Journal of Theoretical and Applied Information Technology
15th February 2024. Vol.102. No 3

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

A METHOD FOR ENHANCING WEB SEARCH RESULTS USING CONTEXT-BASED INDEXING

MENNATOLLAH MAMDOUH MAHMOUD[1], DOAA SAAD ELZANFALY[2], AHMED EL-SAYED YAKOUB [3]

[1] Information Systems Department, Faculty of Commerce and Business Administration, Helwan University, Cairo, Egypt
[2] Information Systems Department, Faculty of Computers and Artificial Intelligence, Helwan University, Cairo, Egypt
[3] Information Systems Department, Faculty of Computers and Artificial Intelligence, Helwan University, Cairo, Egypt
E-mail: [1]mennatollah.mamdouh@commerce.helwan.edu.eg, [2]doaa.elzanfaly@bue.edu.eg, [3]ahmedYakoup@fci.helwan.edu.eg

ABSTRACT

While the current search engines are deemed sufficient, they often overwhelm the user with irrelevant results. This is primarily because most indexing structures fail to incorporate the context of the query. This omission significantly impacts the effectiveness of the search results. Despite extensive research is carried out to enhance search engine indexing outcomes, the problem of retrieving the most relevant results by context still exists. This study attempts to bridge this gap and contributes with a new approach to context-aware indexing, aiming to enhance the relevancy of the retrieved documents to the user's query. Unlike traditional methods that rely solely on keywords, the proposed approach leverages document context. The effectiveness of the proposed method is evaluated based on three criteria: index size, indexing construction complexity, and the relevancy of the results. Furthermore, the proposed method is compared to the Boolean retrieval model employed by the traditional information retrieval systems. The experimental results show that the proposed method outperforms the traditional information retrieval systems in terms of the indexing size and complexity, as well as the relevancy of the results.

Keywords: *Context-based Indexing, Information Retrieval (IR), Modifying Index, Search Engine, Web Document Indexing.*

1. INTRODUCTION

With the internet offering an overwhelming amount of information, the task of finding the most relevant and precise answers to a user's query has become increasingly difficult. The core of the issue lies in the fact that users are often unable to obtain all the relevant documents in one go, necessitating the need to repeatedly modify their search terms to locate the appropriate information. [1].

The process of retrieving relevant information to a user's query, which is represented by a set of terms, from a collection of resources is referred to as Information Retrieval (IR). [2, 3]. The objective of Information Retrieval (IR) systems is to provide outcomes that are closely matched with the user's search requirements. [4].

A search engine is an online application that utilizes a Graphical User Interface (GUI) to explore and retrieve information from the internet. By utilizing keywords and phrases provided by users, it scours through an immense number of web pages and presents hyperlinks to pertinent pages [5-7]. These search engines collect, organize, assess, and process data from the web while offering a user-friendly interface for accessing resources [7]. The search results deemed relevant are those that contain answers specifically related to the user's query, even if they do not precisely match the keywords used [8]. Results that encompass synonyms, which possess the same meaning as the query keywords, are also considered relevant. However, search engines that are based on exact keyword matching do not classify these results as relevant. Consequently, some of the displayed results may lack relevance to the user, despite sharing the same context.

Typical indexing techniques are divided into four main stages. The first stage is the pre-processing stage that involves lemmatizing, tokenizing, and removing unwanted characters from texts, such as stop words, symbols, punctuation, and special characters [4]. Pre-processing is used to prepare the document for the next steps. It is used to clean the document from the unwanted and useless words as well as return the word to its root. Based on a lexical and morphological pattern of words, the potential key phrases are extracted in the second stage [9]. The second step includes eliminating words' inflectional endings and returning them to their core form, or lemma, using a dictionary and morphological analysis [4]. It is possible to lemmatize with or without a Part-of-Speech or POS tag. Each word is given a tag by a POS tag, which improves the lemma's accuracy within the context of the dataset [9]. In the third stage, important phrases that are identified from the text are given relevance scores based on numerous features [4]. Term Frequency-Inverse Document Frequency or TF-IDF is used to give weight or score to each word in the document. It provides the importance of the word in the document or the corpus [10]. Lastly, key phrase selection and ranking are based on relevance score. The words in the index are ranked based on the TF-IDF score [4]. By using conventional Information Retrieval or IR techniques, the documents are indexed based on the terms they contain rather than the concepts that explain them. There is a language gap when users and domain specialists use different terms to express the same idea. This is because there is no dictionary or semantic lexicon to minimize this gap. Ontologies can be used to minimize this gap for example, but ontology is domain-specific and cannot be used as a generic robust solution for all domains [4]. Hence, a new indexing structure is needed to allow minimizing this gap and to allow getting search results that are more relevant to the query context even if the resulted documents do not contain the exact query search keywords.

The development of an index presents several key design challenges, such as merge factors, storage techniques, index size, lookup speed and maintenance. In merge factors, the process of creating an index entail extracting information from documents, incorporating topic attributes, improving navigational capabilities, and facilitating concurrent operations by multiple indexers. All these aspects need to be validated prior to the construction or merging of the index. In storage techniques, the aim is to reduce the amount of storage required by implementing an index using filtering and compression techniques. The index size is a significant design consideration, as it denotes the storage capacity of the index. Smaller sizes can expedite the search process, while larger sizes may necessitate more time and effort. The speed of index lookup is influenced by the pace at which content is retrieved from the index, inserted into the data structure, and either deleted or modified. Index maintenance is a standard procedure that occurs with every update of a repository. The approach to maintaining indexes should be established based on the frequency of updates, whether it is weekly, monthly, or within a specific timeframe [11].

