International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Enhancing Information Retrieval from Unstructured Data Using Lexical Chain Analysis and WordNet Integration with Lucene

Mr. Chattar Singh[1], Dr. Vijay Gupta[2]

Submitted: 15/02/2024 **Revised**: 22/03/2024 **Accepted**: 04/04/2024

Abstract: Big data entails extracting relevant information from unstructured data sources properly. This paper presents a novel approach to increase the efficiency of Lucene search by the application of lexical chain analysis. The purpose of doing this is to increase the precision and relevance of search results. This paper uses the power of lexical chains through WordNet, a large lexical database of synonyms, hyponyms, hypernyms, homonyms, and meronyms. Such chains can be considered to form coherent words in a text, which comprises an important indication to the context associated with a conceptual framework of a given text. The lexical chain analysis that our system does tries to look at and exploit the subtle semantic associations that are there in the search text for better context and improving recall through documents that are semantically more relevant. The system to be proposed, in addition, would support a variety of search modalities: document name, content, and attributes such as type, size, date, and author. The indexing mechanism is based on keyword frequency and concerns the occurrence of various keywords related to the lexical chains in the documents. Furthermore, the system will also apply its search engine with image-based documents, hence taking into account diversified formats that are the characteristic of a contemporary data repository. The facility for providing suggestions for autocompletions by retrieving past search queries and documents will be bound to enrich user interactions through the ability for users to search for things immediately. Furthermore, it will incorporate sentence-based searching capabilities to allow users to break down text and mine for in-depth details. This cross-discipline effort is expected to produce far-reaching innovations in unstructured searching. Through synergy between the power of Lucene in search and the semantic insight provided by WordNet-driven lexical chain analysis, this paper tries to redefine information retrieval to match the changing needs of almost all spheres, making the access to knowledge repositories intuitive and efficient.

Index Terms: *Lucene, lexical chain analysis, WordNet, information retrieval, unstructured data, semantic search, document indexing, image-based search.*

Introduction

In today's digital world, the amount of unstructured data produced has surpassed all the previous figures. The massive amount of data, either it be text documents or images or multimedia contents, faces a single major bottleneck—information retrieval. Traditional search engines, built keeping in mind the stance on structured data, many times fail to yield exact and relevant results due to the implicit variations and complexities coming with unstructured data. Their failure to capture the semantic nuances and contextual relationships embedded in unstructured text brings to the fore the deficiencies of conventional, keyword-based search mechanisms. This gap, therefore, calls for the invention of advanced search methodologies capable of sailing through the complexities of the unstructured data and enhancing precision and relevancy of search outcomes [2]. The application of Lucene with Lexical Chain analysis would be another innovative and applicable solution for these problems. Text A lexical chain, or a semantically connecting fixed word chain, is a sequence of semantically related words used to fulfill the cohesive intent of the writer in text. By applying WordNet—the most commonly used comprehensive lexical database with synonyms, hyponyms, hypernyms, homonyms, and meronyms—the search queries and documents would have more possible ways to be semantically understood. This integration will enable a better search process for documents that are not only keyword-relevant but also contextually relevant; hence, much more detailed

International School of Informatics and Management, Jaipur, Rajasthan 302020, India
Rajasthan Technical University, Kota, Rajasthan 324010, India
International School of Informatics and Management, Jaipur, Rajasthan 302020, India
Rajasthan Technical University, Kota, Rajasthan 324010, India
Corresponding author: (e-mail: chattarsinghr@gmail.com).
Second author: (e-mail: vijaygupta1@gmail.com).

experiences are being received [2]. Various extensions to make the proposed framework be better in functioning than the regular search engines.

The prime search support modalities are the multi-faceted search modalities that allow the users to search in terms of the name of the document, the content, and other attributes like type, size, date, and author of the document [3]. The system suggested is also very flexible since it provides the support needed for varied user needs and search contexts; indexes the set of documents by giving varying keyword priorities within the lexical chains, and not only by keyword frequency. This indexing type can be expected to enhance the accuracy and relevance of search results. [3]. The system further expands the search facility to image-type documents, which find their way into modern data repositories with different file formats [4]. The feature of autocomplete suggestions, based on earlier search queries made by the user and used documents during the search, aims to provide an improved search experience to the user. The addition of search at the level of a single sentence would help the users to search for an individual sentence in documents and prompt the users to explore textual information much more deeply. \t.

By being so deeply interdisciplinary, on the edge between computational linguistics, information retrieval, and data science in the best sense, the current research takes advantage of the best aspects of every discipline involved. By harvesting the power of synergy that it generates with the inexpensive mining of corpora with robust search functionalities and the power of semantic light shed on the algorithms by the analysis of the lexical chains based on WordNet, this research envisages information retrieval afresh. Resulting in a search system responsive not only to users' changing needs from all kinds of domains but also being intuitive and efficient in supplying access to large knowledge repositories. This research is going to give many breakthroughs in the field of unstructured data search to new levels of effectiveness and usability [5]. The important feature of this research is the potential to bring about a revolution in extraction information from sources that is also represented as unstructured data.

