International Journal of Engineering & Technology, 7 (2) (2018) 495-505

SPC

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET
doi: 10.14419/ijet.v7i2.9544
Research paper

SodhanaRef: a reference management software built using hybrid semantic measure

Mandava Kranthi Kiran [1] *, Dr. K Thammi Reddy [2]

[1] *Assistant Professor, Anil Neerukonda Institute of Technology and Sciences*
[2] *Professor, Gitam Institute of Technology, GITAM University*
**Corresponding author E-mail: mkranthikiran.cse@anits.edu.in*

Abstract

Reference management softwares are widely used by the researchers to maintain their collection of scholarly literature that exist in PDF format. Though widely used most of the reference management softwares have no sophisticated Information retrieval except few which offer advanced search that includes search for title, author etc., These softwares in the present day market do not give importance to the semantic similarity or relatedness concept, query expansion and finding the context within the query to find the concept behind the user mentioned query.

With SodhanaRef, a solution is offered to deal with the above-mentioned issues by building reference management software using a mix up of corpus-based and knowledge-based semantic measures. Based on the evolution done on about 200 various scholarly literatures in the PDF form, SodhanaRef shows a good performance over Mendeley when compared between these two reference management softwares for title search. The other evaluations for finding the semantic similarity between the user mentioned query and the existing titles in the title search and for identifying the concept behind the query along with identifying the concept of a research publication have shown good results with an average precision between 0.8 to 1 for each query.

Keywords: *Corpus-Based Semantic Measures; Knowledge-Based Semantic Measures; Reference Management Software; Ontology; RDF.*

1. Introduction

An increase in the number of digitised research articles has lead the computer science research community to take on two issues such as effectively organizing and retrieving these articles. Effective organization can lead to an effective retrieval based on the user's need. There are many digital libraries available online and also a lot of research regarding effective information retrieval in these digital libraries are available and of which many deal with web based search engines rather than desktop based. But most of the time researchers will typically download a number of online scholarly literature from the databases or digital libraries or scholarly literature search engines like Google scholar, IEEE Xplore, ACM etc.; Advanced semantic search is not supported by most of these online scholarly literature databases although they offer some kind of advanced search which includes title search, keyword search etc.; Most preferably researchers prefer either title or keyword search based on the concept they have in mind. After getting the results these articles are downloaded and are managed using reference management softwares on a researchers personal computer.

Reference management softwares, mostly the recent day advanced reference managers help the researcher in organising a number of research articles that exist in a PDF form. They even allow the retrieval of metadata and annotate that metadata to the PDF research article. Basically this metadata include title, author(s), venue, year, publisher etc., As mentioned previously, most of the times researchers prefer to search by title and this is the reason that online Scholarly literature search engine Google scholar as mentioned by Beel Jöran et al [1], gives importance to the title search by ranking the results in the Google scholar based on the existence of search keywords in the title of an article. This is also preferred for reference management software. The reference management software like Mendeley allows search for PDF research articles by only checking the keywords present in the research article and the methodology it follows is the indexing of full text articles. But there is a need for reference management softwares to address semantic similarity and semantic relatedness. Semantic similarity and semantic relatedness are used interchangeably by many authors based on their perspective although a clear distinction is given and according to Zhu, Y et al [2] which explains in the medical terms that diabetes and insulin are semantically related while glucose and blood sugar are semantically similar. So according to our perspective "preparing a manuscript" and "writing an article", an example mentioned by Gabrilovich, E et al [3] are two different similar notations mentioned by different users to represent the "writing a research journal" and According to Ensan et al [4] basic language modeling approaches are dependent on the exact match of terms that are present in the query and documents as well as the documents collection. But according to the authors exact match of terms or words have their own limitations such as vocabulary gaps between documents and the queries, as the users might be choosing some words, which might not be used in the documents but might have the same meaning or similar sense or might use the same word in different context.

For example when a researcher searches for a title (τ_1), which is "gnowsis semantic desktop" the reference manager needs to understand that title (τ_2) which is "Deepa Mehta semantic desktop" is similar to the former title. When the user gives a query the system should stick to strict co-occurrence of words in the query to match title. Sometimes the occurrence of terms in the query might match but according to the users knowledge and perspective the user might

change the terms and use a synonym of it instead. In the above example both gnowsis and Deepa Mehta are the names of various semantic desktops and hence in our view both are similar. If semantic desktop is taken both of the above-mentioned terms should be considered as synonyms of semantic desktop in the technical perspective. Along with this the other terms in the query have to be expanded with their synonyms and the co-occurrence of the terms in the expanded query has to be checked as a whole. If the user mentioned query is not matching any of the available publication titles in the reference management software, then there is a need to understand what exactly a user is mentioning through his query by understanding the context based on the context words as suggested by Bai, Jing, et al [5] and then find the concept behind it. To understand the similarity between the two titles (τ_1) and (τ_2), it needs to understand the relation between terms in a title. There is also a need to understand the context and then the concept to which they belong to, for example titles (τ_1) and (τ_2) belong to the concept semantic desktop, which is part of computer science domain. When a user prefers for a title search, he or she gives a search query that contains words that are related to each other like as said in [5]. Consider a word "program" which has its own ambiguity like a "TV Program" and "java Program", but a user always prefer to mention contextual words like "TV" and "Java" to clarify what he exactly needs. This can make the information retrieval easier. Based on the previous studies it has been stressed by the authors [5] that contextual factors, such as domain of interest to the user, the knowledge and preferences etc.; are very important for the retrieval of relevant results. When considering a reference management software, a researcher's favoured domain and its core concepts, for example computer science domain which has artificial intelligence, semantic web, computer vision etc., as its core concepts, can be obtained by analysing the title and feature words of the incoming research articles that are being added to the reference management software, for finding the contexts around the query. Although domain knowledge is largely available for medicine for most of the years, nowadays vast domain knowledge can be obtained from Wikipedia, which includes many concepts as its articles and provides a link between them. The context within the query can be obtained by considering the relation between the words in a query strictly. For example take a word "desktop" which might be having ambiguity when there is no context word mentioned along with it. If the desktop is mentioned along with a word "semantic" which in turn becomes "semantic desktop" can be helpful to recognize the appropriate concept. In other words, there exists some words in a query which might be called as context words [5] can help determine what an ambiguous word can mean and what concept the user is trying to represent in the query.

This paper focuses on the use of hybrid semantic measure, which is a mix of corpus based, and knowledge based semantic measure. In the following sections we will explain about the building of computer science domain ontology, which is built using Wikipedia, the approach we followed to judge the similarity between the title mentioned in the query and the titles of the stored publications using WordNet [39] along with the approach towards identifying the concept behind the query and concept of the incoming scholarly publication. Finally we present an experimental setup and report the results with analysis.

2. Related work

Researchers while searching for a related article of their interest will either go for a keyword search hoping to retrieve the related articles based on indexed terms or go for a title search which mainly retrieves the articles based on the matching of the terms given in the search query to the title of a research article. As mentioned by Beel Jöran et al [1] Google scholar, which is one of the most famous scholarly literature search engine focuses mostly on the two types of search which are mentioned above. According to the authors [1] Google Scholar's ranking system was analysed and based on their results they have come to the conclusion that the ranking order is given to the research articles in accordance to matching of their terms in the title to the terms in the search query. They have shown that 86% of the produced results have a matching between the search query and their own title. But the major disadvantage is that, Google Scholar does not see the synonyms of the terms [1] mentioned in the query by the user or it even does not consider the concept behind that query. Not only Google scholar, but also most of the online scholarly literature search engines will not care for context within the query, context around the query which help in finding the concept behind the user mentioned query.

As our main motive is to develop a reference manager that works on a personal computer of the researcher to help in his/her daily tasks of organising the scholarly literature, we have looked and even personally experienced different reference management softwares like Mendeley, Docear and Zotero. From the literature [6], [7], [8], [9], [10], [11] and even from the personal experience that we have gained in each of the reference manager we would like to present our findings. Coming to Mendeley this reference manager will help us in citation management along with full text article organisation. When the full texts are stored or imported, its metadata is mined from the article and the metadata is linked to it and stored elsewhere. Mendeley provides full indexing search as it indexes all the full text articles that are imported into Mendeley and when a researcher wants to find an article by its title, Mendeley provides the results based on the presence of the search query terms in the full text articles. No importance is given to the synonyms of the terms that are mentioned in the search query and is strictly restricted to term-to-term matching which might be a disadvantage while analysing the query to know the context within it and concept behind the query. Docear, which is one of the most advanced reference manager has a mind mapping concept but the search goes with the same old style of only matching the term mentioned in the query, and is same with Zotero.

