Smart Search Over Encrypted Cloud Multimedia Data By using Machine Learning

MSc Research Project
Cloud Computing

Husna Mehwish
Student ID: x21168385

School of Computing
National College of Ireland

Supervisor: Aqeel Kazmi

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Husna Mehwish
Student ID:	x21168385
Programme:	Cloud Computing
Year:	2022
Module:	MSc Research Project
Supervisor:	Aqeel Kazmi
Submission Due Date:	01/02/2023
Project Title:	Smart Search Over Encrypted Cloud Multimedia Data By using Machine Learning
Word Count:	7076
Page Count:	23

Signature:	Husna Mehwish
Date:	1st February 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Smart Search Over Encrypted Cloud Multimedia Data By using Machine Learning

Husna Mehwish
x21168385

Abstract

Many people are looking forward to using the cloud to save their data. This is because of the progress in technology in the computing industry. If you want to outsource a sensitive amount of data into the CS, it is best to encrypt the personal data first, to protect its security. However, current search techniques are not much efficient and they don't support documentation and image search simultaneously. Data in the encrypted form is difficult to search because encryption scrambles the content. Ranked keyword searches are receiving a lot of attention because they bring the most relevant data quickly. The present ranked keyword search, however, does not support other multimedia materials and focuses only on text documents and their content. And searching for encrypted multimedia data files on the cloud is a very challenging task. To solve this issue we used a search scheme based on machine learning. The k-mean and b-tree algorithms are used for clustering. Users can add custom tags for all the files he is uploading. We also provided search specifications such as AND, NOT, synonym search, etc. to make the search easier for the user. Without any compromise on accuracy, we used the Customized Query Execution(CQE) algorithm to reduce the search complexity. Our experimental results are also presented in this paper based on accuracy and latency. Evaluation of our scheme proves that the scheme is protecting index privacy and keywords. Experimental evidence from using a real-world dataset has revealed that our plans are accurate and feasible in realistic applications.

1 Introduction

As computers become more commonly used, they have become the primary source of personal data. This results in an increase in the number of files downloaded, saved, and shared with others. And as technology has improved and storage capacities have increased, people find it difficult to store large amounts of information. This is a significant problem because the capacity of a hard drive outside a PC can be very high. Before the discovery of the cloud, people had to save their data locally. This is expensive because you need a lot of storage devices to save your data rather than saving it online. With a vast amount of people willing to shift their data storage to the cloud, everyone will be able to take advantage of the cost benefits and increased levels of security. The on-demand model is quite new, and thus so far it seems convenient. You just pay for what you need and allow the app to connect to the network. Honestly, it is much more affordable than buying tons of software licenses Vaquero et al. (2008). The cloud's flexibility and simplicity make it an ideal choice for organizations to store data. A managed cloud solution eliminates the stress for operations and IT departments because it only requires them to focus their efforts on other aspects of the business.

We need to remember that cloud storage still has risks despite its popularity. However, there are understandable concerns about storing sensitive data in the cloud due to the rise in data breaches that have been occurring. Many people are concerned about storing their data in the cloud for positive reasons. It is important to encrypt the data before uploading sensitive information to the cloud. For a lack of trust in the cloud, there may be different ways to store files depending on the company's security requirements. Song et al. (2000) introduced the idea of searching encrypted cloud data. Unfortunately, Many proposals have been made on how to search for data after a specialist is performed. However, these schemes incur additional storage costs and decrease accuracy.

Recently, Miao et al. (2022) proposed a machine-learning technique to search encrypted data. He proposed two ML methods: ML-RKS reduces search complexity by providing efficient search results, and ML-RKS Plus is used for dynamic updates with forwarding security. If you want to make any changes to the data, you will be able to update it since you still have the token. However, this solution supports only certain types of data. Although they say that it is for all types of data sets, they do not function properly with anything that is not text-based. This is not an effective solution to the problem because most organizations have access to employees' data. This could be their driver's licenses, pictures of their ID cards, photos, or video footage on the job or at work. Employees also want to save images that are meaningful for various reasons, and there should be a system where users can search for images simply by adding tags for specific events. This would allow people to save data without having to worry about them being private.

To solve these problems, in this study, we propose a solution. We used a machine learning-based search scheme and designed a customized query execution algorithm that aims to make it more efficient. K-mean and b-tree are used for clustering. A PDF reader will be able to read PDF files, which is effective because it reduces search complexity without decreasing accuracy, and we used it to reduce search complexity. Our work has made the following major contributions:

- We implemented the k-means-based algorithm to design a way of finding the desired information in encrypted cloud multimedia files.
- We designed a scheme in which someone seeking information fires a query by setting the priority of the keywords. The word ranked highest will be searched for first.
- The files can also be in PDF form, for reading PDF documents we used a PDF reader which can read the document's content.
- A data owner can also take advantage of custom tags to label the documents, images, etc. that they uploaded. This way it becomes easier to search and find what they are looking for based on keyword search.
- The DU could search for the exact keywords he entered in the search query. For example, If you have 5 keywords, and you want to search the document having all 5 keywords, only then the data in that document will be shown. If all 5 words are not in the document, then it will return zero results.

- Data user can also search by mentioning he does not want this word to be searched for his required documents. for example, if he wants cloud computing papers but not including fog, etc., he will not get any paper having fog-related data in it.

- If the user doesn't know what is the exact keyword for the document he can still use the synonym search for that. For example, if he wants to search the pay slips and he doesn't know what can be keyword he can search for wages, salary, etc., and will get the required document.

- We analyzed our system by performing experiments using real-world datasets to evaluate the accuracy and latency.

The remainder of the paper is structured as follows: Section II examines previous research on the search for encrypted cloud data. Section III describes the system methodology. Section IV will present the architectural specifications. Section V will go over all of the implementation and work that was completed. Section VI will evaluate the design and discuss the results, and Section VII will conclude the paper and also explain any future work.

2 Related Work

Due to the growth of computing applications and the internet, all types of data have been growing exponentially, resulting in a gradual increase in user requirements for data access and storage capacity. Meanwhile, cloud computing, a new computing model, meet the user's expectations by offering them on-demand high-quality services and network access in a pay-as-you-go manner. Due to the amazing flexibility and simplicity of the cloud, companies and individuals would like to store their local and private data in a cloud server so they can stop worrying about data management. Despite the significant advantages, storing data on the cloud server removes the data from the data owner's physical control raising questions about the security and privacy of the data. Is the data stored correctly or is the data secured or not?

To guarantee data security, encryption is the best option. But there is a big question when we are saving data in the encrypted form then how will we search for encrypted data in a server? We know it is impossible to search and share encrypted data. Song et al. (2000) was the first one who defined the concept of encrypted search. They think that keeping encrypted data on the cloud can reduce security risks as well as local storage costs. However, with some benefits, they are also worried about the leakage of private information because storing data in the cloud makes the data out of the user's physical control and there is a good chance for hackers to steal the important information.

In recent years most researchers/specialists have proposed their ideas about encrypted search. They worked on different types of searches such as single-keyword search, multi-keyword search, fuzzy keyword search, phrase search, semantic search, etc at the same time they also proposed different techniques such as bloom filter, Machine learning, novel features matching, etc to make the search for encrypted data easy for the data users.

I have divided my research into two parts because, to the best of my knowledge, the researchers have worked on documented data and on image data separately this is why we need to divide our research into two parts namely documented or text data, image data, etc.