Useful and relevant information must be harnessed efficiently in this modern digital era, where data is hugely generated at an unprecedented rate and in diverse formats [6]. Traditional search engines were meant for structured data and have, therefore proven inefficient in case of complexities associated with unstructured data. In this regard, this research work has proposed the current deficiency overcoming new approach that couples a lexical chain analysis with a potent search engine library, namely Lucene [6]. One of the salient contributions of this research is in finding ways to make search outcomes as precise and relevant as possible. With the incorporation of lexical chains and utilization of the semantic clarity of the WordNet [7], the proposed may be able to further understand the context and relationships of the search queries and documents. This better grasp of semantics makes it possible to return documents that are not only relevant based on keywords but also relevant contextually, thus improving the general quality of the search results [7].

The proposed framework has additional significance in that it supports varied search modalities related to the name of a document and its content, as well as other attributes in the concerned domain, giving users the scope of being flexible and having increased control over their queries. In this regard, it can manage diversified users' needs across different domains and search contexts [8].

The indexing mechanism is situated such that keyword variations that are relevant to the lexical chains within the documents take a priority. This kind of indexing gives more nuanced approaches, accuracy, and relevance of the search result, thus being more satisfying for user experience, for instance [8].

Further improving user experience in finding certain pieces of information, the proposed system lends utility to image-based documents by allowing the user to search and by adding features for giving hints for autocompletion and sentence-level search capabilities in the system [9].

As such, this research is very promising for the implementation of information retrieval through its extraction from unstructured sources of data. Leveraging the synergy of high-quality search functionalities found in Lucene and the semantic insights that could be provided by lexical chain analysis, this study is about to define the information retrieval landscape afresh, whereby the same will become much more effective, user-friendly, and intuitive.

This is a multipronged goal, which hopes that the difficulties pertaining to retrieving information from unstructured sources of data can be resolved by exploiting the immense potential of lexical chain analysis and the integration with Lucene. The major research objectives of the study are as follows:

- Enhanced Search Effectiveness: The outcome of this research would raise the levels of precision and relevance to the searching results because sought-after information would be exactly retrieved from information retrieved from unstructured sources of data.
- Implement Lexical Chain Analysis with WordNet: This would lexically chain the related words in WordNet, such as synonyms, hyponyms, hypernyms, holonyms, and meronyms. This makes the search richer in semantics and hence helps the user get a proper contextual understanding.
- Allowing Searching by Multiple Parameters: The study is going to come up with a search mechanism that will allow the capability of searching depending on the document name, content, and other attributes such as type, size, date, and author. This will improve the search system to be more versatile and flexible in ways that it can be tailored to various consumer needs.
- Search Inside PDF Documents: This is another objective to increase the search capability up to the text inside PDF documents. This caters to the general requirement of content searching across disparate document repositories capable of supporting any file format.
- Advanced Indexing Mechanism Development: The research must provide an indexing mechanism that will go beyond the frequency of keywords and investigate the relevance of keywords to lexical chains in every document. The latter sharpens the accuracy and relevance of search results, therefore improving the entire search experience.
- Extending Search Capabilities to Image based Documents: One more goal is the integration of search capabilities to image based documents. The enormously felt need to facilitate search in these is important since it has been hinted above that pieces of information come in the form of images too. The goal is quite ensuring in its implication by offering the guarantee that the system will be able to carry on an extensive search over different file formats.
- Implementing Autocomplete Suggestions: The feature of this performance goal is to provide autocomplete capability for user queries and documents aimed at streamlining the user's experience in interacting with the system of search.
- Enabling Sentence-level Search Functionality: It provides power to the user so that the user may search a sentence within a document using the conventional method of searching. Fine-grained search functionality in textual content might further increase usefulness of the search system by providing capabilities at the sentence level to the end user.

The research looks up to getting the objectives accomplished and to get over the limitations of traditional ways of conducting searches by providing users with searches that are natural and efficient, therefore realizing the advancing field of unstructured data search.

I. Literature Survey

Kanev et al. provide an overview of the most effective domain of semantic search for information retrieval systems and present a comparative study of a semantic search vs. a Lucene-based search in terms of precision and recall with respect to a few distinct query types. They also propose a methodology to compare rankings from search engines and further test its applicability on the Open Corpora text corpus.

Kasmani, F., et al. (2020, March) add to the movement of learning from subject-oriented to topic-oriented, and recommend on Content-Based Search Engine for E-Books. The proposed one has two main functions of Indexing E-books and rating it in addition to results produced based on frequent phrases. This method is more intuitive when searching for the E-books kind than the traditional keyword-based search engines [9].

Jin, D. et al. (2020, June), in a lightweight manner, propose the technique of full database recovery involving JDBC technology to extract the content and Lucene for full-text indexing. Their research demonstrated effectiveness and stability of the developed approach in full data retrieval from relational databases across several tables [10].

Lin et al. (2020, July) cover interoperability between search engines designed with an open source, including the APIs provided, and methods that one can use to compare what is ostensibly an apple-to-apples comparison. The authors suggest methods for sharing indexes among search engines and suggest that adoptive mechanisms support the Common Index File Format, which enables independent innovation and fair comparison in the field [11].

D. Wang: In November of 2021, this same statement comes with the volume of information increasing tremendously from all over the world. They have generated research on the development of a digital archive management system supported by modern

advanced technologies in computer, communication, and network systems. The system architecture designed is client/browser/server component-based to enable cross-platform function [12].