As we have experienced through some of the famous reference manager and found that these reference manager do not address basic need of finding the context around the query, context within the query and concept behind the query, we have gone through the literature to find out what has been done to address this issue of finding the context around the query and within the query and relating it to a concept. According to Bai, J et al [5] many studies have suggested a user profile to deal with finding the context of a user query but a single profile of a user is not sufficient for finding out the context of a query and the concept behind it as most of the time the user might provide varied queries which are unique. So according to the author the context should only be found in the query mentioned by the user.

The authors of [12], [13], [14], [15], [16] mostly deal with user-centric ones like a single user profile is created without distinguishing the topic comments. For example in [12] the authors have proposed the user should identify some interested topic categories of open Directory project and term vectors are created to represent the whole domain of interests based on the classification of documents in the chosen category. This basically does not give any particular focus on a domain rather the focus is mainly on the user. A possible solution was provided by Liu, F et al [17] and Croft W. B et al [18] as they have also created the domain models using open Directory project categories and as said by Bai, J et al [5] there is a fine need for creating a domain model instead of a user centric model.

Li, Dandan et al [19] has stressed on having a domain ontology that is constructed for more accurate results in information retrieval and has proposed the methodology for constructing the ontology for computer science domain. This Idea of constructing domain ontology can be important as pointed out by Harispe et al [20] that several of the large corporations have been adopting ontologies for supporting their large-scale worldwide systems. Taking example of Google which has adopted a knowledge graph, which is built from the collection consisting of billions of non-ambiguous <S,P,O> which represents subject, predicate, object statements that are used to describe general knowledge or domain specific pieces of knowledge [21]. This ontology is being used to benefit the users on a daily basis by enhancing their search engine capabilities. For constructing this ontology they have initially taken encyclopaedia of

computer science and technology and computer science and technology Chinese thesaurus are used as a source for computer domain standards and authority to core professional vocabulary. Using TF-IDF, which is a feature words weight calculation algorithm, they have finalized on 220 feature words as core concepts, which formed a computer science domain concepts hierarchical structure.

Wikipedia nowadays has become a major resource for many researchers who are performing research in improving the Information retrieval. As stressed by Xun, Guangxu, et al [22] and S. Cucerzan [23] Wikification is becoming more popular and the main objective of it is to identify the tokens that are important in a given text in a particular context and link them to a particular entry in an external repository like Wikipedia. As mentioned in [22] if there is a document d with the set of important tokens 't' and a set of Wikipedia articles 'w' then the output of Wikification should be mapping the sets 't' and 'w'. In another Paper [3] the authors have recommended on giving importance to the human cognition and went on to collect the concepts from Wikipedia. These concepts are mapped with each and every word, which appears in them, and are ranked in the order of their relevance (measured by a weight) to a particular word. Their main motive is to find the semantic relatedness between words or text in the same way as humans identify. So they have used a semantic interpreter, which actually takes the help of inverted index to form a weighted vector of Wikipedia concepts for each and every text and compare those vectors to find the relatedness of the two texts. The authors have mainly mentioned about matching text of general sense and not in a technical perspectives. In a general sense Wikipedia articles are a good resource, but coming to technical perspective vast technical knowledge belonging to a certain domain can be obtained from the collection of research articles and combined with Wikipedia can determine the concept of each research journal along with identifying the concept behind a query. As the authors [3] say Wikipedia offers a wide variety of concepts, which also include core concepts of computer science, Wikipedia can act as a good resource to determine the concept to which a research publication belongs. As Stressed by the authors in [20] it will be good to mix up corpus based and knowledge based semantic measures. In this context the authors suggest that using ontologies and WordNet along with Wikipedia can give better results. Based on this suggestion and instead of using the long process mentioned by Li, Dandan et al [19], our approach uses computer science domain ontology, built using Wikipedia articles that are related to computer science using the links between the articles, where each article is treated as a concept in computer science domain.

Here Wikipedia is used for identifying the core computer science domain concepts to construct ontology of computer science domain, which helps in information retrieval. Taking inspiration from [19][3] for information retrieval we have designed a slightly combined process with slight modification and applied it for finding relatedness of two research articles based on the finding of similarity between their titles, for identifying the concept behind the query using the context within the query by taking into consideration, the contextual words and for identifying the concept to which an input research article belongs.

3. Architecture of the proposed approach

Our main motive is to built a reference management software which includes corpus based and knowledge based semantic measures for effective organization and retrieval of relevant research articles as desired by the user. In this regard we considered two aspects to be important in our reference management software, one is advanced semantic search for effective information retrieval and the other is effective storage and organization of metadata retrieved from the incoming research articles. In our earlier work [24] (which is yet to be published, but accepted) in building the SodhanaRef the metadata extraction includes Title, Author, Author Details and Reference for establishing a reference linking mechanism in the reference management software that works on a stand alone personal computer. But in this work we have majorly given importance to the Title and Concept (a piece of knowledge in computer science domain) of a research paper. The second one is storing and organizing the metadata that is obtained through metadata extraction that includes Title and Concept. We decided to represent the metadata obtained, as RDF-triples initially in an XML/RDF file as done in [24], rather than using a traditional relational database. If it is decided to use a relational database, then SQLite is the best choice for this type of application is as it has a cross-platform file format and is apt for local or client storage. But in recent days application developers have been migrating graph databases rather than sticking to the relational databases [25] as there is an increase in data day-by-day and representing this huge amounts of data using a graph databases or RDF-triple should be easy and useful. The necessity to use the semantic technology has also been stressed by the authors of [26,27] for the representation of bibliography. As said above we have initially decided to use XML/RDF, but thinking about the future growth in the number of files has prompted us in giving importance to a database which suports RDF store and in this case MarkLogic [40] was considered as a good choice.

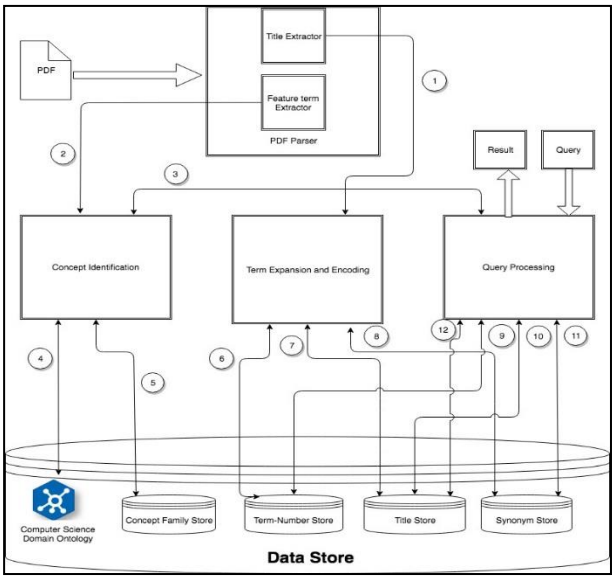


Fig. 1: Showing the Architecture for Organizing and Retrieving the Journal Articles in SodhanaRef.

As Shown in Fig.1, SodhanaRef constitute four main modules PDF Parser, which includes Title Extractor and Feature term extractor, which is helpful for concept identification, which is to be done later, Term Expansion and Encoding, Concept Identification and Query processing module. Initially, an incoming research journal article in PDF form is subjected to PDF parsing where the title extraction and feature term extraction is done and are explained in detail in section 5. The job of Term Expansion and Encoding module is to expand each term from the extracted title with its synonyms if found in the WordNet and store them in the synonym store. If a word is not found in the WordNet, which it means it isn't a word that is generally used and is specifically technical, is also stored in the synonym store as a synonym to the concept of the research paper to which it belongs. Now each term along with its family of synonyms is given an index number and is stored in the term-number store. The job of Concept identification which is explained in detail in section 6 is to analyze the feature terms that are extracted from each incoming research paper and decide, to which concept the research paper belongs by interacting with the computer science ontology built with the help of Wikipedia dump. The job of Query processor, which is explained in detail in section 7, is to analyze whether the query given by the user is a title or a set of important keywords belonging to a concept. Based on this analysis the query processor module will either search for a match using query to title comparison or search for a concept to which the mentioned keywords in the query belongs to and display the results with all the research articles that belongs to that particular concept.

