# A Strategy for Near-Deduplication Web Documents Considering Both Domain &Size of the Document

**MD Zaheer, V. A. Narayana, Gaddameedhi Sreevani**

***Abstract --- The advice on the web is adopting to huge volumes, so a arduous affair to atom near-duplicate abstracts efficiently. The alike and near-duplicate abstracts are breeding a boundless botheration for seek engines, appropriately decelerate or access the amount of confined answers. Elimination of near-duplicates save arrangement bandwidth and reduces the accumulator amount and advances the superior of seek indexes. It aswell decreases the amount on the limited host that is confined such web documents. Server applications are aswell benefited by identification of abreast duplicates. In this avant-garde approach, the crawled web certificate is taken and keywords are acquired and are compared with the keywords accessible in the athenaeum of the accurate domain, again a accommodation of certificate acceptance to a accurate area is absitively adjoin the amount of keywords akin in that accurate domain. After selecting the domain, the admeasurement of the ascribe certificate is advised and the seek amplitude is bargain and calculations of affinity array are aswell diminished. Thereafter the affinity account is affected with abstracts which are acceptance to that accurate area only. This access reduces seek amplitude thereby abbreviation the seek time.***

***Keywords: Search Engine, Storage Management, Time Management, Web Document***

## 1. INTRODUCTION

The advice on the web is exponentially advanced in massive volumes and appeal to use this abundant advice calmly and effectively. The web consists of added no. of assorted copies of aforementioned agreeable [1]. Some advice repositories are mirrored artlessly to accommodate back-up and admission reliability. The seek engine faces a huge botheration due to the all-inclusive bulk of advice and it leads to extraneous answers. The alike and near-duplicate abstracts accept produced an added aerial for the seek engines alarmingly affecting their performance. The apprehension of abreast alike abstracts has afresh become a claiming and a area of abundant interest. A lot of studies accept brought calm on the Apprehension of Near-Duplicate Documents. Several methods and algorithms for Near-Duplicate Apprehension accustomed by advisers are available. Thus partially or absolutely alike web abstracts frequently arise on the web. Some abundant accomplishments advice has been addressed apropos the associated areas which cover Web Mining, Web Scraping, Alike Abstracts and more.

### *1.1 Web Mining*

The adjustment of carrying Data mining on the web is alleged Web mining. Digging the web abstracts and advertent the patterns from it. Web mining is mainly disconnected into 3 audible groups as follows,

Web Content Mining

Web content mining can be acclimated for the acquisition of admired data, information, and acumen from a web page. Web anatomy mining helps to access admired abstracts arrangement from the anatomy of hyperlinks. Due to adverse and blemish of anatomy in web data, automatic analysis of new abstracts arrangement can be arduous to some extent. Web agreeable mining performs scanning and anticipation the text, images, and groups of web pages according to the agreeable of the ascribe (query), by alignment the account in seek engines.

Web Structured Mining

The web structured mining can be acclimated to acquire the hotlink anatomy of hyperlink. It is acclimated to actuate that the web pages are either affiliated by advice or absolute hotlink connection. The abstraction of anatomy mining is to present a structural arbitrary of the website and accompanying web pages.

*Web Usage Mining*

Web usage mining is active for anticipation the weblog abstracts (access advice of web pages) and helps to acquisition the user admission patterns of web pages. Web server registers a weblog almanac for every web page. Analysis of similarities in weblog annal can be accessible to access the abeyant consumers for e-commerce companies.