Hema et al. (2021) aim to propose a domain-specific search application to empower proficient retrieval of chemical documents. The application incorporates specialized indexing and optimized ranking relative to chemical entities and phrases, positively affecting the retrieval accuracy pertaining to patents and chemical abstracts [13].

Chan, H. P., et al. (2021), developed a cloud-based search engine system for rich text content. The system employs Apache Tika for metadata extraction and uses the BM25 algorithm for keyword-text similarity scoring. The system shows a high recall rate and thus tremendously high throughput in search performance.

Safaei introduced a text-based multidimensional medical image indexing technique in 2021, which enhances the process of retrieval in the medical image search engine. This implementation, done with Lucene, enhances terminological-semantic indexing of multidimensional medical image databases for efficiency and effectiveness in query image retrieval.

Zouaoui and Rezeg present a semantic-based search engine for the Qur'an by using an ontology as an index. They develop a new ontology for Qur'anic documents and add semantic relationships between words in queries, which makes it result in higher precision and recall in comparison with now-used search engines [16].

Haddela, P. et al. (2021) where the document highlights the classification methods that are investigated over Sinhalese data with evolved Lucene search queries generated using a Genetic Algorithm. Their method has proven to produce promising results in terms of accuracy and interpretability for document classification compared to other classifiers [17].

Azizan and A., et al.,2022)., developed a search engine based on the Apache Lucene framework for the Malay Quran Translation text. They emphasized on understanding Lucene by beginners because of the limit of its precision and recall rate of moderate. The results hence becomes to be stated a basic approach to a test on Information Retrieval[18].

Singh and Gupta [10] introduced a full-text search engine using Lucene regarding the efficiency of retrieval in the web environment. They are planning to incorporate image search using the Tesseract API and also use WordNet to form lexical chains [19].

Liu, R., et al. (2022, October), in their paper, introduce the MMH-index, a composite index structure that enhances Apache Lucene with state-of-the-art multi-modal data indexing and searching capabilities. Their approach optimizes traditional inverted indexes further using modality bitmaps to achieve significant betterment in space consumption and query performance [20].

The authors present MAR, the model search tool for supporting query-by-example, improved in precision. Several enhancements have been developed in ways of incremental-indexing, keyword-based search, and user-friendly interfaces, developed over processing approximately 600,000 models [21].

II. Research Methodology

The methodology applied in this work entails a number of steps to arrive at an all-rounded file search application that retrieves text content from various file formats.

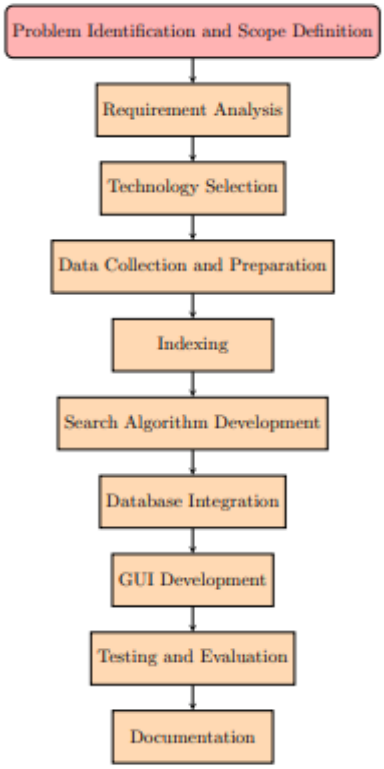


FIGURE 1. Proposed Flow Diagram [Project Implementation]

It starts with the identification of the problem and the scope of work: identifying the need for a file search application and what functionalities it is expected to have. This includes the determination of file types that would be supported and the scope of search functionalities to be offered to users.

First, the exact functionalities and features that the file search application must support are identified, taking into account requirements for supported file formats, different modes of search, database integration, and GUI requirements. This ensures that the implemented application would indeed serve the identified needs.

technologies and libraries for the implementation of the file search application. The technologies used here for this application include PyMuPDF for the extraction of text from PDF documents, python-docx for the extraction of text from DOCX documents, pytesseract for extracting text in images, the Whoosh library for text indexing/searching, MySQL database for storing the results, and Tkinter for the development of the GUI.

The next stage, after having the technologies in place, involves data collection and preparation. The sample data was collected in different file formats—instantiating from text documents to PDFs, DOCX files, and images—and was prepared by extracting text content from each file format using respective extraction techniques identified earlier.

Subsequently, an indexing process is implemented using the Whoosh library to index various file formats' text. A schema handling file paths and text content for efficient search operations shall have been defined in advance.

Subsequently, a search algorithm is developed, which processes the queries from the user. Such algorithms incorporate several kinds of searches like direct search, similar word search, and all word form searches. WordNet helps extend the search query and enhance the accuracy of the search by reducing synonyms, hyponyms, and other semantic relationships.

It is then integrated with the database to store the search results in a MySQL database.Such a schema enables the storage of information that is retrieved from search queries, the paths to matched files, and related statistics in a manner that it allows fast retrieval of the search results and their management.

A graphical user interface has been developed with Tkinter to make the functionality of the application more user-friendly. Directory selection, input of search text, mode of search, and file type selection have been implemented in the model so that it becomes more user-friendly.

Afterwards, functionality is tested rigorously, and performance evaluation is done in order to ensure the reliability and performance of the developed application. This includes indexing speed, search speed, and accuracy, along with some user feedback for further improvements.