4. Building computer science domain ontology

In the process of conceptualizing computer science domain we have chosen to build ontology, as it is defined as the specification of conceptualization. As said in [28] Ontologies provide a common vocabulary for all the researchers who are in need of sharing the information in the domain, in other words constructing an ontology in a machine readable format can help the system we develop, to understand the basic concepts or the terms and the relation between them, used in the particular domain of target. There is also a progress in information retrieval, which has been improved from a keyword-based retrieval to knowledge-based search [19]. As said in the literature survey we are not following a long process mentioned in [19] where the authors have followed a seven-step method explained by [28]. Instead we have considered using Wikipedia which is the free encyclopaedia and consists of best knowledge in the human perspective, as said in [3] Wikipedia Consists of vast amounts of highly organised human knowledge. As Wikipedia is growing day by day, it can also be considered as a source for new concepts that are being included throughout its expansion. Wikipedia is mostly considered for general sense, which might not deal with in-deep scientific knowledge, but it has the capacity to hold many important concepts and the hierarchical links and relations between them for a certain targeted domain, in our example it is about computer science domain.

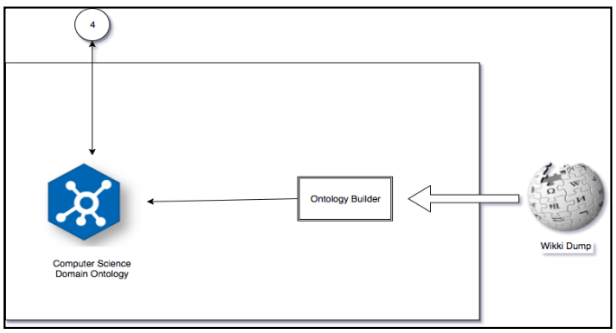


Fig. 2: Building Ontology Using Wikidump.

To help us in our task of building computer science domain ontology, which should consist of important concepts and the relations among the concepts, we have taken the help of Wikipedia dump, which is up to date, dated October, 2017 [29]. Our Ontology Builder module initially parses the Wikipedia dump which is in the form of XML and considers the string between < title> </title> as a concept. From each concept that was found in the above said process, their links were gone through to the other concepts, which are linked to them. The links were easily identified as they are in this form [[link]]. Initially, Ontology was built using RDFS with the help of Jena framework, which helps in modelling ontology in RDF/XML syntax, but finally MarkLogic was chosen as the backend due to its ability to deal with scalability. With the help of MarkLogic, an ontology was built by creating RDF triples in a graph named computer science ontology, expressing each concept and its relation with other concepts as triple (subject, predicate, object) as shown in Fig.3 and Fig.4. Here each concept is considered as a resource.
Since semantic web provides a representational infrastructure for the metadata representation [30] and RDF (Resource Description Framework) is a semantic web data model that is adopted to represent the information of the resources available on web and according to [31] RDF graphs are populating the emerging semantic web and are the core data structure of the big web data and in other words helps in managing huge amounts of data. As the research publications have metadata and an increase in number of research publications that are being stored in the reference manager on a personal computer or a desktop, will increase the metadata that becomes huge day by day. In this particular situation it is desirable to use semantic web technologies for managing desktop data [30] where each research publication can be identified by an URI (Uniform Resource Identifier) and the metadata can be represented using RDF graphs. So, not only for building computer science domain ontology, it has been decided to represent other data that has to be stored, as RDF graphs.

```
<?xml version="1.0" encoding="UTF-8"?>
<sem:triples xmlns:sem="http://marklogic.com/semantics">
 <sem:triple>
  <sem:subject>computerscience</sem:subject>
  <sem:predicate>http://hasLink</sem:predicate>
  <sem:object datatype="http://www.w3.org/2001/XMLSchema#string">Machine Learning</sem:objec
 </sem:triple>
 <sem:triple>
  <sem:subject>computerscience</sem:subject>
  <sem:predicate>http://hasLink</sem:predicate>
  <sem:object datatype="http://www.w3.org/2001/XMLSchema#string">Semantic Web</sem:object>
 </sem:triple>
 <sem:triple>
  <sem:subject>computerscience</sem:subject>
  <sem:predicate>http://hasLink</sem:predicate>
  <sem:object datatype="http://www.w3.org/2001/XMLSchema#string">Query language</sem:object>
 </sem:triple>
</sem:triples>
```

Fig. 3: Show A Part of Computer Science Domain Ontology.

```
<?xml version="1.0" encoding="UTF-8"?>
<sem:triples xmlns:sem="http://marklogic.com/semantics">
 <sem:triple>
  <sem:subject>MachineLearning</sem:subject>
  <sem:predicate>http://hasLink</sem:predicate>
  <sem:object datatype="http://www.w3.org/2001/XMLSchema#string">Artificial intelligence</sem:object>
 </sem:triple>
 <sem:triple>
  <sem:subject>MachineLearning</sem:subject>
  <sem:predicate>http://hasLink</sem:predicate>
  <sem:object datatype="http://www.w3.org/2001/XMLSchema#string">Computational intelligence</sem:object>
 </sem:triple>
 <sem:triple>
  <sem:subject>MachineLearning</sem:subject>
  <sem:predicate>http://hasLink</sem:predicate>
  <sem:object datatype="http://www.w3.org/2001/XMLSchema#string">Computational neuroscience</sem:object>
 </sem:triple>
</sem:triples>
```

Fig. 4: Show Concepts and Their Links in Computer Science Domain Ontology.

5. Parsing PDF research article

Parsing the PDF is done for two main categories, title extractor and feature term extractor. These two categories have their own importance in building up an innovative Storage and retrieval system for reference management software.

5.1. Title extractor

Title extractor first parses the title τ from the research document, which is in the form of PDF and sends it to the tokenizer for tokenization. After the completion of tokenization, title τ is defined as the set of tokens, where these tokens are terms or words $\{t_1, t_2 \ldots t_n | n \geq 1\}$ each token which is now a term in the title is expanded by term expansion module as shown in Fig.6. The term expansion module has a duty to check whether the term already exist in the synonym store or not and if it does exist it expands the term t_i as $\{t_i, t_i m_1, t_i m_2 \ldots t_i m_n | n \geq 1\}$, where $t_i m_n$ is the synonym of term t_i where $t_i \in \tau$. If the term t_i and its synonym set is not available a communication will be done with the WordNet and the synonyms are obtained from it. Hence term t_i is defined as T_i which is a set of t_i and its synonyms $\{t_i, tim1, tim2 \ldots timn | n \geq 1\}$. In any case as said in section 3, if a word is not found even in the WordNet, which it means it isn't a word that is generally used and is specifically technical, is also stored in the synonym store as a synonym to the concept of the research paper to which it belongs, for example a title might contain

a term "Gnowsis" which is the name of a semantic desktop then this term is identified as a synonym for the concept "Semantic Desktop" and is stored in the synonym store. Now the title is represented as $\{T_1,T_2,....,T_n\}$. These new terms and their synonym sets are stored in the synonym store along with sending them parallely to the term encoder. The term encoder checks whether the received term and their synonyms in the subset T_i is present in the term-number store. If at least one term in the subset T_i, $\{T_i \in T_1,....,T_n\}$ matches with the term in the term-number store which is in the index form where every term in the previously added titles will have an index number, that corresponding index number is given to that particular subset T_i. Now title τ is assigned a number combination NC which is of the form N_1^N_2^....^N_k, where N_k is an index number for a term t_k or its synonym in the subset T_k and hence it is awarded to the whole subset. After attaining a number combination for a title it is now stored as RDF triple in a graph using marklogic in the form of <S,P,O> where S is the subject, P is Predicate and O is object. Here number combination for the title NC is regarded as the subject, "isTitleOf" is regarded as a relation or predicate and path of the file is regarded as the object, as shown in the Fig.5.

Fig. 5: Showing the Triple Store in the Title Store Graph in Marklogic.

