Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 48 (2015) 228 – 235

Procedia Computer Science

International Conference on Intelligent Computing, Communication & Convergence

(ICCC-2015)

Conference Organized by Interscience Institute of Management and Technology,

Bhubaneswar, Odisha, India

XNDDF: Towards a Framework for Flexible Near-Duplicate Document Detection Using Supervised and Unsupervised Learning

Lavanya Pamulaparty[a], Dr. C.V Guru Rao[b],Dr. M. Sreenivasa Rao[c]

[a]*Department of CSE, Methodist college of Engg. & Tech., OU, Hyderabad*
[b]*Department of CSE, S R Engineering College, JNT University, Warangal*
[c]*Department of CSE, School of IT, JNT University, Hyderabad*

Abstract

The WWW has witnessed the exponential growth of web documents. People of all walks of life depend on the electronic superhighway, Internet, for retrieving information. Search engines retrieve data. Detecting near duplicate documents and handling them can help search engines to improve performance. In this paper, we proposed two algorithms. The first algorithm is meant for unsupervised probabilistic clustering of documents while the second algorithm is to detect near duplicates that can handle in offline processing of search engines. The clustered documents can avoid unnecessary comparisons while near duplicate detection algorithm involve local feature selection in are given document based on weights assigned to terms. A classifier is built to have supervised learning for discriminating documents. We proposed a framework named eXtensible Near Duplicate Detection Framework (XNDDF) which provides various components that provide room for flexible duplicate detection solutions besides showing offline and online processing required by a search engine. Our future work is to implement the framework components through a prototype application.

Keywords: Web search;document clustering; near duplicate detection;offline processing.

1. Introduction

The World Wide Web became increasingly popular, and it is the universal database of knowledge. Digital

Peer-review under responsibility of scientific committee of International Conference on Computer, Communication and Convergence (ICCC 2015)
doi:10.1016/j.procs.2015.04.175

documents are piled up into WWW in an exponential fashion. People across the world, in all walks of life, use general search engines and domain – specific search engines to collect the required information in a short time. The authors of the documents that contribute to the know-how possessed by WWW might indulge in slight modifications in their documents and again upload to WWW. Thus, the web has a great degree of near-duplicate documents. When two documents are totally identical, they are called duplicates. Search engines crawl tens of billions of documents besides indexing them [2], [6]. Near – duplicate documents are one of the issues of search engines [22]. However, near-duplicate documents are different. If a document is the duplicate of another document with small modifications, those documents are said to be near-duplicates [30]. The modification is made due to editing operations such as insert, update or delete. Due to the provision for editing provided by word processing and other tools near-duplicate documents are prevailing in WWW [31]. It is easier to detect duplicate document but difficult to detect near-duplicate documents [32], [33]. Following are two sentences with near-duplicate characteristics.

1. People with technical **knowledge** can become successful entrepreneurs in the era of global economic reforms in the wake of globalization.
2. People with technical **know-how** can become successful entrepreneurs in the era of global economic reforms in the wake of globalization.

The highlighted word in each sentence reveals the little difference between the two sentences. In this paper, we proposed two algorithms for document clustering and near-duplicate detection respectively. The problem with existing approaches is that they obtain various representations for these two sentences. It leads to computational overhead besides presenting near duplicate documents to end users that cause wastage of time surfing such documents.

There are many approaches towards near-duplicate detection. Bag of words based on features of documents that are extracted using terms and term frequency concept, and similarity measures are employed to detect near-duplicate documents. Bag of words model can be used for document representation [34]. A set of documents denoted by D is represented as D= {d1, d2, d3,…, dn}, where d1, d2, d3 etc. are individual documents. Every document di(1<=i<=n) can be represented by a feature set denoted as fi = {fi1, fi2, …, fin} where fi1, fi2 etc. are features extracted from the document di. Anything that is associated with a given document is known as a feature [35], [36], [37], [38], and [39]. Important terms that appear in given document are taken into consideration for extracting features [40], [41]. A similar degree is computed to know how similar two documents are [42], [43]. Based on the user-given similarity threshold, it is possible to know the near-duplicates. However, it is difficult to determine the threshold through the program.

The contribution of this paper is as follows.

1. We proposed an algorithm or clustering documents prior to subjecting them to near-duplicate detection. Since the clustering can reduce the scope of comparison of documents, we followed unsupervised probabilistic clustering approach, keeping the needs of a search engine for offline processing in mind.
2. We proposed another algorithm that makes use of local features of a document through weighted terms and similarity measures for near-duplicate document detection. This algorithm takes the help of a classifier.
3. We train a classifier that computes discriminative function that is best used by the near-duplicate detection algorithm for expert decision making. The classifier takes training sets of documents that are provided by domain experts in order to compute discriminative function.
4. We proposed a framework named eXtensible Near Duplicate Detection Framework (XNDDF) which makes room for flexible incorporation of clustering algorithms, near-duplicate detection algorithms and classifiers so as to make the prototype application to be extensible and "future proof".

The remainder of the paper is structured as follows. Section II reviews the literature on prior works pertaining to near duplicate detection. Section III presents the proposed approach towards near duplicate detection and introduces two algorithms for document clustering and near-duplicate detection respectively. Section IV presents the framework that has flexible provisions for future proof enhancements for unsupervised learning methods for clustering, supervised learning for classifiers and state of the art near-duplicate detection algorithms. Section V concludes the paper besides providing directions for future work.

