International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 07 Issue: 12 | December - 2023 SJIF Rating: 8.176 ISSN: 2582-3930

Webpage Duplication Detection using Machine Learning

E. Nandhu, E. Badrinath, E. Hasmitha, E. Lohitha, H. Packiaraj

School of Engineering, Malla Reddy University, Secunderabad – 500010, Telangana, India;

Abstract

Web search engines face challenges with nearly identical and duplicate webpages, leading to increased storage requirements and slower results. Researchers have addressed this issue by exploring methods to detect such similarities. One approach involves utilizing sentence-level characteristics alongside fingerprinting techniques. The proposed method employs K-mode clustering for large sets of web documents and compares fingerprints and sentence features to identify almost identical web pages quickly and accurately. Experimental results on web page collections validate the effectiveness of this approach.

1. Introduction

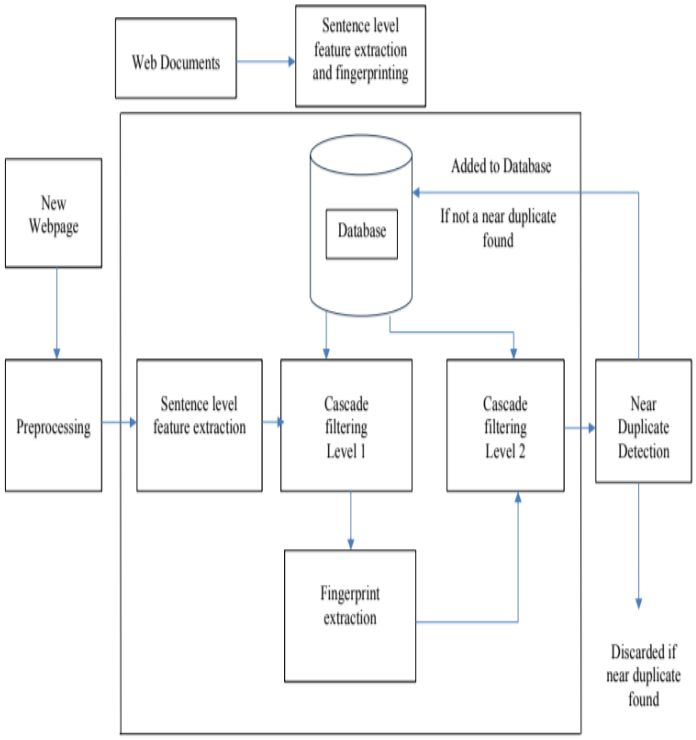
Web mining is a subset of data mining that concentrates on utilising data mining techniques to analyse the World Wide Web and extract information. It incorporates ideas from the Semantic Web, Internet technology, data mining, and the World Wide Web. Web crawling is the methodical exploration of the internet using web crawlers, often known as spiders, which download resources, gather URLs, and add new ones to a frontier until a stopping condition is satisfied. In web crawling, identifying near-duplicate and duplicate web sites is essential. Search engines rely on web crawlers to fill in local indexed repositories for user queries. On the internet, it's typical to see almost duplicate pages with minor changes, such as adverts or timestamps. Eliminating close duplicates improves search index quality, saves storage costs, and conserves network traffic.The burden on distant hosts that provide web pages is also lessened by this procedure. There are difficulties in detecting near duplicates. Particularly now that there are billions of webpages, creating multi-terabyte databases. One major problem is the crawl engine's capacity to process billions of webpages. Identifying duplicates is often accomplished by fingerprinting, which produces a condensed summary of webpage characters. Checksum methods are useful in locating precise copies that result from plagiarism or mirroring. In conclusion, web mining automatically extracts information from the World Wide Web using data mining techniques. Crawlers are the tools used by search engines and other applications to crawl the web. In order to improve search index quality, save storage costs, and facilitate efficient crawling, it is imperative to identify duplicate and almost duplicate web pages. Among the difficulties are the large number of webpages and the especially with the scale of webpages reaching billions, resulting in multi-terabyte databases. The crawl engine's ability to handle billions of webpages is a significant concern. Fingerprinting, generating a succinct digest of webpage characters, is a common method for detecting duplicates. Checksum techniques help identify exact duplicates caused by mirroring or plagiarism. In summary, web mining utilizes data mining techniques to automatically extract information from the World Wide Web. Web crawling, performed by crawlers, is vital for search engines and other applications. Detecting duplicate and near-duplicate web pages is essential for efficient crawling, reducing storage costs, and improving search index quality. Challenges include the vast scale of webpages and the capability of the crawl engine. Fingerprinting and checksum techniques play crucial roles in identifying duplicates.