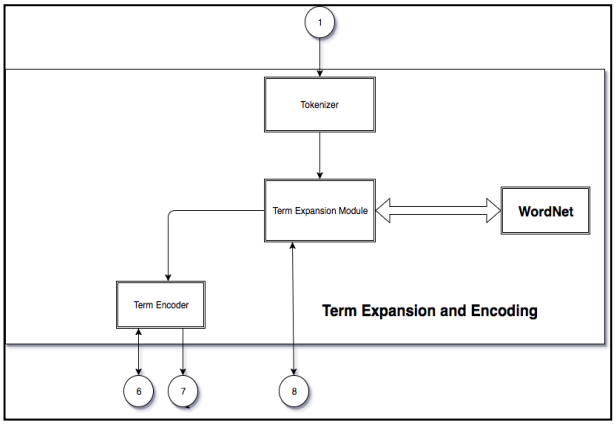


Fig. 6: Term Expansion and Encoding.

5.2. Feature term extraction

The document is parsed and a set of words or terms $\{t_1,t_2,....,t_n|n\geq 1\}$ is created for that particular document. This set of words is now subjected to stop word removal, where all the predefined stopwords numbering 319 are determined by calculating tf-idf for the words in around 200 documents. The words with a weight of zero are defined as stop words. Using these predefined stop words, the stop word removal module now removes unnecessary words from the list of words. The remaining words are now assigned a weight $\boldsymbol{\omega}_i$ based on probability of a term (TP) to occur in a document irrespective of size of the document [32] and a new weighted set (WS) of all the words or terms with their corresponding weight $\boldsymbol{\omega}_i$ is created, and is defined as $(\{t_1,t_2,....,t_n|n\geq 1\}, \{(t_i,\boldsymbol{\omega}_i) \mid 1\leq i\leq n\})$. The expression for TP as suggested by the authors in [32] is as follows.

$$TP = \frac{tf_{ij}}{\sum_{i=1}^{n} tf_{ij}} \quad (1)$$

Where tf is the term frequency and i indicates the i^{th} term in the j^{th} document and n indicates the number of terms in the list of words for the j^{th} document. Here j has no importance as the process is dynamic and every time we consider only one document at a time and its value is constant i.e. j=1.

6. Concept Identification

These weighted set of words WS is now send to concept identifier, which is shown in Fig.7 whose job is to analyse which concept this document or its weighted set of words WS belongs to.

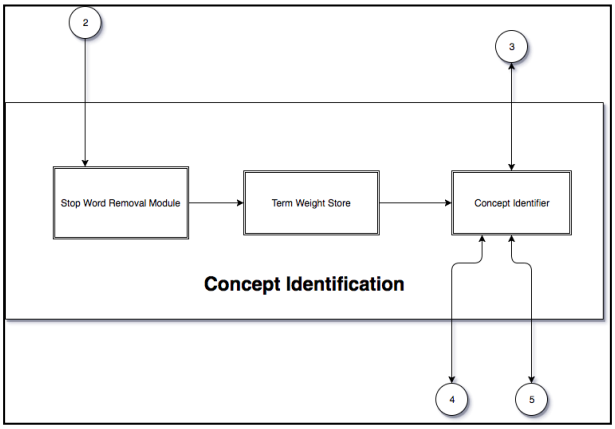


Fig. 7: Concept Identification.

Let C is the set of concepts $\{c_1, c_2,....,c_n|n\geq 1\}$ obtained from the computer science ontology built from wikipedia. Concept identifier iterates through all the concepts in C while parallelly checking them with the incoming weighted terms to find whether any term matches with a concept in C. If a term t_i matches with a concept c_i, $\{c_i \in c_1,c_2,....,c_n\}$ where n is the total number of obtained concepts, then that term t_i is included as a decided concept dc_i, $\{dc_i \in dc_1,dc_2,....,dc_n\}$ in the concept-family store and all the other terms in the weighted set of words WS along with their weights are now related to that decided concept in the concept-family store. For this relation, instead of inverted index a triple is created in a graph G which is a concept-family store, using MarkLogic database with <subject> as the term t_j, <predicate> as the weight $\boldsymbol{\omega}_j$, <object> as the decided concept c_i, where <S,P,O> = < t_j , $\boldsymbol{\omega}_j$, dc_i> as we preferred a graph based storage as shown in Fig.8.

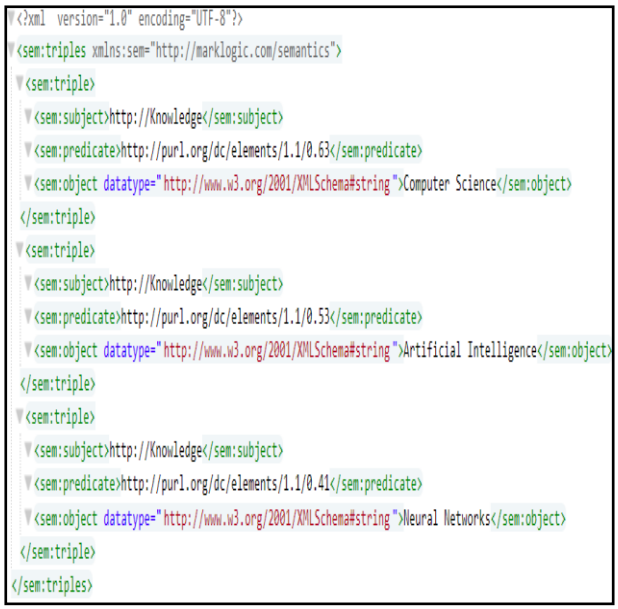


Fig. 8: Showing the Triple Store in Marklogic.

If a term t_j and its weighted relationship with a concept c_j, has already a triple in the graph G, there is a need to update its already existing weight $\boldsymbol{\omega}_j$ with the concept c_j in a normalized way. To get a normalized weight N_w from the old existing weight and new incoming weight, we used a formula $N_w = (\boldsymbol{\omega}_j + \boldsymbol{\omega}_{nj})/2$ where $\boldsymbol{\omega}_j$ is the old existing weight and $\boldsymbol{\omega}_{nj}$ is the new incoming weight. Now $\boldsymbol{\omega}_j$ is updated with the calculated normalized weight N_w.
If the weighted set of words WS contains two or more terms matching with different concepts in concept set C then the concept identifier checks the weight of all the terms for example (t_i, t_j, t_k), $\{(t_i, t_j, t_k) \boldsymbol{\epsilon} WS\}$, that are matching with different concept (c_i, c_j, c_k), where $\{(c_i, c_j, c_k) \boldsymbol{\epsilon} C\}$ and gets the highest weighted term among t_i, t_j and t_k, say for example t_i. A comparison is done between weights of the highest weighted term ti and other terms t_j and t_k to find out the difference between their weights. If the difference between the weights of terms t_j and t_k with the highest weighted term t_i is greater than 0.2 then the terms t_j and t_k are discarded and t_i is decided as the decided concept dc_i. All the terms in the weighted set of words or terms WS of the input document are now related to the decided concept dc_i and stored in the form of triples for each word or term that belongs to WS, in the concept-family store. But if the difference is zero or less than 0.2 both t_j and t_k are also decided as the decided concepts dc_j and dc_k.

6.1. Relating concept to title

As mentioned in the previous section above, every research article document will undergo two parallel activities of extracting the title along with deciding its number combination NC and identifying its concept. The number combination for a title of a document and concept the document belongs to are stored in the title store graph as triple with subject being number combination NC, predicate being "has Concept" and object being the decided concept dc as shown in the Fig.9.

```
<?xml version="1.0" encoding="UTF-8"?>
<sem:triples xmlns:sem="http://marklogic.com/semantics">
 <sem:triple>
  <sem:subject datatype="http://www.w3.org/2001/XMLSchema#string">2^3</sem:subject>
  <sem:predicate>http://hasConcept</sem:predicate>
  <sem:object datatype="http://www.w3.org/2001/XMLSchema#string">Semantic Desktop</sem:object>
 </sem:triple>
</sem:triples>
```

Fig. 9: Showing (Title, Has concept, Decided Concept) Triple Store in Marklogic

7. Querying processing

Query expansion is generally preferred for improvement in information retrieval for better results. It is a technique used for reformulation of user query to stop unwanted or unrelated results [33]. Queries are sometimes short which do not provide complete specification of information need. So in the literature many proposals like expanding documents and queries are proposed [5]. User centric techniques like observing the user and understanding the concept has been proposed but according to the authors in [5] domain knowledge should be given importance to basically understand the context within the query as mentioned by the user using the context words and in turn understand to what concept the given terms and the relationship between them belong to. According to the authors in [34] the expansion of queries were earlier done form the statistical result of the terms co-occurrence in the collections but according to them [34] it has not improved any information retrieval. Some of the authors as in [35] have proposed query expansion using synonym set in word net but all the terms are independent. In [34] the word sense disambiguation is used to get the sense of a word in the given query context. Search terms are actually independently considered and these search terms are extended with some synonyms from thesaurus [36], which will some times bring a large amount of undesirable results. So a statistical co-occurrence is proposed. Which according to [34] is not a better one to improve the information retrieval. Based on these senses the concepts are identified and using these identified concepts a corresponding similar terms are identified and chosen from WordNet. To understand a given query we have considered expanding the terms in the query with their synonyms if the majority of terms are already present in the synonym store, other wise find the concept that relates to the combination of these terms as shown in Fig.10.