In this paper, a context-based index is contributed, which utilizes a unique indexing technique to allow for the retrieval of relevant documents based on their context or meaning, rather than relying solely on keyword matches. The proposed index allows for performing modifications, such as adding new documents to the index and deleting documents from the index that allows making search by context. To evaluate the proposed method, it has been compared with the traditional Boolean retrieval model in terms of the index size, the indexing complexity, and the search effectiveness using F-Measure.

The subsequent sections of this paper are structured as follows. Section 2 provides a succinct overview of the relevant research conducted in this field. The proposed method in this study is elaborated upon in Section 3. In Section 4, a comprehensive analysis of the proposed method is presented, which includes the examination of various parameters and a comparison with the traditional inverted index. Finally, Section 5 serves as the conclusion of the paper, summarizing the findings and suggesting avenues for future work.

2. LITERATURE REVIEW

A broad range of research has been conducted to enhance the indexing techniques of the search engines in order to provide more relevant results to the user's query.

Some research focus on improving the index structure, others use ontologies to enhance building and using the indexes for more relevant results. The following subsections review the work that has been done in each of these directions.

2.1. Improving Index Structure

Meihan Qi et al in [12] proposed an inverted index-based keyword information retrieval system that consisted of an inverted file and dictionary. The

number of inverted items in the inverted list was computed statistically, and the dictionary was separated into multiple subsets based on the order of the dictionary's keywords. Each sub-index file in the inverted index file contained only a portion of the keywords and the inverted list. To identify potential keyword errors during information retrieval and to suggest potential right terms, keyword error correction was utilized. Abdulla Kalandar Mohideen et al in [13] and [14] proposed a new graph-based indexing (GBI) method for big data systems. It makes advantage of a directed graph structure to efficiently detect the simultaneous use of several keywords in the same text. The goal is to effectively retrieve all Boolean AND query results at once by using the relationship between the search phrases recorded in the graph structure.

Zhimin Wei et al in [15] developed a novel text re-generation semantic retrieval approach using a pretraining representation model and an inverted index. The model can accurately complete semantic retrieval while retaining the properties of semantic vectors. In addition, it responds quickly and uses less resources. It features high availability, monitoring tools, and offers distributed solutions in addition to its vector data management function. Dilip Kumar Jang Bahadur Saini et al in [11] proposed a novel inverted index model that comprises three modules: the Inverted module, the main inverted module, and the deleted file list module. These modules contribute to enhancing the efficiency of both index creation and searching processes. In comparison to traditional systems, the proposed method demonstrates superior performance when documents are added, deleted, or modified within the repository. This is due to the continuous monitoring of the repository by the current system, which promptly updates the indices in response to any changes, resulting in improved performance. Wentao Xu et al in [16] developed a searching system named SAES, which is a distributed suffix index scheme that possesses the ability to build online and merge offline, thereby accommodating the scalability requirements of the Elastic Search architecture. They replace the conventional inverted index in Elastic Search with the suffix index, enabling efficient full-text searches on large-scale datasets. The suffix array, constructed by sorting all the suffixes of the data, serves as the fundamental component of the suffix index.

Dany Widiyatmoko and Agus Setiyono in [17] proposed a method for indexing the documents using Term Frequency- Inverse Document Frequency (TF-IDF). In the proposed model, the documents are stored in the index as vectors weighted by TF-IDF. In search process, cosine similarity is calculated between the query and documents vectors.

The problem with most of the current inverted index structures is that they do not provide context to the user's query. They do not return the related documents that relate to the query by context, as well. Graph-based indexing (GBI) disadvantage is the necessity of frequently updating the relationship between words to obtain correct search results. The GBI also consumes more memory and high indexing time than the Inverted Index. Moreover, in both cases, the search process for a query relies on exact matching between the query keywords and the keywords of the documents, it does not consider context for retrieving documents.

2.2. Combining Ontology with Information Retrieval System

Binbin Yu in [18] used a domain ontology based on information retrieval model with a genetic algorithm. The genetic algorithm was used to determine the ideal combination of word frequency weighted parameters. The ontology server, information database, query transition and retrieval agent modules, together with ontology document processing and retrieval, are all included. While Anil Sharma and Suresh Kumar in [4] used both a domain ontology and machine learning to generate a hybrid semantic document indexing method. In this model, a unique concept ranking method based on various aspects, where statistical, semantic, and scientific named entity properties of the idea are employed to weight the annotations' importance. These feature weights' parameters are derived using a fuzzy analytical hierarchy technique. Kamal El Guemmat and Sara Ouahabi in [19] suggested developing a new educational search engine based on traditional, ontological, semantic, and metadata indexing that enables us to index the concepts related to particular use cases while also taking into account the independent uses or generics of a given domain and while also taking the deep extraction of the concepts into consideration. the proposed search engine index first performs preprocessing as the traditional index then the lemmatized terms are represented according to an ontology in RDF. The engine enhances the indexing process through a deeper concept extraction that makes it possible to identify the relationships between the notions of application and specific usage. Nay Nandar Linn and Thinn Thinn Win in [20] proposed a semantically enhanced model for conducting searches over RDF Graph. The model aims to integrate and leverage extensively formalized semantic information, represented in the

