A Comparative Analysis of Identifying Near Duplicate Web Documents

Ankit Kumar Singh

[1] *Dept of CSE, CMR College of Engg & Tech, Hyderabad, India.*

***Abstract*— The internet is a major source of information. Internet users can have access to this data by searching using the keywords as query. The world wide web consists of trillions of web pages, out of which the search engine detects the documents that are the most suitable for the search query by the user. This can often result in an unreasonably large number of results because the web search engine often contains duplicate and near-duplicate documents. In order to decreases the computation time and increase in the relevance of the respective search results, the duplicate documents need to be detected. They can be detected using various techniques that re especially developed to identify two identical or near identical documents in a large set of documents. The following experimentation has been done to compare the efficiency of two such techniques for detecting the near-duplicate web pages. The existing approaches are compared to analyze their performance, time comparison, confusion matrix parameters such as accuracy, sensitivity, and other such factors. The two techniques being demonstrated are "A Novel and Efficient Approach for Near Duplicate Page Detection in Web Crawling" and "A Near-Duplicate Detection Algorithm to Facilitate Document Clustering". The documents have been compared using NC2 combinations. The near-duplicate document detection ultimately results in repositories reduced in size and the search engine quality also improves to a great extent.**

***Keywords*: Search Engine, Web page, Duplicate web page, Near-duplicate web page, Near duplicate detection.**

1. INTRODUCTION

The world wide web as we know it today, has billions of web pages, thousands of new ones created every day. The globalization and demand for new information has inevitably led to a huge increase in the production of digital content. Most of which, are mirrored copies of each other. This can occur due to the need for more accessibility of the documents, and more copies of the same document can somehow increase the reach of the data within the web document. But the side effect of this is that it creates a data redundancy so bad, that it ultimately affects the efficiency and performance of the search engine itself. The duplicate and near duplicate documents created will produce an additional overhead to the repositories. This is the reason why the detection of such duplicate and near duplicate documents has been in demand and a domain of popular interest in the recent years. Numerous studies have been made on this topic already and a lot of algorithms have been developed to solve the near duplicate document detection problem. In order to find the documents of our interest, there is a lot that goes into searching for a web document based on a keyword or query in the search engine, and they include processes like web mining, web scraping, and so on.

To briefly describe about web mining, it is a method that is used to deliver data on the web by digging the documents on the web repository. After doing so, certain patterns can be discovered from the collected documents. Web mining can be mainly divided into three categories as follows-

The first category being Web Content Mining. It is basically collecting the valuable information from web pages in the form of text, images, and hyperlinks. Automated discovery of the new data patterns can be pretty challenging because of the heterogenous nature and inadequacy of the structure in web data. After scanning the web pages according to the context of the query, they are arranged by relevance in the search engine result page.

The second being Web Structure Mining. This category of web mining deals with the obtaining of data patterns from the structure of the hyperlinks used in the web documents. All the web documents are linked to one another, and thus tracing them to find connection between one another can be a good way of scraping data from the web. This process scans and derives text, images and many other groups of web pages based on the query that has been requested by the users. This process can be used to determine whether the web pages are linked by information in the document, or by the direct link connections. And thus, it provides a summary of the structure of the website and the web pages that it is related to.

The third category of web mining is the Web Usage Mining It is used for keeping tab on the patterns in which a user logs into a web page. The web server is the one that solely registers all this information in the records. This information can be very useful for obtaining customers or potential customers in the field of e-commerce.

The three categories of web mining as stated above are thus used for different reasons. The next method employed is the web scraping method. It has many other names such as spidering, or web crawling. It is a very powerful tool for extracting data from a bunch of web pages. Web scraping is followed by web indexing, where the web documents are indexed in the search engine. This process is used for

converting unstructured data which is in the HTML format into structured data which will be in the form of a database or spreadsheets.

In order to properly define what are duplicate and near duplicate documents, it is important to understand how vast the internet repository is. The duplicate web documents are the ones where the content of these documents appear in more than one location in the web, the only difference being the link and address of the website (URL) being different. Other than that, the content part will be the same. If there are multiple versions of the original document, it will be difficult for the search engine to distinguish which document is the original one and which are the copies. This is how the duplicate document can create data mining issues.

This duplicate web document is bothering the users because of the rise in service cost, as well as more storage space. Information site links are ordered by the user seeking to query the answers from the Google search engine based on their most hit website link above. When the user seeks to query the information about "Enum vs set" resulting in different website links but the information in the two websites is replicated accurately and duplicated. So, when crawlers try to avoid indexing such types of duplicate web documents, the unique information available in the worldwide web repository will reduce storage space in the worldwide web repository and increase processing speed when users try to find or query answers.

2. RELATED WORK

The following work has taken inspiration from various other previously existing works on the topic of duplicate and near-duplicate document detection.

The reference paper [1] proposes a system where the web documents are to be registered and the duplicate copies are detected. An algorithm has also been introduced where it detects the near-duplicate documents, and the results have been evaluated using various evaluation metrics such as accuracy, sensitivity, efficiency, and even security. The prototype that was implemented of this system is called COPS, and it was established through the evaluation metrics that this prototype can definitely detect the near-duplicate documents within a database successfully.

In the research paper [2], the syntactic similarity of the collected files is determined efficiently, and this had been applied to every document on the world wide web (www). This mechanism was taken as a basis to perform clustering of all of the documents that are syntactically similar to each other. This mechanism has the following applications- it can be used in the lost and found services, identifying the violations on the intellectual property, and updating the documents along with filtering the search results by removing all the unnecessary and near-duplicate documents from the web.