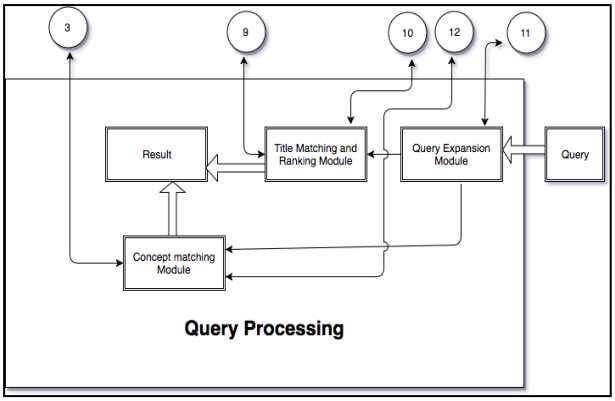


Fig. 10: Query Processing.

Given a user query (Q) which is a set of search terms $\{qt_1,qt_2,...,qt_n|n\geq 1\}$, whose synonyms are obtained from the synonym store by query expansion module, which indicates that the terms or their synonyms in the query Q are already present in an already existing title or different titles that are stored in title store. The query expansion module upon receiving the query, first tokenize the query and brings out individual terms which are now checked for their existence in the synonym store, if they are present in the synonym store then the expansion will take place where instead of term qt_n, QT_n is considered, which is a set of all the synonyms of qt_n including qt_n, $\{qt_n,qt_nm_1,qt_nm_2,....,qt_nm_k|k\geq 1\}$ and now Q is a modified set $\{QT_1,QT_2,....,QT_n|n\geq 1\}$. Now this expanded set is sent to title matching and ranking module which will check each subset QT_i in the query set Q with term-number store and upon matching at least one term in the subset QT_i, $\{QT_i \boldsymbol{\epsilon} QT_1,....,QT_n|n\geq 1\}$ in the term-number store which is in the index form where every term in the previously added titles will have an index number, that corresponding number is given to that particular subset QT_i. Now the modified Q is assigned a number combination NC which is of the form $QN_1\text{^}QN_2\text{^}....\text{^}QN_k$, where QN_k is an index number for QT_k.
The obtained number combination NC is now checked with the already stored title number combination of titles in the title store. If modified Q's NC is represented as a set $\{QN_1,QN_2,.....,QN_k\}$ and let there be a title $\boldsymbol{\tau}$, whose number combination NC is represented as a set $\{\boldsymbol{\tau}N_1,\boldsymbol{\tau}N_2,\boldsymbol{\tau}N_3,....,\boldsymbol{\tau}N_k\}$, then we calculate the deciding factor df which is represented as percentage to determine whether the given query contains keywords that may not be related to the existing titles or not. The deciding factor df is calculated as given below:

$$qmp = \frac{(n(Q)-n(Q\cap\tau))}{n(Q\cap\tau)} * 100 \quad (2)$$

$$tmp = \frac{(n(\tau)-n(Q\cap\tau))}{n(\tau)} * 100 \quad (3)$$

$$df = \frac{qmp+tmp}{2} \quad (4)$$

Where qmp is the query matching percentage, tmp is the title matching percentage and n(Q), $n(\boldsymbol{\tau})$, $n(Q\cap\boldsymbol{\tau})$ are the cardinality of sets Q, $\boldsymbol{\tau}$and $Q\cap\boldsymbol{\tau}$.

According to the condition the df should be less than or equal to 25% ie., df≤25, so that if the query set Q has at-least one deciding factor df less than or equal to 25% (based on our observation of 20 queries, the deciding factor of query and title match is always less than or equal to 25% if most of the terms mentioned in the query as a combination matches with the terms as a combination in a title). Then the results are displayed based on the direct title search.

7.1. Title match

The results of direct title search are divided into two main categories, one is exact match and the other one is partial match.

7.1.1. Exact match

If modified Q's NC is represented as a set $\{QN_1,QN_2,.....,QN_k\}$ and let there be a title $\boldsymbol{\tau}$ whose number combination NC is represented as a set $\{\boldsymbol{\tau}N_1,\boldsymbol{\tau}N_2,\boldsymbol{\tau}N_3,....,\boldsymbol{\tau}N_k\}$, then n(Q) should be equal to n($\boldsymbol{\tau}$) and (Q∩ $\boldsymbol{\tau}$) should be equal to both Q and then the match between Q and $\boldsymbol{\tau}$ as exact match. Here the ranking is given based on the order of discovery of exactly matching titles

7.1.2. Partial match

rank#1: If the number combination NC, $\{QN_1,QN_2,.....,QN_k\}$ of query Q has less cardinality than the number combination NC, $\{\tau N_1,\tau N_2,\tau N_3,....,\tau N_k\}$ of title τ ie., n(Q)<n(τ), where n(τ)=n(Q)+1 and every element of modified set Q matches with every element of title set τ except one element in τ.
rank#2: If the number combination NC, $\{QN_1,QN_2,.....,QN_k\}$ of query Q has greater cardinality than the number combination NC, $\{\tau N_1,\tau N_2,\tau N_3,....,\tau N_k\}$ of title τ ie., n(Q)>n(τ), where n(τ)=n(Q)-1 and every element of title set τ matches with every element of modified set Q except one element in Q.
rank#3: If the number combination NC, $\{QN_1,QN_2,.....,QN_k\}$ of query Q has lesser cardinality than the number combination NC, $\{\tau N_1,\tau N_2,\tau N_3,....,\tau N_k\}$ of title τ ie., n(Q)<n(τ), where n(τ)=n(Q)+2 and every element of modified set Q matches with every element of title set τexcept two elements in τ.
rank#4: If the number combination NC, $\{QN_1,QN_2,.....,QN_k\}$ of query Q has at least 50% match with number combination NC, $\{\tau N_1,\tau N_2,\tau N_3,....,\tau N_k\}$ of title τ, then it is ranked as 4.

7.2. Concept match

With in the given query terms if the majority of them are not matching with the terms in the term number store, and synonym store, as said above if their deciding factor is more than 25% with all the titles in the title store, then the keywords are to be matched with their corresponding concept from the concept-family store, which in turn helps in identifying the titles whose articles might match the mention keywords.
Let there be a set Q consisting of terms, $\{qt_1,qt_2,....,qt_n|n\geq 1\}$and dc_i be the decided concept in the concept family store, which consists of both terms t_i and their correspond weights ω_i with that concept as explained previously in section, $(\{t_1,t_2,....,t_n|n\geq 1\}, \{(t_i,\omega_i) \mid 1\leq i\leq n\})$ and $dc_i\equiv c_i \in C$ (concept set). Now the results are divided into two categories Exact Match and Partial Match

7.2.1. Exact match

In exact match the condition is, every term in Q should be mapped with the terms in concept c_i and the ranking is decided among all the matching concepts, based on concept deciding factor cdf. The cdf is calculated as below

$$\text{cdf} = \frac{sum\ of\ all\ query\ terms\ weights\ with\ a\ concept\ dc_i}{total\ number\ terms\ in\ query} \quad (5)$$

Or

$$cdf = \sum_{j=1}^{n} q\omega_j/n(Q)$$

Where $q\omega_j$ is the weight of the j[th] query term qt_j, $\{qt_j \in qt_1,....,qt_n\}$ and n is the total query terms in set Q, with respect to the i[th] decided concept dc_i. The ranking is provided in descending order.