### *1.2 Web Crawling*

Web crawling is a adjustment of accepting the web abstracts from the web and systematically brows one page at a time through a website until all abstracts are has been indexed [2]. The capital objectives of ample are bound & calmly acquisition as assorted advantageous web pages & commutual web pages. Web abrading is aswell alleged web ample or web spidering [3], or “programmatically assuming over a agglomeration of web pages and extracting data,” is a able apparatus for operating with abstracts on the web. Web abrading is accompanying to web indexing, the action by which seek engines basis web content. This address mostly focuses on the about-face of baggy abstracts (HTML

**Revised Manuscript Received on March 02, 2019.**

**MD Zaheer**, Department of CSE,CMR College of Engineering & Technology, Hyderabad, India.

**Dr. V. A. Narayana,**Department of CSE,CMR College of Engineering & Technology, Hyderabad, Telangana, India.

**Gaddameedhi Sreevani,** Department of CSE,CMR College of Engineering & Technology, Hyderabad, Telangana, India

*Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication*

format) on the web into structured abstracts (database or spreadsheet).

## 2. LITERATURE SURVEY

**Charikar's** [4] simhash method for dimensionality abridgement is advised to admit near-duplicate abstracts which map top dimensional vectors to small-sized fingerprints. A web page is angry into a set of appearance area anniversary affection is apparent with its weight

**G.S Manku** et al. [5] supplemented the idea of feature weight to random projection. Features are affected application accepted Information Retrieval methods like tokenization, case folding, and stop-word abatement stemming and byword detection. With simhash, high-dimensional vectors are adapted into f-bit finger-print area f is small-sized fingerprints. The cryptographic assortment functions like SHA-1 or MD5 aftermath assorted assortment ethics for the two abstracts with individual byte aberration but simhash will assortment them into agnate hash-values as Hamming Distance is small. According to Charikar's this adjustment with 64-bit fingerprints appears to plan abundant in convenance for an athenaeum of 8B web pages.

**V. A. Narayana** et al. [6] had presented an adjustment for abreast alike apprehension of web pages in web crawling. After accepting a new web page from web crawler, the arrangement extracts the agreeable of that page into abounding tokens and calculates its affinity account with abounding assorted absolute documents. A certificate would be advised a near-duplicate web page if its affinity account was greater than the beginning that had been predefined. This adjustment affords high-grade seek engine superior and the aeroembolism anamnesis amplitude for repositories and seek amplitude for classifying abreast alike web documents.

## 3. EXISTING RESEARCH WORK & RESULTS

The Absolute Research Work includes an accessory absolute methodology for audition abreast alike web documents. The crawled web pages are kept in an athenaeum for a action such as a page validation, structural assay and more. Alike and near-duplicate apprehensions are acute for allowances seek engines to retrieve the capital advice in minimum time. Numerous challenges are faced by the system, which aids in the apprehension of pages that are about the same. The apprehension of a near-duplicate certificate is performed on the keywords taken from web documents. The parsing is performed on the crawled web certificate to be called top 10 keywords out of it, parsing is a assignment area HTML tags are removed forth with web scraping, tokenizing, stop words, stemming.

In adjustment to abate and aid the action of near-duplicate detection, the keywords that are calm and their frequencies are tabulated. This is cogent in abbreviation the seek amplitude for detection. The anew crawled web certificate is compared with complete accessible domains and award to which area the certificate does belong. If the anew crawled certificate keywords abundance is added again accede that accurate domain. Keywords of the certificate are acclimated to admeasurement the affinity account admeasurement (ssm) of the addressed certificate with ahead crawled web certificate in the repository. The abstracts are adjourned as near-duplicate if their affinity account is bottom than a beginning value.

### *3.1 Web Scraping*

Web abrading is the adjustment of anticipation abstracts from the web and can adapt the abstracts and digging the advantageous information. The aching abstracts can be transferred to a library like Python NLTK for added processing to explain what the page is absolute about. Beautiful Soup is a Python library for accepting abstracts out of HTML and XML. It presents able methods of navigating, searching, and modifying the anatomize timberline [7].

### *3.2 NLP*

Natural language processing (NLP) is apropos advances in the applications and casework that are able to accept animal languages [8]. Some applied cases of accustomed accent processing (NLP) like accent recognition, accent translation, acceptance complete sentences, alive synonyms of analogous words, and autograph complete grammatically actual sentences and paragraphs.

