# Mining Contents in Web Pages and Ranking of Web Pages Using Cosine Similarity

**Divya C.**

Department of Computer Science Engineering, Christ University Faculty of Engineering, Bangalore, India

**Abstract:** ***Now a day's internet has become a part of life because of which web pages have became a key communication and information medium for various organizations. Web pages typically contain a large amount of information that is not part of the main contents of the pages, e.g.; banner ads, navigation bars, copy right and privacy notices, advertisements which are not related to the main content (relevant information). In this paper the system use HTML Parser to construct DOM (Document Object Model) tree from which Content Structure Tree (CST) is constructed which can easily separate the main content blocks from the other blocks. The paper also introduces a method for calculating the rank of a web page based on the content similarity between the web documents and the user query, since usually when the user searches for web pages using a key word many web pages are retrieved the user might not be knowing which web pages are most relevant to overcome this problem the web pages are ranked using Cosine Similarity and Jaccard Similarity. The Cosine Similarity and Jaccard Similarity are implemented with the stop word removal algorithm. Many experiments were conducted for both Cosine Similarity and Jaccard Similarity. The obtained results have been compared to decide which one work best. The result was that Cosine Similarity retrieved most relevant pages to the user than the Jaccard Similarity.***

**Keywords:** Content mining, DOM tree, CST tree, TF-IDF, Cosine Similarity.

## 1. Introduction

The web is a medium for accessing a great variety of information stored in different parts of the world. Information is mostly in the form of unstructured data. As the data on the web grows at explosive rates, it has lead to several problems such as increased difficulty of finding relevant information, extracting potentially useful knowledge and learning about consumers or individual users. Efforts are being made to make such data available, usually in some structured form such as table, for querying and further manipulation. Web mining is an emerging research area focused on resolving these problems. This is web mining. Some of the techniques of web mining are web content mining, Web usage mining, Web structure mining.

Web content mining extract information from web page content. Two groups of web content mining are those that directly mine the content of documents and those that improve on the content search of other tools like search engine. For Web content mining data can be image, text and links. Any mining method focuses on information extraction and integration. Web content mining extracts information from different web sites for its access and knowledge discovery.

One of the most important functions of the Internet is information retrieval. However, resource discovery on the Internet is still frustrating and inefficient when simple keyword searches can convey hundreds of thousands of documents as results. The continuous growth in the size and use of the Internet thus creates difficulties in the search for information. As a result, when a user enters a keyword in a search, the returned result is often a large list of web pages, many of which are irrelevant pages, moved pages, abandoned pages, etc.

Therefore, a sophisticated method to fetch relevant pages to the user is important, particularly as the Internet grows in size and also a part from the useful information on the web; it usually has such information as navigation panels, copyright notices, banner ads, etc. Although these information item are useful for human viewers and necessary for the Web site owners, they can seriously harm automated information collection and Web data mining, e.g. Web page clustering, Web page classification, and information retrieval. So how to extract the main content blocks become very important.

Therefore we focus on extracting the relevant documents to the user query. The content from the HTML tags such as , <alt>, <a>, <text> are retrieved with the help of DOM ( Document Object Model ) and CST (Content Structure Tree ) and convert into plain text using HTML parser. From the plain text terms are extracted which are called as tokens. For each token the corresponding weight is calculated by using the formula term frequency inverse document frequency then the user entered query is matched with the web documents using cosine similarity formula then the web documents are ranked according to the obtained value

## 2. Literature Survey

### 2.1 Basic Concept

**World Wide Web**

With the dramatically quick and explosive growth of information available over the Internet, World Wide Web has become a powerful platform to store, disseminate and retrieve information as well as mine useful knowledge. Due to the properties of the huge, diverse, dynamic and unstructured nature of Web data, Web data research has encountered a lot of challenges, such as scalability, multimedia and temporal issues etc. As a result, Web users are always drowning in an "ocean" of information and facing the problem of information overload when interacting with the web.

### 2.2 Web Mining

Web mining is the use of data mining techniques to automatically discover and extract information from Web documents and services. This area of research is so huge today partly due to the interests of various research communities, the tremendous growth of information sources available on the Web and the recent interest in e-commerce. Web mining is often associated with IR or IE. However, web mining is not the same as IR or IE.