7.2.2. Partial match

The partial match is considered only when there is a mismatch between the query terms and the terms in the concept, with a condition that mismatching terms should be less than or equal to 20% of the total terms in the Q. Again the concept deciding factor is used for ranking.

$$\text{cdf} = \frac{sum\ of\ all\ matching\ query\ terms\ weights\ with\ concept\ dc_i}{total\ number\ matching\ terms\ in\ query} \quad (6)$$

The ranking is provided in descending order. The list of Exact matches and Partial matches are displayed as shown in Fig.11 in a descending order in a drop box and upon selection of a desired file and a click on open will open a selected file.

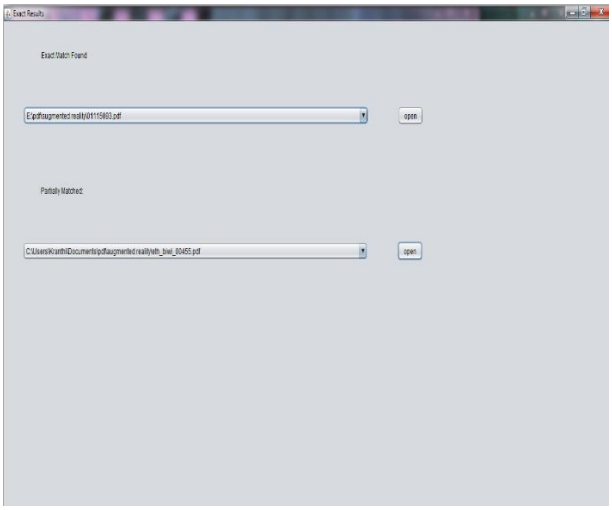


Fig. 11: Showing the Display of List of Exact and Partial Match for A Query in Title Match.

8. Evaluation and results

There are two important aspects of our system which we need to evaluate one is the title matching either exact or partial, if the user wants to search using title. The second one is the concept identification both for input journal article, which is subjected to parsing, and for a raw query posed by the user.
For evaluating the Title Matching, we have initially taken five concepts like semantic web, semantic desktop, artificial intelligence, image processing, and augmented reality belonging to computer science domain. For each selected concept, we have downloaded five different scholarly publications, a total of 25 scholarly publications in PDF form, which were then parsed for their metadata, which includes title and concept of the journals and stored into our system. We have involved five volunteers who are considered good at a particular concept and these five volunteers are the faculty belonging to computer science department. Each volunteer is given the copies of those five different downloaded scholarly publications and was asked to read the paper and the title of each publication. This will give an idea of which publications are stored in the system. After a day the volunteers are called upon and were asked to search the publications that were stored in our system using a title search by recalling their titles. Each volunteer has given five queries one query for one journal and a total of 25 queries (but the results were presented with only 20 selected queires). The volunteers assumed each query as title of the publication. Unknown to the volunteers we have also added few other articles, which are having identical titles like for example if a publication with the title "The social semantic desktop" is present in our system, we have also added another publication with the title "The Networked semantic

desktop" and also other identical publication with a title that can be partially matching for example "Gnowsis semantic desktop" and "DeepaMehta – A Semantic Desktop" indicating that their work might be identical. So this has increased the total number of articles present in our system to 60. Considering a hypothetical case as shown in the Table.1 below.

Table 1: A Hypothetical Case

Query given by user assuming title	Result sin rank based on our algorithm	Average precision
The Collaborative Semantic Desktop	Exact Match: #Rank 1 The Social Semantic Desktop The Networked Semantic Desktop Partial Match: #Rank 1 Gnowsis Semantic Desktop DeepaMehta – A Semantic Desktop	1

The average precision is calculated based on the below formulae which is taken from [37] and original paper. Equation (7)

$$AvgP = \frac{1}{Rl} \left(\sum_{k=1}^{n} P(k) * rel(k)\right)$$

Where Rl is the total number of relevant publications, P(k) is the precision value up-to the position k, rel(k) is an indicator function which is equivalent to '1' if the publication at position k is in its correct position and equivalent to '0' if it is not in the correct position.

The order is important while calculating the average precision. Take for example in the Table.1 in which a hypothetical is presented. In this the rank ordering for exact match and partial match is presented. This ordering is important. If a publication with a title "Gnowsis Semantic Desktop" is not present in #Rank 1 and is wrongly presented as #Rank 2 or #Rank 3 or is not returned as a result then the precision goes to zero as the desirable position k is not attained while getting the result. The attained results are presented in the Table.2 given below which shows the AvgP(k) and MAP(c) where 'c' is the concept.

Table 2: List of Queries for Each Concept Along with Each Concepts Average Precision

	query	expected	obtained	AvgP(k)	MAP
semantic desktop	q1	2(partial match rank 1) 2(partial match rank 3)	2(partial match rank 1) 2(partial match rank 3)	1	4/5=0.8
	q2	1(partial match rank 1)	1(partial match rank 1)	1	
	q3		no output	0	
	q4	1(Exact match)	1(exact match)	1	
	q5	1(partial match rank 4) 1(exact match)	1(partial match rank 4) 1(exact match)	1	
semantic web	q6	1(partial match rank 1) 1(partial match rank 3)	1(partial match rank 1) 1(partial match rank 3) 1(partial match rank 4)	3/4=0.75	3.75/4= 0.9375
	q7	1(partial match rank 2)	1(partaila match rank 2)	1	
	q8	1(partial match rank 2)	1(partial match rank 2)	1	
	q9	1(partial match rank 2)	1(partial match rank 2)	1	
image processing	q10	1(partial match rank 3) 1(partial match rank 4)	1(partial match rank 3) 1(partial match rank 4)	1	1
	q11	1(partial match rank 4)	1(partial match rank 4)	1	
	q12	1(partial match rank 4)	1(partial match rank 4)	1	
artificial intelligence	q13	1(exact match)	1(exact match)	1	3/4=0.75
	q14	1(partial match rank 1)	1(partial match rank 1) 1(partial match rank 4)	1/2=0.5	
	q15	1(partial match rank 3)	1(partial match rank 3)	1	
	q16	1(partial match rank 3)	1(partial match rank 3) 1(partial match rank 4)	1/2=0.5	
augmented reality	q17	1(Exact match)	1(exact match)	1	1
	q18	1(partial match rank 3)	1(partial match rank 3)	1	
	q19	1(partial match rank 3)	1(partial match rank 3)	1	
	q20	1(partial match rank 1)	1(partial match rank 1)	1	

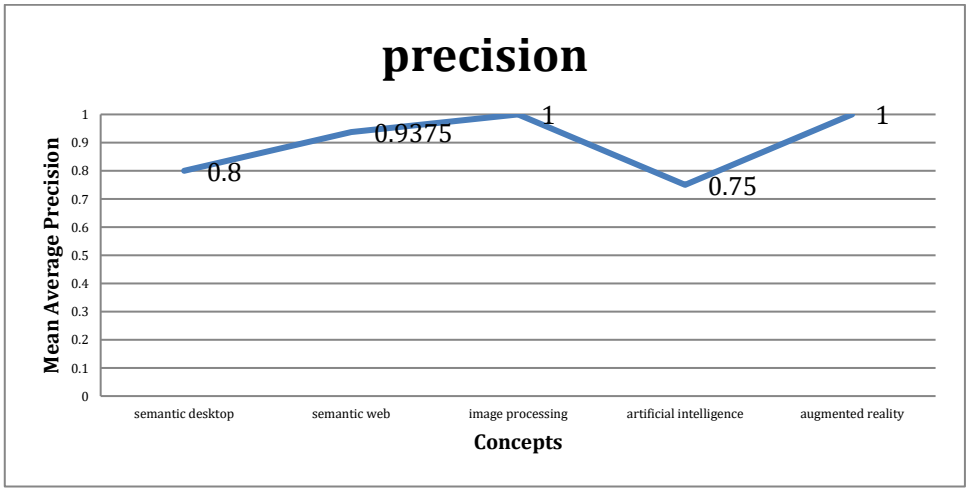


Fig. 12: Average Precision for Each Concept.

An evaluation is done for concept identification both for input journal article, which is subjected to parsing, and for a raw query posed by the user.