In research paper [3], in a large dataset of documents, the extent and the types of duplication has been established first. This research has been mainly divided into three distinct parts. Firstly, 50 million documents have been studied for patterns in the different types of duplication and how they are distributed in the dataset. The next part is to establish signatures to all the original documents so that they can be easily differentiated from the duplicate ones and the sensitivity of collecting the following updates was examined. The final part is the mechanism being used to characterize and compare the documents with one another in order to determine whether to permit the document into the repository or identify the document as a near-duplicate document. Thus, this method has delivered promising results after which an elaborate evaluation using a collection for testing based on production was created.

In the research paper [4], unlike the other mechanisms, this mechanism focuses of exploring and measuring the intermediate similarities of a document, and to detect where a certain piece of text has originated from. This will aid in knowing whether the web document is from a legitimate source. The detection of similarity is at sentence level, and there are a number of approaches that are used to combine the sentence-level evidence with the document-level evidence. And thus, the algorithms used have been incorporated into a prototype information flow analysis tool called RECAP.

In the research paper [5], this method mainly focuses on automating the process of identifying the near-duplicate web documents. Using clustering technique on simple text and retrieval algorithms to identify the near-duplicate documents, especially for identifying the public comments. Here, there is documents are retrieved based on the feature and clustering is based on the similarity of these documents to discover the near-duplicates from the database. This method was tested and evaluated on a large dataset of public comments collection to use the EPA rule.

In the research paper [6], two approaches to find the near-duplicate documents have been compared. These two documents are namely the shingling algorithm approach [2] and the projection-based [7] approach. This comparison has been done on an extensively large set of data, which has a size of 1.6B distinct web pages. Although, the results stated that neither of the two approaches was suitable enough to find the near-duplicate documents on the same site, but were perfectly capable to do so on different sites. This led to the creation of a hybrid model which combines the aspects of both the approaches and results in a precision rate of 79% with the other algorithms in consideration.

In the research paper [8], a previous approach to find the near-duplicate documents has been refined to produce a new approach called the DURIAN, short for Duplicate Removal In large collection. This DURIAN approach uses identifies the form letters and the edited copies of them using document attributes such as metadata, content structure, and so on in the public comment section on different sites. The results of this approach have claimed that the similarity measures statistically and constrained clustering on an instance level can be very effective in finding the near-duplicate web pages.

In an attempt to make a system to detect the nearduplicate web documents for a multi-billion-page repository, the

research paper [9] gives two contributions to this cause. Firstly, it approves of the fingerprinting technique of the paper [7] and says that it is appropriate enough for the goal. And secondly, a new algorithmic technique has been proposed to identify the existing f-bit fingerprints that are different from the given fingerprints, mostly in the k bit positions where the k is small. This method works for both online and batch queries, or single and multiple fingerprints.

In the research paper [10], it focuses on mainly solving the problem of DUST, which is short for Different URLs with Similar Text. An algorithm was proposed, called the Dust Buster, which uncovers or discovers the rules that can likely change a given URL into the ones that are having very similar content, thus reducing the amount of duplicate web documents on the world wide web. The Dust Buster effectively removes all the near-duplicate documents from the previous crawling logs or even the web server logs without the need to examine the whole content of the page.

In research paper [11], a new technique was proposed where the similarity of the algorithms and filtering of the documents based on the position is done. This technique takes advantage of the fact that the similarity scores of the document have an upper bound or maximum threshold, and the ordering of tokens leads to the upper bound estimates of a web document. The results display an exuberant performance of the said algorithms where the algorithms which are based on the prefix filtering mechanism on different datasets was surpassed successfully.

The following research has been done to find the near-duplicate web documents where the main motivation behind the creation of this technique is the various existing works that have already attempted to eliminate this problem. Thus, in order to present a more efficient technique, efficient algorithms have been proposed in [12, 13, 14, 15, 16, 17, 18, 19] where they compute the clusters of the duplicate documents. The first of these algorithms was proposed by Manber Heintze, where the detection of near-duplicates was done with reduced number of comparisons [20] and [21].

Both of these approaches attempt to work on the adjustment of characters and their sequences. Brin [34] in his research has used sequences of words in order to detect the violations of copyrights, if any. This can help in avoiding plagiarism and other such malpractices. Garcia Molina along with Shiva Kumar [22] have worked on continuing the research on expanding the scale of the databases to multi-gigabytes. Broder et all [23] used sequences of words find near-duplicate web documents. Charikar's Simhash technique [24] is used for reducing the dimensionality to identify the pages that are similar to each other. Henzinger [25] compared two approaches Broder et al.'s [14] algorithm of shingling and Charikar's [24] approach of random projection-based method, which is to be tested on a large set of data, and to be specific, on a set of 1.6B web pages that are ideally distinct.

In methods that use the lexical approach, just like the I-Match [27] approach, a lexicon is generated using a very large text corpus. I-Match can be varying in stability sometimes and changes the texts [17]. Jun Fan et al. [15] combined three approaches, namely the shingling, I-Match and the simhash approaches.

3. EXPERIMENTAL ENVIRONMENT AND SETUP

3.1 Dataset collection

Over 800 text documents each consisting of an average of 6K words are gathered from various sources. All the 1.6K text documents are unique in nature. They are stored as .html files in the system. These documents are used for testing and analysis. Both the NPG and DC algorithms use these documents in sets of 100, 200, and 300 unique documents to produce the percentage of various factors such as accuracy, sensitivity, specificity, etc., which determine the performance of the algorithms. Later, combination of unique and near duplicate documents is taken into consideration to measure the effectiveness of the algorithms.

