Article

Not peer-reviewed version

A Novel Hashcode-based Duplication Reduction via Thresholding Approach for Large-scale Web Documents

Sana Ejaz , Asma Naseer , asma Ahmad , maria Tamoor * , Samina Naz

Posted Date: 6 August 2024

doi: 10.20944/preprints202408.0443.v1

Keywords: Duplicate detection; Hash keys; Information Retrieval; Threshold; Web documents; and Web pages

 doi:10.20944/preprints202408.0443.v1

Article

A Novel Hashcode-Based Duplication Reduction via Thresholding Approach for Large-Scale Web Documents

Sana Ejaz [1], Asma Naseer [1], Asma Ahmad Farhan [1], Maria Tamoor [2,*] and Samina Naz [3]

[1] National University of Computer and Emerging Sciences, FAST School of Computing, Lahore, Pakistan
[2] Department of Computer Science, Forman Christian College University
[3] Muhammad Nawaz Sharif university of engineering and technology
* Correspondence: Email: mariatamoor@fccollege.edu.pk; ORCID: 0000-0002-3023-6706

Abstract: Modern search engines encounter a significant challenge when it comes to handling duplicate and nearly identical web pages, particularly during the indexing process for vast amounts of web content. This issue can lead to slow search results and increased costs due to the accumulation of storage space necessary for storing indexes. To tackle this issue, different techniques have been proposed to find similar websites. However, it has long been a challenge in research to distinguish between web pages. In the current study, sentence-level features i.e., hashcode and thresholding are used to determine the nearly identical web pages. We employ an adaptive threshold that enables the application of our model in both large- and small-scale settings. The benchmark datasets consisting of Shakespeare's collections, free text, job descriptions, and Reuters-21578 are used to test the proposed approach. With an accuracy of 0.99 and an F1-score of 0.97, the proposed technique outperforms existing methods.

Keywords: duplicate detection; hash keys; information retrieval; threshold; web documents; web pages

1. Introduction

The duplication of data over the internet has been an issue for a long time. The digital document databases include duplicate documents. The duplication of pages makes it much harder to retain indexes which reduces the effectiveness of the results, and aggravates users. Duplicate page identification is crucial for both querying for similar documents and filtering out unnecessary material in document databases. Duplicate page elimination leads to lower storage expenses and higher-quality search indexes. Information retrieval from the web is to get a web page to extract useful content from it. The content initially extracted is in raw form, and needs to be preprocessed and normalized. On the internet, duplicate web pages are common. Even though they are not bit-for-bit identical, near duplicates are exact replicas of the other pages and there aren't many changes between the documents, such as adverts, counters, and timestamps. Numerous renderings of the same physical object, typographical errors, versioned, similar web pages, spam emails created using the same template, and many other circumstances may also cause the removal of a nearly identical web page. The standard of search indexes is improved, storage costs are reduced, and safe network bandwidth is achieved by removing near duplicates. Additionally, the demand on servers that host web pages remotely is reduced. Systems that identify duplicate and almost duplicate web pages must perform several intricate operations. The primary issue is with the size of the online pages; the search engine has a big number of web pages, which is why it has a massive database that spans many terabytes. Another problem is the search engine's capacity to browse many web pages and for each page it allows to find duplicates by computing hash key values that provide a concise description of the characters on that web page. When two website documents have the same hash key value, it is necessary to determine if the page value is the same and, if it is, to conclude that one is a duplicate copy of the other page. In order to identify duplicate web pages, check-summing techniques are utilized. The identification of close duplicates is useful for many applications such as relative documents retrieval or clustering multiple documents as a single topic etc. The quality and diversity of the search results are improved by locating and deleting the nearly identical websites. This can further aid in the

 doi:10.20944/preprints202408.0443.v1

detection of spam content. Different web mining applications, such as document clustering, filtering, recognizing repeated online collections, finding huge dense networks to identify duplication, and social network site community mining, are required to fully and properly determine near copies. An important application is to identify fake user profiles which are generated from other profiles for forgery and other malicious purposes. The elimination of nearly duplicate pages contributes to significant bandwidth reduction, lower storage costs, and improved search index quality. The first step in preprocessing crawled web pages is parsing, which removes the HTML, tags, and scripts from the online resource. Utilizing the stemming technique, suffixes and prefixes are removed. The next stage is to group the various documents into relevant clusters to enable the exploration of a document collection via document clustering or to present the search engine's results in response to a user's request. This can significantly increase the memory and accuracy of information retrieval systems, and it is an accurate method of identifying a document's closest neighbors. Here, a useful method for detecting duplicate web pages based on similarity and employing hash values is proposed [1–5].

This research aims to adequately avoid duplicates and near duplicates, which results in depletion in storage costs and time. In this method, the hash is used along with sentence segmentation. Instead of generating one hash of a whole page or fixed-sized chunks of data, we applied hash on sentence level. The advantage of this approach is that if you generate the hash value of an entire document and modify even a single line, the entire hash value will change. This yeilds false results and allows the document to be uploaded when it should not have been uploaded. By breaking down a document and generating a sentence-level hash it can be made sure that each sentence is unique, resulting in a unique document instead of a copy. In many cases reuse of data is required such as fixed terms of a particular field, for that, a value is set in the algorithm to leave room for reusing data. This value allows only copying some percentage of text from the whole web. Each field has a growing set of terms that are unchangeable, they can't be added to a dictionary as the term lengths are various and in huge numbers. Firstly, a dataset is preprocessed and each document is broken down into sentences and a hash is created against each sentence. Then using these sentence hashes, duplication is tested on any new data. If the requested data has duplication higher than the allowed value, then it should not be uploaded. If not, then it can be uploaded. This allows minimization of the same documents being uploaded on the web multiple times. And also, can detect if a page is simply a combination of multiple existing pages or documents [6].

The following are four main methods that are applied to datasets before they are processed by the models in order to help the model to make extract useful information and provide better results. But not all steps are always required. It depends on the nature of the application about which steps are required and which are unnecessary. The fifth method is performed later for filtration purposes and to reduce time.