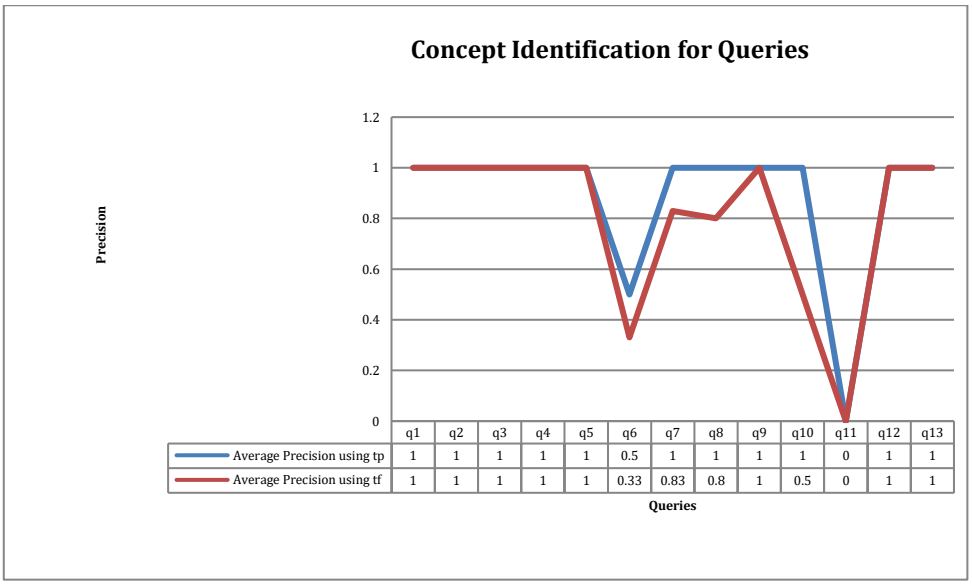
As mentioned earlier, for concept identification, the term weight plays an important role. So, to evaluate concept identification for query, which is posed by the user, we have used two different ways to measure term weight, one is tf (term frequency) form tf/idf and the second one is tp (term probability) from tp/idf as suggested by [32]. For this evaluation, we have invited the above said volunteers to provide some n number of queries, where each query might belong to a concept and which consists of a keyword combinations that belongs to a particular concept in the computer science domain. This evaluation takes place using 13 queries that we have selected from the user provided queries of varied lengths and the results are provided below along with a graph of comparision between the results obtained for 13 queries using tp and tf. Based on our observation from the below Table.3 and graph in the Fig.13 the difference between the (MAP) Mean average precision of concept identification using tp and MAP of concept identification using tf is not too big, but it can be observed that there is a slight increase in MAP of concept identification using tp. This slight increase is because the retrieved concepts for some queries using tf are not in their exact position k and it can be observed in the Table.3 that some unwanted concepts are ranked high and appears in the top position and this is due to their varied ways of assigning weights to the terms.

Table 3: Obtained Average Precision of Each Query and Mean Average Precision Using both Tf and Tp

s.no	Query	Concept Based on (Term Probability) Weight	Average precision	Concept Based on (Term Frequency) Weight	Average precision
1.	Gnowsis	Semantic desktop	1	Semantic desktop	1
2.	Xcosim	Semantic desktop	1	Semantic desktop	1
3.	Finding a Seamless desktop	Semantic desktop	1	Semantic desktop	1
4.	Will gnowsis interact socially	Semantic desktop (Artificial intelligence)	1	Semantic desktop (Artificial intelligence)	1
5.	Virtual reality	Augmented reality Image processing	1	Augmented reality Image processing (Semantic desktop)	1
6.	Aeronautical maintenance	(Augmented reality) Artificial intelligence (Semantic web)	0.5	(Semantic web) (Augmented reality) (Semantic web) Artificial Intelligence	0.33
7.	Relation between image processing and augmented reality	Augmented reality Image processing	1	Augmented reality (Semantic web) Image processing	0.83
8.	Advances in information retrieval	Artificial intelligence Semantic web Image processing Semantic desktop (Augmented reality)	1	Artificial intelligence (Augmented reality) Image processing Semantic web Semantic desktop	0.8
9.	Argumentation in artificial intelligence	Artificial intelligence Augmented reality	1	Artificial intelligence Augmented reality	1
10.	Ontology learning in semantic web	Semantic Web	1	(Semantic Desktop) Semantic Web	0.5
11.	Automated composition in semantic web	No result	0	No Result	0
12.	Social behavior	Artificial intelligence Augmented reality Semantic desktop Semantic web Image processing	1	Artificial intelligence Augmented reality Semantic desktop Semantic web Image processing	1
13.	Overview and outlook of ontology	Semantic desktop Semantic web	1	Semantic desktop Semantic web	1
14.	MAP		0.88(88%)		0.80(80%)

From the Table above it can be observed that there is no much difference in ecquiring the precision for identifying the concepts for the queries.

A comparision between SodhanaRef a reference management software which has been developed and the Mendeley reference management software has been done using 15 queries, taken from 5 participating volunteers each giving 3 queries. The results have been presented in the Table.4 form and a graph has been presented in Fig.14 showing the comparision of average precision values of Mendeley and SodhanaRef for each query. This comparision has been basically done for the title search where the users are expected to give the title with all the words exactly matching or replacing few words with the synonyms of the existing words in the title. The observation has been done that Mendeley has sometimes given less preference to a publication which has all the words in the title matching with all the words in the search query. But SodhanaRef could detect the synonyms and match with the exact title even though the words are not exactly matching.



	q1	q2	q3	q4	q5	q6	q7	q8	q9	q10	q11	q12	q13
Average Precision using tp	1	1	1	1	1	0.5	1	1	1	1	0	1	1
Average Precision using tf	1	1	1	1	1	0.33	0.83	0.8	1	0.5	0	1	1

Fig. 13: Comparision of Average Precisions Using Tp and Tf.

Table. 4: Average Precision Values of 15 Queries for Sodhanaref and Mendeley

Queries	SodhanaRef Average Precision for each query	Mendeley Average Precision for each query
Q1	0.83	0.5
Q2	1	0
Q3	1	1
Q4	1	0
Q5	1	0.33
Q6	1	1
Q7	1	0
Q8	1	0
Q9	0.8	0
Q10	1	0.5
Q11	0.83	1
Q12	0.66	0
Q13	1	1
Q14	1	1
Q15	1	0
MAP	0.941	0.422

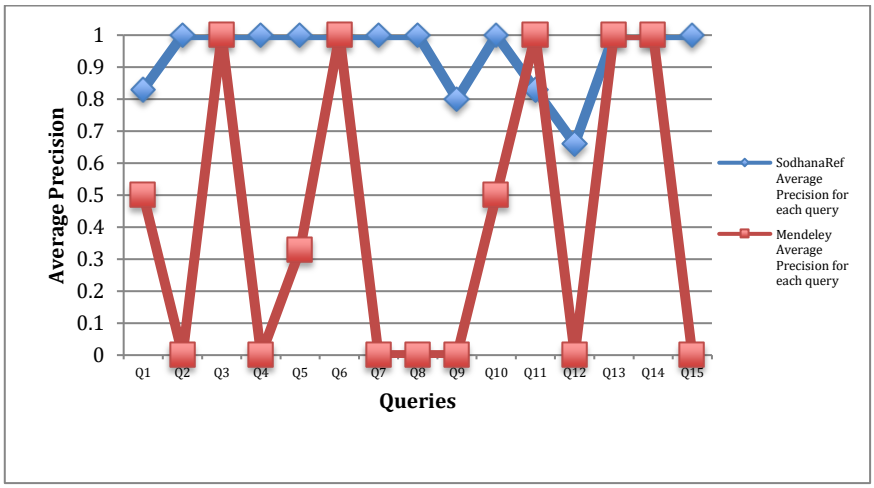


Fig. 14: Comparison Graph for Average Precision of Sodhanaref and Mendeley

9. Conclusion

In this paper we have used a mix of corpus based and knowledge based semantic measures where Wikipedia, WordNet and Ontologies are involved to find the semantic similarity between the queried titles and existing titles of the publications during a title search, along with identifying the concept of a publication and the concept behind the user mentioned query. Explicit semantic analysis, which inspired our work, stress on human cognition. But ESA address the problem of computing semantic relatedness in a more general sense. This ESA type of approach can also be applied for online scholarly search engines or reference management softwares where lot of publications are stored and retrieved by the researchers, so a computation of semantic relatedness and similarity is addressed using the research publications in the same way as ESA uses Wikipedia articles as concepts and computes the semantic relatedness between two texts. Using the data from Wikipedia an ontology that includes the concepts belonging to computer science domain was built and each input paper is parsed to identify the concept to which the research publication belongs. All the data that is required to be stored during this process is stored as RDF triples using MarkLogic.