2.1 Documented Or Text Data

2.1.1 Single-Keyword Search Schemes:

For the first time, Boneh et al. (2007) discussed the issue of untrusted routing and developed a public-key encryption system that permits PIR (private information retrieval) on documents that have been encrypted. This method can address the issue of searches without disclosing user query information. But this solution is computationally very expensive and everyone cannot get it. Recently Wang et al. (2010) investigated secure ranked search over encrypted cloud multimedia data. They developed a crypto primitive OPSE technique and drive efficient one-to-many order-preserving mapping functions. which was secure and efficient in providing a ranked and efficient keyword search. However, this scheme is only based on a single keyword search, and with the popularity and increasing amount of data, there is a need for multiple keyword search methods. single keyword search schemes can't meet the user's actual requirements.

2.1.2 Multi-keyword search schemes:

To facilitate more spatial queries, multi-keyword or conjunctive search schemes over encrypted cloud data have been proposed. multi-keyword search allows users to input multiple keywords in a request to query to get the required results containing these words. Cui et al. (2016) use a linear secret-sharing scheme to achieve efficient search as well as data access control over encrypted data but the scheme was only working for AND and OR gates. Ali and Lu (2016) proposed a method for symmetric encryption that facilitates connecting keyword searches, and this method does not have to identify the keyword's precise location. Additionally, bloom filters and pseudo-random functions are employed to assure index security in the method. This scheme's efficiency was just 70 percent which is not good enough for the user. Yang and Ma (2015)offered a new approach to searching for conjunctive keywords for systems for electronic health records. This approach encourages the automated delegating revocation, the first searchable encryption method that permits proxy re-encryption. But there are still many deficiencies, especially in the tracking and revocation of malicious users.

In another research, Fu et al. (2013) first proposed a multi-keyword ranked scheme based on a synonym search for encrypted cloud multimedia data. where the users can also perform a search by sending the synonyms of the predefined keywords, and they will get the required results. But they didn't support spelling mistakes. Later Fu, Wu, Wang and Ren (2017) proposed central keyword-based semantic search schemes. An effective trade-off between search functionality and efficiency was achieved by extending the primary search term but this scheme only supports documented search and not supports image search. Metkari and Sonkamble (2016) suggested a system that supports semantic and multi-keyword searches at the same time. For the encryption, he was using the KNN algorithm. Multi-threading technique was used to parallelize the indexes for their fast generation of them. Xia et al. (2015a) suggested a system that allows for the exact match of the multi-keyword searches, prioritized by relevance. Their scheme also can add or remove anything from the documents. These schemes can achieve more accurate results. However, updating cost of the above-mentioned schemes is excessive like if the data owner wants to update any documents then he needs to reconstruct the searchable index trees and all the encrypted index vectors to guarantee the cloud server works normally.

2.1.3 Fuzzy keyword search schemes:

For Tolerating keyword spelling mistakes in search queries, specialists proposed some techniques to search fuzzy keywords. So, when data user mistakenly adds some wrong words about their required documents they still get the required result. Or when the exact query match fails, it will return the closest possible file match. Li et al. (2010) was the first who proposed the fuzzy keyword search scheme. They used edit distance in this scheme. Fuzzy keyword sets are created using a wildcard-based technique to measure the similarity between terms. But this scheme only supports a single keyword and gives the documents not the multi-keyword. Yang and Ma (2015) designed a scheme of fuzzy keyword search over a large number of encrypted data sets. In the scheme, they used an LSH and other hashing algorithms to improve the efficiency of the query and reduce the utilization of space. This system supports valid search authorization for a limited period. Based on Wang et al. (2012), and Fu, Wu, Guan, Sun and Ren (2016) proposed an efficient multi-keyword fuzzy ranked search scheme, this scheme is able to search for more common spelling mistakes. Chen et al. (2019) suggested a strategy for multiple data owners. they coupled LSH and Bloom Filter techniques to enhance the fuzzy search. Zhang et al. (2021) designed a fuzzy keyword search scheme under a hybrid cloud architecture. But returns many irrelevant results.

2.1.4 Phrase search schemes:

Recently, some scholars also have proposed phrase search schemes for encrypted cloud data. unlike supporting multi-keyword or conjunctive keyword, fuzzy keyword search schemes, and phrase search schemes requires consideration of each keyword information, like semantics, order, location information, etc. Poon and Miri (2015) suggested an encrypted search scheme that supports both the multi-keyword search and the phrase search at the same time. In order to reduce the storage cost that scheme statistically considers the properties of natural language. But this scheme was not efficient enough for the phrase search so in their next research Poon and Miri (2017) they proposed a phrasing technique based on bloom filter, which supports phrase-independent query. Which supports a single phrase only. Tang et al. (2012) suggested a symmetric searchable encryption model then they suggested a symmetric encryption phrase search technique that can perform secure and effective phrase search over encrypted data. This necessitates two rounds of communication between the server and the client, which takes far too long to return results. Zittrower and Zou (2012) developed a method for phrase and multiple-keyword searches on encrypted data sets. A thoroughly encrypted search feature is made possible by keeping the keyword location information. There are too much of its information is available to the cloud server.

2.1.5 Semantic search schemes:

To meet the needs of the users, some specialists have worked on semantic-based searches. In this search, a user can search for words with the same meaning and will get the required results. The research purpose is to give the user more effective results based on their queries. Wang et al. (2012) suggested a method based on the trie-traverse searching index. which only supports the single keyword which does not work with the increasing amount of data. Fu et al. (2015) proposed a simhash-based similarity search method for encrypted documents. In this scheme, the data user will submit the query and will get related encrypted documents that are stored in the cloud. Because of the poor sensor communication technology, it is tough for the Sensor cloud to assure reliable connections and real-time facilities. Then, Fu, Huang, Sun, Vasilakos and Yang (2016) and Fu, Huang, Ren, Weng and Wang (2017) introduced a conceptual graph, and then Fu, Huang, Ren, Weng and Wang (2017) proposed a search scheme named content-aware, which is used to make the semantic search more smart and secure and designed new methods that convert conceptual graphs to vectors. They can extract the most important and simplified topic sentences from documents by using them. They were not efficient enough to return good results.

2.2 Image database

Hu et al. (2016) proposed a practical scheme for searching image data. Their novel technique SIFT was made by using secure protocols BSMP and BSCP. This technique was mainly based on the extraction of features of the image which was successfully done on a cloud environment without exposing the image data of the data owner. For using this scheme user needs to perform pre-computation of queries which increases computation cost. Shashank et al. (2008) proposed a private content-based image retrieval scheme(PCBIR) which was suggested to protect the privacy of the image database, but it was exposing the unencrypted image database directly to the server. Lu et al. (2009) was the first one who presented the CBIR scheme for the encrypted images. In order to compare two sets of visual words and their accompanying images, the authors first extract the words used to describe the images. They then compute the Jaccard similarity between the two sets of extracted words. The information contained in visual words is protected using the order-preserving encryption and min-hash method. Search efficiency still remains the main issue. This scheme failed to account for dishonest query users who might illegally distribute the retrieved images.

Using two parallel CNNs Alzu'bi et al. (2017) proposed a new bilinear CNN-based architecture as a feature extractor. Convolutional layers are used to retrieve image attributes at various scales and locations. Although it was end-to-end training, the accuracy of feature learning was not good enough. Mistry et al. (2018) suggested a hybrid feature-based efficient CBIR system by using various distance measures. Color and edge directivity descriptor features are used to improve image features. so that they can provide user efficient CBIR system. This system was not efficient enough to extract the color feature.