#### 2.2.1 Web Mining Taxonomy

Web mining field consists of main three categories, Web usage mining, Web structure mining, and Web content mining. Web usage mining refers to the discovery of user access patterns from Web usage logs. Web structure mining tries to discover useful knowledge from the structure of hyperlinks. Web content mining aims to extract/mine useful information or knowledge from Web page contents.

### 2.3 Related papers

From time to time, many extraction systems have been developed. In [1], C. Li et al. propose a method to extract informative block from a web page based on the analysis of both the layouts and the semantic information of the web pages. They needed to identify blocks occurring in a web collection based on the Vision-based Page Segmentation algorithm. In [2], L. Yi et al. propose a new tree structure, called Style Tree to capture the actual contents and the common layouts (or presentation styles) of the Web pages in a Web site. Their method can difficult to capture the common presentation for many web pages from different web sites.

In [3], Y. Fu et al. propose a method to discover informative content block based on DOM tree. They removed clutters using XPath. They could remove only the web pages with similar layout. In [4], P. S. Hiremath et al. propose an algorithm called VSAP (Visual Structure based Analysis of web Pages) to exact the data region based on the visual clue (location of data region / data records / data items / on the screen at which tag are rendered) information of web pages.

In [5] S. H. Lin et al. propose a system, InfoDiscoverer to discover informative content blocks from web documents. It first partitions a web page into several content blocks according to HTML tag <TABLE>. In [6] D. Cai et al. propose a Vision-based Page Segmentation (VIPS) algorithm that segments web pages using DOM tree with a combination of human visual cues, including tag cue, color cue, size cue, and others. In [7], P. M. Joshi propose an approach of combination of HTML DOM analysis and Natural Language Processing (NLP) techniques for automated extractions of main article with associated images form web pages. Their approach did not require prior knowledge of website templates and also extracted not only the text but also associated images based on semantic similarity of image captions to the main text.

In [8], Y. Li et al. propose a tree called content structure tree which captured the importance of the blocks. In [9], R, R, Mehta propose a page segmentation algorithm which used both visual and content information to obtain semantically meaningful blocks. The output of the algorithm was a semantic structure tree. In [10], S. Gupta proposes content extraction technique that could remove clutter without destroying webpage layout. It is not only extract information from large logical units but also manipulate smaller units such as specific links within the structure of the DOM tree. Most of the existing approaches based on only DOM tree.

## 3. Methodology

### 3.1 Architecture based on Cosine Similarity

The steps in the architecture are as follows;

- Term extraction
- Pre-Processing
- TFIDF calculation
- Similarity calculation between the user query and the web documents

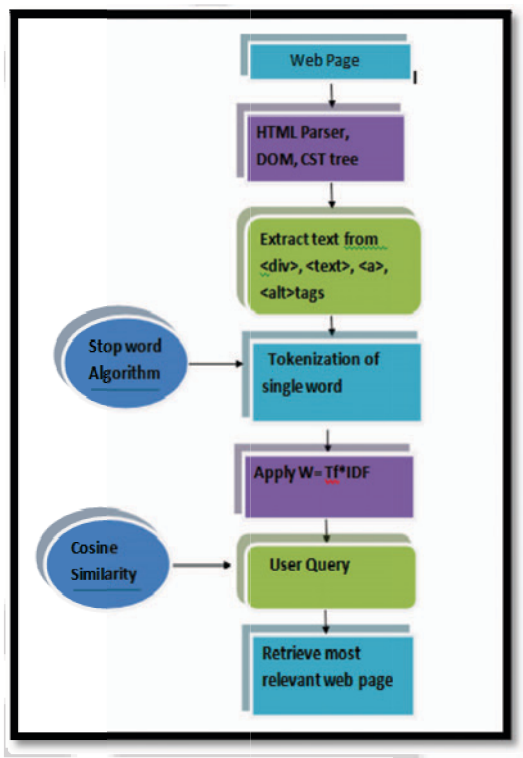


**Figure 1:** Architecture based on Cosine Similarity

#### 3.1.1 Term extraction (Tokenization )

The architecture is based on the analysis of actual contents of the Web pages in a given Web site. A web page usually contains main content blocks and noise content blocks. Only the main content blocks represent the informative part that is really we want to know. Thus, in our first task, we will use the HTML Parser to create a DOM tree representation of the original html source.