4. NEAR DUPLICATE WEB PAGE DETECTION

In order to solve the near duplicate document problem, a novel approach has been introduced, which can increase the efficiency of the search engine. During the process of web crawling, the webpages that have been crawled will be stored in the repository and they are processed further to find the structural analysis, page evaluation, notification update, search engine formation, and even visualization. Detection of these duplicate and near-duplicate documents is very necessary because it facilitates the search engine to provide results that are free of data redundancy and plagiarism. This results in the search results being distinct and actually relevant from the query that has been searched. There are a number of problems that are usually faced in order to solve this nearduplicate document identification problem, some of which include the size of the database to be considered. Since the world wide web is very vast and there are millions of web pages in it, indexing every one of them on the search engine is a challenge in itself. The second problem that is faced is that the web crawler has to crawl new web pages every day that increase billions in number, and it can be very difficult to mark the web page as near-duplicate. All of these processes should be done at a faster pace for better results. The system should also focus on using minimum number of machines as stated in [9].

Before every approach used on the data of the documents to be classified as duplicate or near-duplicate there has to be a certain step that is to be performed, and that is none other than the pre-processing of the textual data. This pre-processing is performed on the keywords that are extracted from the web documents. The web documents are firstly crawled and then parsing is performed to successfully extract all the different keywords that are frequently occurring in the document. Parsing is done to remove all the HTML tags, and java script as well. Then the keywords undergo stemming and stop word elimination. These keywords, now that they have undergone the pre-processing, will be extracted one after the other and

stored in tables with their frequencies in the descending order. N number of keywords are collected from each document for comparison. The similarity score of one of the web documents has to be calculated against an already existing web document in the repository. Ultimately, the document that has a greater similarity score with respect to a threshold value will be treated as a near-duplicate document, and thus will not be added to the document repository.

4.1 A NEAR-DUPLICATE DETECTION ALGORITHM TO FACILITATE DOCUMENT CLUSTERING

5. PRE-PROCESSING

Before testing the datasets on either of the algorithms, the data needs to undergo pre-processing first. This involves different steps that help in processing the information in the web document. Especially by finding the similarity in the number of keywords in each of the document.
The different steps involved in pre-processing of data are as follows-

5.1 Web Crawling

Web crawling can be defined as - the understanding and analysis of the informatics and structure of the web documents which is possible with the data collection technique known as web crawling. There are a number of websites that use web crawling for collecting web pages from the internet and then index them according to relevance. Some of the basic features of the web crawler are the automatic traversal of web sites, document downloading from the web, and following the traces of the links to other pages. The main intent of web crawling is to collect as many web pages relevant to the query as possible, with their interconnected web page links in the fastest and most proficient method possible. Web crawling is extremely useful, but it can also be a very tedious task because of the rapidly changing number of pages, which are also huge in number. The pages being added and removed everyday makes it difficult to perform web crawling.

A certain type of URLs is called the seed URLs, and these are the set of URLs that are the first URLs that the crawler can work with. These all URLs are in an orderly manner in the form of queues which are obtained by the crawler. After they are queued, the web crawler is supposed to download the page, after which the URLs are extracted from the page. This cycle continues until the web crawler decides to stop. This loop consists of the following steps- the URL has to be obtained from the queue, then the file corresponding to it should be downloaded with the help of HTTP, and then the page has to be traversed for some more new URLs.

5.2 Web Document Parsing

The web document parsing means basically analyzing and converting a program into an internal format. This internal format is to be run in a real runtime environment. The browser that is used will parse the HTML text into a DOM tree. The parsing process can range from being as simple as the extraction of a few hyperlinks or URL extraction or it can also be as complex as the analysis of HTML tags by cleaning the HTML content itself. The

5.3 Stop Words Removal

Stop words are the words that are frequently used, but have no particular meaning of their own. Examples of stop words are ‘to’, ‘from’, ‘and’, ‘it’, ‘can’, ‘an’, ‘a’, ‘by’, ‘for’, ‘of’, ‘the’, and so on. These words can be eliminated from the text to reduce the number of words to be considered for the comparison. The following procedure is named as stop listing because it helps with the reduction of the size of the files for the indexing. It also helps in enhancing the efficiency of the whole process.

5.4 Stemming Algorithm

The words that are left after the removal of the stop words are the keywords, but all of them are not in their root word form. They either have a prefix or suffix attached to them. Even if two words can be seemingly different, if their root word is the same, then the meaning of the sentence will be the same. Even if the terms hold identical meaning, they can seem to be morphologically dissimilar to the information retrieval system. Thus, stemming allows the multiple words having the same root word to one root word itself. This can be easily achieved by removing the suffixes and prefixes of the word if any. **For example:**

- rehearsals, rehearsal -> rehearse
- marketing, marketed, markets -> market
- walks, walking, walker ->walk

6. ARCHITECTURE AND FORMULAE USED IN THE TWO APPROACHES

6.1 Similarity score algorithm

In this approach, a certain number of keywords are considered, between 15 to 20 keywords. The prime keywords of the web

T_1	K_1	K_2	K_4	K_5	……..	K_n
	C_1	C_3	C_2	C_4	……..	C_n

T_2	K_1	K_3	K_2	K_4	……..	K_n
	C_1	C_3	C_2	C_4	……..	C_n

page in the table are compared to the prime keywords of the new web page. If the keywords do not match, then the web page will be added to the repository. But if the prime keywords match with the page that is already in the repository, then the web page is considered as near-duplicate and it is not included

HTML parsing consists of tokenization and construction of trees. It is not avoidable for the parser to traverse the entire web to find a lot of errors. And thus, the parser sometimes does not consider small words like 'a', 'an', 'the' and so on.