2. Related work

Near duplicate web page detection techniques have been very popular recently, and a lot of research is being done in this important field. This review of the literature highlights several studies that address the identification of almost identical web pages. An effective method for identifying nearly duplicate web pages during web crawling has been reported by V. A. Narayana et al. It involves extracting keywords from the crawled pages and calculating the similarity score between two pages. A document is deemed almost identical if its similarity score exceeds a predetermined threshold. The two "state-of-the-art" algorithms—Charikar's random projection based technique and Broder et al.'s shingling algorithm—have been compared by Monika Henzinger method to identify very identical webpages.Either popular web search engines or developers employed both algorithms. They conducted a large-scale comparison between the two algorithms using a set of 1.6 billion unique web pages. The findings demonstrated that while both methods had good precision for near-duplicate pairs on distinct sites, they were unsuccessful in locating near-duplicate pairs on the same site. Due to the fact that Charik's method produced many near-duplicate pairs on various sites with an overall higher precision of 0.50 as opposed to 0.38 for Broder et al.'s technique. They have developed a combination algorithm that achieves 79% of the recall of the other algorithms with a precision of 0.79.Gurmeet Singh Manku et al. made two scientific contributions in the area of creating a near-duplicate detection system for a multi-billion page repository. They first claimed that Charikar's fingerprinting method worked for their purpose. Subsequently, they demonstrated an algorithmic method for determining which f-bit fingerprints currently exist that diverge from a given fingerprint in a maximum of k bit-positions, for small k. Their method is effective for all batch queries as well as online inquiries. The feasibility of their approach was validated by empirical assessment utilising authentic data. With a focus on IMatch, A. Kolcz and A. Chowdhury proposed a randomization-based technique to enhance its signature stability. The proposed methodology consistently outperformed conventional I-Match, exhibiting a 40–60% relative gain in near-duplicate recall.Remarkably, only modest increases in processing power were needed to counteract the significant gains in detection accuracy. They also addressed incorrect matches, a secondary challenge that was critical to I-Match's ability to fingerprint large documents. The large amount of data made it thought-provoking to identify nearly identical webpages on the web. Therefore, a method for identifying duplicate data must be offered in order to give the user relevant search results. In order to identify duplicates and near duplicates, Ranjna Gupta et al. presented a concept that offers techniques that will function both online and offline based on favoured and disfavored user questions.A modest yet efficient sentence-level statistics-based approach for identifying near-duplicate texts has been presented by Hung-Chi Chang and Ten-Hour Wang. The method is independent of language. The results of the experiment demonstrated the suggested approach's high efficiency and good accuracy in identifying near-duplicates in news archives. For a specific domain, Hannaneh Hajishirzi et al. developed a nearduplicate document detection approach. Using this approach, each document was viewed as a real-valued sparse k-gram vector, and the weights were trained to maximise for specific similarity functions, like the Jaccard coefficient or the cosine similarity. Additionally, using a locality-sensitive hashing approach, these vectors were transferred to a limited set of hash values as document signatures for efficient similarity computation.The occurrence of near duplicates is a severe matter, as stated by Bingfeng Pi et al. Then, they went over a number of discoveries, including the nature of the issue and the advantages of the SimHash-based method, and they gave an example of how a SimHash functions and the advantages it has for locating near duplicates.

3. Proposed Methodology

This section covers a method for identifying almost identical web pages. In digital libraries or large document repositories like the government declassification project, duplicate papers are commonly discovered. The frequency of near duplicates raises the amount of storage space needed for the index, reduces providing outcomes, and annoy users. It's critical to have capable duplicate document detection not simply to enable searching for related papers and filtering out extraneous data from the large document databases. Reducing the nearly identical pages results in lower storage expenses and higher-quality search indexes along with extensive bandwidth control. In the suggested method, we make use of using the simhash technique and sentence level extraction, near duplicates can be found.Our suggested method enables faster comparison and search while significantly reducing the storage space. At first, the web pages that are crawled are When parsed, it eliminates stop words and frequent words, Java scripts, HTML elements, and stems words that are left. The near duplicates are then found and eliminated using the sentence level function and simhash. Our suggested method now includes the K-Mode clustering algorithm to produce faster and better results.