**(a) DOM tree**

We will use open source HTML Parser that builds a DOM tree from a page using its HTML code. In a DOM tree, tags are internal nodes and the detailed texts, images or hyperlinks are the leaf nodes. Figure 2 shows some html

segments and its corresponding DOM tree in figure 3. In the DOM tree, we need to tidy some unnecessary nodes, such as script, style or other customized nodes. HTML Web pages begin from the BODY tag since all the viewable parts are within the scope of BODY. If we need to extract useful informative block from the web pages, we need a more powerful structure tree called Content Structure Tree (CST) as shown in figure 4 [11].

Then we can find which content block is more important in the CST tree.

```
<body>
  <div id = "wrapper">
    <a href="#"></a>
    <img src="img"/ >
    <span>text</span>
  </div>
<img src="img"/>
  <a href="#"></a>
<table>
.....
</table>
<span>text</span>
</body>
```

**Figure 2:** HTML page

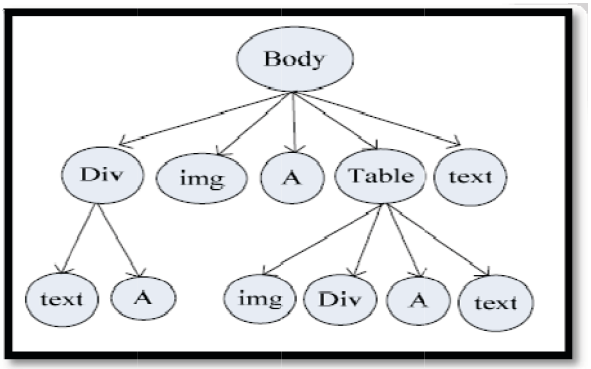


**Figure 3:** Segment of HTML code and its DOM tree

Second, our task is to find relevant information from the web pages in the site. So, we construct a Content Structure Tree (CST) based on the DOM tree.

**(b) Content Structure Tree**

A Page Content Structure Tree consists of two types of nodes: HtmlItem node and content node that contains text node, image node, link node, etc. An HtmlItem node represents a block which is generated by body tag, div tag and table tag. A content node represents the actual content Cosine of the web page such as text, image and link [11].

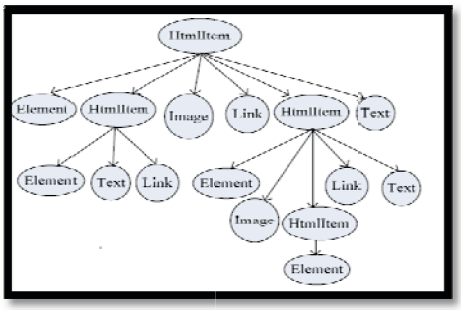


**Figure 4**: Content Structure tree

In addition to text content it also contains tag information this tag information is removed using the HTML parser and punctuation should be removed from the web document to extract meaningful terms. Tokenization operation is performed. Given a web document, tokenization breaks it into pieces called tokens, perhaps at the same time throwing away certain characters, such as punctuation. Here is an example of tokenisation:

Input: Friends, Romans, Country men, lend me your ears;
Output: Friends Romans, Countrymen lend me your ears
Contents from the web documents are extracted by removing the tags and special symbols with the use of tokenisation. Extracted terms are given as an input to the next step.

**3.1.2 Pre-Processing (stop word removal)**

Some common stop words, such as is, was, are, were, what, etc., do not provide any useful information in the process of identifying the similarity among the pages. Therefore, they are removed to avoid confusion. For stop word removal, initially the file of stop words is created and the terms extracted from the web pages are compared with the file of stop words. Stop words found in the extracted collection of terms are removed.

**3.1.3 TFIDF calculation**

Calculation of term frequency which is nothing but the content weight estimation, because the user searching token might be present in <alt> tag, <a> tag and <text> tag adding all together results in content weight estimation.