The keywords are stored in the tables as shown above. If the keywords in both the tables match, then the similarity score of the keywords is calculated using the formula below.

$$a = \Delta[K_i]_{T_1}(1) \quad b = \Delta[K_i]_{T_2} (2)$$

$$S_{Dc} = \log (\text{count}(a) / \text{count}(b)) * \text{Abs} (1+(a-b)) \quad (3)$$

In the above formula, the letters 'a' and 'b' are the index of the respective keyword in the table. The similarity score can be calculated if $T_1 / T_2 \neq \varphi$.
In order to calculate the SDT1, all the keywords that are present only in the table T1, and not present in the table T2 are considered.

$$S_{DT1} = \log (\text{count}(a)) * (1+|T_2|) \quad (4)$$

Similarly, the SDT2 score is calculated by taking the keywords that are only present in the table T2, but not present in the table T1.

$$S_{DT2} = \log (\text{count}(b)) * (1+|T_1|) \quad (5)$$

Finally, the three scores SDc, SDT1 and SDT2 are added to find the sum, and this sum is divided by 'N', which is the average of the sum of absolutes of the number of terms present in tables T1 and T2. This will result in the similarity score of the two documents considered.

$$SS_M = \frac{\sum_{i=1}^{|N_C|} S_{DC} + \sum_{i=1}^{|N_{T_1}|} S_{DT1} + \sum_{i=1}^{|N_{T_2}|} S_{DT2}}{N} \quad (6)$$

Where $n = |T_1 \cup T_2|$ and $N = (|T_1| + |T_2|) / 2$

The resulting similarity score is to be compared with the threshold value. If the score is more than the threshold value, then it means that the web document is not a nearduplicate document, and thus can be added to the into the repository. The similarity score of the common keywords is calculated as follows- Consider two tables T1 and T2, where each table consists of the top 'n' keywords in a web document, and their count.

repository. But if the score is less than the threshold value, it is a near-duplicate document, and must be eliminated.

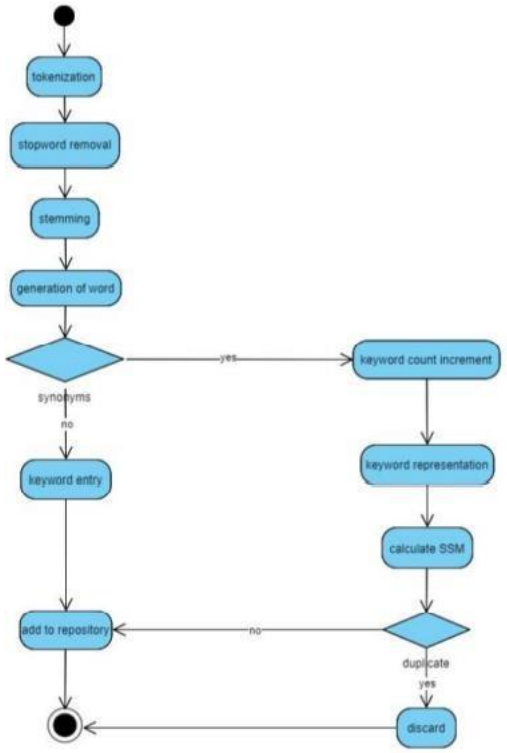
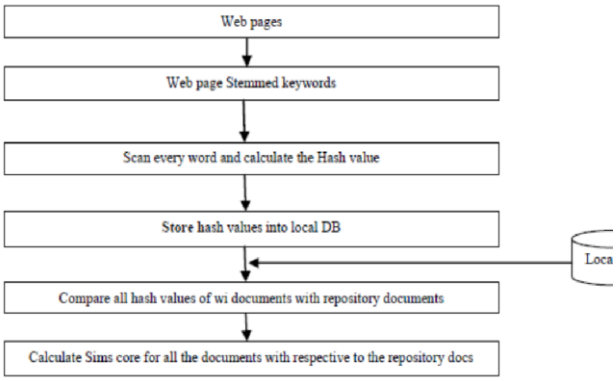


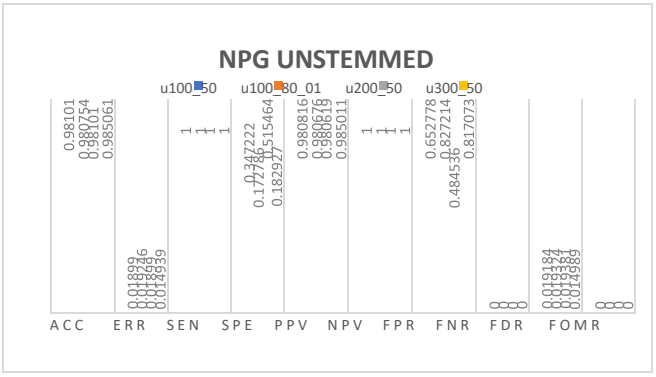
Fig 6.1 Activity diagram

6.2 Duplicate Detection (DD) Algorithm

This is the second approach that is considered for this experiment to detect the near-duplicate documents. There are certain steps involved in this approach, which are as follows-

Step 1: Perform pre=processing on the text of the document, including tokenization, stop word elimination, stemming of the keywords.
Step 2: All the obtained keywords must be scanned to check for any new keywords. The database is initially empty, but as the keywords are added, it is necessary to check for any new keywords, and generate a hash value for the respective keywords.
Step 3: The above steps have to be repeated until all the keywords are covered.
Step 4: All the hash values that are generated must be stored in the local database.
Step 5: All the above steps have to be repeated for all the sample documents that are taken, say if there are 'N' samples, then all the steps should be applied on the 'N' samples. **Step 6:** The similarity score should be calculated on the samples in percentage. If the score is more than 80%, then it is considered as a near-duplicate document, and is excluded from the repository.