1.1. Preprocessing:

Previously people have utilized preprocessing, which involves turning a web page into keywords also called segmentation. Segmentation can be at page,sentence, word or character level [7–11]. Crawling, parsing, stemming, and the elimination of stop words are processed during the preprocessing stage. The database is searched for the web pages. When parsing, a grammar is used to explain the linear structure of the web pages. The processed papers are then cleaned up by removing stop words and linking words like "is," "as," "are," and more. A procedure known as stemming is used to reduce the words in the document to their most basic forms.

1.2. Parsing:

Parsing is a technique for organizing a linear representation in accordance with a predetermined grammar. After obtaining a webpage, they had to parse its content in order to obtain data that would likely feed and direct the crawler's future course. In parsing, the extraction of URLs may be straightforward [12] or it may entail the more difficult task of picking up the HTML content and

 doi:10.20944/preprints202408.0443.v1

looking at the HTML tag tree. To eliminate stop words and execute stemming on the remaining words, turn the URL into a recognized form before parsing.

1.3. Stop Word Removal:

Stop words, which are terms that are utilized in online documents but have less significant meanings than the keywords are frequently eliminated. Search engines often remove these stop words from sentences and provide the appropriate response to the query. To improve search performance, all stop words are eliminated from searches, even the most frequently used ones like "a" and "the." Stop words like "it," "can," "an," "and," "by," "for," "from," "of," "the," "to," and "with" are frequently the most well-known and widely used stop words. Stop-word elimination is a step that is taken while parsing a text to learn more about its content.

1.4. Stemming:

Words with similar morphological variations frequently share the same meaning explanations and are regarded as equal in many contexts. To achieve this goal of reducing a word to its root form, numerous stemming algorithms or stemmers have been devised. Instead of the actual word, a query's stems reflect its important terms. Lemmatization, an algorithm that aims to reduce each word to its grammatically suitable root, ultimately helps to minimize the number of words. Condensing each word's inflectional and derivational suffixes allowed for the creation of this. For instance, "eat" is the fundamental form of the words "eating," "eatings," and "eaten."

1.5. Cascade Filter:

Following the preprocessing phase, duplicate web pages are found using the defined cascade filtering. Two filtering methods are used sequentially in cascade filtering. Sentence-level feature hash comparison is the mechanism employed in this strategy. By using the hash values comparison approach and sentence level feature filtering to speed up execution, these strategies aim to boost the precision of the outcome. With the help of these two techniques, a result that is efficient, exact, and quick is produced. The time reduction is achieved at the sentence level because it just combines the sentence elements and not the keywords. On particularly large web pages, the sentence-level feature is used to filter out web publications that don't fit the criteria. After comparing sentence features, fewer papers are obtained, thus saving time. The technique computes the processed document using the hash algorithm and compares the results to find duplicate web pages [2].

Major Research Contributions: The proposed approach presents a promising solution to the challenge of dealing with duplicate and nearly identical web pages in search engine indexing and offers an improved accuracy compared to the existing approaches. The major research contributions can be summarized as:

- The current study suggests a novel approach based upon sentence-level hash-code extraction and a thresholding mechanism to distinguish between identical web pages.
- An adaptive threshold is employed, allowing the proposed model to be effective in both large- and small-scale settings.
- Benchmark datasets, including collections of Shakespeare's works, free text, job descriptions, and Reuters-21578, are used to test the proposed approach.
- The proposed technique demonstrates impressive performance, with an accuracy score of 0.99 and an F1-score of 0.97 thus outperforming existing methods.

The subsequent sections in the rest of the article are structured in a way that ensures coherent organization of information. Section II offers a comprehensive overview of the relevant technical details and provides a bird eye-view of the existing techniques. On the other hand, section III provides an insight into the methodology employed in this research while section IV depicts the results. Finally section V provides discussion and comparison with the state-of-the-art, followed by a conclusion in section 7.

 doi:10.20944/preprints202408.0443.v1

2. Literature Review

These days internet is the main source for searching which aims to get information based on queries. Everyone can find information or solutions related to their problems or a specific topic [13,14] and get results within a large number of document collections. A large number of data is available on the web specifically, a huge number of pages are now available due to which finding relevant and correct information have become a challenging task. Information Retrieval(IR) is a process or a technique to retrieve the information that matches the queries. However, relevance is an immensely important aspect of Information Retrieval

Data mining is also used for IR as it has higher technology-level audits. It can be applied to research web IR to improve its processing capabilities to a higher intelligence level.

IR uses different models to retrieve information. However, till date, IR methods are not mature enough to make use of semantic knowledge within documents and retrieve precise results. According to [15], web will evolve to yield correct results in near future, currently, new methods [16,17] have been introduced and are being used in the advancement of the field.

To enhance the efficiency of user information retrieval, a model based on information retrieval (IR) is constructed. This model incorporates a domain ontology and a knowledge base to augment traditional information retrieval methods, aiming to improve efficiency. The integration of ontology-based IR system enhances semantic retrieval along with keyword-based information retrieval, resulting in improved recall and precision outcomes. It is common for users to utilize two to three keywords when conducting searches, yet a vocabulary gap often exists between the keywords in the documents and those used in the query [18,19]

In [20], Particle Swarm Optimization(PSO) algorithm is used that assigns weights to the keywords. The system is divided into three parts i.e., first is web page databases which are arrangements of web pages that are stored from distinctive Sources. For search queries, these pages are retrieved from the source. Then there is a query submission subsystem where the user can specify needed data which is then provided by the framework, for this user's query is submitted in a sequence of keywords in the target database. Lastly, there is a matching mechanism, in which there are stored feature sets that are searched using query strings. This returns the most relevant web page. These three steps are used to enhance accuracy, memory consumption, and time consumption by providing content-based results and using a static database to deal with the same words of query keywords.

[21] proposed two models of Web Information retrieval, a set of premises and an algorithm, these are used for ranking the documents retrieved related to user queries. "Information retrieval model is a quadruple $[D, Q, F, R(qi, dj)]$, where D is a set of logical views for documents in the collection; Q is a set of logical views for the user queries; F is a framework for modeling document representations queries, and their relationships; $R(qi, dj)$ is a ranking function which associates a real ranking with a query qi document dj." Retrieval models can describe the computational process, for example, how the documents are ranked and how documents or indexes are implemented.