**ContentWeight = TextWeight+ ImageWeight + LinkWeight**

**(a) Basics of Model**

Term Frequency/Inverse Document Frequency model is based on the principle that if a term occurs more within a document(TF) and rare within the corpus of documents(IDF), then that term would be having high discriminative power to distinguish between relevant and non-relevant documents[**12**]. TF/IDF is a type of keyword search based algorithm. The document having high TF/IDF value is having strong relationship with the query and would be more relevant for the user. Inverse Document Frequency (IDF) is based on the fact that a term which occurs in many documents is not a good discriminator and should be given less weight than one which occurs in few documents [13].

Let there are N documents in the collection, and that term ti occurs in ni of them. IDF is calculated as

$$idf(t_i) = \log \frac{N}{ni} \qquad (1)$$

Term Frequency/Inverse Document Frequency model incorporates local and global information. Encoding TF/IDF is simple.

$$w_i = tf_i * \log \frac{D}{df_i} \qquad (2)$$

Where
tfi = term frequency (term counts) or number of times a term i occurs in a document. This accounts for local information.
dfi = document frequency or number of documents containing term i
D = number of documents in a database.

The *dfi /D* ratio is the probability of selecting a document containing a queried term from a collection of documents. This can be viewed as a global probability over the entire collection. Thus, the *log(D/dfi)* term is the *inverse document frequency, IDFi* and accounts for global information.

The **tf–idf** weight (term frequency–inverse document frequency) is a weight often used in information retrieval and text mining. This weight is a statistical measure used to evaluate how important a word is to a document in a collection or corpus. The importance increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus. Variations of the tf–idf weighting scheme are often used by search engines as a central tool in scoring and ranking a document's relevance given a user query.

One of the simplest ranking functions is computed by summing the tf-idf for each query term; many more sophisticated ranking functions are variants of this simple model. Many search engines use combination of keyword search based algorithm e.g. tf-idf based ranking and link based algorithm e.g. PageRank based ranking **[**14**].**

**(b) Term Weight calculation Example**
Suppose there is a set of English text documents and user wants to determine which document is most relevant to the query "the brown cow." A simple way to start out is by eliminating documents that do not contain all three words "the," "brown," and "cow," but this still leaves many documents. To further distinguish them total number of terms in a document is counted and summed together; the number of times a term occurs in a document is called its *term frequency*. However, because the term "the" is so common, this will tend to incorrectly emphasize documents which happen to use the word "the" more, without giving enough weight to the more meaningful terms "brown" and "cow". Also the term "the" is not a good keyword to distinguish relevant and non-relevant documents and terms like "brown" and "cow" that occur rarely are good keywords to distinguish relevant documents from the non-relevant documents. Hence an *inverse document frequency* factor is incorporated which diminishes the weight of terms that occur very frequently in the collection and increases the weight of terms that occur rarely.

Consider a document containing 100 words wherein the word cow appears 3 times. Following the previously defined formulas, the term frequency (TF) for cow is then 0.03 (3 / 100). Now, assume we have 10 million documents and cow appears in one thousand of these. Then, the inverse document frequency is calculated as in (10000 / 1 000) = 9.21. The TF-IDF score is the product of these quantities: 0.03 * 9.21 = 0.28

The results of applying Term Frequency-Inverse Document Frequency (TF-IDF) to determine what words in a corpus of documents might be more favourable to use in a query can be calculated with this model. As the term implies, TF-IDF calculates values for each word in a document through an inverse proportion of the frequency of the word in a particular document to the percentage of documents the word appears in. Words with high TF-IDF numbers imply a strong relationship with the document they appear in, suggesting that if that word were to appear in a query, the document could be of interest to the user. This algorithm efficiently categorizes relevant words that can enhance query retrieval. In this study, tf-idf is used as the base technique for calculating the token frequency. The use of tf-idf with cosine similarity will be explained in detail in further pages.