form of RDF and Knowledge Base (KB), into conventional Information Retrieval (IR) ranking models. The semantic information retrieval model can be perceived as a progression from the conventional keyword-based retrieval techniques, wherein a semantic knowledge base is employed to substitute the keyword-based index. The model accepts keyword queries pertaining to the domain of computer science research fields from the user. Shivani Jain et al [21] in proposed a fuzzy ontology framework in the process of query expansion for information retrieval systems. It establishes a lexicon of concepts within a particular domain (solar domain) and assigns fuzzy membership utilizing the ConceptNet Global Ontology. The fuzzy membership function incorporates semantic weights and ConceptNet edge weights to compute membership values, subsequently employed as edge weights connecting various concepts within the fuzzy ontology. The proposed framework facilitates the recognition of interconnected concepts within a domain, thereby expanding queries and establishing a semantic web for query context. Assma Boughoula et al in [22] introduced a novel semantic structure, known as the Browsable Concept Index (BCI), which aims to enhance the organization of educational content by incorporating semantic relationships. The BCI is composed of a set of representative concepts that are interconnected in a graph-like structure. The strength of association between two concepts is indicated by weighted edges, while each concept node is linked to specific text segments within a collection of documents that pertain to that concept. The BCI effectively consolidates dispersed textual content and facilitates efficient access for users through various means, such as semantic browsing, exploratory and targeted information seeking, and collaborative information seeking. Additionally, they proposed an optimization-based algorithm that automates the creation of a BCI using a collection of educational textual content. Shanshan Jiang et al in [23] proposed a hybrid indexing approach, which integrates both automated and manual linking techniques to establish connections with concepts. The proposed methodology exhibits enhanced search efficacy, particularly in cases where dataset descriptions are deficient or ambiguous. Moreover, it diminishes the necessity for manual linking and domain expertise. The proposed approach is universally applicable to any ontology, and the process of linking concepts to diverse lexical databases facilitates the provision of multi-language support. Consequently, datasets in distinct natural languages can be explored within a unified framework.

The main drawback of using ontology is that ontology is limited to one domain or field. So that we need multiple ontologies to enhance the whole search engine in different fields.

This research aims to solve this gap by putting forth a new methodology for context-aware indexing, with the purpose of elevating the retrieval of the relevant documents based on user queries.

3. THE PROPOSED METHOD

The proposed method is based on a Context-based Web Document Indexing method (CWDI) that enhances the relevancy of the web query search results. The proposed method includes three main steps: Document preprocessing, document categorization and labeling, and building and modifying the index. The proposed method is shown in Figure 1.

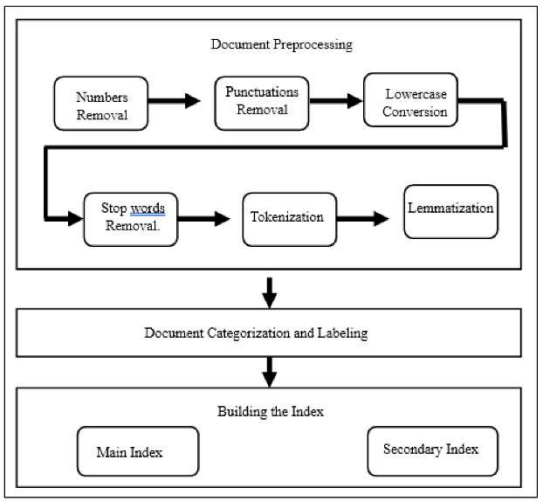

Figure 1: The Proposed Method

As shown in the Figure 1, the web document is first preprocessed. Preprocessing includes removing numbers and punctuation from the document, converting the document letters to lowercase, and removing the stopwords from the document. After that, the words of the document are tokenized, or the document text is segmented to words and then lemmatization is performed or considering grammatical properties for converting the words to their root. After preparing the document or preprocessing it, the document is then categorized in the second step. Categorization means classifying the documents according to a known class or category. The document is categorized into two

levels, domain, and area. For example, the document is categorized in "Computer Science" domain and "Machine Learning" area. The last step is building and modifying the index. The index is built based on the preprocessed documents with their domain and areas categories. Modifying the proposed index includes performing index updates, such as inserting new document concepts and terms in the index, updating document contents in the index, and deleting documents. After building the index, it is now ready to take the query from the user and provide him with the results. Each step is discussed in detail in the following subsections.

3.1. Document Preprocessing

In this step, the significant keywords are extracted for building the index. Document preprocessing includes several processes, such as removing numbers, white spaces, punctuations, converting text to lowercase, and removing the stop words [15]. After that, each chunk of text is divided into tokens by the process of tokenization, which can be done with words, phrases, symbols, or other significant components. Finally, Lemmatizing the text by combining inflected forms into a single fundamental form. This method entails removing prefixes and suffixes from a word's basic form. Using the dictionary and morphological analysis, it involves stripping words of their inflectional ends and restoring them to their basic form, or lemma. Because various lemma forms may or may not arise often during training, lemmatization's primary objective is to reduce sparsity [24].