In a work proposed by [22], IR retrieves the information in two parts i.e., one is the document process, and the second is the document retrieval. In the Information retrieval model, a document is grouped into tags and indexes based on semantic structure. A large number of documents are available on the Internet so users must find information accurately. For making sure that users find accurate data IR uses mathematical models. The classical IR models are the Boolean probability vector binary retrieval model.

The Boolean model is a classical IR model based on theory and Boolean expression but it gives accurate results on whether the documents are related or not. However, the Boolean model lacks the ability to correctly describe the situation [23].

The vector space model is based on vector space and vector linear algebra operation which abstracts the query conditions and text into a vector in the multidimensional vector space [24].

Link analysis [25,26] which used hyperlinks on pages is used successfully for deciding which web pages will be correct to add it to the document collection. It is useful for determining which pages to

 doi:10.20944/preprints202408.0443.v1

crawl and for ordering the documents according to the user's query (ranking). Also, it has been used to find relative pages of the current pages and to find duplicate pages. It has mainly been used for ranking but this hyperlink structure needs to be further analyzed.

Few of the previous researches [27,28], presented ideas of user modeling in four paradigms i.e., ad-hoc Information Retrieval, information hypertext browsing, filtering, and visualization. Ad-hoc returns a list of ranked documents to retrieve relevant information. In filtering, the user gets a selection of the most relative documents after a long-term search from a flow of documents going through a filtering system. In hypertext, a collection of related documents from a collection is returned which are linked together as in the link analysis above. In the visualization method, a user can select relevant information by interaction with a 2− or 3−dimensional visualized set and observe to select. For better results, it was considered to combine all four of these techniques. Although it was later discussed as four relevant paths to achieving and constructing better tools.

However, current IR methods lack speed and precision in comparison to their large-scale information needs. A Natural Language Processing(NLP) based semi-supervised algorithm for learning was introduced to deal with various issues [29]. NLP techniques ensure that the input data is correct grammatically and in spelling. In the semi-supervised algorithm, some undirected graphs are made and labeled to reduce the complexity of the Information Retrieval process. Both labeled and unlabeled data are used for the semi-supervised technique where unlabeled data is further divided and processed for prediction and the labeled data simplifies the undirected graphs. The text rank algorithm extracts keywords that are used for page ranking. Web and its pages on it are a directed graph where the pages are nodes. The graph shows the relation between all interlinked nodes. After that, a page rank score is computed. The text rank and page rank are similar as in both cases some data is extracted and after some calculations, its score is achieved which shows its importance. Experiments were done in 5GB and 15GB sized data blocks with 4 sample sets each using two search methods: Information Retrieval based on Text Rank Summarization(TRS), and Information Retrieval based on Natural Language Processing. On small-scale data, TRS gave better results but on large scale, NLP gave better results in terms of time (0.496 seconds instead of 0.645) but hit rate remains similar. The proposed algorithm provided better retrieval time hence good performance.

For unstructured data, storage requirements are large. Despite the low cost of storage, few issues still need to be addressed that includes power consumption, space required physically for all the storage devices etc. The duplicate data reduction, in this case, can help reduce the need for storage and power. Features for all unstructured data are computed and similar files are deleted reducing duplicates. For structured data, the hash is an efficient way, but not for unstructured data such as audio, video, images, and graphs. The experiments done on images by extracting features like edges and colors and comparing them show that the method was efficient [30]

In a work proposed by [28] it is stated that similar or the same data is present at many locations on the web and needs to be standardized. To standardize, they proposed three phases. They created a structure to standardize the records which are flexible and can adopt new methods according to new needs. They investigated standardization techniques applied and decided what should be incorporated. They also used a weight-based Borda method which according to the measurements shown to work better than the pattern techniques.

Data deduplication can be applied on any form of storage such as primary, secondary, cloud, or any other for image, text, video, audio, etc. An increase in bandwidth, space and a deduction in cost are some of its advantages. The deduplication process is done on chunks of data in the storage. These chunks are compared to other data if it is unique, it will be saved if not then only a pointer that leads to one single instance is maintained and other copies are eliminated. This is also useful for the backup process. During the backup process, only the recently updated data is sent for backup instead of backing up the whole storage again. The data that was once backed up is not considered for backup again only the newly added data or the modifications are done on previous data. this reduced the time taken for backup. It is beneficial for large-scale storage systems like client-server systems.

 doi:10.20944/preprints202408.0443.v1

Near duplicate detection [31] refers to a duplicate with minimal changes in them. Their study conducted by [31] is based on pairs of 493K pages from 6K websites. These pairs are categorized into three categories clone, near duplicates, and distinct. Then boundaries are defined which are used to find 10 near duplicate detection methods from information retrieval, computer vision, web testing, and some web apps. They also discussed about the hash method smash used by Google with the fingerprint technique used for indexing and another fingerprint and hath technique TLSH. Results show that no model can perfectly detect all duplicates without some compromise such as coverage.

In a min hash-based algorithm that works on passages from long texts, the problem discussed is aligning the near duplicate passages in long texts. the problems identified are heuristic alignment due to high cost, which reduces recall. The model generates some hash values on chunks from a paragraph and compares them with others and returns similar values. to reduce the complexity of the computation of pairs of text compact windows are introduced which checks for overlapping paragraphs with multiple same-value pairs and eliminate the ones that don't match which reduces the huge number of pairs. The computation for creating windows is done using divide and conquer and the window size is variable, it takes a position in the text centers it, and expands to the text around that point to check if it matches other paragraphs. To increase efficiency, the longest matching pairs are computed, as also sentence-level paragraph duplication. The experimentations show that this method is better than the state of the art.

Another significant work, Copycat-21 [32], aims to reduce the computational and physical work costs. The study shows that $14 - 52\%$ of documents that a crawler crawls are near duplicates. First, the model compiles the near duplicate documents. For each document, it calculates its Simhash fingerprint. If more than one document has the same fingerprint, only one instance is kept. The rest of the fingerprints are divided, and lastly it calculates the difference among them. It is a precision-based model. However, the model is page-based and does not works document-based.