**3.1.4 Similarity calculation between the user query and the web documents**
The similarity in vector space models is determined by using associative coefficients based on the inner product of the document vector and query vector, where word overlap indicates similarity. The inner product is usually normalized. The similarity measure used in this dissertation is the cosine similarity, which measures the angle between the document vector and the query vector.

$$Cosine(\mathrm{d}_i, \mathrm{q}) = \frac{d_j . q}{\| d_j \| \| q \|} = \frac{\sum_{i=1}^{t} w_{i,j} * w_{i,q}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^2} * \sqrt{\sum_{i=1}^{t} w_{i,q}^2}} \qquad (3)$$

**3.2 Architecture based on Jaccard Similarity**

The architecture is based on the analysis of actual contents of the Web pages in a given Website. A web page usually contains main content blocks and noise content blocks. Only the main content blocks represent the informative part that is really we want to know. Thus, in our first task, we will use the HTML Parser to create a DOM tree representation of the original html source. Second, task is to find relevant information from the web pages in the site. So, we construct a Content Structure Tree (CST) based on the DOM tree. In addition to text content it also contains tag information this tag information is removed using the HTML parser and punctuation should be removed from the web document to extract meaningful terms. Tokenization operation is performed. Given a web document, tokenization breaks it into pieces called tokens, perhaps at the same time throwing away certain characters, such as punctuation.

During tokenization some common stop words, such as is, was, are, were, what, etc., do not provide any useful information in the process of identifying the similarity among the pages. Therefore, they are removed to avoid confusion. For stop word removal, initially the file of stop words is created and the terms extracted from the web pages are compared with the file of stop words. Stop words found in the extracted collection of terms are removed. Then the weight of each token is calculated, finally Jaccard similarity is used to measure the similarity between the query and the web documents whose computed value ranks the web pages according to the highest value to lowest value. The top most web pages are considered as the most relevant pages or documents to the user query.

### 3.2.1 Explanation of Jaccard Similarity

The Jaccard Similarity calculates the similarity between sets and is defined as the size of intersection of A and B divided by the size of the union of A and B

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} \qquad (4)$$

## 4. Analysis of Cosine and Jaccard Similarity

Analysis of Cosine and Jaccard Similarity are based on the two experiments done, for the first experiment input was the 3 web pages whose contents within the page were only few lines. The result observed was that initially search for token one both Cosine and Jaccard Similarity showed the same result , as the tokens number were increased for search for example two, three, five and seven as shown in figure 5.28-5.31 the cosine similarity showed the most relevant page when compared to the Jaccard Similarity. Cosine Similarity retrieved web pages having most of the searched token than the Jaccard Similarity.

For the second experiment input was the 10 web pages whose contents within the web pages were more than few lines. The result observed was that initially search for token one showed the same result for both Cosine and Jaccard Similarity which is shown in figure 5.32, as the tokens were increased for search for example two, three, four, five, six, seven and nine which are shown in figure 5.32-5.39 respectively the cosine similarity showed the most relevant page when compared to the Jaccard Similarity because Cosine Similarity retrieved web pages having most of the searched tokens when compared to the Jaccard Similarity.