3.2. Document Categorization and Labeling

After preprocessing the documents, documents are then categorized based on their domain. Each domain has its class of related documents. For each domain class, the area is obtained. In other words, there are two levels of categorizing the document. The first level is categorizing the document based on their domain. The second categorization level is within each domain where documents are categorized based on the area. The documents are classified based on predefined domains and areas. To illustrate the difference between the domain and the area, consider this example: "data mining" concept belongs to "Machine Learning" area under the domain of "Computer Science".

3.3. Index Building

The following sub sections describe the process of building the main index, the secondary index and merging both indexes.

3.3.1. Building the main index

The preprocessed documents with their domain and areas categories are the input of the index building stage. First, the concepts are extracted from the preprocessed documents. The concepts are extracted through getting the representative terms provided by the web document and usually separated by a separator. Then the extracted concepts are used to construct the index. The index is constructed as an inverted index. For fast retrieval, the inverted index takes the form of a dictionary of Keys and values. As illustrated in Figure 2, the important concepts that represent the domain are the keys and the values contain the sub-domain concepts. The important concepts are extracted by a concept extraction technique as in [25, 26].

Key	Value
annotation capture	['CS , Structured Storage ']
annotation tool	['Psychology , Nonverbal communication ']
annoyance	['Psychology , Person perception ']
annual cost	['ECE , Electrical network ', 'ECE , PID controller ']
annual cost system	['ECE , PID controller ']
annual load	['Civil , Water Pollution ']
annual payroll	['Psychology , Attention ']
annual relapse rate	['Medical , Multiple Sclerosis ']
annual simulation design	['Civil , Green Building ']
annual survival	['ECE , Satellite radio ']
annual wellness visit	['Medical , Medicare ']
annualized cost system	['MAE , Internal combustion engine ']
annualized relapse rate	['Medical , Multiple Sclerosis ']
annular boundary	['MAE , Fluid mechanics ']
annular combustion chamber	['CS , Image processing ']
annular domain	['Civil , Suspension Bridge ']
annular leu metallic fuel	['MAE , Hydraulics ']
annular nozzle ejector	['ECE , Electric motor ']
annular ozone generator	['ECE , Electrical generator ']
annular ring antenna	['ECE , Satellite radio ']

Figure 2: A Sample of the Proposed Inverted Index Structure

Another version of the proposed index is built in the form of B-Tree data structure.

B-tree is another data structure that is used for storing a large amount of data, so it is used for indexing purposes. It consists of root node, leaf nodes with keys and child nodes. Figure 3 shows a sample of the proposed index in B-tree structure. The sample of the proposed index structure in figure 2 is illustrated in the B-tree structure in figure 3. B-tree is used in Database Management Systems for indexing and making the search more efficient. The advantages of using B-Tree are that it is fast in retrieving the data from the index, the index size is small and allows the insert, update and delete operations to be executed in logarithmic time, which is fast.

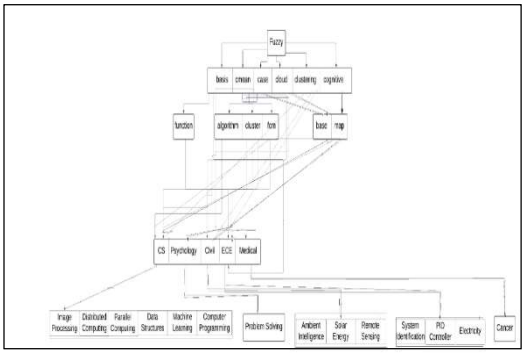

Figure 3: A Sample of the Proposed Index in B-Tree Structure

Algorithm 1: Constructing the Proposed Main Inverted Index

Input:
Class Labels or domain (CL)
Class Areas (AR)
Classes containing Preprocessed Documents (C)
Output:
Inverted Index (I)
Variables:
Unique concepts of class j (UCj)=Ø
Unique concepts of all classes (UC)=Ø
Inverted Index (I)=Ø
Method:
Repeat:
/ Extract unique concepts uc from each class */*
N= Total number of classes.
Loop (C from 1 to N)
Loop (Doc in C)
uc=Doc.Split (doc, separator)
UCj . Add (uc)
End loop
End loop
*/*Build the index*/*
UC=UCj of all classes
Loop (UC)
Loop (u in UC)
If u not in I
I. Add (u, UC_CL, UC_AR)
End if
End loop
End loop

3.3.2. Maintainting the proposed index

The main index is the primary structure from where the concepts and the terms are matched, and the relevant documents are retrieved consequently. All the terms and concepts of the documents are stored in the main index and any users' queries are searched in the main index. Any updates in the documents, such as inserting a new document in the index or updating document content are conducted in the secondary index. Then the secondary index is merged with the main index for further search.

Maintaining the proposed index includes performing index updates, such as inserting new document concepts and terms in the index, updating document contents in the index, and deleting documents.

3.3.2.1. Inserting new documents

When inserting a new document in the index, the same steps made in constructing the main index are performed again for each new document. The new documents are first preprocessed. The important concepts are then extracted from the preprocessed documents by obtaining representative terms from the web document, which are typically separated by

separators. The extracted concepts are then used to construct the secondary index, that is structured as an inverted index data structure, with a dictionary as the main index. The secondary inverted index is composed of the relevant concepts as keys, and the following values (such as class domain and concept area/sub-domain) within a dictionary data structure.

3.3.2.2. Updating existing documents

The procedures of updating document content in the index are the same as inserting a new document to the index. In traditional index, updating documents means removing the old postings of the document and inserting the new ones as discussed in [11]. In the proposed indexing model, updating the document process means keeping the old concepts in the index and inserting the new ones. This keeps the context of the document.