A large amount of mirror pages on the web are taking up many resources and are reducing user experience [33,34]. Further, few of them are also malicious. The Simhash-based deduplication model is used to identify such pages that can be used in any harmful way. The fingerprint similarity is calculated using Hamming distance. Mirror pages can be exact or partial copy of the original page. These mirror pages can be illegally used to rent IP hosts, phishing/harmful websites, and spread illegal activities far beyond their reach. The Simhash algorithm of creating fingerprints of pages and finding mirrors has high precision and recall as compared to previous studies. But the attackers can attack and modify the code to avoid detection thus compromising it for which some protection needs to be implemented and improved further.

Unlike other models, that detects near duplicates offline, model proposed in [35] works online news articles to reduce duplication created by editors. The previous hash methods created a hash of documents but it could not quantify the documents similarity. For near-duplicate detection, a similarity measure is explored. Specifically, Cosine similarity is explored. The presented model takes the HTML page extracts the original text from, them and then finds out near duplicates. If the original document is rightfully modified, the new one is linked to the original parent document. The model has 96% accuracy. It is fast, online, and can handle both HTML text and plain text.

In a study, an analysis of academic data from Russian schools is performed. The Russian cursive way of writing makes the problem harder. This handwritten text analysis algorithm uses near duplication to detect plagiarism in exams. The model applies to images, does not need word recognition, and is compared to the English handwritten text analysis algorithm. The words in documents are segmented without actually recognizing them. The model is comparable to state-of-the-art handwriting recognition algorithms. Table 2 summarizes the significant work covered in this literature review.

The current study introduces a technique using sentence-level features to identify nearly identical web pages, overcoming challenges- handling duplicate and nearly identical web pages, faced by the existing techniques. As nearly duplicate documents are hard to identify hence, the sentence-level hash

 doi:10.20944/preprints202408.0443.v1

coding along with threshold value makes the proposed approach outperform the existing methods with high accuracy and F1-score.

Table 1. Comparison of studies.

Paper	Title	Algorithm / Model	Segmentation	Results
1	Near-Duplicate Detection in Web App Model Inference	Simhash	Threshold-based	F1 = 0.45
2	Allign: Aligning All-Pair Near-Duplicate Passages in Long Texts	Allign /min-hash	Window pairs of size O(n)	F1-score from 0.595 to 0.672
3	CopyCat: Near-Duplicates Within and Between the ClueWeb and the Common Crawl	Simhash/ChatNoir-CopyCat-21	Page based	F1 = 0.94
4	An improved Simhash algorithm based malicious mirror website detection method	Simhash	128 bit strings	Indicates degree of similarity
5	Online Near-Duplicate Detection of News Articles	Shingling/min-hash	N-grams N= 3,4	F1 = 0.955 for 3-gram
6	A Duplication Reduction Approach for Unstructured Data using Machine Learning Method	SIFT features for images	8x8 grids	Effective comparison
7	Research on Information Retrieval Algorithm Based on TextRank	Semi-supervised learning/TextRank	5 and 15 GB blocks	Up to 56% hit rate
8	Near-duplicate handwritten document detection without text recognition	Series analysis/DTW	Fast DTW	Recall 87-96% for DTW
9	Normalization of duplication records from multiple sources	Weighted Borda	Record, field and value level	N/A
10	Data De-Duplication Engine for Efficient Storage Management	De-dupe engine	128 KB chunks	N/A
11	Web Information Retrieval Using Island Genetic Algorithm	Island genetic algorithm	Multiple	Similarity measure greater than 0.8
12	A near-duplicate detection algorithm to facilitate document clustering	Simhash	Words segmentation	Similarity less than 60% allowed
13	Near-duplicate web page detection: an efficient approach using clustering, sentence feature and fingerprinting	K=mode clustering, fingerprint extraction	Sentence level	F1 = 0.80

 doi:10.20944/preprints202408.0443.v1

Table 1. *Cont.*

Paper	Title	Algorithm / Model	Segmentation	Results
14	Efficient near duplicate document detection for specialized corpora	Simhash	Page level	N/A
15	Informational Retrieval on the Web	Boolean models	Words segmentation	N/A
16	Challenges in Web Information Retrieval	SVM	Whole Page	An Overview
17	Information Retrieval on the Web and its Evaluation	Shingling	Canonical sequence of tokens	Precision = 0.8 Recall = 0.05
18	Fuzzy logic based similarity measure for information retrieval system performance improvement	FLBSM, Cosine, Euclidean and Okapi	Page level	Best by FLBSM = 0.13
19	Web searching and information retrieval	Vector space model	Page level	Web is not a Digital Library
20	Webpage relationships for information retrieval within a structured domain	Hyperlink Structure	Page level	Prec@1 for retrieval= 0.668
21	Preface to special issue on user modeling for web information retrieval	WIFS	Keywords	N/A
22	Link analysis in web information retrieval	Hyperlink	Words segmentation	ranking query
23	A PSO Algorithm Based Web Page Retrieval System	PSO algorithm	Avg Accuracy 91%	
24	State of the art in Web Information Retrieval	Boolean models,Fuzzy Model,Vector Space Models and Probabilistic Models	Words segmentation	Temporal analysis is supported
25	Research on information retrieval model based on ontology	Domain ontology model	Words segmentation	Threshold of 0.55 gives Precision and Recall = 95%
26	After the Dot-Bomb	Classification	Index segmentation	Theoretical%
27	What is this page known for? Computing Web page reputations	Random Walk	Term on page	Needed improvement
28	Learning to Understand the Web	Hidden Markov model	Words segmentation	N/A
29	Context in Web Search	Context model	Keywords	Context is better than One size fit all
30	Next Generation Web Search: Setting Our Sites	Hyperlinks	Word Segmentation	N/A

 doi:10.20944/preprints202408.0443.v1

3. Proposed Methodology

The proposed approach uses hash algorithms and sentence-level extraction to find duplication. Our objective is to reduce the storage space and increase the speed of comparison and search. First of all, the documents are parsed and segmented into sentences, and a new hash is created against each sentence. When a new document is given as input with or without duplication these steps of sentence-level features and the hash algorithm is applied to identify if the input is a duplicate or not and remove the pages if duplication exists.