Another research Guo et al. (2020) suggested an innovative and useful classification and retrieval method (for pandemic situations like covid-19) for locating pertinent cases in encrypted images. Mainly they constructed a CNN framework that allows the data to classify and search for secure, large-scale images based on content with homomorphic encryption. Janani et al. (2021) proposed a methodology to execute the secure encryption of pictures and search for medical images efficiently over the encrypted cloud image data. They also introduced a recovery mechanism ROI in order to recover the obtained data. And they introduced a scheme to locate authorized users known as copyright protection. The avalanche effect, which occurs when a small modification to the original image or video causes a significant change in the hash that emerges, is what distinguishes this technique.

To the best of my knowledge, specialists have worked on different techniques to search for encrypted data over the cloud most efficient and secure for the data owner and the data user so that they can outsource their data without any problem. But when the specialists worked or suggested any technique their main point of focus is the documented or text data Or they worked only on image data. what if the data owner needs to store the data in documented and image form simultaneously? Here we can take the example of a doctor willing to save his patient's data on a cloud server. But the data is in different formats such as the patient's registration form, after checking he can have the patient's X-ray reports, insurance documents, etc. In short, we can say that the data is in text form, image form, and video form. So, there is a need to search each type of document by using the same technique to make the work easier in this chaotic world. At the same time, the data owner can add custom tags for his documents which can make his work easier, he just needs to input those tags and will get his required documents. This is the reason behind the idea of working on all types of data so the data owner and data user can access it easily.

3 Methodology

The main objective of this system is to provide a robust search for encrypted multimedia data stored in the cloud. This section presents all explanations and supporting data required to demonstrate the superiority of this particular research strategy. The client-server architecture underlies our system. The three main entities in this system are the cloud server, Data Owner, and data user.

The research methodology is mainly divided into two parts.

- Indexing
- Searching

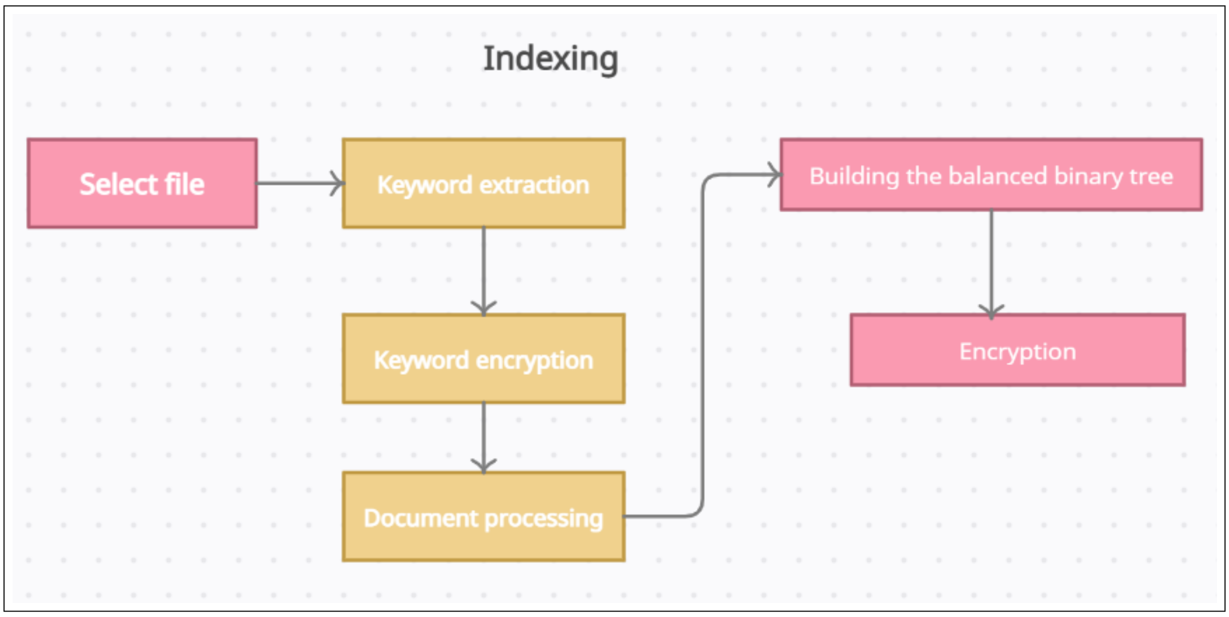

Figure 1: **Block diagram for indexing**

3.1 Indexing

We needed to create indexes of the keywords for search purposes. We divided the indexing part into the following subsections: keyword extraction, Multimedia content keywords handling, keyword encryption, document processing, building a balanced binary tree, and data encryption.

3.1.1 keywords extraction

To create relevant keywords for the documents, we must fetch the keywords from the documents. For this purpose, we used TF*IDF. TF*IDF is a measure that can be used to retrieve important words from documents. Before using TF*IDF, we used stop words to remove all the extra words such as a, an, and, etc. At the same time, we used lemmatization to remove all the words that are grouped into different inflected forms of words and convert them to their root form, having the same meaning. So after removing all the extra words from the documents now we have a document with most of the important words so by using TF*IDF we will fetch the keywords.

3.1.2 Multimedia content keywords handling:

For a text document, we can easily read the document and retrieve the keywords. However, we cannot read the content from pdf, images, and videos. We used a pdf reader to read pdf files. And for the image and video files, we provided the user with the facility to add custom tags. After uploading the images or video files, the user can add custom tags to a specific file. Then, we will create the indexes with these custom tags. This facility can also be applied to text-based documents. After extracting the keywords from the documents, we allow the end user to edit those keywords and add more keywords for convenience.

3.1.3 Keyword Encryption

We encrypted the keywords beforehand to ensure maximum security. we cannot use them as they are so at first we encrypt them. For that purpose We used the SHA algorithm because it can't be decrypted, making it a one-sided encryption.

3.1.4 Document processing

This involves the processing of the entire document. We extracted important keywords from the documents. After the extraction of keywords, we found appropriate clusters for documents. which cluster group is suitable for a specific document type. We are grouping the documents with the help of keywords and we have more than one cluster so it divides our data into multiple groups. It will be helpful for further searching. Once we have fetched all keywords from the documents, it is now time to generate the indexes for the keywords.

3.1.5 B-Tree building

To lessen the complexity of the search without reducing accuracy, our basic plan is to use the k-means clustering algorithm Likas et al. (2003), and Miao et al. (2022). The entire file is split up into k clusters. Presented a group of file vectors $\{d_1, d_2, \ldots, d_n\}$

for the documents $F\{f_1, f_2, ..., f_n\}$, DO at a very first point chooses the k initial clusters $k\{k_1, k_2, ..., k_n\}$then he will add all file vector d_y to the i-th cluster in such a way that the relevance score for S$(d_y, c_i)$is as high as possible. Finally, DO terminate when all k clusters are achieved. He again adds all files d_y to the i-th cluster such that the relevance score for (d_y, c_i) is as high as possible. Finally, obtain the k clusters when all files are added to clusters. we will use a balanced binary tree to get a sublinear search time. Xia et al. (2015b) to build the index structure, we construct the i-th balanced binary tree in Algorithm 1

```
Algorithm 1: Building balanced binary tree
   Input: fileset $F\{f_1, f_2, \ldots f_n\}$ from cluster Ci
   Output: Binary tree B-tree(F)
 1 Read file vectors $\{d_1, d_2, \ldots, d_n\}$ for documents $F\{f_1, f_2, ..., f_n\}$
 2 Let h = $2^{n+1} - 1$
 3 Iteration = $2^n - 2^{h-1}$
 4 for i=1 to iteration do
 5     Create node = $n_1$
 6     Node [value] =$v_1$
 7     Create immediate node as D[i- iteration/2] as inode
 8     inode [ value] = max(node [i].value, node[i+1].value - node
         [i+1].value)
 9     Connect inode and node
10 end
11 return tree
```

3.1.6 Data encryption

The data used in the proposed system was encrypted. Therefore, data must be encrypted. We used the AES algorithm to encrypt the data. The original order of this information is lost in AES encryption. Only equality comparisons are supported by AES.AES is a type of encryption algorithm for symmetric data keys. The exact key is used for the encryption and decryption of the data every time. It's deterministic so that the same ciphertext will always be produced from the same plain text and key, over and over again.