3.3.2.3. Deleting documents

When deleting the documents from the corpus, their concepts are not deleted from the index. This gives more accurate result when searching a query, because the document context exists in the index. In contrast to the traditional index that when removing the document, the corresponding postings are deleted.

The secondary index is an index that is constructed in case of any document insertion or updating. After constructing the secondary index, it is then merged with the main index. The secondary index is considered as a transitive index that holds only any new insertion or updates. The secondary index consists of two phases. The first phase is constructing the secondary index, its pseudocode is the same as building the main index, shown in Algorithm 1. The second phase is taking the data from the secondary index and putting it in the main index or updating the main index for further queries. Algorithm 2 demonstrates the pseudocode for updating the main index or the merging process.

Algorithm 2: Updating The Main Index

Input:
- The Main Index (MI)
- The Updating Index (UI)

Output:
- Updated Main Index (UMI)

Method:
/ Updating the main index */*
MI.update (UI)

Return UMI

4. EXPERIMENTAL RESULTS AND EVALUATION

The datasets that are used to evaluate the proposed main index and the secondary index are WOS benchmark dataset downloaded from Mendeley Data [27] and Research papers from WOS, Scopus, IEEE, ACM and Google Scholar benchmark datasets, downloaded from Kaggle [28] and Zenodo [29, 30], respectively. The three datasets are then combined to form one integrated dataset. The Web of Science (WOS) dataset is a document classification dataset that contains 46,985 documents with 7-parent categories and 134 sub-categories. The dataset contains research papers from different domains, such as Computer Science (CS), Electrical Engineering (ECE), Psychology, Mechanical Engineering (MAE), Civil Engineering, Medical Science, and biochemistry. Each domain has areas or subdomains such as computer graphics in CS domain. The second dataset is a combination of three benchmark datasets, called miscellaneous dataset. They are all research papers from WOS, Scopus, IEEE, ACM, and Google Scholar. It contains 1,353 documents with 4 parent categories and 17 sub-categories from different domains, such as Computer Science (CS), Chemistry, Multidisciplinary Science and Engineering. Each domain has areas or subdomains such as physical area in chemistry domain.

4.1. Evaluation Measures

The proposed method is evaluated using three criteria which are the index size, the indexing complexity, and the search effectiveness. Search effectiveness is measured using F-Measure.

4.1.1. Index size metric

It is important to calculate the index size to determine the quality of the search engine index. The index size is calculated as a ratio between the index size and the whole dataset. Small index size indicates better quality. This is because it doesn't take much space in the desk and its speed in retrieving the documents is faster. The index size is described in equation 1.

$$Index\ Size\ in\ KB = \frac{Index\ size\ in\ KB}{The\ Dataset\ Size\ in\ KB} \quad (1)$$

4.1.2. Index construction complexity

The complexity for index is evaluated using Big O notation. The big O notation is used to discuss an algorithm's running time, it describes how long it

takes to complete an action. The running time big O terms measures the worst-case running time, or the longest that an input of a given length might possibly take [31, 32].

4.1.3. Search accuracy metric

The fundadamental fundamental metrics used to assess search strategy effectiveness are recall and precision [33]. F-measure mearges presision and recall into a sigle score [34].

Recall is the ratio of relevant records that were successfully retrieved to all relevant entries in the database. Typically, a percentage is used to indicate it. Recall is described in equation 2 [33].

$$Recall = \frac{Relevant\ Document\ Retrieved}{Relevent\ Document\ in\ the\ Collection} \quad (2)$$

Precision is the ratio of the number of relevant records to the total number of relevant and irrelevant records retrieved. Typically, a percentage is used to indicate it. Precision is described in equation 3 [33].

$$Precision = \frac{Relevant\ Documents\ Retrieved}{Total\ Retrieved\ documents} \quad (3)$$

Recall is based on the relevant records in the collection, precision is dependent on the documents that were retrieved.

High recall involves determining that all the retrieved records are relevant, whereas high precision simply involves retrieved records [33].

F-Measure is a weighted harmonic mean between precision and recall. It is a measurement that trades off precision versus recall. F-Measure is described in equation 4 [34, 35].

$$F - Measure = \frac{2*(Precision * Recall)}{Precision+Recal} \quad (4)$$

The F-Measure ranges from 0 to 1, with higher values indicating better performance [35].

4.2. Experimental Results

To evaluate the proposed method, its performance has been compared to the standard Boolean retrieval model in terms of the index size, its indexing complexity and search effectiveness. The enhanced indexing algorithm is implemented using Python.

4.2.1. Index size evaluation

Tables 1 and 2 illustrate the evaluation in terms of the index size of its creation and after deleting 10% of the documents from it, resprctively. They show the comparison between the proposed index in the form of inverted index dictionary structure, the proposed index in the form of B-Tree structure and the traditional inverted index. This comparison is conducted for the two datasets based on equation 1.

It is noticed from Table I that the proposed B-Tree Index has the smallest index size. This has the advantage of increasing the speed of retrieving the document from the index. The proposed B-Tree Index has the smallest index size, because the concepts, the domain, and the areas are stored without any redundancy. The data is linked together through the tree pointers. While in the traditional index in dictionary data structure, the dictionary value (or the location of the word) is stored in a redundant manner based on each word location in documents.