3.1. Datasets

For the evaluation of the proposed technique multiple datasets i.e., Free Text, Shakespeare's Plays, Job Description [36] and Reuters-2157 [37] are used.

- Free text: During the development process, the model was tested on free-form text such as clean paragraphs taken from Wikipedia or any piece of writing that is in a clean normalized form.
- Shakespear: For model evaluation, scenes are taken from Shakespeare's plays, available online. All the three categories of Shakespeare's plays i.e., comedies, histories, and tragedies, are included in this dataset. In total there are 34,895 sentences consisting of 884,421 words out of which 28,829 words are unique.
- Job Description: A new dataset [36] is also considered for the experiment and utilised in testing before and after the normalization of the dataset. The dataset contains HTML tags, dashes, and white spaces etc. hence required a thorough preprocessing.
- Reuters-21578: The Reuters-21578 dataset [37] is well-known and is considered a benchmark for the detection of duplicates as it is available with the ground truth. It is a document collection, consisting of news articles. The original collection contains 10,369 articles and 29,930 unique words. There are three splits of the dataset, each is made by one of the authors. The ground truth is available for the Levis-split of this dataset which is originally available in SGM format. Hence, the split with ground truth is used for the evaluation of the proposed technique.

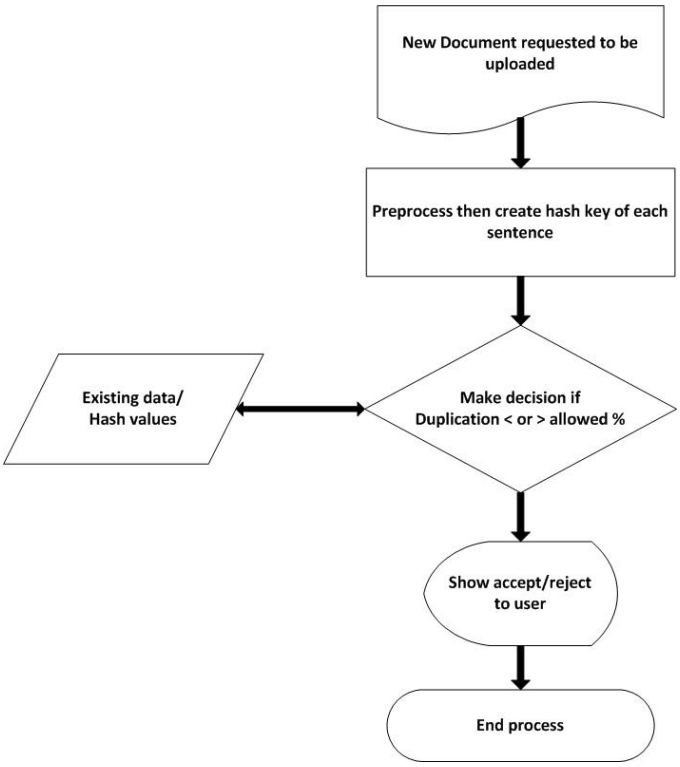

Figure 1. Process Flowchart.

 doi:10.20944/preprints202408.0443.v1

3.2. *Preprocessing:*

The datasets taken from different resources is preprocessed for which multiple steps are performed such as removal of HTML tags, conversion into a text file, normalization of all the data in one format. In addition to these setpes, segmentation is applied using different delimiters consisting of sentence boundaries such as '.' and '?', and other special characters like '!'. Both training and testing documents go through the same preprocessing. To evaluate the results of normalized data, both normalized and unnormalized datasets are used for the experiments.

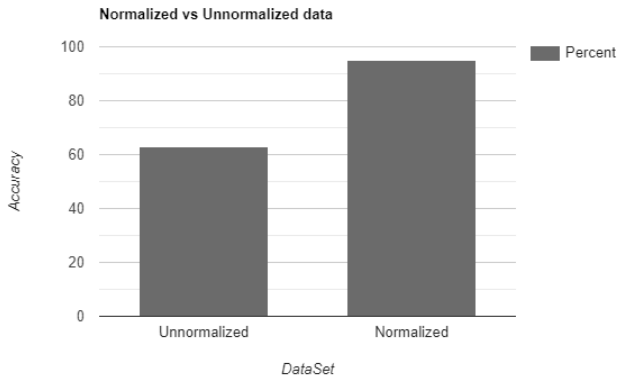

Figure 2. Unnormalized vs Normalized dataset results

3.3. *Hash Value Comparison:*

The dataset is in .sgm format which is in coded form so we first extract the actual text from the files and convert from .sgm format to .txt file format. Each file had a thousand documents which we extracted and gave to the algorithm. Previously filtered documents of web documents are converted into a set of keywords. Each keyword has some weight which is the count of how many times the keywords appear in the documents. If the hash values are the same then the documents will not be uploaded at a website and if the new documents and other pages have unique hash key values, then the new documents will become a part of the website that employs this technique. Previously, the proposed technique are applied to the whole page which result in a different hash of a document with even a single word difference. The page-level hash changes even if a single word is changed and would require a distance calculation to determine similarity level. As for keywords, multiple documents can be on the same topic and would have the same keywords but that does not necessarily mean that they are duplicates of each other.

Hence, our proposed technique calculates hash value at sentence level in combination with its tentative duplicates. Segmenting document into multiple sentences results in unique hash values for each sentence. As a result, changing words in a document will still rate it high as a duplication. Using the hash of an entire document does not yield good duplicate detection. However, this approach helps eliminate flaws more effectively compared to the techniques that depend on chunks with fixed-size or detect duplication at the whole-page level.