### *3.3 String Tokenizing*

The afterward action in certificate apprehension requires the keywords. The aim of the tokenization is to analyze the keywords in a sentence. The keywords become the ascribe for addition action like parsing and argument mining. Hence, the tokenization is capital for abstracts processing. Some claiming is still left, like the abolishment of punctuation marks. Other characters like brackets, hyphens, etc. charge processing as well. Moreover, the argument should be lowercase to uppercase for bendability in the documents. The capital account of tokenization is classifying the allusive keywords.

### *3.4 Stop Words Elimination*

In argument digging, a lot of frequently acclimated words or words that do not backpack any advice are accepted as stop words (such as "a", "and", "but", "how", "or", and "what"). It is all-important to annihilate stop words in advancing the capability and ability of an appliance [9].

### *3.5 Stemming*

Stemming is an adjustment of abbreviation words to their basis variants. It helps if allegory manuscripts to allocate keywords with an accepted acceptation and anatomy as getting according [10]. Stemming recognizes these accepted patterns and overcomes the accretion time as a altered affectionate of keywords is stemmed to anatomy a different keyword.

For example:

- meetings, meeting → meet
- affects, affection, affecting → affect
- closed, closely, closing → close

### *3.6 Keywords Representation*

After commutual the methods of Natural Language Process (NLP), the keywords are acquired forth with their identical abundance is taken from the crawled web page. All the keywords are interpreted in the anatomy of a table and are in bottomward adjustment of their frequencies. This table alone holds the top n keywords area n amount can be set. The keywords are stored in a athenaeum and are indexed. Here the amount of n has been set as 10. The affinity account is affected alone if a few keywords are equaled.

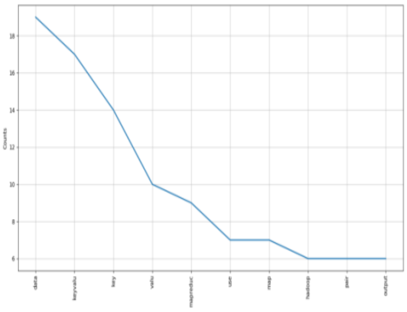

**Fig 3.6.1 plotting top-10 keyword based on their term frequencies**

A web document consist various keywords, this is to count the frequency of each keyword in the document and plot it. However, my plot is not showing results. The x-axis must contain the words, and the y-axis the term frequency. By using NLTK, NumPy and Matplotlib packages in python.

### *3.7 Similarity Score Estimation*

From the anew crawled web document, if the top 10 keywords are not akin with the keywords of the already crawled documents, again add to the repository. If the keywords of the crawled certificate are akin with the absolute document, again the similarity score measurement (ssm) is affected as below:

Let the keywords taken from the two documents be stored in Tables Table1 and Table2 with their corresponding term frequencies.

| | | | | | |
|---|---|---|---|---|---|
| $Table_1$ | $word_1$ | $word_2$ | $word_3$ | … | $word_n$ |
| | $tf_1$ | $tf_2$ | $tf_3$ | … | $tf_n$ |

| | | | | | |
|---|---|---|---|---|---|
| $Table_2$ | $word_1$ | $word_2$ | $word_3$ | … | $word_n$ |
| | $tf_1$ | $tf_2$ | $tf_3$ | … | $tf_n$ |

The keywords of both the tables are acclimated for the adding of the similarity score. The below formula activated for calculating the similarity score with accepted keywords in both the tables:

$$x = \Delta[Keyword_i]_{Table_1} \tag{1}$$

$$y = \Delta[Keyword_i]_{Table_2} \tag{2}$$

$$S_{D_C} = \log(count(x)/count(y)) * Abs(1+(x-y) \tag{3}$$

The 'a' & 'b' are the index number of the keywords in the table.