**Experiment 1:** Few web pages with less content

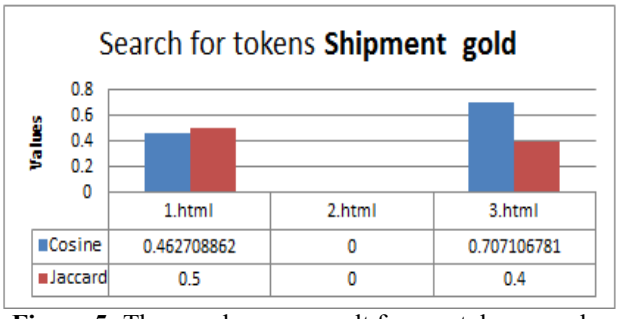


**Figure 5:** Three web pages result for one token search

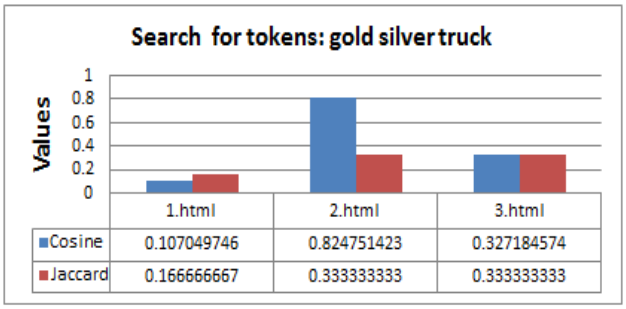


**Figure 6:** Three web pages result for three tokens search

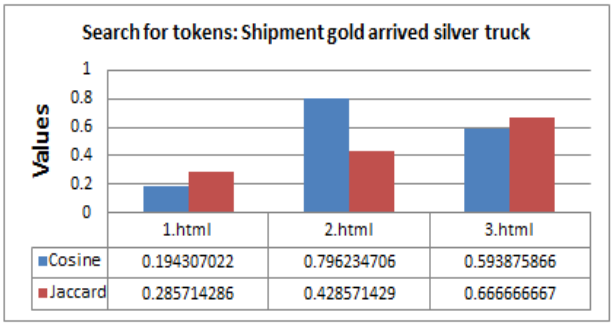


**Figure 7:** Three web pages result for five tokens search

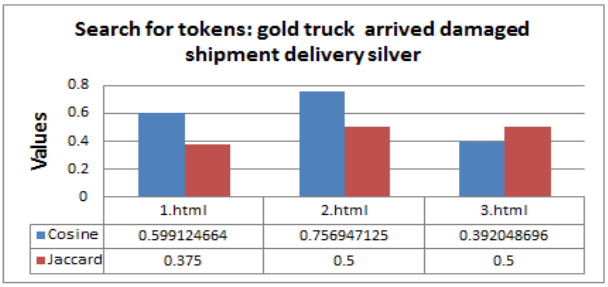


**Figure 8:** Three web pages result for seven tokens search

**Experiment 2: More number of web pages with huge contents**

Dataset consist of 10 web pages taken form computer hardware industry such as capedge.html, capmkt.html, contact.html, equipfin.html, equisale.html, home.html, info.html, iword.html, plain.html, sgi_info.html.

