeyword search on distributed data towards dynamic updates of the outsourced data. it supports not only the multiple keyword search but also the dynamic updates of the outsourced data , flexible learning of the data under various category. The automated trap door generation established using TF*IDF model. The efficient data index has been achieved to obtain the better retrieval efficiency to user queries. In addition, query vector construction in order to improve the query efficiency. Furthermore relevance score calculation between encrypted index and query vectors provide high accurate results. Finally data owner provided with flexible retrieval provision. Experimental results demonstrate the efficiency and accuracy of our proposed scheme on various measures.

References

[1] S. Yu, K. Ren, and W. Lou, “FDAC: Toward fine-grained distributed data access control in wireless sensor networks,” IEEE Transactions on Parallel and Distributed Systems, vol. 22, no. 4, pp. 673–686, 2011.

[2] S. Chatterjee, S. Roy, A. K. Das, S. Chattopadhyay, N. Kumar, A. G. Reddy, K. Park, and Y. Park, “On the Design of Fine Grained Access Control with User Authentication Scheme for Telecare Medicine Information Systems,” IEEE Access, vol. 5, no. 1, pp. 7012–7030, 2017.

[3] Q. Chai, G. Gong, Verifiable symmetric searchable encryption for semihonest- but-curious cloud servers, in: Communications (ICC), 2012 IEEE International Conference on, IEEE, 2012, pp. 917–922.

[4] C. Wang, N. Cao, K. Ren, and W. J. Lou, “Toward Efficient Multi-Keyword Fuzzy Search Over Encrypted Outsourced Data With Accuracy Improvement,” in IEEE Transactions on Parallel and Distributed Systems, vol. 23, No. 8, pp. 1467-1479, 2012.

[5] A. Swaminathan, Y. Mao, G.-M. Su, H. Gou, A. L. Varna, S. He, M. Wu, and D. W. Oard, “Confidentiality-preserving rank-ordered search,” in Proc. ACM Workshop Storage Security Survivability, 2007, pp. 7–12

[6] C. Wang, N. Cao, K. Ren, and W. Lou, “Enabling secure and efficient ranked keyword search over outsourced cloud data,” IEEE Trans. Parallel Distrib. Syst., vol. 23, no. 8, pp. 1467–1479, Aug. 2012.

[7] J. F. Wang, X. F. Chen, J. Li, J. L. Zhao, and J. Shen, “Towards achieving flexible and verifiable search for outsourced database in cloud computing a¸ î,” in Future Generation Computer Systems, vol. 67, pp. 266-275, 2016.
[8] B. Wang, S. Yu, W. Lou, and Y. T. Hou, “Privacy-preserving multikeyword fuzzy search over encrypted data in the cloud,” in Proc. IEEE INFOCOM, 2014, pp. 2112–2120.
[9] Y. C. Chang, M. Mitzenmacher, Privacy Preserving Keyword Searches on Remote Encrypted Data, in: International Conference on Applied Cryptography and Network Security, Springer, 2005, pp. 442-455.
[10] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, “Searchable symmetric encryption: Improved definitions and efficient constructions,” in Proc. 13th ACM Conf. Comput. Commun. Secur., 2006, pp. 79–88.
[11] Z. Wan, J. Liu, and R. H. Deng, “HASBE: A hierarchical attribute-based solution for flexible and scalable access control in cloud computing,” IEEE Trans. Inf. Forensics Security, vol. 7, no. 2, pp. 743–754, Apr. 2012.
[12] Q. Zheng, S. Xu, G. Ateniese, VABKS: verifiable attribute-based keyword search over outsourced encrypted data, in: INFOCOM, 2014 Proceedings IEEE, 2014, pp. 522–530.
[13] J. F. Wang, X. F. Chen, and J. Li, “Verifiable Search for Dynamic Outsourced Database in Cloud Computing,” presented at the International Conference on Broadband and Wireless Computing pp. 568-571, 2015.
[14] L. X. Chen, and N. Zhang, “Efficient Verifiable Multiuser Searchable Symmetric Encryption for Encrypted Data in the Cloud,” presented at the International Conference on Security and Privacy in New Computing Environments pp. 173-183, 2016.
[15] C. Guo, X. Chen , Y. Jie , F. Zhang, M. Li, B. Feng, Dynamic Multiphrase Ranked Search over Encrypted Data with Symmetric Searchable Encryption, IEEE Transactions on Services Computing, vol. 30, no. 10, pp. 1-12, 2017.
[16] E. Boopathi Kumar, V. Thiagarasu, “Segmentation Using Fuzzy Membership Functions: an Approach”, International Journal of Computer Sciences and Engineering, pp. 101 - 105, Volume 5, Issue 3, 2017.
[17] E. Boopathi Kumar, V. Thiagarasu, “Segmentation using Masking Methods in Color Images: an Approach”, International Journal of Engineering Sciences & Research Technology, pp. 104 - 110, Volume 6, Issue 2, 2017.
[18] E. Boopathi Kumar and M. Sundaresan, “Fuzzy Inference System based Edge Detection using Fuzzy Membership Functions”, International Journal of Computer Applications, Volume 112, Issue: 4, February 2015.