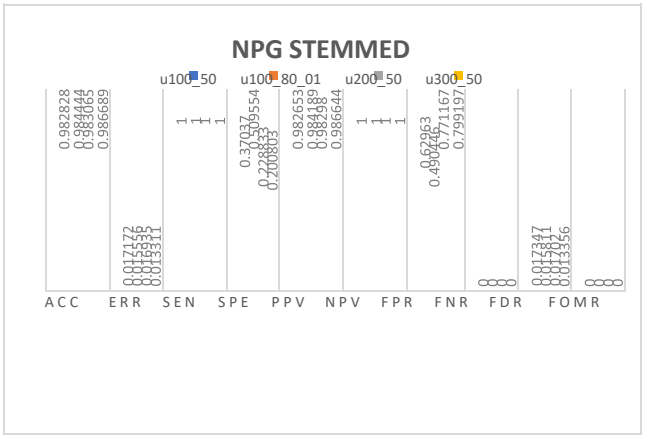




Graph 7.2 NPG unstemmed Fig 6.2 Data flow diagram

7. Results

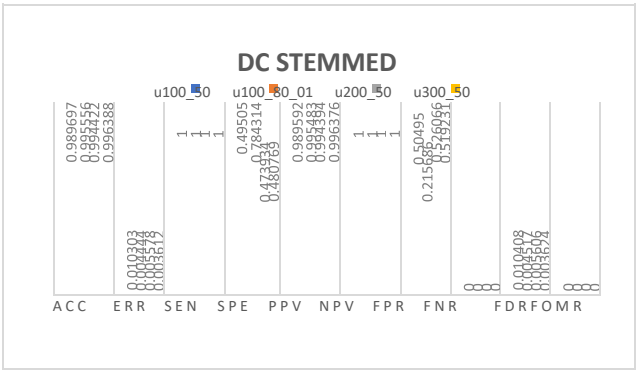
The following bar graph illustrates the performance of the NPG algorithm for different sets of html text documents in terms of factors like accuracy, sensitivity, specificity, etc. The keywords used are all stemmed using the stemming algorithm.



Graph 7.1 NPG stemmed

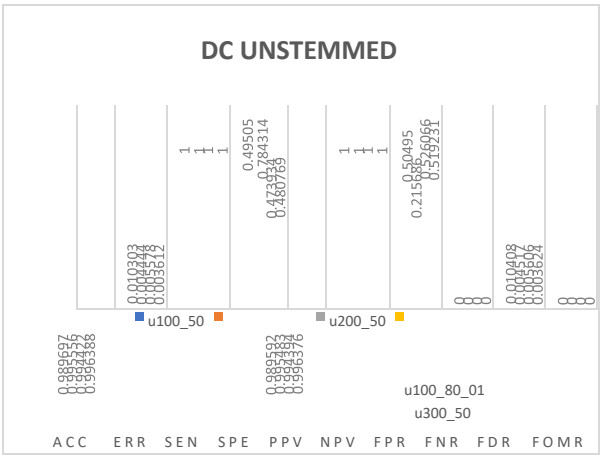
The first set has 50 unique and 50 near duplicate documents, a total of 100 documents. Similarly, the second set also has 100 documents, but with 80 unique and 20 near duplicate documents. Whereas, the third set has 200 documents, with 50% unique and 50% near duplicate documents. The fourth set has 300 documents with 50% unique and 50% near duplicate documents.

The NPG algorithm is also tested on the dataset with unstemmed keywords, and the graph illustrating the results is as follows.



Graph 7.3 DC stemmed

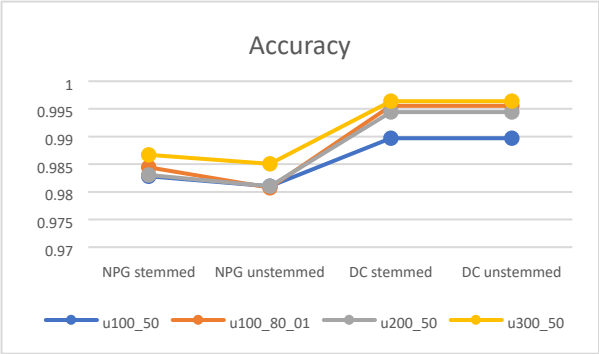
The graph illustrating the result of testing the DC algorithm on datasets with unstemmed keywords is as follows.



Graph 7.4 DC unstemmed

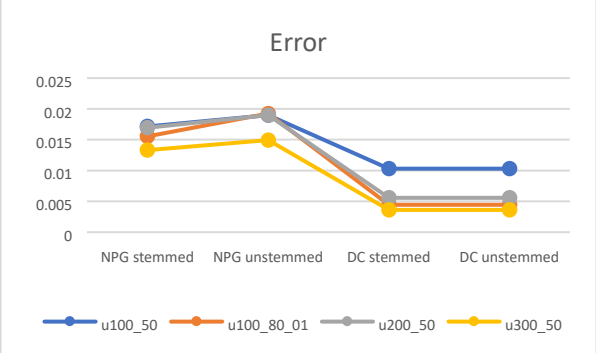
As mentioned above, there are various factors that determine the performance of the two methods. These parameters include Accuracy, Error, Sensitivity and Specificity, which constitute of the most important parameters. The rest of the parameters including Positive predictive value (PPV), Negative predictive value (NPV), False positive rate (FPR), False negative rate (FNR), False discovery rate (FDR), and False omission rate (FOR).