It involves the documentation of implementation details regarding the structure of the code, functions, usage guidelines, deployment of the application for real-world use, user documentation, and support.

III. Proposed Approach

A. WordNet:

WordNet has been one of the primary resources in the area of computational linguistics, offering a huge, deep lexical repository encapsulating semantic relations and interlinking words within natural language. WordNet was created at Princeton University as a central resource for very many tasks in natural language processing by offering a rich lexical repository. At base, the aim is to group words into synonyms sets called synsets, which describe a complex of interrelated concepts. The resulting network of synonyms greatly expands into a complex structure of other relations: hyponyms (subordinate terms), hypernyms (superordinate terms), meronyms (wholes-parts relation), and holonyms (parts-wholes relation). This complicated deal of relation makes it possible for WordNet to capture the nuanced meanings and semantic nuances that inhere in language [22].

Additionally, the potential of WordNet goes far beyond the extremely stuffy notion of synonymy and rather lies in the fact that the structure gives information about the hierarchy of lexical concepts. Through this hierarchy, it becomes quite possible to look at concepts that are broader or narrower, thereby yielding a better understanding of the meaning of words and the semantic relations thereof. This hierarchical structure does not only help in word sense disambiguation but also provides a rich context for a host of tasks in language understanding, such as

information retrieval, text summarization, and machine translation [22].

In addition to this hierarchical structure, WordNet contains many other linguistic features, such as lexical categories (nouns, verbs, adjectives, and adverbs), word senses, and linguistic relations. All this rich linguistic information makes it possible to employ WordNet in a great variety of different NLP applications. The researchers and developers use WordNet as a baseline resource for different NLP tasks, such as word sense disambiguation, semantic similarity measure, and semantic analysis.

The fact that WordNet is constantly updated and extended further strengthens its constant pertinence in the field of natural language processing. The updating and enhancing of the lexical database of WordNet being a continual process, it makes WordNet a dynamic resource that evolves as the handling mechanism for the ever-increasing complexities of the use of the language. In this manner, WordNet remains an important tool for sustained research, development, and work by language aficionados in natural language understanding and computational linguistics [23].

B. Lexical Chains:

Lexical chains are essentially a methodological approach designed within computational linguistics to help unravel meticulously complex semantic relations embedded in textual data. A lexical chain is a simple linear sequence of words or phrases that are related in meaning and serve to connect and maintain a coherent thread within a passage of text. These chains provide essential information about the conceptual framework on which the text is built, giving in explicitly the semantic links and thematic cohesion coiled within the discourse.

Normally, lexical chains are created with the help of identifying and linking a word with a semantic relation or similarity to the previous one. There are various ways in which these semantic relations can be established—for instance, through WordNet-driven analysis or via distributional semantic models. Once the lexical chains had been identified, they are useful tools in the revealing of the innate semantic structure of textual data—hence, provisioning a deeper understanding of themes and coherence [24].

One of the key advantages of lexical chains, therefore, is that they can handle the hosting of subtle nuances of meaning, which otherwise may pass unnoticed under the more keyword-driven, traditional approaches. Lexical chains trace semantic links between words, offering a global view of the text, while the ability to find meaning patterns and identify the theme or topics serves to allow researchers to come to these conclusions.

Besides, lexical chains are of broad applicability to a huge variety of tasks related to natural language processing at large, beginning from information retrieval to document summarization and text categorization. In each of the above-stated tasks, a chain analysis adds appropriateness and accuracy to the results for a more effective and meaningful analysis of textual data.

In a nutshell, lexical chains can prove useful in the unfolding of the semantic structure that hides behind the text data, to understand what a text is actually about. Moreover, with the latest techniques in natural language processing, lexical chain analysis remains a robust analytic tool in unlocking the richness in meaning carried by text corpora..

C. ALGORITHM FOR PROPOSED APPROACH

Text Extraction Module

Algorithm 1 Extract Text from PDF Files

```
function EXTRACT_TEXT_FROM_PDF(pdf_file_path)
    OPEN pdf_file_path using PyMuPDF
    text ← ""
    for each page in pdf_document do
        page_text ← EXTRACT_TEXT_FROM_PAGE(page)
        text ← text + page_text
    end for
    CLOSE pdf_file
    return text
end function
```

Algorithm 2 Extract Text from DOCX Files

```
function EXTRACT_TEXT_FROM_DOCX(docx_file_path)
    OPEN docx_file_path using python-docx
    text ← ""
    for each paragraph in docx_document do
        paragraph_text ← EXTRACT_TEXT_FROM_PARAGRAPH(paragraph)
        text ← text + paragraph_text
    end for
    CLOSE docx_file
    return text
end function
```

Algorithm 3 Extract Text from Images

```
function EXTRACT_TEXT_FROM_IMAGE(image_file_path)
    OPEN image_file_path
    ocr_result ← PERFORM_OCR_ON_IMAGE(image)
    text ← EXTRACT_TEXT_FROM_OCR_RESULT(ocr_result)
    CLOSE image_file
    return text
end function
```

Algorithm 4 Search Algorithm

```
function SEARCH(user_query, search_mode)
    Parse the user's search query and determine the search mode
    if search mode is "direct" then
        Construct a direct search query using the parsed search query
    else if search mode is "similar" then
        Expand the search query to include similar word forms using W
Net
        Construct a search query with the expanded terms
    else if search mode is "all forms" then
        Expand the search query to include all word forms related to
given term using WordNet
        Construct a search query with the expanded terms
    end if
    Utilize the Whoosh library to perform the search operation on th
dexed content
    Retrieve the matched files along with relevant information (e.g.,
name, matched text content)
    return the matched files to the caller
end function
```