3.1 Preprocessing

Before the fingerprint and feature comparisons at the sentence level, preprocessing is carried out. A web page can be preprocessed to provide a list of keywords. Crawling, parsing, stop word removal, and stemming make up the preprocessing. The site pages are retrieved through crawling from the database. The web pages are represented in parsing by a linear structure with a predetermined language. After the online documents have been parsed, linking terms like "is," "as," and so forth are eliminated using the stop word removal technique. The process of stemming reduces the words in the document to their most basic form. For additional processing, the preprocessed web pages are sent to cascade filters.

a.Web Crawling:

A web crawler is essentially an automated script or programme that "crawls" across websites, carefully gathering data from each page to create an index. Other names for web crawlers include web spiders, web robots, crawlers, and automatic indexers. Web crawling is frequently used in conjunction with search engines. Web crawlers are compelled by search engines to gather data from publicly accessible webpages.

The new set of variables obtained are now trained using recurrent neural network algorithm (back propagation). After the training of data is completed the third task is to apply LSTM over the trained data to obtain the predictions and to generate the error rate (RMSE value). The working of LSTM is defined briefly below.

The back propagation is through time as the output gradient depends on the previous values other than the current values only. LSTM uses different functions in calculation of hidden states. LSTM contains a memory cell, having four main elements

Their main objective is to gather information so that when users enter a search phrase on their website, they can promptly provide the user with relevant websites. Notwithstanding the wide range of uses for Web crawlers, they are all essentially the same. This is the primary method by which Webcrawler does its work. After downloading the website, navigate through it to find all the links, and Repeat the procedure for every link that is obtained. This procedure will persist till the crawler is shut down. A crawling loop consists of pulling a URL from the queue, using HTTP to download the corresponding file, navigating the page to find new URLs, and adding the URLs that haven't been visited to the queue.

b.Parsing:

The process of organising a linear representation in accordance with a specified grammar is known as parsing. Once a webpage has been retrieved, we must parse its content to gather data that could potentially influence and direct the crawler's future steps. In parsing, this could mean extracting a hyperlink or URL simply, or it could mean going through a more

involved process of cleaning up the HTML content so that the HTML tag tree can be examined. In addition to removing stop words and stemming the remaining words, parsing entails converting the extracted URL to a canonical form.

c. Stop word Removal:

Certain frequently used words in a web content have less meaning than keywords; therefore, it is required and advantageous to delete them. Usually, in order to provide the most relevant result, search engines eliminate these frequently used words—also referred to as "stop words"—from a keyword phrase. To improve search performance, all stop words—for example, often used terms like "a" and "the"—are removed from multiple word queries. Common stop words include "it", "can", "an", "and", "by", "for", "from", "of", "the", "to", and "with".

Stop words are eliminated while parsing a document to gather content information or when identifying new URLs that the page suggests.

d. Stemming:

For the sake of many applications, morphologically similar terms might be regarded as equal because they often have similar semantic implications. Because of this, numerous stemming algorithms and stemmers have been created to simplify a word to its stem or root form. Instead of using the original word, the stems are used to indicate the important terms in a document or query. Lemmatization [30] is an algorithm that aims to reduce a word to its linguistically correct root, hence making it easier to reduce all words with the same root to just one.

3.2 Cascade Filtering

The intended cascade filtering is used to identify duplicate web pages following the preparatory stages. The two filtering methods used in the suggested cascade filtering are applied sequentially, one after the other. Our method compares features at the sentence level and uses fingerprint comparison. The goal of applying these techniques is to reduce execution time by using sentence level feature filtering approach and to increase result precision by using fingerprinting comparison technique. We can get a solution that is efficient, accurate, and takes less time by combining these approaches. Sentence features alone are compared in sentence level feature comparison, saving time over keyword comparison. By using this sentence-level functionality on a vast amount of web pages, it eliminates web documents that don't meet the requirements, which reduces the amount of documents filtered in for the next steps and serves as a filter while saving time. Using the Sim-Hash algorithm, the fingerprint of the document is computed in the fingerprint technique, and comparison is performed to find near copies.