Table 1: Main Index Size Of The Two Datasets

Datasets/Indexes	Percentage of the Index Size to the Whole Dataset		
	The Proposed Inverted Index	***The Proposed B-Tree Index***	***The Traditional Inverted Index***
Miscellaneous Dataset	12.1%	1.3%	4.7%
WOS Dataset	55.5%	2.8%	12.6%

Table 2 reveals that the B-Tree Index proposed exhibits the smallest index size, thereby offering the advantage of enhancing the retrieval speed of documents from the index. Even after removing postings from the conventional index, its size remains larger than that of the proposed B-Tree Index. The B-Tree data structure employed in the proposed index ensures a compact index size and more precise outcomes by refraining from deleting concepts from the index when documents are deleted.

Table 2: Main Index Size Of The Two Datasets After Deleting 10% Of Data

Dataset/ Indexes	Percentage of the Index Size to the size of the whole dataset		
	The Proposed Inverted Index	***The Proposed B-Tree Index***	***The Traditional Inverted Index***
Miscellaneous Dataset	12.1%	1.3%	3.88%
WOS Dataset	55.46%	2.8%	11.3%

4.2.2. Index complexity evaluation

It is shown from Table 3 that, in both datasets, the proposed inverted index and the traditional inverted index have the same time complexity $O(n^2)$ where n is the size of input concepts extracted from documents. While the proposed B-Tree index has smaller time complexity O (log n). Comparing the time complexity calculated in Table 3 indicates that the proposed B-Tree index has the best indexing complexity. The proposed B-Tree index has smaller time complexity because the B-Tree data structure maintains balance, which means that each node has a minimum number of keys, so the tree will always be balanced. In the worst case of constructing the B-Tree, for inserting several new keys, represented in concepts, the tree can split to leaf node which takes O (log n) time complexity. In other words, the number of construction steps is reduced when the number of concepts (or keys) increases. While in the proposed inverted index in dictionary structure and the traditional index, nested for loops are required to insert keys or concepts or construct the index. Nested for loop requires quadratic time complexity represented as $O(n^2)$.

Table 3: Main Index Building Complexity Of The Two Datasets

Index complexity of:	The Proposed Inverted Index	The Proposed B-Tree Index	The Traditional Inverted Index
The miscellaneous dataset	$O(n^2)$	O (log n)	$O(n^2)$
The WOS dataset	$O(n^2)$	O (log n)	$O(n^2)$

It is shown from table 4 that, in both datasets, the proposed inverted index and the traditional inverted index have the same time complexity $O(n^2)$ for creating the secondary index and rebuilding the inverted index respectively, where n is the size of input concepts extracted from documents. While the proposed B-Tree index has smaller time complexity O (log n). Comparing the time complexity calculated in table 4 indicates that the proposed B-Tree index has the best indexing complexity. The proposed index requires two measurements for the secondary index complexity, one for constructing the secondary index $O(n^2)$ and the other for merging the main index and the secondary index O (1). Merging the main and the secondary index in the proposed inverted index is faster than updating the traditional index reconstruction.

Table 4: Secondary Index Building And Merging Complexity Of The Two Datasets

Index complexity of	The Proposed Inverted Index	The Proposed B-Tree Index	The Traditional Inverted Index
The miscellaneous dataset	$O(n^2) + O(1)$	O (log n)	$O(n^2)$
The WOS dataset	$O(n^2) + O(1)$	O (log n)	$O(n^2)$

4.2.3. Search accuracy evaluation

After applying the proposed method on each of the previously mentioned datasets, four Search Queries are used to evaluate the performance of the search of proposed index using the WOS dataset. The Proposed index is also evaluated after deleting 10% of its documents using the same four queries compared to that of the traditional index.

4.2.3.1. The WOS dataset

The WOS Dataset consists of research papers from WOS with different domains and areas. Evaluating the proposed method is done through four queries. The queries are not benchmark, instead they are made based on many different criteria. Criteria that are used to generate the queries involves generating queries with different Term Frequency (TF) for the query terms, different query length, and different query types in IR systems.The queries that are used to evaluate the proposed method are:

i. Machine Learning Supervised, Semi-supervised, Unsupervised Learning Methods.
ii. Distributed Computing
iii. Machine Learning Algorithms and Applications
iv. Database Applications

The proposed method is evaluated based on the relevant result set for the queries. They are obtained by experts.

Figure 4 demonstrates the evaluation of the four queries on the WOS dataset of the proposed index, traditional inverted index using (AND) and traditional inverted index using (OR) in terms of F-measure. Figure5 demonstrates the evaluation of the four queries on the WOS dataset of the proposed index, traditional inverted index using (AND) and traditional inverted index using (OR) in terms of F-measure after removing 10% of the documents from the index.