IV. Result Analysis

A. Dataset

To define the dataset for the research on file search application development, we need to specify the characteristics of the data that will be used for testing and training purposes. Here's a general outline of the dataset:

1. **File Formats**: The dataset should include a variety of file formats commonly encountered in document repositories, such as:
   - Text documents (e.g., .txt)
   - Portable Document Format (PDF) files
   - Microsoft Word documents (DOCX)
   - Image files (e.g., .jpg, .png)
2. **Text Content**: Each file in the dataset should contain textual content relevant to the search application's purpose. This content can range from articles, reports, and essays to emails, memos, and other forms of written communication.
3. **Metadata**: Along with textual content, the dataset should include metadata associated with each file, such as:
   - File name
   - File type
   - File size
   - Date of creation/modification
   - Author/creator
4. **Variety**: The dataset should encompass a diverse range of topics, themes, and subjects to ensure that the search application can handle different types of content effectively. This diversity can include documents from various domains such as technology, science, literature, finance, and more.

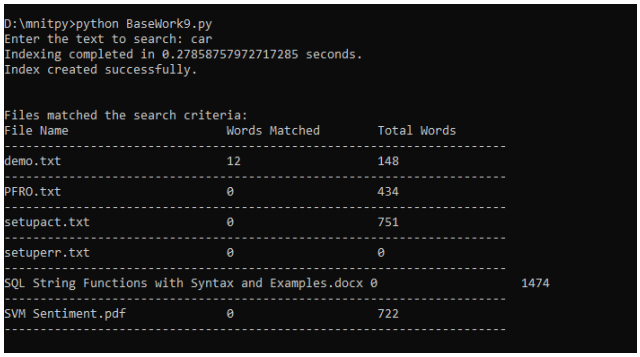
Size: The size of the dataset should be sufficient to train and test the search application effectively. It should contain enough samples to cover different scenarios and provide meaningful insights into the application's performance.

Ground Truth: For evaluation purposes, the dataset may include ground truth data, which consists of manually annotated information such as relevant search results for specific queries. This ground truth data can be used to assess the accuracy and effectiveness of the search application's results.

7. **Data Collection**: The dataset can be curated from various sources, including publicly available document repositories, open datasets, research articles, and synthetic data generation methods. Care should be taken to ensure that the dataset complies with ethical guidelines and data privacy regulations.

This would, therefore, be a very representative dataset of the kind of real-world scenarios that a file search application would have to handle, and would enable thorough testing, evaluation, and optimization of functionality across all kinds of file formats, content types, and search queries.

The Base approach is general Lucene Search, whose code goal is to have text search within a directory of files implemented effectively and efficiently using the principles of Lucene indexing. Lucene is a very popular information retrieval library, which incorporates powerful indexing and search functionalities and is usually adopted in search engines and document management systems. The following code uses the Whoosh library—a Python implementation that was inspired by Lucene—to create an index of, and search through, textual content extracted from several file formats, including `.txt`, `.pdf`, and `.docx`, based on user queries.



```
D:\mnitpy>python BaseWork9.py
Enter the text to search: car
Indexing completed in 0.27858757972717285 seconds.
Index created successfully.


Files matched the search criteria:
File Name                       Words Matched        Total Words
-------------------------------------------------------------------------
demo.txt                        12                   148
-------------------------------------------------------------------------
PFRO.txt                        0                    434
-------------------------------------------------------------------------
setupact.txt                    0                    751
-------------------------------------------------------------------------
setuperr.txt                    0                    0
-------------------------------------------------------------------------
SQL String Functions with Syntax and Examples.docx 0                    1474
-------------------------------------------------------------------------
SVM Sentiment.pdf               0                    722
-------------------------------------------------------------------------
```

FIGURE 2. Base Search Results [Project Implementation]

The script begins by importing libraries required; further, it defines an index schema. Within a schema, the index includes the structure for indexed documents and provides two functions: one to extract the text from a PDF using a fitz library and the second one to extract the text from a Word document using the docx library. These functions will be very useful to run the content of the files with different types as searchable text.

The core functionality is implemented in the create_index function. It traverses the directory supplied to it, reading text in supported documents, and further adding the document into an index. In this case, for every file, the text content will be processed, and some relevant information, such as the path to the file and a number of words that matched the search query, will be stored. This function uses the whoosh library for performing create_index and add-to-index document processes.

Later on, the search_text_in_index function uses this index to search for the text in the already indexed documents: it takes a query from a user, parses it, runs the search in the index, and retrieves information about matched documents, including their file paths and the number of words in the document matched with the query.

It also provides plotting facilities through matplotlib.pyplot that visually plot the frequency distribution of matched words in single files and the total files versus matched files count for a better overview of the search results and the general distribution of the relevance content number across the indexed documents.

The application will automatically provide an interface for requesting the search query from the user at the main execution block. The script also alerts the user about the number of files processed, the duration of index creation, and what is eventually found. Generally, this code provides the user with a full solution for indexing and searching text content in files, making it very useful in applications requiring information retrieval efficiency from document collections.