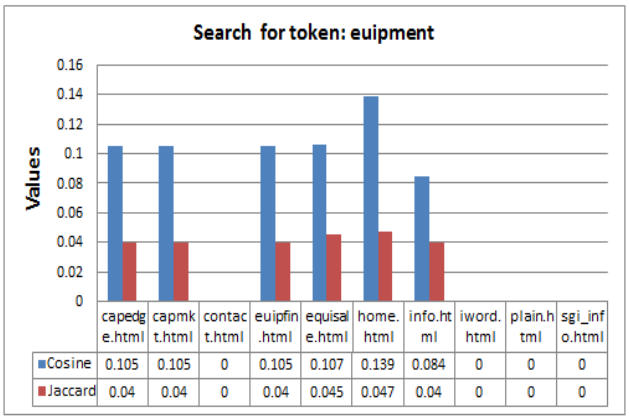


**Figure 9:** Ten web pages result for one token search

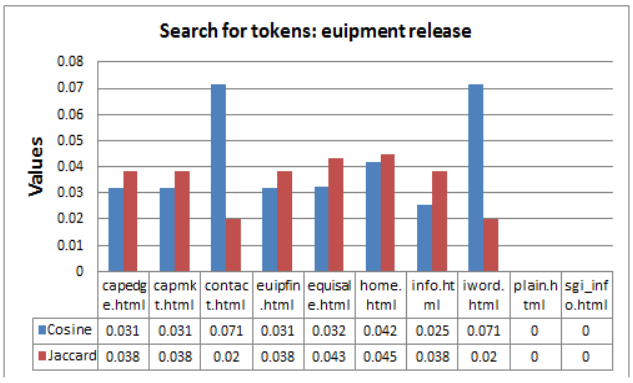


**Figure 10:** Ten web pages result for two tokens search

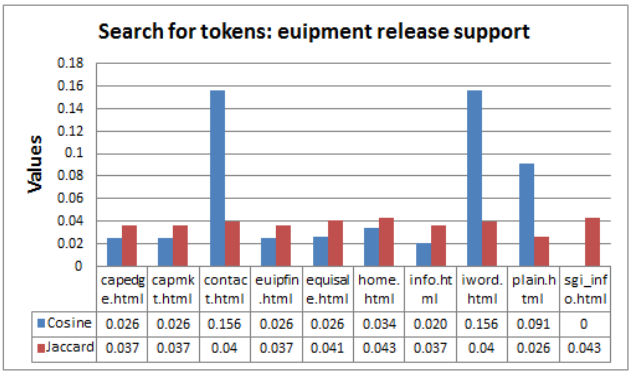


**Figure 11:** Ten web pages result for three tokens search

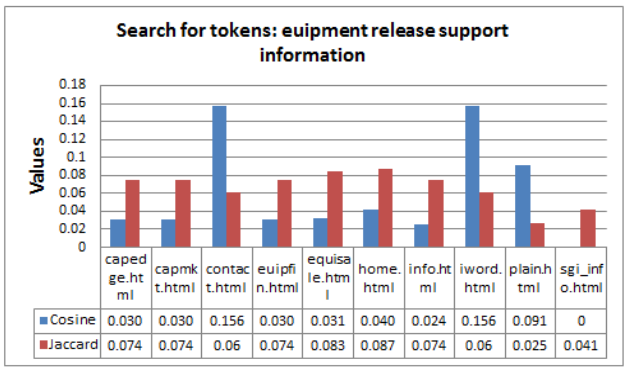


**Figure 12:** Ten web pages result for four tokens search

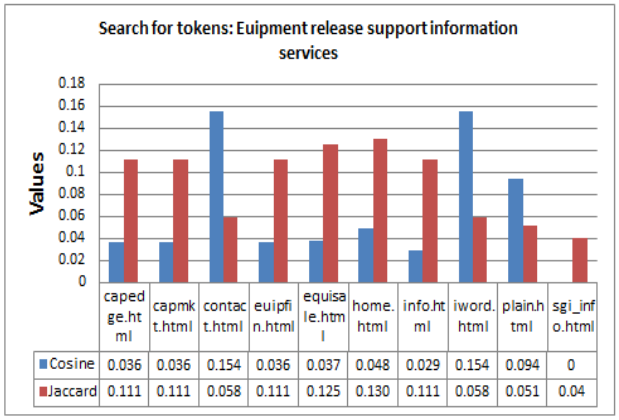


**Figure 13:** Ten web pages result for five tokens research

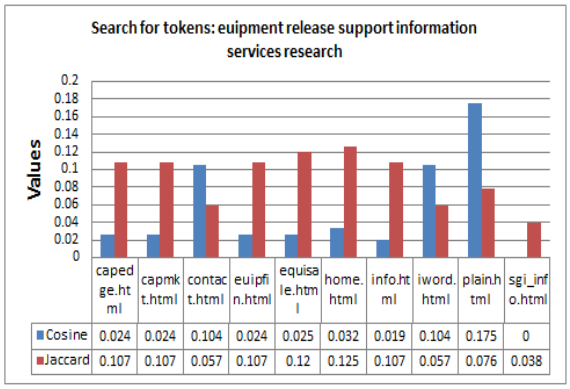


**Figure 14:** Ten web pages result for six tokens search

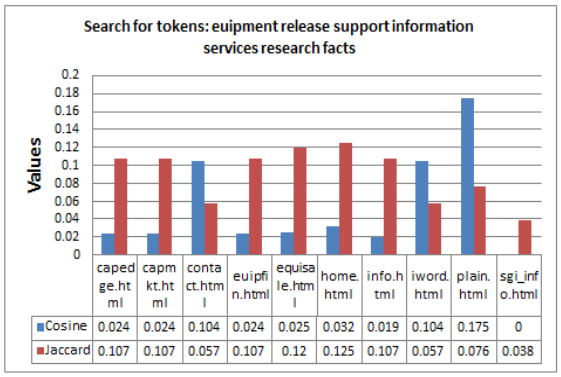


**Figure 14:** Ten web pages result for seven tokens search

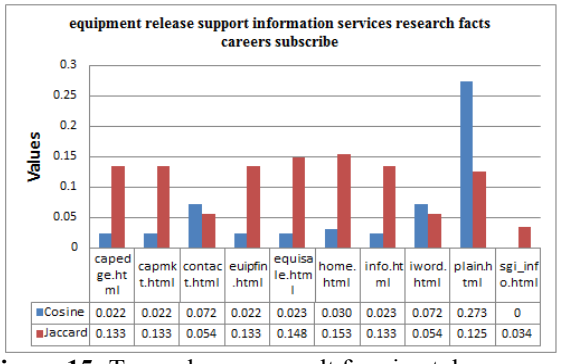


**Figure 15:** Ten web pages result for nine tokens search

## 5. Conclusion

TF-IDF is an efficient and simple algorithm for matching words in query to web pages that are relevant to that query. TF-IDF returns web pages that are highly relevant to a particular query. If a user wishes to input a query for a particular topic, TFIDF can find web pages that contain relevant information on the query. Furthermore, encoding TF-IDF is straightforward, making it ideal for forming the basis for more complicated algorithms and query retrieval systems. Despite its strength, TF-IDF has its limitations. In terms of synonyms, notice that TF-IDF does not make the jump to the relationship between words. If the user wanted to find information about, say, the word “priest”, TF-IDF would not consider documents that might be relevant to the query but instead use the word “reverend”.

By using Vector Space Model with TF-IDF Model, the web pages can be ranked more accurately than probabilistic model. Cosine similarity and vector length calculation of Vector Space Model combined with tf and idf value and similarity based on tf*idf. TF-IDF Model

provides a better model for distinguishing between relevant web pages from non-relevant web pages.

Cosine Similarity returns most relevant web pages to the user then the Jaccard Similarity for any number of keyword or tokens search where as Jaccard Similarity works well for single token search, sometimes for two tokens search as the number of search tokens increases the performance of Jaccard Similarity degrades when compared to Cosine Similarity.