The hash model used is a cryptographic function and has manipulation detection code; a keyless hash model for which equation 1 to 6 are used.

$$C(X,Y,Z) = (X \wedge Y) \bigoplus (\bar{X} \wedge Z) \tag{1}$$

$$M(X,Y,Z) = (X \wedge Y) \bigoplus (X \wedge Z) \bigoplus (Y \wedge Z) \tag{2}$$

$$\sum_{0}(X) = RotR(X,2) \bigoplus RotR(X,13) \bigoplus RotR(X,22) \tag{3}$$

 doi:10.20944/preprints202408.0443.v1

$$\sum_{1}(X) = RotR(X,6) \bigoplus RotR(X,11) \bigoplus RotR(X,25) \tag{4}$$

$$\sigma_0(X) = RotR(X,7) \bigoplus RotR(X,18) \bigoplus RotR(X,3) \tag{5}$$

$$\sigma_1(X) = RotR(X,17) \bigoplus RotR(X,19) \bigoplus RotR(X,10) \tag{6}$$

```
The COUNT's palace.
Enter BERTRAM, the COUNTESS of Rousillon, HELENA, and LAFEU, all in black.
```

Figure 3. Sample strings from dataset

```
"2fb023b0f5ef2a109cd70e65e18e988d981e29d243f560f2851912b5d6efb8fe",
"8cd9bdf98131921bcf1c8cdbce840325a743a6502fb8ffe4f08c4ea4ee2cf4af",
```

Figure 4. Hash Key value of sample string in Figure 3

Regardless of the length of a sentence, the hash generated using equations 1 to 6, is of the same size for every sentence.

3.4. Duplicate Detection:

To detect duplicate documents, the hash of all files is compared with the hash of the new input file. Since the hash values are stored at the sentence level, comparing them becomes an easier task. The model takes the hash values of the input file and compares them to the hash values of all files. It then calculates the percentage of matching hash values and determines whether it is below or above the allowed threshold. Based on this calculation, the model provides the information that whether the file can be uploaded or not. The complete algorithm of text to hash generation is provided in Algorithm 1.

Algorithm 1: Text to Hash

```
EXTRACTHASH()
    file ← file(ReadTextStream)
    file ← file.tostring()
    file ← file.split('.')
    total_length ← file.length
    indexer_dict ← file(ReadTextStream())
    indexer_dict ← indexer_dict.to_json()
    for line of file
        hash ← generateHash(line)
        hash_list.push(hash)
    for hash of hash_list
        indexer_dict.findIndex(hash) ← index
        If index ≠ 0
            matches_count ← matches_count + 1
    compare_percent ← matches_count/total_length
    If compare_percent > allowed
        return Not allowed
    return Allowed
```

3.5. Evaluation Metrics

For evaluating the performance of the proposed technique multiple evaluation metrics including accuracy 7, precision 8, recall 9 and f-measure 10, are calculated.

$$Accuracy(A) = \frac{TP + TN}{TP + TN + FP + FN} \tag{7}$$

Where TP is True Positive, FP is False Positive, TN is True Negative and FN is False Negative.

 doi:10.20944/preprints202408.0443.v1

$$Precision(P) = \frac{\#Retrieved\&Relevant}{\#Retrieved} = \frac{TP}{TP + FP} \tag{8}$$

$$Recall(R) = \frac{\#Retrieved\&Relevant}{\#Relevant} = \frac{TP}{TP + FN} \tag{9}$$

$$F - measure = \frac{2PR}{P + R} \tag{10}$$

4. Results

Our evaluation comprised three datasets i.e., Reuters, Shakespeare and Job Description data. For evaluation, we tested a combination of data, extracted from all the three datasets. Some glimpse of the results produced by the proposed hash code based technique are depicted in the Figures 5–7.

```
72.72727272727273  Percent duplication found!
We cannot upload your file.
Accuracy =  0.9821428571428572
Precision =  0.9375
Recall =  0.9900990099009902
F1-score =  0.9630818619582665
```

Figure 5. Duplication above 50%

```
37.5  Percent duplication found!
We cannot upload your file.
Accuracy =  0.9876543209876544
Precision =  0.9803921568627452
Recall =  0.9970089730807578
F1-score =  0.9886307464162136
```

Figure 6. Duplication below 50%

```
16.666666666666664  Percent duplication found!
You can upload your file!
Accuracy =  0.9917355371900827
Precision =  0.9900990099009901
Recall =  0.9985022466300548
F1-score =  0.9942828734775043
```

Figure 7. Duplication below the allowed value of 20%

The confusion matrix (please refer to Table 2) provides an insight into the performance of the proposed algorithm. From a total of 400 evaluated documents across three datasets, it is observed that 388 documents are correctly classified. Among the remaining documents, 12 are falsely classified as non-duplicates, while 7 documents are incorrectly identified as duplicates.

A summarised overview of the performance evaluation against each dataset is provided in Table 4. The dataset derived from Shakespeare plays exhibits the highest accuracy, while the Reuters dataset demonstrates the highest precision. On average, the evaluation across all three corpora yields a 97% accuracy. The average precision, recall, and f-measure values are 0.98, 0.99, and 0.99 respectively. Figure 8 visually represents the average results.

 doi:10.20944/preprints202408.0443.v1

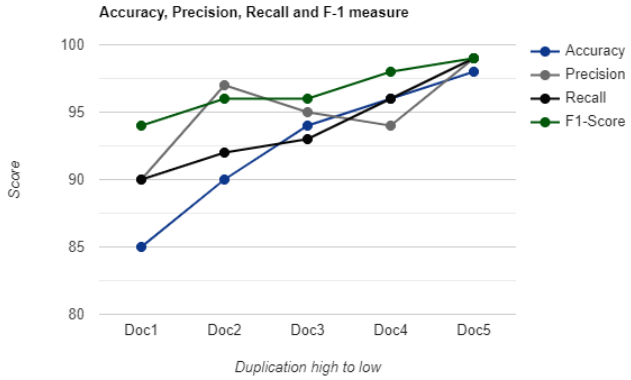

Figure 8. Results of Duplication against documents

Table 2. Confusion Matrix for all the three datasets.

		Predicted	
		Positive	Negative
Actual	Positive	388	12
	Negative	7	10

Table 3. A snapshot of the test results against some documents and their duplicates.

Source ID	Duplicate IDs	TP	FN
519	11422,1120	2	0
522	3164,7769,3735	3	0
3729	6044,10859,9972	2	1
5344	9857	1	0
7025	1969	0	1
7204	8343,7764	2	0
10459	2678	1	0
12456	1971,12471	2	0
5123	5281	1	0
16090	16199	0	1
16094	16357	1	0
16624	6236	1	0

5. Discussion

The proposed algorithm outperforms the state-of-the-art techniques by depicting higher accuracy in recognizing duplicate documents. It successfully identifies the majority of documents as duplicates. However, in the failed cases, the level of duplication is relatively very low thus making it difficult to classify them as duplicates. The results indicate that the level of duplication in these documents is minimal and are catered as false negative considering the ground truth.