a. Sentence Level Feature Extraction:

The first filtering method that is essential for near duplicate web page detection is sentence level feature extraction. Here, the feature vector is the total number of sentences in a web page. Let Sk represent the total number of sentences on the recently added web page and Sdi represent the total number of sentences in the di th document. By obtaining the total amount of phrases in each web document, a comparison is done between the newly uploaded webpage and every other webpage in the database. The compared page is filtered in if there is a difference between the new and compared webpages' total number of lines that is less than the sentence threshold (Ts) di k s | S - S | < T. This function functions as a filter since it reduces the amount of web documents that are entered for the second stage of filtering.Here, just numerical values—rather than string values—are compared, which reduces calculation time and disc space usage. We can get rid of the documents that are really identical using this method. As a result, the fingerprint comparison process will require fewer inputs, resulting in the identification of the fingerprints of a smaller number of web pages rather than all of the database's pages, which will save time. The fingerprint may then be found by using the simhash technique on each of the bits filtered in.

b. Fingerprint Comparison:

The sentence level comparison filter is followed by a second filter made from fingerprints extracted from a web page. The Sim-Hash algorithm is used in fingerprint calculation to identify close duplicates.

Sim-Hash is a great method for identifying near copies because of two significant, though rather contradictory, features. They are as follows: (1) A document's fingerprint is a "hash" of its features; and (2) Hash values are similar across related documents. By comparing the corresponding hash values of the two documents, it helps ascertain whether or not they are similar. Every web document that is screened in is first transformed into a set of keywords. The weight of each keyword is assigned based on how many times it appears in the document. Next, we convert a high-dimensional vector into a fingerprint of f bits, where 'f' is quite modest in relation to the original dimensionality.

In this case, let's preprocess the input document "D" and create a keyword list. A f dimensional vector V is initialised with zeros in each dimension. The document's keywords are then hashed into a f bit hash value for each one. The weight of that word determines how much the f bits increase or decrease the vector's f f components. Ultimately, the components' signatures identify the bits that correlate to the final fingerprint on the document. To find close duplicates, the process is repeated for every document and the fingerprint of the new webpage is compared to those filtered webpages.

3.3 Near Duplicate Detection

Following cascade filtering, we receive the fingerprints of every document that was filtered in as well as the newly submitted web document, which is used to determine whether or not it is nearly identical to an already-existing web page in the database. For the

comparison process, a threshold is established, typically based on user input. A piece-by-piece comparison is done between the fingerprints of the new online document and the other documents. The bit-by-bit difference measure, or bit-by-bit difference (b f) in number of bits, is considered and used to determine whether or not two web pages are near duplicates. Documents that have bit-by-bit differences that are less than or equal to the threshold (TB) are deemed near duplicates and are deleted. In a similar manner, the recently added webpage is compared to every other webpage fingerprint. If it is discovered that the newly added webpage is not nearly identical to any of the existing webpages, it is chosen and added to the database.Taking into consideration the following fingerprints, one of which belongs to the new webpage and the other to another website that is stored in the database.

4. Enhancement of our Proposed approach

Finding the near copies of a web page involves a laborious process that takes time when dealing with databases that contain a big number of web documents. This technique compares the web page to all other web pages in the database. Rather than comparing with all the pages, a set number of defined online documents must be chosen based on some similarity metric and compared with each other in order to get beyond this challenge. A useful technique for organising a huge number of documents into manageable groupings based on predetermined criteria is clustering. Thus, K-mode clustering is chosen in this instance to group the online documents. We start by removing the keywords from every web page and grouping the results based on those keywords. The new webpage gets classified with the most similar cluster. Lastly, we completed the comparison process, which involved the near duplication technique and the filtering that had been previously mentioned. It is only completed using the papers that are part of the cluster, making calculation easier.

4.1 Clustering of Large Web Page Documents

The technique of grouping data based on a similarity metric is called clustering. In this case, a collection of clusters represents the major online articles in the database, allowing near duplicate detection to be performed just on the web pages included in the most pertinent clusters.