According to the requirements of the user, the further improvements in this algorithm can be better removal of stop words can be made dynamic and better technique for stemming. The Cosine Similarity can be compared with other similarity techniques such as Dice Co-efficient, Euclidean Distance. After Ranking the web pages using Cosine Similarity, a data mining technique such as classification can be use to classify the web pages based on the keyword search for example business, education etc or by using some threshold values

## References

[1] C. Li, J. Dong, and J. Chen, "Extraction of Informative Blocks from Web Pages Based on VIPS", 1553-9105/ Copyright January 2010.
[2] L. Yi, B. Liu, and X. Li, "Eliminating Noisy Information in Web Pages for Data Mining", in Proc. ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD-2003).
[3] Y. Fu, D. Yang, and S. Tang,"Using XPath to Discover Informative Content Blocks of Web Pages", IEEE. DOI 10.1109/SKG, 2007.
[4] P. S. Hiremath, S. S. Benchalli, S. P. Algur, and R. V. Udapudi, "Mining Data Regions from Web Pages", International Conference on Management of Data COMAD, India, December 2005.
[5] S. H. Lin and J. M .Ho, "Discovering Informative Content Blocks from Web Documents", in Pro. ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp.588-593, July 2002.
[6] D. Cai, S. Yu, J. R. Wen, and W. Y. Ma, "VIPS: a Vision- based Page Segmentation Algorithm", Technical Report, MSR-TR, Nov. 1, 2003.
[7] P. M. Joshi, and S. Liu, " Web Document Text and Images Extraction using DOM Analysis and Natural Language Processing", ACM, DocEng, 2009.
[8] Y. Li and J. Yang, "A Novel Method to Extract Informative Blocks from Web Pages", IEEE. DOI 10. 1109/ *JCAI*, 2009.
[9] R. R. Mehta, P. Mitra, and H. Kamick, "Extracting Semantic Structure of Web Documents Using Content and Visual Information", ACM, Chiba, Japan, May 2005.
[10] S. Gupta, G. Kaiser, D. Neistadt, and P. Grimm,"DOM-based Content Extraction of HTML Documents", *Pro.* 12 th International Conference on WWW, ISBN: 1-58113-680-3, 2003.
[11] Swe Swe Nyein, "Mining Contents in Web Page Using Cosine Similarity" University of Computer Studies, Mandalay, 2011
[12] P. M. Joshi, and S. Liu, " Web Document Text and Images Extraction using DOM Analysis and Natural Language Processing", ACM, DocEng, 2009.
[13] Stephen Robertson- Microsoft Research "Understanding Inverse Document Frequency: On theoretical arguments for IDF".
[14] Y. Li and J. Yang, "A Novel Method to Extract Informative Blocks from Web Pages", IEEE. DOI 10. 1109/ *JCAI*, 2009.

## Author Profile

**Divya C.** received her B.E in Information Science from VTU in 2009. She received her M.Tech in Computer Science from Christ University in 2013.