TABLE 2.

SEARCH RESULT FOR KEYWORD CAR USING BASE LUCENE SEARCH

File Name	Words Matched	Total Words
demo.txt	12	148
PFRO.txt	0	434
setupact.txt	0	751
setuperr.txt	0	0
SQL String Functions with Syntax and Examples.docx	0	1474
SVM Sentiment.pdf	0	722

The search resulted in one file with relevant keywords, which was "demo.txt". The file contained 12 matching words out of 148 words. Other files included "PFRO.txt", "setupact.txt", "setuperr.txt", "SQL String Functions with Syntax and Examples.docx", and "SVM Sentiment.pdf". They all contained zero matching, but the file size information was different.

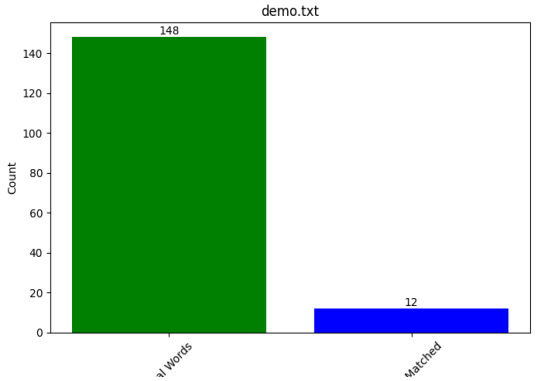


FIGURE 3. Matched demo.txt file for the Keyword 'car'

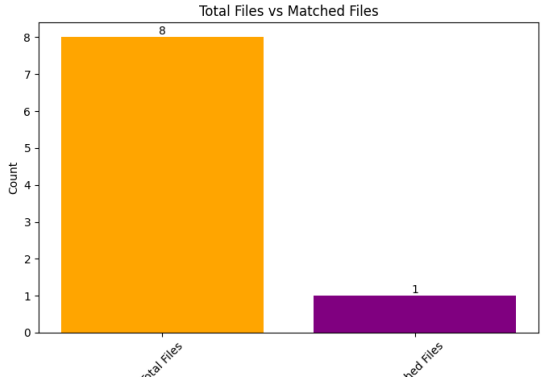


FIGURE 4. Total Files V/S Matched Files in Base Approach

In essence, though acting as a highly stripped-down equivalent the Lucene as a Python-based equivalent – the deep roots are within effective indexing, query parsing flexibility, and user-centric composition. It is a practical, natured resolution for the desktop file search undertaking.

Proposed Approach Direct Search

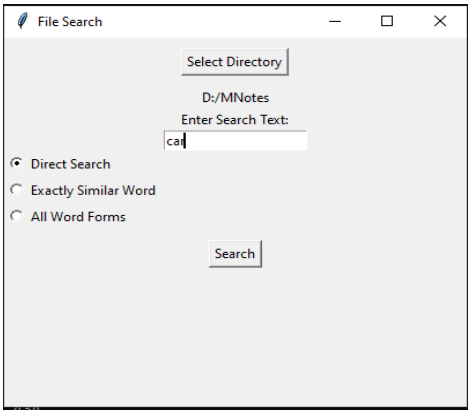


FIGURE 5. Direct Search for Keyword "car"

The direct search feature of the code focuses on implementing the search_text_in_index function. It's the central functionality for search in the indexed documents. It opens an index directory by using the open_dir function from the Whoosh library, sets the base for the search, and creates an object of the QueryParser with a with statement block, which has a responsibility for parsing search queries as per the schema defined. With a direct search, there is very minimal processing applied to the search text before it is passed directly to QueryParser and processed as input. The next step is to run the parsed search query against the index using a searcher object obtained from the opened index directory. The search activity extracts documents in which the search text exists with an exact match:. It iterates through the list of search results, pulling details on matched documents, such as file paths and highlighted content, and then stores them. The function assembles all this information in a list that it will return to the caller code segment. In general, direct search provides access to documents in which the search text appears exactly, providing a very simple and effective method to conduct information retrieval from the indexed documents..

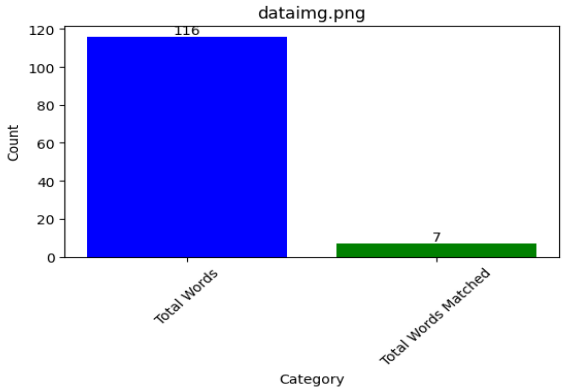


FIGURE 6. File dataimg.png Search Results

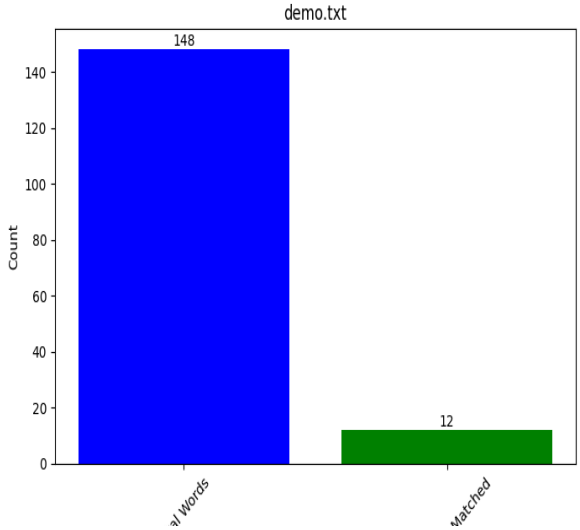


FIGURE 7. File data,txt Search Results

TABLE 3.