Query expansion techniques were employed using WordNet to understand the context within a query that is mention by the user, to understand the concept behind it. Reference management software SodhanaRef is built with this approach and our experimental results show that it performs well when compared to Mendeley during title search. Our future work includes the implementation of our approach in distributed reference management software using MarkLogic Distributed database.

10. Acknowledge

We thank G Sai Murali and Arige Maheswari for their immense support in evaluating our system.

References

[1] Beel, Jöran, and Bela Gipp. "Google Scholar's ranking algorithm: an introductory overview." *Proceedings of the 12th International Conference on Scientometrics and Informetrics (ISSI'09)*. Vol. 1. 2009.

[2] Zhu, Yongjun, Erjia Yan, and Fei Wang. "Semantic relatedness and similarity of biomedical terms: examining the effects of recency, size, and section of biomedical publications on the performance of word2vec." *BMC medical informatics and decision-making* 17.1 2017: 95.

[3] Gabrilovich, Evgeniy, and Shaul Markovitch. "Computing semantic relatedness using wikipedia-based explicit semantic analysis." *IJcAI*. Vol. 7. 2007.

[4] Ensan, Faezeh, and Ebrahim Bagheri. "Document retrieval model through semantic linking." *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 2017. https://doi.org/10.1145/3018661.3018692.

[5] Bai, Jing, et al. "Using query contexts in information retrieval." *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2007.

[6] Hensley, Merinda Kaye. "Citation management software: features and futures." *Reference & User Services Quarterly* 50.3, 204-208. 2011. https://doi.org/10.5860/rusq.50n3.204.

[7] Basak, Sujit Kumar. "A Comparison of Researcher's Reference Management Software: RefWorks, Mendeley, and EndNote." *Journal of Economics and Behavioral Studies* 6.7, 561, 2014.

[8] Gilmour, Ron, and Laura Cobus-Kuo. "Reference management software: a comparative analysis of four products." *Issues in Science and Technology Librarianship* 66.66, 63-75, 2011.

[9] Beel, Joeran, et al. "Docear: An academic literature suite for searching, organizing and creating academic literature." *Proceedings of the 11th annual international ACM/IEEE joint conference on Digital libraries*. ACM, 2011. https://doi.org/10.1145/1998076.1998188.

[10] Ray, Aswini Kumar, and D. B. Ramesh. "Zotero: Open Source Citation Management Tool for Researchers."International journal of library and information sciences. 2017

[11] Parabhoi, Lambodara, Arabinda Kumar Seth, and Sushanta Kumar Pathy. "Citation Management Software Tools: a Comparison with Special Reference to Zotero and Mendeley." *Journal of Advances in Library and Information Science* 6.3 2017: 288-293.

[12] Chirita, Paul Alexandru, et al. "Using ODP metadata to personalize search." *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2005. https://doi.org/10.1145/1076034.1076067.

[13] Dumais, Susan, et al. "Stuff I've seen: a system for personal information retrieval and re-use." *ACM SIGIR Forum*. Vol. 49. No. 2. ACM, 2016. https://doi.org/10.1145/2888422.2888425.

[14] Kim, H. R., and Philip K. Chan. "Personalized ranking of search results with learned user interest hierarchies from bookmarks." *WEBKDD'05 Workshop*. 2005.

[15] Schamber, Linda, Michael B. Eisenberg, and Michael S. Nilan. "A re-examination of relevance: toward a dynamic, situational definition∗." *Information processing & management* 26.6 1990: 755-776. https://doi.org/10.1016/0306-4573(90)90050-C.

[16] Teevan, Jaime, Susan T. Dumais, and Eric Horvitz. "Personalizing search via automated analysis of interests and activities." *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2005.

[17] Liu, Fang, Clement Yu, and Weiyi Meng. "Personalized web search by mapping user queries to categories." *Proceedings of the eleventh international conference on Information and knowledge management*. ACM, 2002. https://doi.org/10.1145/584792.584884.

[18] Croft, W. Bruce, and Xing Wei. *Context-based topic models for query modification*. CIIR Technical Report, University of Massachusetts, 2005.

[19] Li, Dandan, Jianwei Du, and Shuzhen Yao. "Research on Computer Science Domain Ontology Construction and Information Retrieval." *Knowledge Engineering and Management*. Springer, Berlin, Heidelberg, 2011. 603-608. https://doi.org/10.1007/978-3-642-25661-5_74.

[20] Harispe, Sébastien, et al. "Semantic similarity from natural language and ontology analysis." *Synthesis Lectures on Human Language Technologies* 8.1 2015: 1-254. https://doi.org/10.2200/S00639ED1V01Y201504HLT027.

[21] Singhal, Amit. "Introducing the knowledge graph: things, not strings." *Official google blog* 2012.

[22] Xun, Guangxu, et al. "A survey on context learning." *IEEE Transactions on Knowledge and Data Engineering* 29.1 2017: 38-56. https://doi.org/10.1109/TKDE.2016.2614508.

[23] S. Cucerzan, "Large-scale named entity disambiguation based on Wikipedia data," in Proc. EMNLP-CoNLL, 2007, vol. 7, pp. 708–716.

[24] Mandava Kranthi Kiran and K. Thammi Reddy. " An Approach Towards Establishing Reference Linking in Desktop Reference Manager", Journal of Information and Knowledge Management, World Scientific Publication (in Press)

[25] Vicknair, Chad, et al. "A comparison of a graph database and a relational database: a data provenance perspective." *Proceedings of the 48th annual southeast regional conference*. ACM, 2010. https://doi.org/10.1145/1900008.1900067.

[26] Casanovas, Pompeu, et al. "Semantic web for the legal domain: the next step." *Semantic Web* 7.3 2016: 213-227. https://doi.org/10.3233/SW-160224.

[27] Di Iorio, Angelo, et al. "Describing bibliographic references in RDF." *SePublica*. 2014.

[28] Noy, Natalya F., and Deborah L. McGuinness. "Ontology development 101: A guide to creating your first ontology." 2001.

[29] http://dumps.wikimedia.org/enwiki/20171001

[30] Decker, Stefan, and Martin Frank. "The social semantic desktop." *Digital Enterprise Research Institute, DERI Technical Report May* 2 2004: 7.

[31] Cuzzocrea, Alfredo, et al. "MapReduce-based algorithms for managing big RDF graphs: State-of-the-art analysis, paradigms, and future directions." *Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*. IEEE Press, 2017. https://doi.org/10.1109/CCGRID.2017.109.

[32] Reddy, K. Thammi, M. Shashi, and L. Pratap Reddy. "Hybrid Clustering Approach for Concept Generation." *International Journal of Computer Science and Network Security (IJCSNS)* 7.4 2007: 62-69.

[33] [33] Klyuev, Vitaly, and Yannis Haralambous. "Query expansion: Term selection using the ewc semantic relatedness measure." *Computer Science and Information Systems (FedCSIS), 2011 Federated Conference on*. IEEE, 2011.

[34] Zhang, Jiuling, Beixing Deng, and Xing Li. "Concept based query expansion using wordnet." *Proceedings of the 2009 international e-conference on advanced science and technology*. IEEE Computer Society, 2009. https://doi.org/10.1109/AST.2009.24.

[35] Varelas, Giannis, et al. "Semantic similarity methods in wordNet and their application to information retrieval on the web." *Proceedings of the 7th annual ACM international workshop on Web information and data management*. ACM, 2005. https://doi.org/10.1145/1097047.1097051.

[36] Boubacar, Abdoulahi, and Zhendong Niu. "Concept Based Query Expansion." *Semantics, Knowledge and Grids (SKG), 2013 Ninth International Conference on*. IEEE, 2013. https://doi.org/10.1109/SKG.2013.10.

[37] Bhavani, M., K. Thammi, and M. Shashi. "A rough set based approach to detect plagiarism." *TENCON 2009-2009 IEEE Region 10 Conference*. IEEE, 2009.

[38] Qiu, Yonggang, and Hans-Peter Frei. "Concept based query expansion." *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 1993.

[39] https://wordnet.princeton.edu/.

[40] https://www.marklogic.com/.