3.2 Searching

This is the main component of the proposed system. To obtain the required document efficiently and effectively without losing any privacy, we have divided this part into several sections, namely query generation, Searching strategies, query execution, search over the binary tree, decryption, etc.

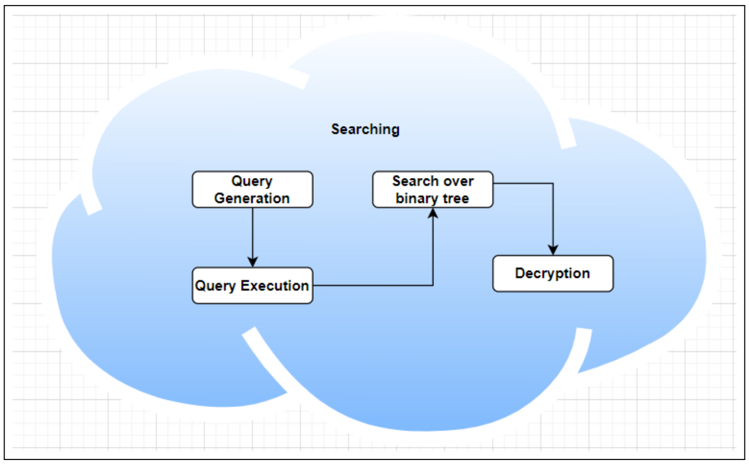

Figure 2: **Block diagram for searching**

3.2.1 Query generation

In this section, the user will fire a query to the cloud server with a priority value. With the query word for example "cloud" the data user can add priority by adding " cloud:0.07" so the server checks the priority value and will search with the highest priority. When the data user adds the search query, it can be any type of query because, with all the other search options, we have the option of synonym search as well because the user does not need to remember all the keywords, so he can add keywords that are not the root form of the word, so how will he get any clusters matched as we are processing lemmatization and stop words to create the document before applying TF*IDF? To avoid these situations, in this system, when the data users add any query, lemmatization will be performed on that query as well to generate the proper keyword that will be used to find the related clusters.

3.2.2 Searching Strategies:

We have provided different search facilities for the data user to make his search more efficient and accurate. we have provided search operators such as NOT, AND, and synonym search, etc. we used the WordNet library to get the synonym of the queried words. If the data user doesn't remember the exact keyword so by adding any word he can apply for a synonym search and will get all the related documents for the queried word. And if the user wants the most specific document then he can select AND operator to get the exact result.

3.2.3 Query Execution

In this section, we describe the execution of the query. It will check which cluster belongs to this query and will search only in that cluster.

3.2.4 Search over Binary tree

Figure 3 shows how we will search the binary tree Miao et al. (2022). we have a query q = (0.1,0.5,0.2) . we have given the keyword sets $\{w_1, w_2, w_3\}$ and a set of files $\{v_1, v_2, v_3\}$

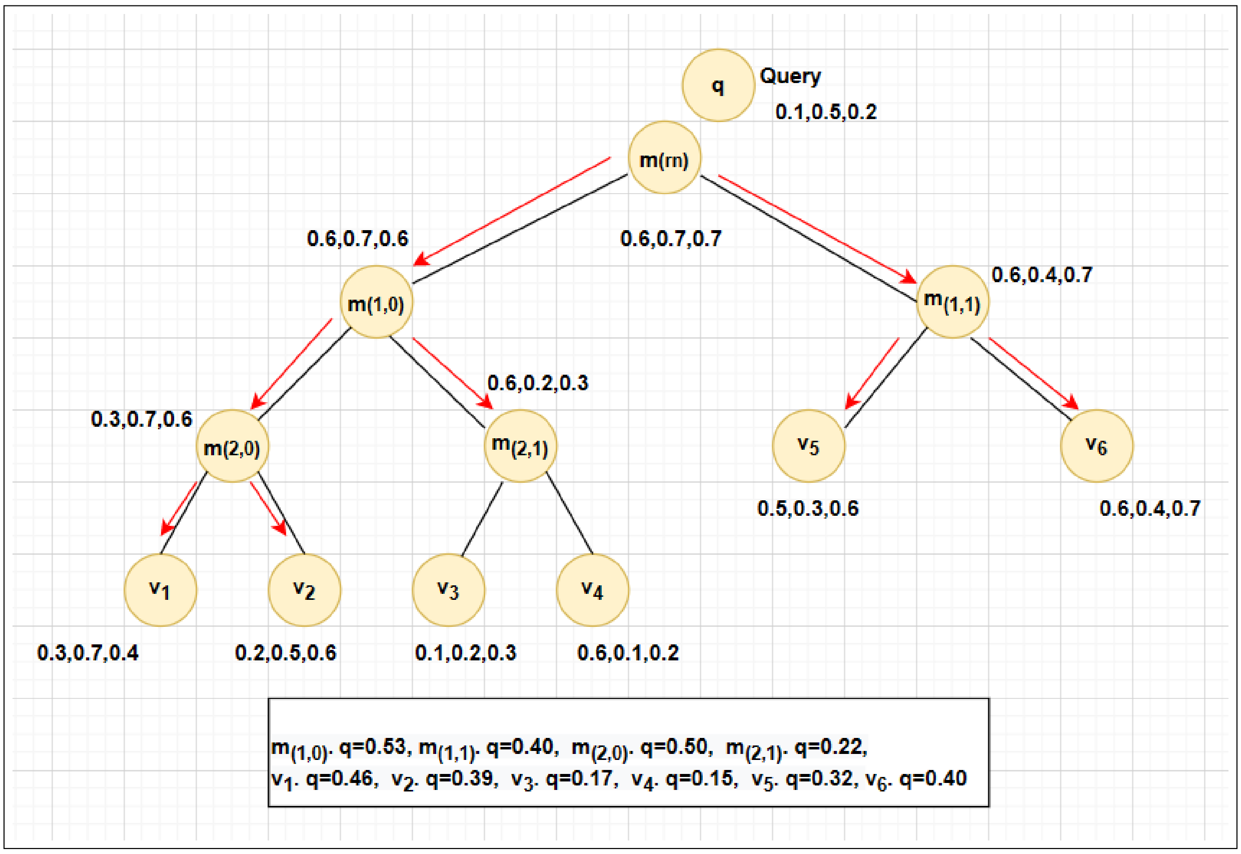

Figure 3: **Example of searching a binary tree**

. This binary tree will be traversed using the root node vector n(rn) = (0.6,0.7,0.7) to find the top results. The procedure is as follows:

- By calculating the values we have seen that $(m_{(1,0)}.q) > (m_{(1,1)}.q)$. Thus with root node vector $m_{(1,0)}$ we will search the subtree, then continue to search the root node vector from $m_{(2,0)}$ due to $(m_{(2,0)}.q) > (m_{(2,1)}.q)$.
- Next, we compare the scores $(v_1.q)$=0.46 and $(v_2.q)$= 0.39. Due to $(m_{(2,1)}.q) = 0.22$ which is smaller than the 0.39 value so we do not need to calculate the value for d3 and d4.
- Due to $(m_{(1,1)}.q)$= 0.40 is grater than the $v_2.q$ value, so we need to compare $(v_5.q)$= 0.32 and $(v_6.q)$ =0.40 . now the top results v_1 and v_6 will return.