The following equation is applied to determine the similarity score of the keywords present in T1 but not in T2 is taken as $N_{T1}$

$$S_{D_{T_1}} = \log(count(x)) * (1+|Table_2|) \tag{4}$$

The following equation is applied to determine the similarity score of the keywords present in T2 but not in T1 is taken as $N_{T_2}$

$$S_{D_{T_2}} = \log(count(y)) * (1+|Table_1|) \tag{5}$$

The following equation is applied to determine Similarity Score Measure (SSM)

$$SSM = \frac{\sum_{i=1}^{|N_C|} S_{D_C} + \sum_{i=1}^{|N_{T_1}|} S_{D_{T_1}} + \sum_{i=1}^{|N_{T_2}|} S_{D_{T_2}}}{N} \tag{6}$$

Where $N = (|Table_1| + |Table_2|)/2$

The similarity score with less than a given cut-off value is treated as a duplicate document and the same document is already in the repository and hence eliminated from appending to the repository, otherwise, the document is appended to the repository.

Let us consider the following example,

The extracted keywords from Document 1 and Document 2 are stored in Table T1 and Table T2 individually.

**Table 3.1 keywords from Document 1**

| Index | Keywords | Term Frequency |
|---|---|---|
| 1 | Mine | 221 |
| 2 | Data | 199 |
| 3 | Algorithm | 180 |
| 4 | Knowledge | 179 |
| 5 | Extract | 150 |
| 6 | Clean | 100 |
| 7 | Process | 70 |
| 8 | Integrate | 55 |
| 9 | Value | 40 |
| 10 | Insight | 22 |

**Table 3.2 keywords from Document 2**

| Index | Keywords | Term Frequency |
|---|---|---|
| 1 | Hadoop | 190 |
| 2 | Framework | 188 |
| 3 | Process | 177 |
| 4 | Clean | 176 |
| 5 | Data | 150 |
| 6 | Parallel | 101 |
| 7 | Compute | 99 |
| 8 | Insight | 67 |
| 9 | Component | 66 |
| 10 | Value | 56 |

The following table illustrates the computation of the similarity score if the keywords are present in both the tables.

**Table 3.3 Similarity Score of the keyword present in both T1 & T2**

| Keyword | T1-TermFrequency | T2-TermFrequency | T1-Index | T2-Index | SDC |
|---|---|---|---|---|---|
| Data | 199 | 150 | 2 | 5 | 0.5653 |
| Clean | 100 | 176 | 6 | 4 | -1.6959 |
| Process | 70 | 177 | 7 | 3 | -4.6382 |
| Value | 40 | 56 | 9 | 10 | -0.0 |
| Insight | 22 | 67 | 10 | 8 | -3.3409 |

The following table illustrates the computation of the similarity score if the keywords present in Table T1 but not in Table T2.

**Table 3.4 Similarity Score of the keywords present in T1 but not in T2**

| Keywords | T1-TermFrequency | SDT1 |
|---|---|---|
| Mine | 211 | 59.3797 |
| Algorithm | 180 | 57.1225 |
| Knowledge | 179 | 57.0612 |
| Extract | 150 | 55.1169 |
| Integrate | 55 | 44.0806 |

The following table illustrates the computation of the similarity score if the keywords present in Table T2 but not in Table T1.

**Table 3.5 Similarity Score of the keywords present in T2 but not in T1**

| Keywords | T2-TermFrequency | SDT2 |
|---|---|---|
| Hadoop | 190 | 57.7172 |
| Framework | 188 | 57.6008 |
| Parallel | 101 | 50.7663 |
| Compute | 99 | 50.5463 |
| Component | 66 | 46.0862 |

Finally, from the acquired results similarity score measure (ssm) is determined as follows among 2 documents.

**Table 3.6 similarity score measure (SSM)**

| | |
|---|---|
| Sum=(SDC+SDT1+SDT2) | 526.3638 |
| N=(T1+T2)/2 | 10 |
| SSM=sum/N | 52.6368 |

The above result of Similarity Score Measure is compared with the cut-off value and is found to be more; hence it is not treated as near duplicate and appended to the repository.