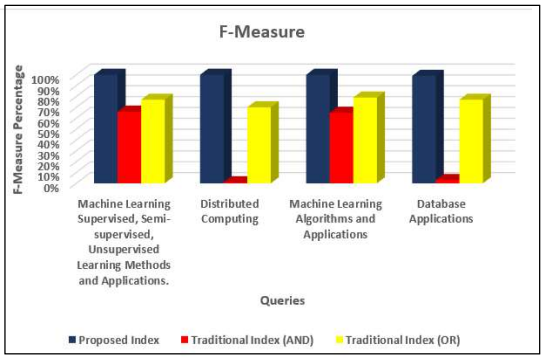

Figure 4: The Evaluation of The Four Queries of the WOS dataset of The Proposed Index, Traditional Inverted Index Using (AND) and Traditional Inverted Index Using (OR) In Terms of F-Measure

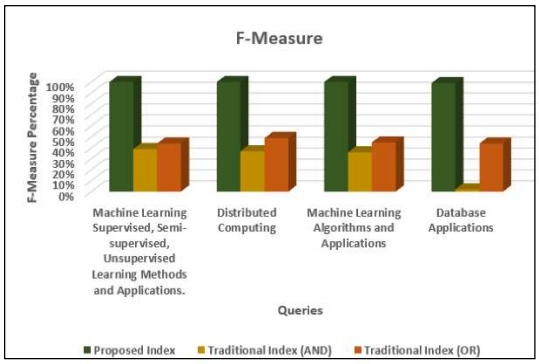

Figure 5: The Evaluation of The Four Queries of the WOS dataset of The Proposed Index, Traditional Inverted Index Using (AND) and Traditional Inverted Index Using (OR) In Terms of F-Measure after removing 10% of docs

As shown in Figures 4 and 5 the proposed method has the highest F-measure value. High F-measure value indicated better performance or the retrieved documents are relevant to the user query. The proposed index provides the highest F-measure value because the retrieved documents are relevant to the query by context. So that the proposed method is better than the traditional inverted index using both Boolean search (AND) and (OR). The proposed index retrieves the relevant documents by context. This is done by retrieving the area that is relevant to the query. It does not perform exact keyword matching with the documents. So that the proposed method is better than the traditional inverted index using both Boolean search (AND) and (OR).

5. CONCLUSION AND FUTURE WORK

In this paper, a Context-based Web Document Indexing (CWDI) method is proposed for enhancing the results of the search engines to get the most relevant information according to received query. The method is built based on a modified indexing method. The proposed method uses context or concepts in the index building algorithm which provides the user with the most relevant information in English documents. Two benchmark datasets are used to build and evaluate the index. Many different evaluation measures are used to evaluate the proposed method including the index size, the indexing complexity as well as the quality of results. The evaluation is also conducting when the index has been modified or updated. The experimental results show that the proposed method outperforms the traditional Information Retrieval systems. The proposed method offers an improved indexing algorithm to deliver the most pertinent results.
As a future work, we are planning to adopt a new searching and ranking algorithm to provide better ranking of the returned documents.

REFERENCES

[1] S. P. Panda and J. P. Mohanty, "A Domain Classification-based Information Retrieval System," in 2020 IEEE International Women in Engineering (WIE) Conference on Electrical and Computer Engineering (WIECON-ECE), Bhubaneswar, India, 2020, pp. 122-125.
[2] Y. Djenouri, A. Belhadi, D. Djenouri, and J. C.-W. Lin, "Cluster-based information retrieval using pattern mining," Applied Intelligence, vol. 51, pp. 1888-1903, 2021.
[3] D. Mahapatra, C. Maharana, S. P. Panda, J. P. Mohanty, A. Talib, and A. Mangaraj, "A Fuzzy-Cluster based Semantic Information Retrieval System," in 2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 2020, pp. 675-678.
[4] A. Sharma and S. Kumar, "Machine learning and ontology-based novel semantic document indexing for information retrieval," Computers & Industrial Engineering, vol. 176, pp. 1-13, 2023.
[5] S. Shahi, A. Shukla, and S. Rastogi, "Search Engine Techniques: A Review," Journal of Natural Remedies, vol. 21, pp. 48-55, 2020.
[6] R. S. T. Lee, "Ontological-Based Search Engine," in Artificial Intelligence in Daily Life, ed Singapore: Springer Singapore, 2020, pp. 193-241.
[7] D. Sharma, R. Shukla, A. K. Giri, and S. Kumar, "A Brief Review on Search Engine Optimization," in 2019 9th International Conference on Cloud Computing, Data Science