3.2.5 Decryption

It involves the decryption of the documents obtained after searching. The document we received is encrypted, and we need to decrypt it before using it. We used the AES algorithm for this purpose. The same key that we used for the encryption will be used to decrypt the document.

4 Design Specification

In our approach, the data owner, cloud server, and user are the main entities. The system will be running in a client-server architecture. The end-user is the person in ownership or possession of the data. To make use of data, the owners need to upload it and the users are able to access it using queries. Cloud servers maintain data and respond to queries from authorized users.

4.1 Index Generation

When you upload your data to your server then the data will be processed, and generate the indexes and the indexes will be saved in a cloud server. The user who owns the data is responsible for uploading it. In this approach, when the data owner uploads the data then the server will read the documents and find the keywords for each document. He will process document D to remove the stop words and then will apply the lemmatization and calculate the TF*IDF. After finding the TF*IDF value he will create the clusters based on inverse frequency. We divided all keywords into different k clusters. And encrypt keywords using SHA. Create the binary tree for each cluster and encrypt the document set using AES, and key. Now upload the document and cluster on the cloud. The cloud server will have all of the information in encrypted form, and it cannot read any information in encrypted form. While sharing the document with the data user, the data owner can share the secret key without losing its security.

Algorithm 1: Index generation

```
   Input: Document set D, cluster count K, Encryption key-ek
   Output: Encrypted documents D, K- clusters
1  for Each document d in D do
2      Apply Lemmatization;
3      Remove stopwords;
4      Calculate the frequency of each word;
5  end
6  Generate word matrix W.
7  Find inverse document frequency.
8  Create k-clusters based on Inverse document frequency.
9  for each cluster c in K-clusters do
10     encrypt word set using SHA
11     create binary tree
12     for each document in cluster do
13         encrypt documents using AES
14     end
15     upload cluster
16     upload encrypted documents
17 end
18 return
```

4.2 Customised Query execution

The facility for firing a range query over secured cloud data is available to the user. The user of our data can set the priorities for each of their queries. After firing the queries he will apply lemmatization and stop words to the keywords to get the exact keywords. Then, will check the query conditions qc. If the query condition is synonym search then update the query with synonym using word net. Now update the query with Q to the cloud. Encrypt query words using SHA. The cloud server will link your query to all the files available at its end, find the required cluster based on Q, and then travel the binary tree to find the matched root node. Then he will apply query conditions and sort the nodes and give you the matched documents DS from the matched nodes. To get a document that has been encrypted with AES, the user needs to enter the Access Key and decrypt it.

Algorithm 1: Customised Query Execution

```
Input: Query keyword with importance Q
Document set D
decryption key-ek
Query condition-qc
clusters with binary tree-T
Output: Matched document set-DS
for Each word W in Q do
    Apply Lemmatization;
    Apply stopwords;
    if qc contains synonym search then
        Find synonyms using word net.
        Apply lemmatization
        Update words in Q.
    end
    encrypt word W using SHA-1
end
Share encrypted Q with the cloud.
for Each cluster T do
    travel binary tree and match the words in Q with nodes.
    Apply query conditions qc.
    sort the matched results.
    return respective documents from matched nodes DS.
end
for Each document D in DS do
    decrypt the document using ek.
    save and display the result.
end
return
```

4.3 System Architecture

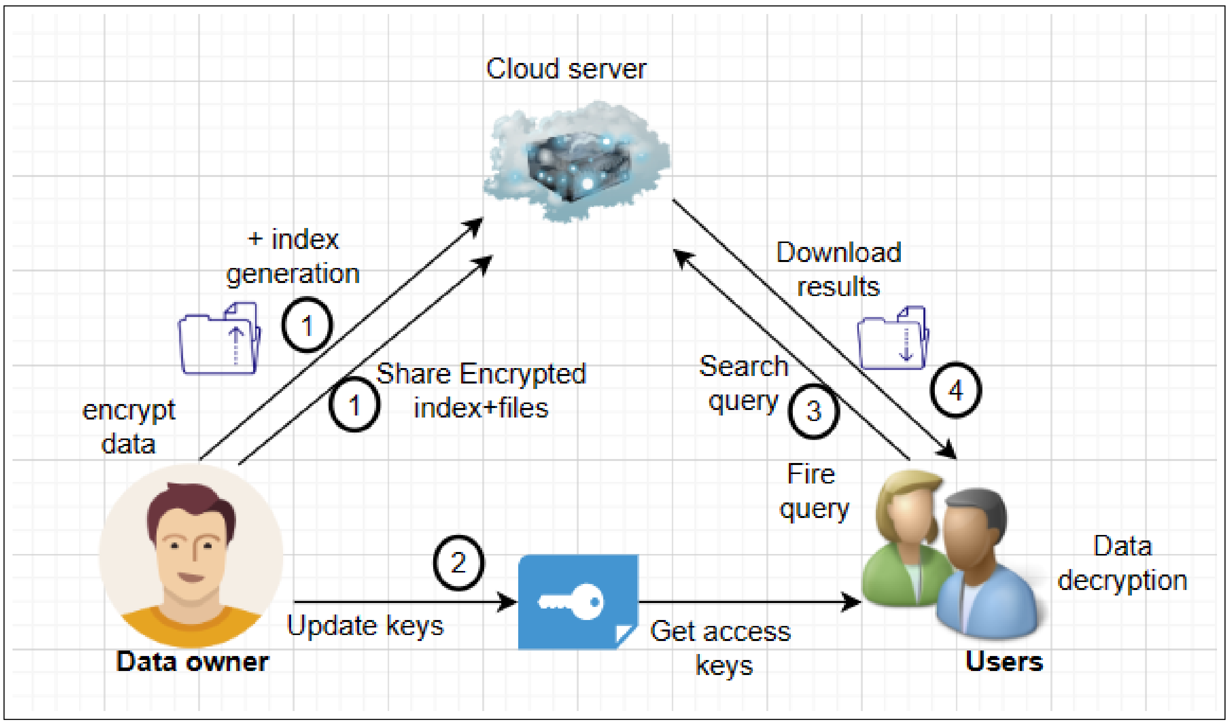

Figure 4: **System Architecture**

Figure 4 shows the system architecture with three main entities: Data-owner, Data-User, and Cloud Server. The owner of the data is responsible for uploading all the data files to the cloud server. The clustering and mapping of clusters, as well as data encryption, was done by the user end. The cloud server's responsibility is to store the data, while the key is responsible for computing security. The data user is responsible for firing a query. Query encryption as well as result display after the decryption is performed at the user's end. The cloud server's responsibility is to check the specification of the query such as he will check if the query is for a synonym search or any other search. Then he will process all the search queries based on their priority values with the previously stored dataset and return matched results.

5 Implementation

In this study, our main aim is to develop an effective model for smart search over encrypted cloud multimedia data. We have divided this section into different parts to explain languages, tools, datasets, and the models developed in this search.

5.1 Language and tools used

We used the client-server architecture for the implementation of our system.

5.1.1 Client Side:

- **Development Environment:**

The system was created in Java using JDK 11. For desktop system development Netbeans 8.2 IDE is used. We have also installed wordNet. Using java swing components, a desktop application is created for the end user. The HTTP protocol is used by this application to communicate with the cloud server.