To validate the performance of the proposed algorithm for non-duplicate documents, the same set of documents are used and majority of them are classified as non-duplicates. The model successfully classifies them as non-duplicates, and manual testing also confirms their absence of duplication.

 doi:10.20944/preprints202408.0443.v1

Table 4. Accuracy, Precision, Recall and F-Measure Scores for all the three datasets.

Dataset	No. of Docs	Accuracy	Precision	Recall	F-measure
Shakespeare	34	0.98	0.93	0.99	0.96
Job Desc.	50	0.97	0.98	0.99	0.98
Reuters	316	0.97	0.99	0.99	0.99
Average	**400**	**0.97**	**0.98**	**0.99**	**0.99**

The initial testing on unnormalized data yields a low accuracy result. However, when data is preprocessed and normalized, a significant improvement is observed. The important factor regarding normalization is that the process does not modify the actual data; rather, it optimizes it for better performance. Furthermore, incorporating more data further improves the accuracy of the model, spotlighting its ability to cater large-scale tasks. Another important factor is the model's parameter are determined from a smaller dataset, which is in contrast to the vast number of documents available on the web. Nevertheless, the flexibility depicted by the model allows for seamless implementation on the web with minimal adjustments, making it adaptable to varying data sizes and sources.

Table 5. Comparison of the proposed technique with the state-of-the-art.

Title	Dataset	Algorithm /Model	Segmentation	F1-Score
Near-Duplicate Detection inWeb App Model Inference [31]	Randomly crawled websites	Simhash	Threshold-based	0.45
Allign: Aligning All-PairNear-Duplicate Passagesin Long Texts [38]	Pan 11 and News	Allign /min-hash	Window pairs of size O(n)	0.59 - 0.67
CopyCat: Near-DuplicatesWithin and Between theClueWeb and theCommon Crawl [32]	ClueWeb09 ClueWeb12	Simhash/ChatNoir-CopyCat-21	Page based	0.94
Online Near-DuplicateDetection of NewsArticles [35]	SpotSigs dataset	Shingling/min-hash	N-grams N= 3,4	0.95
Proposed Technique	Shakespeare acts, Job description(SpotSigs), Reuters- 21578	Secure Hash Algorithm	Sentence level	0.97

In order to evaluate the performance of the proposed technique, the obtained results are compared with state-of-the-art methods, as presented in Table 5. The threshold-based method, utilizing the Simhash algorithm, achieved only a 0.45 F1-score. Conversely, the min-hash and alignment-based approaches were able to improve the F1-score from 0.59 to 0.67. Some techniques were able to achieve F1-scores as high as 0.94 and 0.95 by employing the Simhash/ChatNoir-CopyCat-21 technique at the page-level, and the Shingling/min-hash approach with N-gram, respectively.

Through analysis, it is evident that the current approach, employing sentence-level hashcoding and thresholding, significantly outperforms all existing techniques.

6. Conclusion

In this novel hascode-based duplication detection technique we opted for sentence-level segmentation while performing a hash key comparison to identify duplicate documents, to minimize duplication more effectively on web. With the help of sentence level segmentation, even minor change within a document results in a distinct hash code at sentence level only while keeping the hash code of other sentences the same thus categorizing it as a duplicate of another document. Additionally,

 doi:10.20944/preprints202408.0443.v1

sentence-level segmentation allows for natural occurrences of identical sentences such as references and quotes from other papers can get tolerance during duplication detection.

We chose to avoid word segmentation because our objective is document duplication detection. In documents from same domain there can be many similar words and keywords, which definitely don't make them the same documents. By focusing on the entire document and utilizing sentence-level segmentation, duplication can be significantly reduced.

7. Future Work and Direction

This algorithm can be applied at a larger scale, allowing for fine-tuning of the model based on the acquired results. The supplementary datasets may also be made available as it holds significant importance for duplication detection. Additionally, accurate ground truth is very essential for the researchers to enhance the accuracy of duplication detection. Furthermore, by leveraging our model, we can utilize existing datasets and potentially offer comprehensive ground truth in the future.

Author Contributions: SE proposed the methodology, implemented it and prepared the first draft of the article. AN and MT refined the proposed methodology, supervised and improved the write-up of the article. AA and SN evaluated and refined the article.

Funding: Not applicable.

Data Availability Statement: The datasets used in this research, are publicly available.

Conflicts of Interest: There is no competing interests between the authors.