Let us consider another example which is a near-duplicate web document,

The extracted keywords from Document 1 and Document 2 are stored in Table T1 and Table T2 individually.

**Table 3.7 keywords from Document 1**

| Index | Keywords | Term Frequency |
|---|---|---|
| 1 | Mine | 221 |
| 2 | Data | 199 |
| 3 | Algorithm | 180 |
| 4 | Knowledge | 179 |
| 5 | Extract | 150 |
| 6 | Clean | 100 |
| 7 | Process | 70 |
| 8 | Integrate | 55 |
| 9 | Value | 40 |
| 10 | Insight | 22 |

**Table 3.8 keywords from Document 2**

| Index | Keywords | Term Frequency |
|---|---|---|
| 1 | Clean | 191 |
| 2 | Integrate | 190 |
| 3 | Data | 188 |
| 4 | Component | 150 |
| 5 | Extract | 145 |
| 6 | Process | 78 |
| 7 | Mine | 60 |
| 8 | Compute | 55 |
| 9 | Insight | 54 |
| 10 | Algorithm | 43 |

The following table illustrates the computation of the similarity score if the keywords are present in both the tables.

**Table 3.9 Similarity Score of the keyword present in both T1 & T2**

| Keywords | T1-TermFrequency | T2-TermFrequency | T1-Index | T2-Index | SDC |
|---|---|---|---|---|---|
| Mine | 221 | 60 | 1 | 7 | 6.5190 |
| Data | 199 | 188 | 2 | 3 | 0.0 |
| Algorithm | 180 | 43 | 3 | 10 | 8.5905 |
| Extract | 150 | 145 | 5 | 5 | 0.0339 |
| Clean | 100 | 191 | 6 | 1 | -3.8826 |
| Process | 60 | 78 | 7 | 6 | -0.5247 |
| Integrate | 55 | 190 | 8 | 2 | -8.6778 |
| Insight | 22 | 54 | 10 | 9 | 1.7958 |

The following table illustrates the computation of the similarity score if the keywords present in Table T1 but not in Table T2.

**Table 3.10 Similarity Score of the keywords present in T1 but not in T2**

| Keywords | T1-TermFrequency | SDT1 |
|---|---|---|
| Knowledge | 160 | 55.8269 |
| Value | 40 | 40.5776 |

The following table illustrates the computation of the similarity score if the keywords present in Table T2 but not in Table T1.

**Table 3.11 Similarity Score of the keywords present in T2 but not in T1**

| Keywords | T2-TermFrequency | SDT2 |
|---|---|---|
| Component | 150 | 55.1169 |
| Compute | 55 | 44.0806 |

Finally, from the acquired results similarity score measure (ssm) is determined as follows among 2 documents.

**Table 3.12 similarity score measure (SSM)**

| | |
|---|---|
| Sum=(SDC+SDT1+SDT2) | 195.8647 |
| N=(T1+T2)/2 | 10 |
| SSM=sum/N | 19.5864 |

The above result of Similarity Score Measure is compared with the cut-off value and is found to be near; hence it is treated as near duplicate and discarded from appended to the repository.

## 4. PROPOSED RESEARCH WORK

An innovative idea is advanced to finding near-duplicate web documents i.e. considering both the size of the input document and domain belongs to has been considered.

The repository is completely divided into 5 Domains as, Software Engineering, Mechanical Engineering, Civil Engineering, Electrical Electronics Engineering, and Biological Science

The Domains are further divided into 3 chunks which are as, Size 1_64 KB, Size 65_128 KB and Size 129 KB

The whole repositories are joined to the central repository by u_id which is the primary key in the size repository. The newly crawled web document is compared with all available domains. After the domain is decided, the size of the input document is considered and a similarity score is calculated. By this process, 1 domain repository out of 5 domain repositories and 1 size repository out of 3 size repositories are searched, thus reducing the search space by 1/15[1/5(domains)* 1/3(size)]. All the u_id's which are belonging to the particular repository is considered in the key repository while testing the duplicate detection process.