- **Execution Environment:**

For desktop applications, We have created a Java executable jar that can be executed with the help of JRE 11 or greater.

Table 1: **Execution environment System specification for client side**

Operating System	Windows 10
CPU	Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz processor
RAM	16 GB
system Type	64-bit operating system, x64-based processor

5.1.2 Server Side:

- **Development Environment:**

To set up a server environment Apache Tomcat-7 and MySQL 8.1 is used. For server-side application development Eclipse, and JEE Neon is used.

- **Execution Environment:**

The server-side application is hosted on Amazon EC2 t2*2xlarge instance. where we used MYSQL 8.1 and Apache Tomcat-7.

Table 2: **Execution environment System specification for server side**

Operating System	Windows 10
CPU	Intel(R) Xeon(R) CPU E5-2686 v4 @ 2.30GHz
RAM	32 GB
system Type	64-bit operating system, x64-based processor

5.2 Data set

A system is tested using a multimedia file dataset. It contains text files, audio, video, and images. Compressed files are also employed in testing. All the data sets we used for the testing are taken from the base paper. Miao et al. (2022) Basically we need a text document for the processing, keyword extraction, etc. For the text document datasets, we referred to the correlated documents in which more keywords are matching and will be more helpful for clustering. If we use any random dataset without words matching or completely different files then there will be no intersection and no clustering. For the image or video datasets, we added custom tags because we cannot fetch the keywords from them. So for the videos of the same person or occasion, we can add matching keywords which can be helpful for clustering. Our system is tested using 3 different datasets:

- **Enron Dataset:** This is an email dataset named Enron [1]. It contains email data collected from 150 users. The dataset contains sent emails, deleted emails, inboxes, etc. They are Correlated and can make clusters as well.
- **Synthetic dataset:**Our System is working on a multimedia dataset that's why we created a synthetic data set. This dataset contains multimedia files(images, videos, etc). The sizes of files range from 2KB to 10 KB. We need to add custom tags for these files. And files for the correlated tags can make clusters as well which makes these files easier to search.
- **20newsgroup dataset:** This dataset [2] is made up of around 20,000 newsgroup posts and divided across 20 groups. This dataset contains 20 different categories related to news, sports, etc. and all the files in one category can make clusters with the other related files and it will be easy to search for more than one file.

5.3 Models Developed

A desktop client-side application on the user's end allows for system interaction. With the help of this application, users can register and log in. Following authentication, the user has access to upload, download, search, and create folders with shareable folder rights. The user has the option to upload one or more files at once. Users can change the keyword list while the machine collects keywords from files.

The system records the processing time as it uploads files and keywords to the cloud after encryption. A variety of file formats and sizes are tested on the system. Users can additionally group their uploaded data into clusters. The clustering data is utilized to effectively search the relevant file. Users can use the index keywords to search through uploaded data. The user can choose from a number of logistical search options offered by the system, including AND-Matching, OR-Matching, NOT matching except for certain keywords, and synonym matches. Finding the most relevant search results is made easier by these search operations.

5.4 Code Developed

The major building elements implemented utilizing a client-server architecture are data uploading, clustering, and searching. Encryption is done at the client end and the data is uploaded to the cloud server at the time of data upload, index generation, and data encryption. The k-means and b-tree algorithms are used for clustering. Based on the index keywords, clustering is carried out, and the results are uploaded to the server. On a server, clustering results are searched, and indexed files that match the search are sent back to the client. The client system decrypts the data it receives and saves it where it is needed.

6 Evaluation

In this section, we evaluated the actual performance of the system. For the study, we have implemented different experiments over a real-world dataset Enron, the 20newsgroup

[1]Enron dataset `https://www.cs.cmu.edu/~enron/`

[2]20 newsgroup dataset `https://www.kaggle.com/datasets/crawford/20-newsgroups`

dataset, and our synthetic dataset. Basically, we have added extra search features in our system so we need to evaluate those features. Will they provide the most accurate results than the existing systems, or are they taking more time for processing than the existing system?

To evaluate the system performance we decided to find the results for the accuracy and latency of the system. Is our system working more accurately than the existing systems? or Whether they impact the accuracy and latency or not? For accuracy, we have used precision and for latency, we compared the time. Precision is defined as the proportion of true positives to total classified positives. Time is the total time taken for index generation, cluster creation, search, etc. Each result is discussed here.

6.1 Experiment 1 / Index Generation time

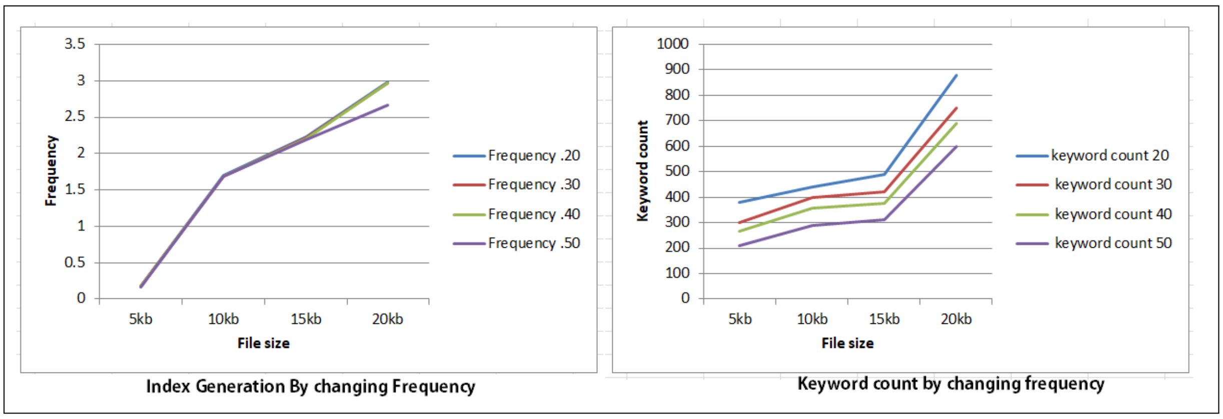

Figure 5: **Index generation and Keyword count**

In this experiment, we examined the index generation time and keyword count. Index generation time is mainly the time taken for processing the file such as fetching keywords, building a balanced binary tree, uploading the files, etc. For this experiment, we uploaded files for different sizes by changing the frequency and examined the processing time and keyword count for each file upload. Frequency and keyword counts are inversely proportional to each other. The increase in frequency is not mainly affecting the index generation time as seen in Figure 5 but it affects the number of keywords found. with the increase in frequency, We will get less number of keywords which will affect the search accuracy. On the other hand, If we have less frequency then we will get a more number of keywords which makes the search efficiency effective and accurate. A decrease in frequency count may lead to a high number of keywords set for a document. which may increase an index size. There should have a trade-off between the number of keyword counts and the frequency. This frequency count is restricted to documented data only. It is applicable for only text data processing because, for multimedia data such as images, video, etc. The user is responsible to provide a keyword set so it is independent of the frequency.

6.2 Experiment 2/ Cluster Creation Time

In this section, we examined the cluster creation time for different files. If we upload 30-100 files then how much time they will take to create the clusters?