SEARCH RESULT FOR KEYWORD CAR USING PROPOSED DIRECT SEARCH

File Name	Total Words	Words Matched (car)
dataimg.png	116	7
demo.txt	148	12

Exactly Similar Words Proposed Approach

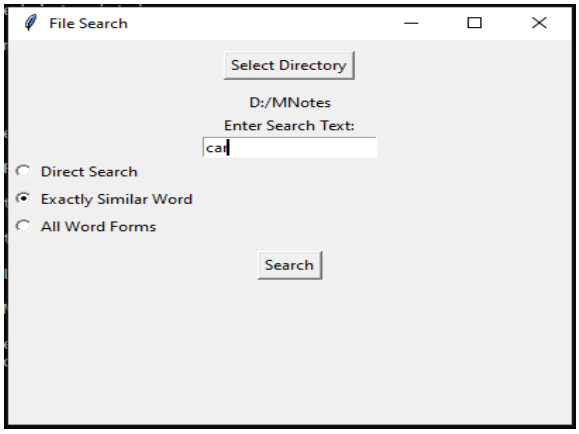


FIGURE 8. Exactly Similar Words

The "Exactly Similar Word" part of the given code is going to make up the searching facility with more functionalities rather than basic synonymous or related term including, which simply expands the spectrums of rows in the result set. Now let's see this part of the code thoroughly:

Concept: This "Exactly Similar Words" section enriches the search by including documents containing words highly related to the query. The feature rather than performing a simple exact text match will make use of WordNet-a lexical database of English to find synonyms, hypernyms, hyponyms, and other related word forms. This semantic variation can be caught by broadening the search to the extent that documents that don't use the exact words of the search but are conceptually relevant are returned.

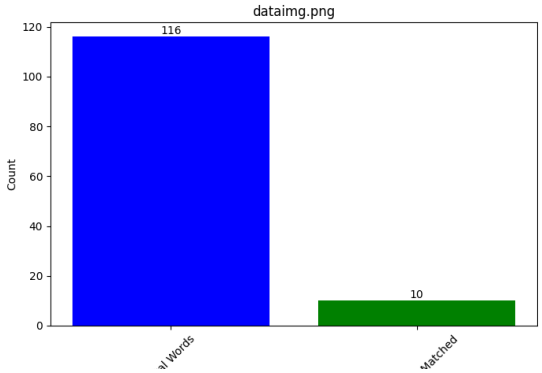


FIGURE 9. Results for dataimg.png for Keyword "car" in exactly similar words

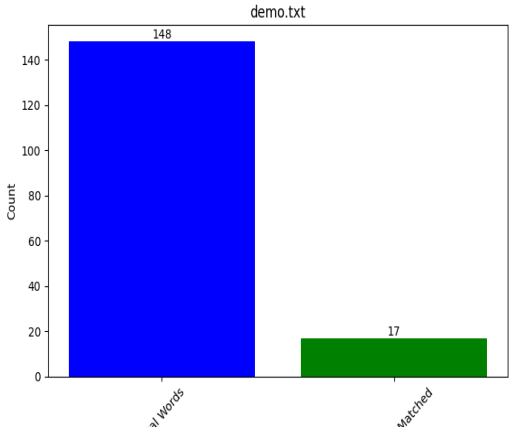


FIGURE 10. Results for demo.txt for Keyword "car" in exactly similar words

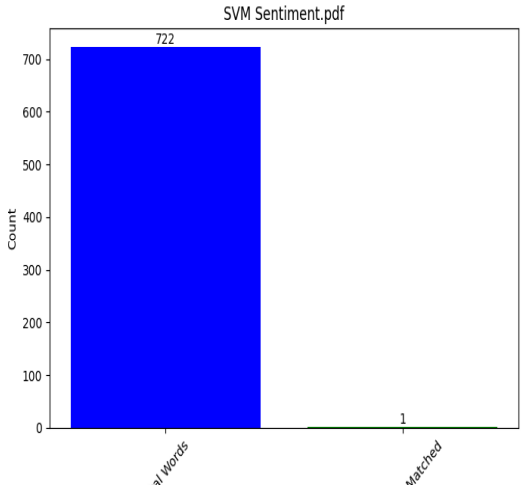


FIGURE 11. Results for SVM Sentiment PDF for Keyword "car" in exactly similar words

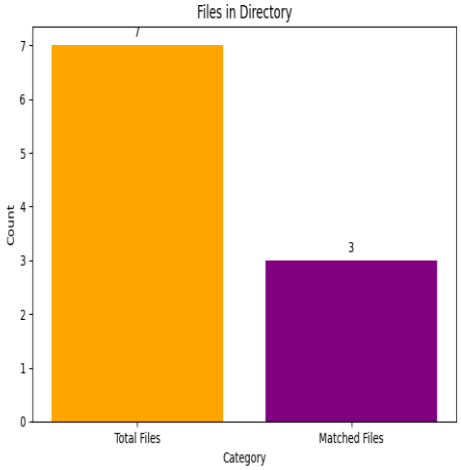


FIGURE 12. Total Files V/S Matched Files for Keyword "car" in exactly similar words