When comparing this technique to earlier methods, the time complexity is immediately reduced. In order to cluster a large number of web documents, we first extract the keywords, which are a collection of important terms that describe the content of the web pages in a document. The web page document obtained after the preprocessing processes is represented by the notation O { d,d...,d } s 1 2, n ^\, where n is the total number of web page documents. Next, each web page document is searched for keywords using the following formula: di = { kw1, kw2,.... kwm}, where m is the total number of keywords in the document (i.e., d). We next determine the frequency of each unique keyword for each keyword extracted from web page document i d using the formula d = { (k,f),(k, f),…….,(k,f) i w1 1 w2 2 wl l, where l is the total number of unique keywords and l f denotes the frequency of the l th keyword. According to the keywords' frequency, the unique keywords are arranged in decreasing order. Lastly, we choose the top f keywords, which are referred to as each document's representations (i R). These representatives are then employed to locate the cluster containing a sizable collection of online documents. The following are the fundamental steps in clustering: Establish a starting point for each cluster containing k -centroids, and then compute the similarity s亖R d 攵 i k i, l, ^\ 1,2,□, between each k -centroid and the representatives of each web page document. (3) Assign document ID of the web page to cluster l C, which has a high similarity measure. (4) Refresh the k representations (5) Iterate from Step 2 to Step 4 until the web page documents remain in their current clusters.

4.2 Detection of Near Duplicate Web Pages

A huge collection of web documents is subjected to k-Means, which produces "k" number of clusters based on the keywords found in each document. These clusters are then composed of web pages with the highest level of similarity. A newly crawled webpage is assigned to the cluster that has the highest similarity after having its similarity with all k cluster representatives assessed. Time savings and computational ease are two benefits of integrated clustering. Because a newly added webpage must compare its sentence-level features with those of only the webpages in its cluster, not with all of the webpages. Using the fingerprint and the sentence level extraction technique, cascade filtering is used when the new webpage is added to the appropriate cluster. The new webpage's fingerprint is compared to the fingerprints of other webpages in the cluster, and the bit-by-bit difference measure, or b f, is computed. When the bit-by-bit difference measure b f exceeds a predetermined threshold value TB, it is deemed to be near duplicate webpages and is ignored. If not, the repository receives the new webpage.

5. Results

The findings and a description of the suggested method for locating nearly identical webpages are presented in this section. We test our suggested methodology on a system equipped with an Intel Core 2 Duo CPU and three gigabytes of RAM. Initially, each web page document's entire sentence count is calculated and saved as a sentence level feature. Next, the SimHash method is used to obtain each document's fingerprint. In order to identify almost duplicate web pages, we start with a newly uploaded webpage called Wpnew and use a cascade filtering procedure to determine whether or not it is a near duplication. Using the cascade filtering process, the 360 GB hard drive is capacitated. Microsoft Access is used as the backend and the Net Beans 6.9.1 IDE as the frontend while implementing this strategy. Java is the programming language used to write the programme.

For the input dataset, the experimental outcomes of the suggested method are provided in the ensuing subsections. According to the conducted trials, our suggested strategy is effective and requires less time, as confirmed by the analysis of the experimental results.

a. Experimental Results

This section provides an example of the outcomes from each intermediate phase of the suggested methodology. We use three web page documents from the web crawling process to provide sample findings. The first step in the preprocessing procedure for the webpages is to remove stop words and stem them. After preprocessing, webpage documents are represented by a list of keywords. Here, the new webpage document and the features saved for other web pages are compared in terms of the sentence level feature. The fingerprints of each document chosen from the previous procedure are compared to the fingerprint on the new web page document if the comparison result is less than or equal to the. Once the fingerprints of the just added web content have been located, a comparison of the fingerprints is calculated and processed using a previously saved threshold value, TB. The practice of comparing bits by bits is carried out in order to identify duplicates and almost duplicates. The total number of bits that differ between the new web page document and the other web pages is calculated after determining the bit-by-bit difference. The predetermined threshold TB is compared with the bit-by-bit difference value. A webpage is deemed nearly duplicate if the bit difference value is less than or equal to the criterion. These webpage documents are then taken into consideration for additional processing, while the remaining webpage documents are dismissed, based on a bit-by-bit comparison of the fingerprint of the Wpnew with that of the Wp2 and Wp3.

b. Performance Evaluation

We used a dataset made up of web page documents that were found using web crawling to assess the effectiveness of the suggested approach. There are twenty-one duplication pages out of the total one hundred pages. Evaluation criteria including precision, recall, and F-measure are used to assess how well the suggested strategy performs. The three criteria that determine accuracy are as follows:

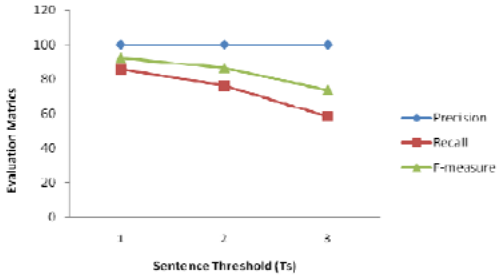
$$\text{Precision}(P) = \frac{\text{No. of true duplicates detected}}{\text{Total No. of duplicates detected}}$$

$$\text{Recall}(R) = \frac{\text{No. of true duplicates detected}}{\text{Total No. of duplicates in the dataset}}$$

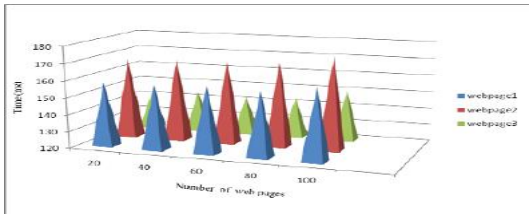
$$\text{F-measure}(F) = \frac{2 \times P \times R}{P + R}$$

1. Accuracy: The suggested method is applied to the input dataset in order to assess the accuracy that is determined for various sentence threshold (Ts) values. The calculated values are displayed as a graph in Figure 5.

We may deduce from the graph that the threshold chosen affects both the recall and the f-measure. Here, we can observe that the precision stays constant despite changes in the sentence threshold, Ts. Furthermore, we can infer that the recall and the f-measure decline as the threshold is raised. Therefore, a threshold value that allows for web duplicate detection as well as high recall and f-measure must be set.



2.Computation time: The computation time for detecting near duplicates varies depending on the type of webpage and the quantity of documents compared to the newly created webpage. By comparing the online documents stored in the database with the newly created webpage, near duplicates are identified. Thus, the time required varies depending on how many webpage documents are stored in the database. A larger database of documents implies a greater number of online documents with which to compare the newly created webpage. In the instance mentioned above, a fixed number of web papers in the database are combined with three fresh web pages to process them for near duplication. Every case's temporal response is plotted. An illustration of performance by our proposed approach is described in fig 6.



6.Conclusion

Nearly identical webpages are a major hazard to web crawlers and are now the main source of worry for search engines. Near duplicates influence both execution time and accuracy since they slow down the process, add to the expense of supplying results, and need a lot of space to keep the indexes. Additionally, it makes the query result less relevant to the users. Numerous methods based on similarity scores and signatures have been developed for the detection of near duplicates. In this work, we have put forth an effective approach that combines the fingerprint technique with sentence level data to identify close duplicates. These two methods are applied to the preprocessed

preprocessed web page documents and function as cascade filters. To achieve better results in less computing time, we have improved our approach by applying clustering when a big number of documents is taken into consideration. The outcomes of the experiment demonstrate how accurate and time-efficient the suggested method is.

7. References

[1]V. A. Narayana., P. Premchand., and A. Govardhan., "Fixing the Threshold for Effective Detection of Near Duplicate Web Documents in Web Crawling", Advanced Data Mining and Applications, Vol. 6440, pp : 169-180, 2010.

[2]Monika Henzinger, "Finding Near-Duplicate Web Pages: a Large-Scale Evaluation of Algorithms", In Proceedings of the 29th annual international ACM SIGIR conference on Research and Development in Information retrieval, pp : 284-291, 2007.

[3]Andrei Z. Broder, Steven C. Glassman, Mark S. Manasse, and Geoffrey Zweig, "Syntactic Clustering of the Web", In Proceedings of the Sixth International Conference on World Wide Web, pp : 1157-1166, 1997.

[4]Moses S. Charikar, "Similarity Estimation Techniques from Rounding Algorithms", In Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing, pp: 19-21, 2002.

[5]Gurmeet Singh Manku, Arvin Jain, and Anish Das Sarma, "Detecting Near-Duplicates for Web Crawling", In Proceedings of the 16th International Conference on World Wide Web, pp : 141- 150, 2007.