The following line graph illustrates the comparison of accuracy of the NPG stemmed, NPG unstemmed, DC stemmed and DC unstemmed test cases as follows.



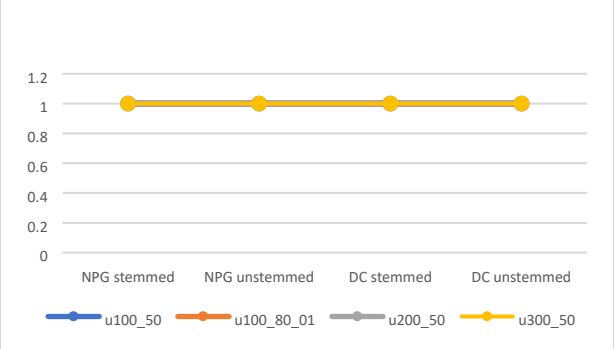
Graph 7.5 Accuracy

The following line graph illustrates the comparison of error of the NPG stemmed, NPG unstemmed, DC stemmed and DC unstemmed test cases as follows.



Graph 7.6 Error

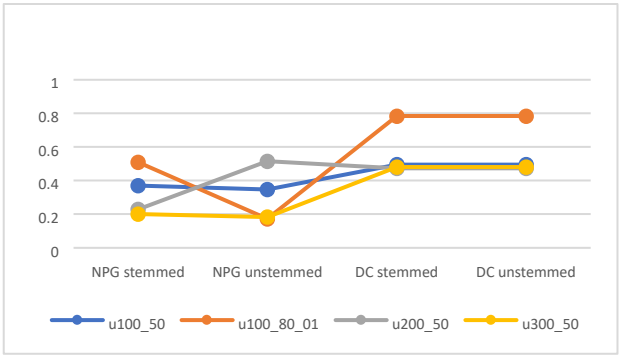
The following line graph illustrates the comparison of sensitivity of the NPG stemmed, NPG unstemmed, DC stemmed and DC unstemmed test cases as follows.



Sensitivity

Graph 7.7 Sensitivity

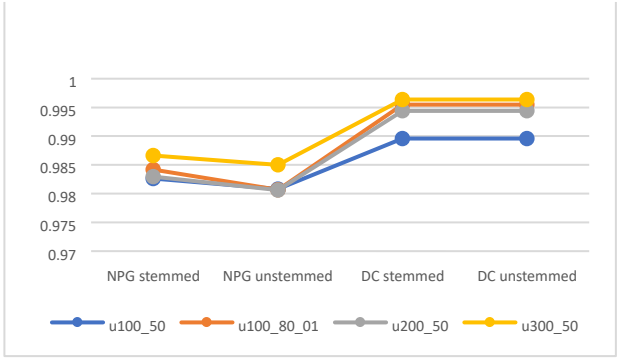
The following line graph illustrates the comparison of specificity of the NPG stemmed, NPG unstemmed, DC stemmed and DC unstemmed test cases as follows.



Specificity

Graph 7.8 Specificity

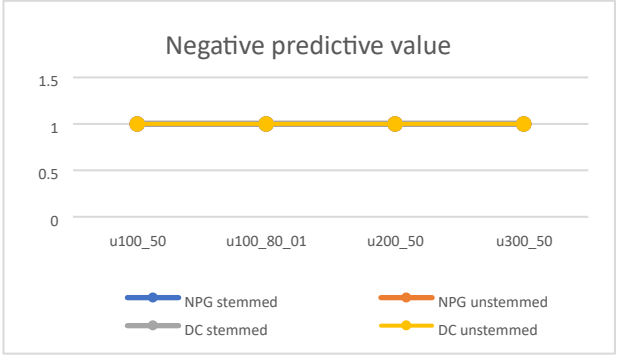
The following line graph illustrates the comparison of Positive predictive value of the NPG stemmed, NPG unstemmed, DC stemmed and DC unstemmed test cases as follows.



Positive predictive value

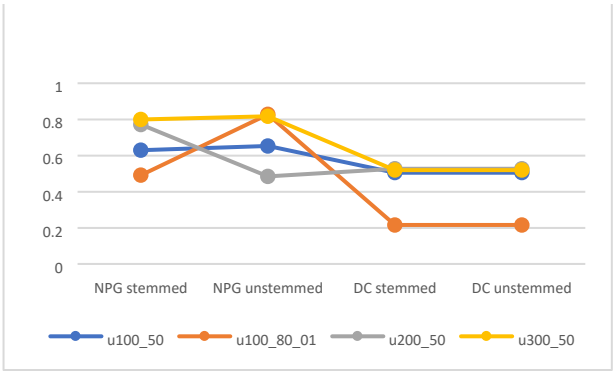
Graph 7.9 Positive predictive value

The following line graph illustrates the comparison of Negative predictive value of the NPG stemmed, NPG unstemmed, DC stemmed and DC unstemmed test cases as follows.



Graph 7.10 Negative predictive value

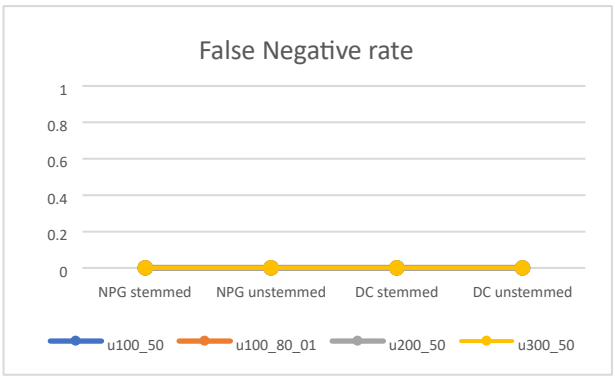
The following line graph illustrates the comparison of False positive rate of the NPG stemmed, NPG unstemmed, DC stemmed and DC unstemmed test cases as follows.