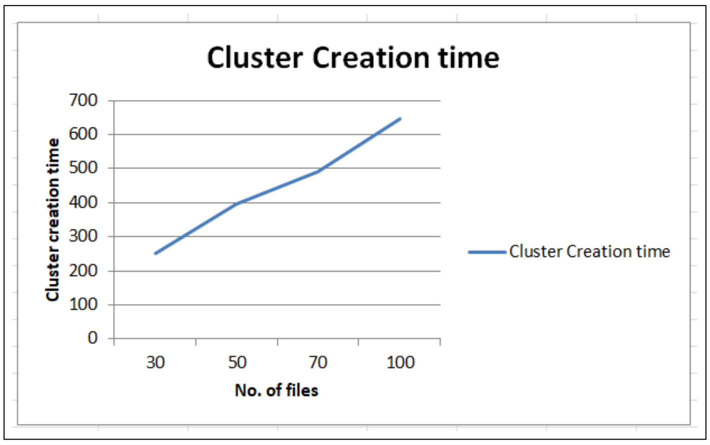

Figure 6: **Cluster creation time**

For this experiment, we have uploaded some folders having a different number of files in them ranging from 30 to 100 document files as seen in figure 6. we have the frequency count as .30 for extracting the keywords and This is the ideal frequency in this case because it will give us the ideal amount of keywords that will not affect the processing time and we will get more keywords. Once the keywords are extracted from a large number of document files, then keyword indexes are generated. we can say that number of files is directly proportional to the clustering time. The increase in the number of files increases the file clustering time. Because the number of document files is increased and it will take more time for processing all the files.

6.3 Experiment 3 / Search Time

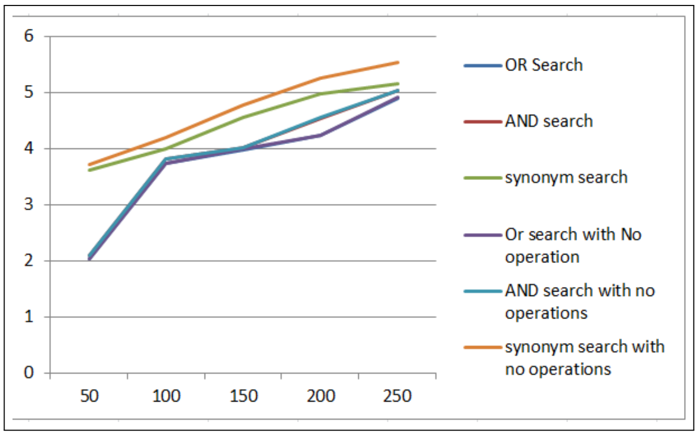

Figure 7: **Search time**

In this section, we have examined the total search time for the specified keywords(OR, AND, NOT, and synonym, etc.) Considering figure 7, we have applied the search operation with different search specifications on different numbers of document files ranging from 50-250. After analyzing we found, Synonym search with No operation took more time for processing because they need to process a large number of files than all the other search specifications to return the specific results. Synonym search took more search time but they return all the relevant documents. AND search will give the relevant results but AND with NOT operation gives us the most limited and specific results. Because NOT operations allow us to filter the data. Synonym search will take higher time as compared to the other search because it needs to find synonyms first and then search all the keywords in documents. With the increase in the number of keywords in search the time also increases.

6.4 Experiment 4/ Precision

In this section, we have created the precision matrices. These metrics depend on the outcomes of the results retrieved from the system such as true positives and false positives. False positives are the total results retrieved at the time of searching and the true positives are the intended result that we actually expected. Precision is defined as the proportion of true positives to total classified positives.

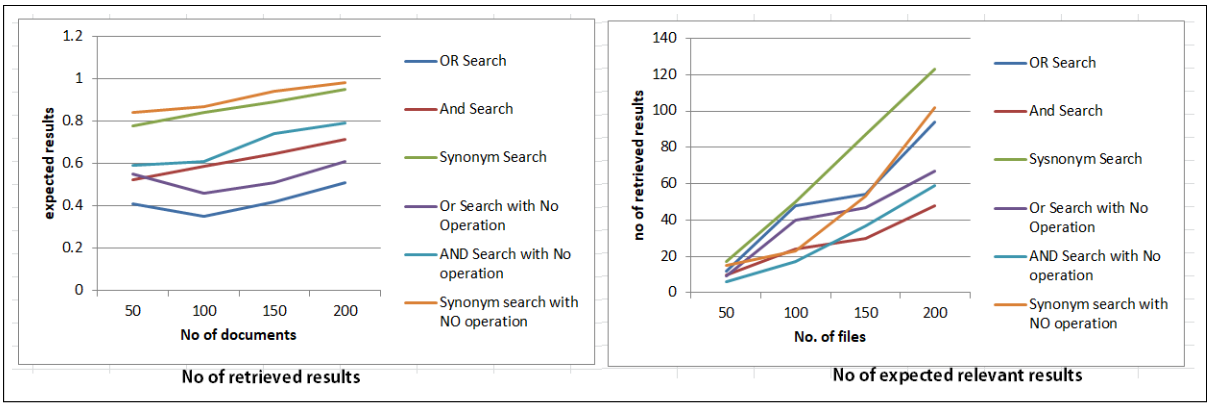

Figure 8: **Precision**

Precision = TP ÷ TP + FP

Considering figure 8, We have searched for the keywords with different search specifications and with the changing number of documents. we performed our search by adding four keywords for each search specification(AND, OR, NOT, and synonym, etc.) we had to check the true positive and false positive, and for that purpose, we verified the data manually. we verified each folder manually and then calculated the precision. we can see in figure 8 that we got too many results but the best results are for the synonym search and with NO operation we received the more accurate results.

The graph represents the total number of results retrieved with the help of different search criteria. we are getting a maximum number of search results with the help of synonym search and limited results with the help of AND with NO operation results. we can limit the number of records based on these criteria then synonym search helps us to find the documents with similar words existence and hence, provides higher precision NO operator helps us to reduce the false positive counts, and AND search provides us the most precise and exact documents having all the keywords.

6.5 Discussion

In this study, we aimed to develop a robust model for smart searches of encrypted cloud multimedia data. Our model has shown better performance in synonym search, AND, OR, and NOT operations; to generate good results, we tested different aspects and came up with some conclusions, which show that the index generation time varies depending on the frequency of the document. There should be a trade-off between keyword count and frequency. Defining the ideal frequency can be difficult, but it should not be too high. Because of clustering and binary trees, our search time does not increase linearly, and a synonym search often yields more accurate results because the words are more relevant to your search. You can use synonyms instead of keywords, freeing up some memory space and opening up a world of possibilities. OR results will give you relevant results and with NOT operation it will work better. If you want an accurate and limited amount of data sets then AND search with NOT operation is the best choice because it is giving us the most relevant and specific to the query keyword results. In our model, we use compound conditions, that's why we obtain the most accurate, expected, and precise results.

We have the ability to add custom keywords to each type of dataset, which is most valuable in the case of multimedia datasets like images, audio, or video. Because multimedia data can be searched on the basis of the index and not content. Multimedia datasets are independent of frequency compared to documents because, in document retrieval, the keywords are fetched, whereas, in multimedia datasets, they are created explicitly. This is why uploading the image dataset will take less time than the text data because the system will not take too much time to process the document. It will not read the image so the data user needs to add the custom tags for the image data set. And he can add as many tags as he wants there is no limitation for tags.

7 Conclusion and Future Work

The main aim of this research is to make the search for encrypted multimedia data easy for users. There is an increasing number of businesses are operating online, so when it comes to storing personnel information, there has been a growth in technology. However, with any progress in time, there are concerns. In this case, they are especially warranted: data breaches seem to be all over the news these days, and that's why we proposed our solution using the balanced binary tree and the k-means clustering technique. Users can add custom tags for all the files he is uploading. we also provided search specifications such as AND, NOT, and synonym search, etc. to make the search easier for the user. without any compromise on accuracy, we used the customized Query Execution algorithm to reduce the search complexity. we have made encryption available to data users. This enables them to save and share content without worry. Results demonstrate that our search promises accuracy and latency for the encrypted search. Synonym search will take more time but will give more results related to the query and AND search with NOT operation will provide the most accurate and the most relevant results.