[6]A. Kołcz, and A. Chowdhury, "Lexicon randomization for Near-Duplicate Detection with I-Match", The Journal of Supercomputing, Vol. 45, No. 3, pp: 255-276, 2008.

[7]Ranjna Gupta, Neelam Duhan, A.K. Sharma and Neha Aggarwal, "Query Based Duplicate Data Detection on WWW" , International Journal on Computer Science and Engineering, Vol. 2, No. 4, pp: 1395-1400, 2010.

[8]Hung-Chi Chang, and Jenq-Haur Wang, "Organizing News Archives by Near-Duplicate Copy Detection in Digital Libraries", Asian Digital Libraries, Vol. 4822, pp: 410-419, 2007.

[9]V.A. Narayana, P. Premchand, and A. Govardhan, "Effective Detection of Near Duplicate Web Documents in Web Crawling", International Journal of Computational Intelligence Research, Volume 5, No. 1 , pp.83–96, 2009.

[10]Berendt, B, Hotho, A., Mladenić, D, Spiliopoulou, M, and Stumme, G. "A Roadmap for Web Mining: From Web to Semantic Web", Lecture Notes in Artificial Intelligence, Springer, Vol 3209, pp. 1-22, 2004.

[11]J Prasanna Kumar, and P Govindarajulu,"Duplicate and Near Duplicate Documents Detection: A Review", European Journal of Scientific Research, Vol 32, No. 4, pp.514-527, 2009.

[12]Spertus, E., "Parasite: Mining structural information onthe web", Computer Networks and ISDN Systems: The International Journal of Computer and Telecommunication Networking, vol. 29: pp. 1205 – 1215.,1997.

[13]Balabanovic, M., and Shoham, Y. "Learning Information Retrieval Agents: Experiments with Automated Web Browsing", In On-line Working Notes of the AAAI Spring Symposium Series on Information Gathering from Distributed, Heterogeneous Environments, 1995.

[14]Kobayashi, M, and Takeda, K, "Information retrieval on the Web", ACM Computing Surveys (ACM Press), Vol. 32, No. 2, pp. 144–173, 2000.

[15]J Prasanna Kumar, and Dr. P Govindarajulu, "Efficient Web Crawling By Detecting and Shunning Near Duplicate Documents", Georgian Electronic Scientific Journal, Vol 22, No. 5, pp.109-115, 2009.

[16]F. McCown, and M.L. Nelson,"Evaluation of Crawling Policies for a Web-Repository Crawler", In Proceedings of 17th ACM Conference on Hypertext and Hypermedia, pp: 157-168, 2006.

[17]Rajashree Shettar, Dr. Shobha G, "Web Crawler on Client Machine", In Proceedings of the International Multi Conference of Engineers and Computer Scientists, Vol 2, pp. 1121-1124 , 2008.

[18]Cho, J, Garca-Molina, H. and Page, L. "Efficient Crawling Through URL Ordering", Computer Networks and ISDN Systems, Vol. 30, No. 1-7, pp: 161-172.1998.

[19]Xiao, C, Wang, W. Lin, X. Xu Yu, J., "Efficient Similarity Joins for Near Duplicate Detection", Proceeding of the 17th international conference on World Wide Web, pp: 131-140. , 2008.

[20]Conrad, J. G., Guo, X. S, and Schriber, C. P, "Online Duplicate Document Detection: Signature Reliability in a Dynamic Retrieval Environment", Proceedings of the Twelfth International Conference on Information and knowledge Management, New Orleans, LA, USA, pp. 443 - 452, 2003.

[21]Fetterly, D, Manasse, M, and Najork, M, "On the Evolution of Clusters of Near-Duplicate Web Pages", Proceedings of the First Conference on Latin American Web Congress, pp. 37.2003.

[22]Shoaib. M, Arshad.S, "Design & Implementation of Web Information Gathering System," NITS e-Proceedings, Internet Technology and Applications, King Saud University, pp.142-144.

[23]Lovins, J.B, "Development of a Stemming Algorithm". Mechanical Translation and Computational Linguistics

[24]Dick Grune, and Ceriel Jacobs, "Parsing Techniques: APractical Guide ", pp. 13, 1998.

25.Bayardo, R. J., Ma, Y, and Srikant, R, "Scaling up all Pairs Similarity Search", In Proceedings of the 16th International Conference on World Wide Web, pp.131-140,2007.