References

1. Pamulaparty, L.; Rao, C.G.; Rao, M.S. A near-duplicate detection algorithm to facilitate document clustering. *International Journal of Data Mining & Knowledge Management Process* **2014**, *4*, 39.
2. Kumar, J.P.; Govindarajulu, P. Near-duplicate web page detection: an efficient approach using clustering, sentence feature and fingerprinting. *International Journal of Computational Intelligence Systems* **2013**, *6*, 1–13.
3. Naseer, A.; Tamoor, M.; Azhar, A. Computer-aided COVID-19 diagnosis and a comparison of deep learners using augmented CXRs. *Journal of X-ray Science and Technology* **2022**, *30*, 89–109.
4. Tamoor, M.; Younas, I. Automatic segmentation of medical images using a novel Harris Hawk optimization method and an active contour model. *Journal of X-ray Science and Technology* **2021**, *29*, 721–739.
5. Pokorny, J. Web searching and information retrieval. *Computing in Science & Engineering* **2004**, *6*, 43–48.
6. Chughtai, I.T.; Naseer, A.; Tamoor, M.; Asif, S.; Jabbar, M.; Shahid, R. Content based image retrieval via transfer learning. *Journal of Intelligent & Fuzzy Systems* **2023**, *44*, 8193–8218.
7. Naseer, A.; Zafar, K. Comparative analysis of raw images and meta feature based Urdu OCR using CNN and LSTM. *International Journal of Advanced Computer Science and Applications* **2018**, *9*.
8. Naseer, A.; Zafar, K. Meta features-based scale invariant OCR decision making using LSTM-RNN. *Computational and Mathematical Organization Theory* **2019**, *25*, 165–183.
9. Naseer, A.; Hussain, S.; Zafar, K.; Khan, A. A novel normal to tangent line (NTL) algorithm for scale invariant feature extraction for Urdu OCR. *International Journal on Document Analysis and Recognition (IJDAR)* **2022**, *25*, 51–66.
10. Nasreen, G.; Haneef, K.; Tamoor, M.; Irshad, A. A comparative study of state-of-the-art skin image segmentation techniques with CNN. *Multimedia Tools and Applications* **2023**, *82*, 10921–10942.
11. Wali, A.; Ahmad, M.; Naseer, A.; Tamoor, M.; Gilani, S. Stynmedgan: medical images augmentation using a new GAN model for improved diagnosis of diseases. *Journal of Intelligent & Fuzzy Systems* **2023**, *44*, 10027–10044.
12. Rafiei, D.; Mendelzon, A.O. What is this page known for? Computing web page reputations. *Computer Networks* **2000**, *33*, 823–835.
13. Cohen, W.W.; McCallum, A.; Quass, D. Learning to understand the web. *IEEE Data Eng. Bull.* **2000**, *23*, 17–24.
14. Naseer, A.; Zafar, K. Meta-feature based few-shot Siamese learning for Urdu optical character recognition. *Computational Intelligence* **2022**, *38*, 1707–1727.

15. Cabanac, G.; Chevalier, M.; Chrisment, C.; Julien, C.; Soulé-Dupuy, C.; Tchienehom, P.L. Web information retrieval: Towards social information search assistants. In *Social information technology: Connecting society and cultural issues*; IGI Global, 2008; pp. 218–252.
16. Chuklin, A.; Markov, I.; De Rijke, M. *Click models for web search*; Springer Nature, 2022.
17. Cambazoglu, B.B.; Baeza-Yates, R. *Scalability challenges in web search engines*; Springer Nature, 2022.
18. Yu, B. Research on information retrieval model based on ontology. *EURASIP Journal on Wireless Communications and Networking* **2019**, *2019*, 1–8.
19. Lawrence, S. Context in web search. *IEEE Data Eng. Bull.* **2000**, *23*, 25–32.
20. Deo, A.; Gangrade, J.; Gangrade, S. A PSO Algorithm Based Web Page Retrieval System. *Proceedings of Recent Advances in Interdisciplinary Trends in Engineering & Applications (RAITEA)* **2019**.
21. Gupta, Y.; Saini, A.; Saxena, A.; Sharan, A. Fuzzy logic based similarity measure for information retrieval system performance improvement. International Conference on Distributed Computing and Internet Technology. Springer, 2014, pp. 224–232.
22. Kobayashi, M.; Takeda, K. Informational Retrieval on the Web. *IBM Japan* **2000**, *47*.
23. Arora, M.; Kanjilal, U.; Varshney, D. Challenges in Web Information Retrieval. In *Innovations in Computing Sciences and Software Engineering*; Springer, 2010; pp. 141–146.
24. Garg, D.; Sharma, D. Information Retrieval on the Web and its Evaluation. *International Journal of Computer Applications* **2012**, *40*, 26–31.
25. Tam, V.W.; Shepherd, J. Webpage relationships for information retrieval within a structured domain. Proceedings of the 21st ACM conference on Hypertext and hypermedia, 2010, pp. 307–308.
26. Henzinger, M.R.; others. Link analysis in web information retrieval. *IEEE Data Eng. Bull.* **2000**, *23*, 3–8.
27. Hearst, M.A. Next generation web search: Setting our sites. *IEEE Data Eng. Bull.* **2000**, *23*, 38–48.
28. Brusilovsky, P.; Tasso, C. Preface to special issue on user modeling for web information retrieval. *User Modeling and User-Adapted Interaction* **2004**, *14*, 147–157.
29. Xu, C. Research on Information Retrieval Algorithm Based on TextRank. 2019 34rd Youth Academic Annual Conference of Chinese Association of Automation (YAC). IEEE, 2019, pp. 180–183.
30. Qian, L.; Yu, J.; Zhu, G.; Mei, F.; Lu, W.; Ge, B.; Wang, L.; Mei, Z.; Pang, H.; Xu, M.; others. A Duplication Reduction Approach for Unstructured Data Using Machine Learning Method. 2019 International Conference on Intelligent Computing, Automation and Systems (ICICAS). IEEE, 2019, pp. 515–519.
31. Yandrapally, R.; Stocco, A.; Mesbah, A. Near-duplicate detection in web app model inference. Proceedings of the ACM/IEEE 42nd international conference on software engineering, 2020, pp. 186–197.
32. Fröbe, M.; Bevendorff, J.; Gienapp, L.; Völske, M.; Stein, B.; Potthast, M.; Hagen, M. CopyCat: Near-Duplicates within and between the ClueWeb and the Common Crawl. Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2021, pp. 2398–2404.
33. Chen, G.; Chen, G.; Wu, D.; Liu, Q.; Zhang, L.; Fan, X. An improved Simhash algorithm based malicious mirror website detection method. Journal of Physics: Conference Series. IOP Publishing, 2021, Vol. 1971, p. 012067.
34. Seshasai, S. Efficient near duplicate document detection for specialized corpora. PhD thesis, Massachusetts Institute of Technology, 2009.
35. Rodier, S.; Carter, D. Online near-duplicate detection of news articles. Proceedings of The 12th Language Resources and Evaluation Conference, 2020, pp. 1242–1249.
36. Burk, H.; Javed, F.; Balaji, J. Apollo: Near-duplicate detection for job ads in the online recruitment domain. 2017 IEEE International Conference on Data Mining Workshops (ICDMW). IEEE, 2017, pp. 177–182.
37. empty. Reuters-21578 Dataset, empty.
38. Feng, W.; Deng, D. Allign: Aligning all-pair near-duplicate passages in long texts. Proceedings of the 2021 International Conference on Management of Data, 2021, pp. 541–553.