False positive rate

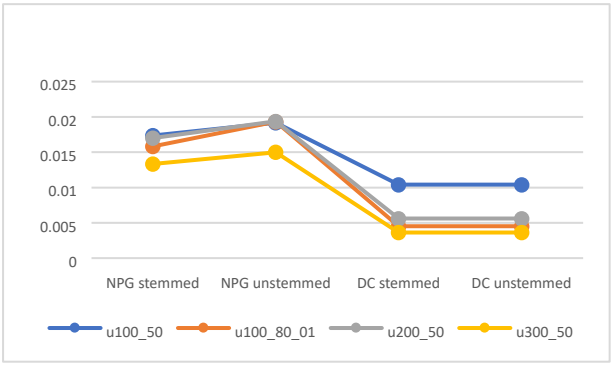
Graph 7.11 False positive rate

The following line graph illustrates the comparison of False Negative rate of the NPG stemmed, NPG unstemmed, DC stemmed and DC unstemmed test cases as follows.



Graph 7.12 False Negative rate

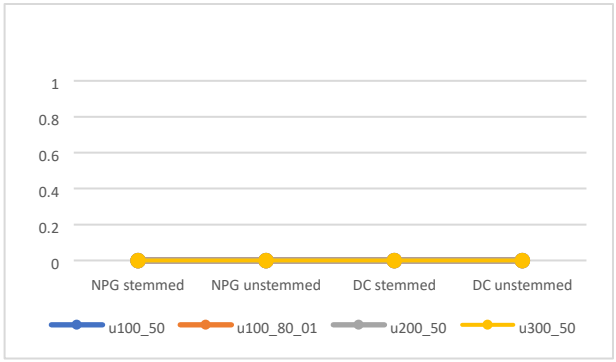
The following line graph illustrates the comparison of False discovery rate the NPG stemmed, NPG unstemmed, DC stemmed and DC unstemmed test cases as follows.



False discovery rate

Graph 7.13 False discovery rate

The following line graph illustrates the comparison of False omission rate the NPG stemmed, NPG unstemmed, DC stemmed and DC unstemmed test cases as follows.

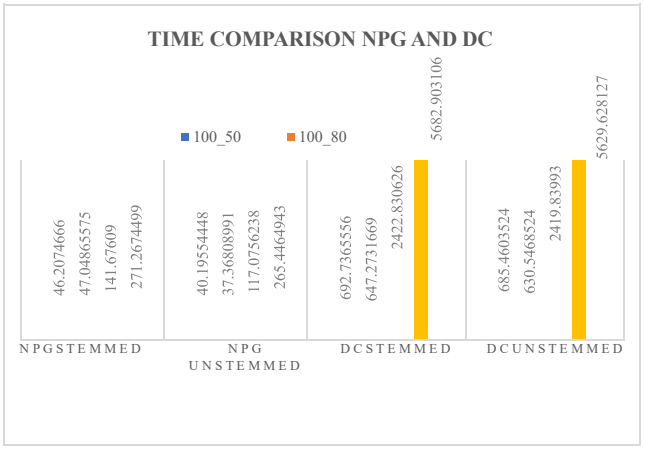


False omission rate

Graph 7.14 False omission rate

Time comparison between NPG and Document Clustering methods

Here, the time taken (In sec.) by both the NPG normal method and he Document Clustering method are compared with one another and the graph depicting the time taken (In sec.) by each of these methods is as follows.



Graph 7.15 Time comparison between NPG and Document Clustering methods

8. Conclusion

It can be concluded that the NPG normal method is comparatively better than the Document clustering in terms of the time taken to perform the near-duplicate documents within various sets of documents. Furthermore, it has been established that the NPG normal method can scan up to 1600 documents set at a time without taking excessive time to compile, whereas the Document Clustering method has been observed to take greater amounts of time as the number of documents are increased. This could be due to the usage of brute force approach to find all the possible ways.

9. Future enhancements

In consideration to the future enhancements, the two methods that have been compared in the performance analysis, namely the NPG normal method and the Document Clustering method can be combined to create a hybrid model of these two methods. The best aspects of both these methods can be taken and a singular model can be produced where the performance of

this method in areas where one or the other method couldn't do well will be significantly improved, resulting in a plausibly better solution to the near duplicate document detection problem.

10. References

[I] Chuan Xiao, Wei Wang, Xuemin Lin, "Efficient Similarity Joins for Near-Duplicate Detection", Proceeding of the 17th international conference on World Wide Web, pp. 131 – 140. April 2008.

[1] Brin, J. Davis, and H. Garcia-Molina. "Copy detection mechanisms for digital documents". In Proceedings of the Special Interest Group on Management of Data (SIGMOD 1995), PAGES 398-409. ACM Press, May 1995.

[2] A. Z. Broder, S. C. Glassman, M. S. Manasse, and G. Zweig. "Syntactic clustering of the web". In Proceedings of WWW6 '97, pages 391–404. Elsevier Science, April 1997.

[3] J. Conrad and C. P. Schriber. "Online duplicate document detection: signature reliability in a dynamic retrieval environment." Proceedings of the twelfth international conference on Information and knowledge management, Pages: 443 - 452 New Orleans, LA, USA, 2003.