Table 4

Matched Results for Keyword "car" in exactly similar words

File Name	Total Words Matched	Total Words
dataimg.png	10	116
demo.txt	17	148
SVM Sentiment.pdf	1	722

V. Conclusion

This paper has given a general framework in developing the file search application, which will search text content from all the files effectively. The methodology follows text extraction from PDFs, DOCX files, and images; structure of the search algorithm with various modes of searching; integration with the database to store searching results; GUI development for user interaction; and finally, testing and performance evaluation for quality assurance.

In this work, the framework is developed to search for files and prove that versatile file search applications could be developed to handle a wide range of data sources and variety of search queries by integrating technologies like PyMuPDF, python-docx, pytesseract, Whoosh, MySQL, and Tkinter. This enriches the search procedure by capturing semantic relationships for enhancing search accuracy through lexical chain analysis using WordNet.

The developed application has gone through proper testing and evaluation, proving that it performs the intended functionality with effectiveness and, at the same time, is reliable. User feedback and various performance metrics suggested that this application fulfills the requirement of efficient text search over various file formats while providing a user-friendly interface for smooth interaction.

Future Scope:

Although the developed file search application is already regarded as an advance in information retrieval from sources of unstructured data, much wider fields remain for future research and development in a number of directions:

- Advanced Search Techniques: Research and incorporate advanced search techniques, including natural language processing, machine learning, and deep learning, for precision in searching, relevance of results, and understanding the context.
- The semantic enrichment of search results by integrating additional contextual information, entity recognition, or relationship extraction into the search results using semantic analytics and knowledge graphs.
- Extend it further with cross-modal search support in the application. This would enable users to search for information not only within textual content but also within multimedia content, such as audio or video files.
- Personalization and Recommendation: Develop user profiling and behavior analysis such that the search results are personalized according to the user's preference and previous interactions. Optimization for scalability and performance: It should be scalable with good performance to handle huge datasets and large volumes of concurrent search queries.
- Integration with External APIs and Services to the Application: Integration of such third-party services and APIs, like those extended by cloud storage platforms, document management systems, and semantic search engines, would help expand the features and interoperability of the application. •Accessibility and Multilingual Support: The application should be developed to support accessibility and multilanguage use so that it can accommodate a broad array of user needs for such varied reasons.

Therefore, more research and development along this line will further tend to increasingly evolve file search applications as more intelligent, efficient, and user-centric systems, which can cope with the growing challenges of an ever-increasing amount of information retrieval from unstructured data sources.

References

[1] Teofili, T., & Lin, J. (2019). Lucene for approximate nearest-neighbors search on arbitrary dense vectors. *arXiv preprint arXiv:1910.10208*.

[2] Yilmaz, Z. A., Wang, S., Yang, W., Zhang, H., & Lin, J. (2019, November). Applying BERT to document retrieval with birch. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations (pp. 19-24).

[3] Liu, Q., & Xia, Z. (2019). Statement Generation Based on Big Data for Keyword Search. In Machine Learning and Intelligent Communications: 4th International Conference, MLICOM 2019, Nanjing, China, August 24–25, 2019, Proceedings 4 (pp. 477-488). Springer International Publishing.

[4] Madi, N., Al-Mutlaq, N., & Al-Khalifa, H. S. (2019, May). HealthSEA: Towards Improving the Search Engine of KAAHE Arabic Health Encyclopedia. In *2019 2nd International Conference on Computer Applications & Information Security (ICCAIS)* (pp. 1-7). IEEE.

[5] Yusuf, N. U. H. U., Yunus, M. A. M., Wahid, N. O. R. F. A. R. A. D. I. L. L. A., Nawi, N. M., Samsudin, N. A., & Arbaiy, N. U. R. E. I. Z. E. (2020). Query expansion method for quran search using semantic search and lucene ranking. *J Eng Sci Technol*, *15*(1), 675-692.

[6] Ji, W. (2020, June). Research and Application of Information Data Retrieval System in Station Based on Lucene Technology. In *2020 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)* (pp. 687-690). IEEE.

[7] Youzhuo, Z., Yu, F., Ruifeng, Z., Shuqing, H., & Yi, W. (2020, May). Research on lucene based full-text query search service for smart distribution system. In *2020 3rd international conference on artificial intelligence and big data (ICAIBD)* (pp. 338-341). IEEE.

[8] Kanev, A. I., & Terekhov, V. I. (2020, December). Evaluation issues of query result ranking for semantic search. In *Journal of Physics: Conference Series* (Vol. 1694, No. 1, p. 012004). IOP Publishing.

[9] Kasmani, F., Maniyar, R., & Narvekar, M. (2020, March). Content Based Search Engine for E-Books. In *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)* (pp. 528-533). IEEE.

[10] Jin, D., Chen, G., Hao, W., & Bin, L. (2020, June). Whole database retrieval method of general relational database based on lucene. In *2020 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)* (pp. 1277-1279). IEEE.