This research can be improved by adding more search specifications for the multimedia data such as fuzzy search, phrase search, etc. by using the same technique. Then it will be easier for the users to search for the data that is related to or misspelled by the user.

References

Ali, F. S. and Lu, S. (2016). Searchable encryption with conjunctive field free keyword search scheme, *2016 International Conference on Network and Information Systems for Computers (ICNISC)*, IEEE, pp. 260–264.

Alzu'bi, A., Amira, A. and Ramzan, N. (2017). Content-based image retrieval with compact deep convolutional features, *Neurocomputing* **249**: 95–105.

Boneh, D., Kushilevitz, E., Ostrovsky, R. and Skeith, W. E. (2007). Public key encryption that allows pir queries, *Annual International Cryptology Conference*, Springer, pp. 50–67.

Chen, L., Liao, X., Mu, N., Wu, J. and Junqing, J. (2019). Privacy-preserving fuzzy multi-keyword search for multiple data owners in cloud computing, *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, IEEE, pp. 2166–2171.

Cui, H., Wan, Z., Deng, R. H., Wang, G. and Li, Y. (2016). Efficient and expressive keyword search over encrypted data in cloud, *IEEE Transactions on Dependable and Secure Computing* **15**(3): 409–422.

Fu, Z., Huang, F., Ren, K., Weng, J. and Wang, C. (2017). Privacy-preserving smart semantic search based on conceptual graphs over encrypted outsourced data, *IEEE Transactions on Information Forensics and Security* **12**(8): 1874–1884.

Fu, Z., Huang, F., Sun, X., Vasilakos, A. V. and Yang, C.-N. (2016). Enabling semantic search based on conceptual graphs over encrypted outsourced data, *IEEE Transactions on Services Computing* **12**(5): 813–823.

Fu, Z., Shu, J., Wang, J., Liu, Y. and Lee, S. (2015). Privacy-preserving smart similarity search based on simhash over encrypted data in cloud computing, *Journal of Internet Technology* **16**(3): 453–460.

Fu, Z., Sun, X., Xia, Z., Zhou, L. and Shu, J. (2013). Multi-keyword ranked search supporting synonym query over encrypted data in cloud computing, *2013 IEEE 32nd International Performance Computing and Communications Conference (IPCCC)*, IEEE, pp. 1–8.

Fu, Z., Wu, X., Guan, C., Sun, X. and Ren, K. (2016). Toward efficient multi-keyword fuzzy search over encrypted outsourced data with accuracy improvement, *IEEE Transactions on Information Forensics and Security* **11**(12): 2706–2716.

Fu, Z., Wu, X., Wang, Q. and Ren, K. (2017). Enabling central keyword-based semantic extension search over encrypted outsourced data, *IEEE Transactions on Information Forensics and Security* **12**(12): 2986–2997.

Guo, C., Jia, J., Choo, K.-K. R. and Jie, Y. (2020). Privacy-preserving image search (ppis): Secure classification and searching using convolutional neural network over large-scale encrypted medical images, *Computers & Security* **99**: 102021.

Hu, S., Wang, Q., Wang, J., Qin, Z. and Ren, K. (2016). Securing sift: Privacy-preserving outsourcing computation of feature extractions over encrypted image data, *IEEE Transactions on Image Processing* **25**(7): 3411–3425.

Janani, T., Darak, Y. and Brindha, M. (2021). Secure similar image search and copyright protection over encrypted medical image databases, *IRBM* **42**(2): 83–93.

Li, J., Wang, Q., Wang, C., Cao, N., Ren, K. and Lou, W. (2010). Fuzzy keyword search over encrypted data in cloud computing, *2010 Proceedings IEEE INFOCOM*, IEEE, pp. 1–5.

Likas, A., Vlassis, N. and Verbeek, J. J. (2003). The global k-means clustering algorithm, *Pattern recognition* **36**(2): 451–461.

Lu, W., Swaminathan, A., Varna, A. L. and Wu, M. (2009). Enabling search over encrypted multimedia databases, *Media Forensics and Security*, Vol. 7254, SPIE, pp. 404–414.

Metkari, S. S. and Sonkamble, S. (2016). Multi-keyword ranked search over encrypted cloud data supporting synonym query, *Int. J. Sci. Res* **5**(6): 2044–2048.

Miao, Y., Zheng, W., Jia, X., Liu, X., Choo, K.-K. R. and Deng, R. (2022). Ranked keyword search over encrypted cloud data through machine learning method, *IEEE Transactions on Services Computing* .

Mistry, Y., Ingole, D. and Ingole, M. (2018). Content based image retrieval using hybrid features and various distance metric, *Journal of Electrical Systems and Information Technology* **5**(3): 874–888.

Poon, H. T. and Miri, A. (2015). An efficient conjunctive keyword and phase search scheme for encrypted cloud storage systems, *2015 IEEE 8th International Conference on Cloud Computing*, IEEE, pp. 508–515.

Poon, H. T. and Miri, A. (2017). Fast phrase search for encrypted cloud storage, *IEEE Transactions on Cloud Computing* **7**(4): 1002–1012.

Shashank, J., Kowshik, P., Srinathan, K. and Jawahar, C. (2008). Private content based image retrieval, *2008 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, pp. 1–8.

Song, D. X., Wagner, D. and Perrig, A. (2000). Practical techniques for searches on encrypted data, *Proceeding 2000 IEEE symposium on security and privacy. S&P 2000*, IEEE, pp. 44–55.

Tang, Y., Gu, D., Ding, N. and Lu, H. (2012). Phrase search over encrypted data with symmetric encryption scheme, *2012 32nd International Conference on Distributed Computing Systems Workshops*, IEEE, pp. 471–480.

Vaquero, L. M., Rodero-Merino, L., Caceres, J. and Lindner, M. (2008). A break in the clouds: towards a cloud definition.

Wang, C., Cao, N., Li, J., Ren, K. and Lou, W. (2010). Secure ranked keyword search over encrypted cloud data, *2010 IEEE 30th international conference on distributed computing systems*, IEEE, pp. 253–262.

Wang, C., Ren, K., Yu, S. and Urs, K. M. R. (2012). Achieving usable and privacy-assured similarity search over outsourced cloud data, *2012 Proceedings IEEE INFOCOM*, IEEE, pp. 451–459.

Xia, Z., Wang, X., Sun, X. and Wang, Q. (2015a). A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data, *IEEE transactions on parallel and distributed systems* **27**(2): 340–352.

Xia, Z., Wang, X., Sun, X. and Wang, Q. (2015b). A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data, *IEEE transactions on parallel and distributed systems* **27**(2): 340–352.

Yang, Y. and Ma, M. (2015). Conjunctive keyword search with designated tester and timing enabled proxy re-encryption function for e-health clouds, *IEEE Transactions on Information Forensics and Security* **11**(4): 746–759.

Zhang, H., Zhao, S., Guo, Z., Wen, Q., Li, W. and Gao, F. (2021). Scalable fuzzy keyword ranked search over encrypted data on hybrid clouds, *IEEE Transactions on Cloud Computing* .

Zittrower, S. and Zou, C. C. (2012). Encrypted phrase searching in the cloud, *2012 IEEE Global Communications Conference (GLOBECOM)*, IEEE, pp. 764–770.