[4] D. Metzler, Y. Bernstein and W. Bruce Croft. "Similarity Measures for Tracking Information Flow", Proceedings of the fourteenth international conference on Information and knowledge management, CIKM'05, October 31st November 5, 2005, Bremen, Germany

[5] H. Yang and J. Callan, "Near-duplicate detection for eRulemaking," Proceedings of the 2005 national conference on Digital government research, pp: 78 - 86, 2005.

[6] M. Henzinger, "Finding near-duplicate web pages: a large-scale evaluation of algorithms," Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval, pp. 284-291, 2006.

[7] M. Charikar. "Similarity estimation techniques from rounding algorithms". In Proc. 34th Annual Symposium on Theory of Computing (STOC 2002), pp: 380-388, 2002.

[8] Hui Yang, Jamie Callan, Stuart Shulman, "Next steps in near-duplicate detection for eRulemaking," Proceedings of the 2006 international conference on Digital government research, Vol. 151, pp: 239 - 248, 2006.

[9] Gurmeet Singh Manku, Arvind Jain, Anish Das Sarma, "Detecting nearduplicates for web crawling," Proceedings of the 16th international conference on World Wide Web, pp: 141 - 150, 2007

[10] Ziv Bar-Yossef, Idit Keidar, Uri Schonfeld, "Do not crawl in the dust: different urls with similar text," Proceedings of the 16th international conference on World Wide Web, pp:111- 120, 2007.

[11] Chuan Xiao, Wei Wang, Xuemin Lin, Jeffrey Xu Yu, "Efficient Similarity Joins for Near Duplicate Detection", Proceeding of the 17th international conference on World Wide Web, pp:131--140, 2008.

[12] Gurmeet Singh Manku, Arvind 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, Banff, Alberta, Canada, 2007.

[13] A. Broder, on the resemblance and containment of documents, Proc. Compression and Complexity of Sequences (SEQS: Sequences97). pp. 21–29.

[14] A. Broder, S. Glassman, M. Manasse, G. Zweig, Syntactic clustering of the web, 6th International World Wide Web Conference, Apr. 1997, pp.393–404.

[15] Jun Fan, Tiejun Huang "A fusion of algorithms in near duplicate document detection".

[16] Henzinger, M., (2006) "Finding near-duplicate web pages: a large-scale evaluation of algorithms," Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval, pp. 284291.

[17] A. Kolcz, A. Chowdhury, J. Alspector, Improved robustness of signature-based near-replica detection via lexicon randomization, Proc. KDD'04(eds. W. Kim, R. Kohavi, J. Gehrke, W. DuMouchel) Seattle, 2004, pp.605–610.) 216, 217

[18] Conrad, J. G., Guo, X. S., and Schreiber, C. P., (2003) "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.

[19] Bayardo, R. J., Ma, Y., and Srikant, R., (2007) "Scaling up all pairs similarity search", In Proceedings of the 16th International Conference on WorldWideWeb, pp.131-140.

[20] U. Manber. Finding similar files in a large file system, Proc. of the USENIX Winter 1994 Technical Conference, 1994, pp. 1–10.) 216

[21] N. Heintze, Scalable document fingerprinting, Proc. of the 2nd USENIX Workshop on Electronic Commerce, 1996, pp. 191-200

[22] Cho, J., Shiva Kumar, N., and Garcia-Molina, H., (2000) "Finding replicated web collections", ACM SIGMOD Record, Vol. 29, no. 2, pp. 355-366.

[23] Dennis Fetterly, Mark Manasse and Marc Najork, Detecting phrase-level duplication on the World Wide Web, In Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval, pp.170 - 177, Salvador, Brazil, 2005.

[24] Charikar's M., 2002. "Similarity estimation techniques from rounding algorithms", In Proc. 34th Annual Symposium on Theory of Computing (STOC 2002), pp. 380-388.

[25] A. Broder, Identifying and filtering near-duplicate documents, Proc. Annual Symposium on Combinatorial Pattern Matching, Lecture Notes in Computer Science (eds. R. Giancarlo and D. Sankoff) 1848, 2000, pp. (1–10.)

[26] W. Pugh, M. Henzinger, Detecting duplicate and nearduplicate files, United States Patent 6658423 (December 2, 2003).) 216, 217

[27] A. Chowdhury, O. Frieder, D.A. Grossman, M.C. McCabe, Collection statistics for fast duplicate document detection, ACM Transactions on Information Systems, 20, 2 (2002) 171–191.)

[28] A. Broder, M. Charikar, A.M. Frieze, M. Mitzenmacher, Min-wise independent permutations, Proc. STOC, 1998, pp. 327–336.)

[29] Muhammad Shoaib, Shazia Arshad, "Design & Implementation of web information gathering system," NITS e-Proceedings, Internet Technology and Applications, King Saud University.

[30] Pant, G., Srinivasan, P., Menczer, F., "Crawling the Web". Web Dynamics: Adapting to Change in Content, Size, Topology and Use, edited by M. Levene and A. Poulovassilis, Springer- verlog, pp: 153- 178, November 2004.

[31] Castillo, C., "Effective web crawling", SIGIR Forum, ACM Press, Volume 39, Number 1, N, pp.55-56, 2005.

[32] Lovins, J.B. 1968: "Development of a stemming algorithm". Mechanical Translation and Computational Linguistics, 11, 22-31(1968).

[33] M. Bacchin, N. Ferro, Melucci M.,"Experiments to evaluate a statistical stemming algorithm", Proceedings of the intl' Cross-Language Evaluation Forum Workshop, 2002.

[34] S. Brin, J. Davis, H. Garcia-Molina, Copy detection mechanisms for digital documents, 1995 ACM SIGMOD International Conference on Management of Data 1995, pp. 398–409.