& Engineering (Confluence), Noida, India, 2019, pp. 687-692.
[8] L. J. Sankpal and S. H. Patil, "Rider-Rank Algorithm-Based Feature Extraction for Re-ranking the Webpages in the Search Engine," The Computer Journal, vol. 63, pp. 1479–1489, 2020.
[9] T. Sreemany. (2022, 3-12-2023). Essential Text Pre-processing Techniques for NLP! Available: https://www.analyticsvidhya.com/blog/2021/09/essential-text-pre-processing-techniques-for-nlp/
[10] riturajsaha. (3-12-2023). Understanding TF-IDF (Term Frequency-Inverse Document Frequency). Available: https://www.geeksforgeeks.org/understanding-tf-idf-term-frequency-inverse-document-frequency/
[11] D. K. J. B. Saini, P. Patil, K. D. Gupta, S. Kumar, P. Singh, and M. Diwakar, "Optimized Web Searching Using Inverted Indexing Technique," in 2022 IEEE 11th International Conference on Communication Systems and Network Technologies (CSNT), Indore, India, 2022, pp. 351-356.
[12] M. Qi, W. Fang, Y. Zhao, Y. Sha, and V. S. Sheng, "Research on Key Word Information Retrieval Based on Inverted Index," in Advances in Artificial Intelligence and Security 8th International Conference on Artificial Intelligence and Security, ICAIS, China, Qinghai, 2022, pp. 392-404.
[13] A. K. Mohideen, S. Majumdar, M. St-Hilaire, and A. El-Haraki, "A Graph-Based Indexing Technique to Enhance the Performance of Boolean AND Queries in Big Data Systems," in 2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID), Melbourne, VIC, Australia, 2020, pp. 677-680.
[14] A. K. Mohideen, S. Majumdar, M. St-Hilaire, and A. El-Haraki, "A Data Indexing Technique to Improve the Search Latency of AND Queries for Large Scale Textual Documents," in 2020 IEEE/ACM International Conference on Big Data Computing, Applications and Technologies (BDCAT), Leicester, UK, 2020, pp. 37-46.
[15] Z. Wei, X. Xu, C. Wang, Z. Liu, P. Xin, and W. Zhang, "An Index Construction and Similarity Retrieval Method Based on Sentence-Bert," in 2022 7th International Conference on Image, Vision and Computing (ICIVC), Xi'an, China, 2022, pp. 934-938.
[16] W. Xu, H. Chen, Y. Huan, X. Hua, and G. Nong, "Full-text search engine with suffix index for massive heterogeneous data," Information Systems, vol. 104, pp. 1-11, 2022.
[17] D. Widiyatmoko and A. Setiyono, "Information Retrieval of Physical Force Using the TF-IDF," in 2019 International Conference on Information and Communications Technology (ICOIACT), Yogyakarta, Indonesia, 2019, pp. 519-522.
[18] B. Yu, "Research on information retrieval model based on ontology," EURASIP Journal on Wireless Communications and Networking, vol. 2019, pp. 1-8, 2019.
[19] K. E. Guemmat and S. Ouahabi, "Towards a new educational search engine based on hybrid searching and indexing techniques," in 2019 1st International Conference on Smart Systems and Data Science (ICSSD), Rabat, Morocco, 2019, pp. 1-5.
[20] N. N. Linn and T. T. Win, "Efficient Semantic Web Data Searching Using Virtual Documents Algorithm," International Journal of Innovative Science and Research Technology, vol. 5, pp. 298-303, 2020.
[21] S. Jain, K. R. Seeja, and R. Jindal, "A fuzzy ontology framework in information retrieval using semantic query expansion," International Journal of Information Management Data Insights, vol. 1, pp. 1-15, 2021.
[22] A. Boughoula, K. Ros, and C. Zhai, "An Optimization Approach to Automatic Construction of Browsable Concept Index for Organizing Online Educational Content," in 2022 IEEE International Conference on Knowledge Graph (ICKG), Orlando, FL, USA, 2022, pp. 22-31.
[23] S. Jiang, T. F. Hagelien, M. Natvig, and J. Li, "Ontology-Based Semantic Search for Open Government Data," in 2019 IEEE 13th International Conference on Semantic Computing (ICSC), Newport Beach, CA, USA, 2019, pp. 7-15.
[24] A. Kathuria, A. Gupta, and R. K. Singla, "A Review of Tools and Techniques for Preprocessing of Textual Data," in Computational Methods and Data Engineering Proceedings of ICMDE 2020, Singapore, 2021, pp. 407-422.
[25] J. Yu, R. Chen, L. Xu, and D. Wang, "Concept extraction for structured text using entropy weight method," in 2019 IEEE Symposium on Computers and Communications (ISCC), Barcelona, Spain, 2019, pp. 1-6.

[26] L. Zhao, Z. Miao, C. Wang, and W. Kong, "An Unsupervised Keyword Extraction Method based on Text Semantic Graph," in 2022 IEEE 6th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), Beijing, China, 2022, pp. 1431-1436.
[27] K. Kowsari, D. Brown, M. Heidarysafa, K. J. Meimandi, M. Gerber, and L. Barnes. (2018, 20-7-2023). Web of Science Dataset (6 ed.). Available: https://data.mendeley.com/datasets/9rw3vkcfy4/6
[28] Z. Saad. (2022, 20-7-2023). Textual Dataset of Articles From WOS and Scopus. Available: https://www.kaggle.com/datasets/zakriasaad1/learn-to-prepare-data-for-machine-learning
[29] M. Mafruchati. (2022, 20-7-2023). dataset of scopus. Available: https://zenodo.org/record/6375278#.Y_DPXojRZzA
[30] L. Heiland, M. Hauser, and J. Bogner. (2023, 20-7-2023). Design Patterns for AI-based Systems: A Multivocal Literature Review and Pattern Repository (2 ed.). Available: https://zenodo.org/record/7733302
[31] S. Rubinstein-Salzedo, "Big O Notation and Algorithm Efficiency," in Cryptography, ed Cham: Springer International Publishing, 2018, pp. 75-83.
[32] S. Bae, "Big-O Notation," in JavaScript Data Structures and Algorithms: An Introduction to Understanding and Implementing Core Data Structure and Algorithm Fundamentals, ed Berkeley, CA: Apress, 2019, pp. 1-11.
[33] M. Arora, U. Kanjilal, and D. Varshney, "Evaluation of information retrieval: precision and recall," International Journal of Indian Culture and Business Management, vol. 12, pp. 224-236, 2016.
[34] C. D. Manning, P. Raghavan, and H. Schütze, "Evaluation in information retrieval," in An Introduction to Information Retrieval, ed Cambridge, England: Cambridge University Press, 2009, p. 192.
[35] (1-12-2023). F-score. Available: https://deepchecks.com/glossary/f-score/