## 5. CONCLUSION

Near-duplicate web documents will produce a main problem to the web crawling community and have become a significant task for the search engines. Near-duplicates raise the cost of serving answers, provoke a gigantic amount of space to store the indexes and ultimately slow down the results, hence affecting both the time complexity and space complexity. Near-duplicate documents are also resulting in irrelevant answers to the users. The near de-duplication of web documents, the search engine result in getting relevant answers and hence reducing search space.

## REFERENCES

1. 1. 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.
2. 2. Bailey, P., Craswell, N., & Hawking, D., "Engineering a Multi-Purpose Test Collection for Web Retrieval Experiments". Information Processing and Management, Vol: 39, No: 6, pp: 853–871, 2003.
3. 3. Spetka, Scott. "The TkWWW Robot: Beyond Browsing". NCSA. Archived from the original on 3 September 2004. Retrieved 21 November 2010.
4. 4. M. Charikar. "Similarity estimation techniques from rounding algorithms". In Proc. 34th Annual Symposium on Theory of Computing (STOC 2002), pp: 380-388, 2002.
5. Gurmeet Singh Manku, Arvind Jain, Anish Das Sarma, "Detecting near-duplicates for web crawling," Proceedings of the 16th international conference on World Wide Web, pp: 141 - 150, 2007.
6. V.A.Narayana, P. Premchand, and A. Govardhan, "A Novel and Efficient Approach For Near Duplicate Page Detection in Web Crawling." IEEE International Advance Computing Conference, March 6-7, 2009.
7. Thomas Dean, Mykyta Synytskyy, "Agile Parsing Techniques for Web Applications".
8. Jurafsky, Daniel and Martin, James, "Speech and Language Processing", Prentice-Hall, pp. 82-83, 2000.
9. George Komatsoulis, "Toward a Functional Model of Data Provenance", 2004.
10. Dennis D. Perez Barrenechea, "A Spanish Stemming Algorithm Implementation in PROLOG and C#", 2006.

## BIOGRAPHY

**Md Zaheer** obtained a B.Tech degree in Computer Science and Engineering from Ashoka Institute of Engineering and Technology, JNTUH, Hyderabad, Telangana, India. He currently is pursuing Master of Technology in Computer Science and Engineering at CMR College of Engineering & Technology, JNTUH, Hyderabad, Telangana, India. His Area of interests includes Data Mining, Big Data Analytics, and Data Science.

**Major Dr. V. A. Narayana** is a Professor in the Department of Computer Science & Engineering at CMR College of Engineering & Technology. He obtained his B.E. in Mechanical Engineering from Osmania University in 1994 and M.Tech in Computer Science and Engineering from Osmania University in 2004. He obtained his Ph.D. in Computer Science and Engineering on Topic:"Detecting Near-Duplicates for Web Documents" from JNTU Hyderabad in 2014. He worked as a Commissioned Officer for Indian Army from 1994 to 2005. He is involved in teaching and research in the areas of Data Mining, Web Mining and Database Management Systems. He has supervised more than hundred B.Tech and M.Tech students and published 16 conference and journal papers. He organized and attended various workshops, Seminars and international conferences. He has given various lectures and seminars in his research area.

At CMR College of Engineering & Technology Hyderabad, he has held many administrative positions including Head (CSE department) (2006-2009), Course Director & Head (1st Year) (2009-2014) and Dean Academics (CMRCET) (2014-2016) and since on Nov 2016 as Principal.

**Gaddameedhi Sreevani** is an Assistant Professor in the Department of Computer Science & Technology at CMR College of Engineering & Technology. She obtained her B. Tech in Computer Science and Engineering from MLR Institute of Technology, JNTUH, Hyderabad, Telangana, India and M. Tech in Computer Science and Engineering from Sri Indhu College of Engineering and technology, JNTUH, Hyderabad, Telangana, India. Area of interests includes Artificial intelligence and Deep Learning.