2. Related Works

This section provides reviews of the literature on prior works. Duplicate and near – duplicate documents occur due to many reasons such as plagiarism, mirror sites, spam, versions and copies [7]. Many researchers contributed to detecting near-duplicate documents. Raveena and Nandini [15] applied supervised learning and document level features for reducing Page Rank for replicas. They used both syntactic and semantic approaches to compare documents. DOM tree was used to navigate to parts of the document and compare them. Arunprasath and Paul [1] presented a new method of crawling Internet forums. Similar kind of solution was explored in [3], [10], [16], [25], and [29]. Umamahoro and Zuping [28] proposed an algorithm for near-duplicate document detection based on word position. Joshi and Gadge [4] proposed an algorithm named NDupDet for detecting near-duplicates in web pages.

Naseer et al. [19] proposed an approach that uses page size for detecting duplicate query results. Srivatsava and Li [18] proposed hashing algorithm used in search based solutions. Ray and Kushwaha [27] proposed NLP and text mining for detecting duplicate content over web pages. Ho and Kim [20] presented a solution to near-duplicate detection problem using fingerprint feature of biometrics. Singh [26] presented an intelligent mechanism for redundant result removal. Santhiya and Preethi [9] presented an approach using ontologies for duplicate content detection. Similar work is found in [17]. Eyk and Leeuwen [21] focused on exploring the performance evaluation of near-duplicate detection algorithms for Crawljax. Fraudulent replica detection was explored in [8]. Williams et al. [13] explored web services for information extraction and resolve issues related to it. He et al. [23] explored the concept of crawling entity pages of deep web. Various crawling algorithms for web were explored in [5], [11]. Field Programmable Gate Arrays are used in [24] for near-duplicate document detection. Williams and Giles [12] explored near-duplicate detection using academic digital library.

Manning et al. [45] considered a document as a series of strings. Each string can have k words and known as k-gram. They considered such k-gram to represent characteristics of a given document. This approach resulted in a huge feature set. Li et al. [46] later overcame this problem by skipping some words. Theobald et al. [47] used stop words concept to solve this problem. The popular stop word list is presented in Table 1. Wang and Chang [48] considered a document as a series of k-grams. In each sentence, the unions of such k-grams are considered to be feature. For finding the similarity of two documents, the similarity function is used. For instance, consider f1= {f1.1, f1.2, …, f1.n} and f2= {f2.1, f2.2, …, f2.n} are feature sets of documents d1 and d2. The following are some of the popular similarity functions.

Jaccard Function

$$sim(d1,d2) = j(d1,d2) = f1 \cap f2 / f1 \cup f2 \quad -(4)$$

Cosine Function

$$sim(d1,d2) = C(d1,d2) = f1.f2 / \|f1\|.\|f2\| \quad -(5)$$

Euclidean Distance

$$sim(d1,d2) = Ec(d1,d2) = \sqrt{(f1-f2).(f1-f2)} \quad -(6)$$

Extended Jaccard Function

$$sim(d1,d2) = EJ(d1,d2) \frac{f1.f2}{f1.f1+f2.f2-f1.f2} \quad -(7)$$

Dice Function

$$sim(d1,d2) = D(d1,d2) = \frac{2f1.f2}{f1.f1+f2.f2} \quad -(8)$$

Conventionally threshold values are used in order to know whether two documents are near-duplicates. However, it is not the correct approach as it cannot be fixed. It needs to be determined by human experts. For this reason, in this paper, we use a learning classifier that takes training documents containing ground truth in order to know the discriminative function**.**

3. Proposed Algorithms for Clustering and Near Duplicate Detection

We believe that clustering of documents before near-duplicate detection can reduce computations pertaining to comparison. Moreover, we do this keeping a flexible framework that helps in offline and online processing of web documents in mind. If all documents are given for an algorithm for near-duplicate detection, comparison takes more time, causing unnecessary computations and cause scalability problems. We use probabilistic fuzzy cluster analysis [44] along with a Fuzzy C Means algorithm for clustering documents. The outline of the algorithm is as shown in Figure 1.

$$J_f = \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^{m} d_{ij}^{2} \quad \text{- (1)}$$

$$u_{ij} = \frac{1}{\sum_{i=1}^{c} \left(\frac{d_{ij}^{2}}{d_{ij}^{2}} \right)^{\frac{1}{m-1}}} = \frac{d_{ij}^{\frac{-1}{m-1}}}{\sum_{i=1}^{c} d_{ij}^{\frac{-1}{m-1}}} \quad \text{- (2)}$$

$$c_j = \frac{\sum_{j=1}^{n} u_{ij}^{m} x_j}{\sum_{j=1}^{n} u_{ij}^{m}} \quad \text{- (3)}$$

Table 1 – shows popular list of stop words [47]

a, about, above, after, again, against, all, am, an, and, any, are, aren't, as, at, be, because, been, before, being, below, between, both, but, by, can't, cannot, could, couldn't, did, didn't, do, does, doesn't, doing, don't, down, during, each, few, for, from, further, had, hadn't, has, hasn't, have, haven't, having, he, he'd, he'll, he's, her, here, here's, hers, herself, him, himself, his, how, how's, I, i'd, i'll, i'm, i've, if, in, into, is, isn't, it, it's, its, itself, let's, me, more, most, mustn't, my, myself, no, nor, not, of, off, on, once, only, or, other, ought, our, ours

The objective function, computation of membership weights, and deriving new cluster centers are performed using equations (1), (2) and (3) respectively.

Algorithm: Fuzzy C means for Document Clustering **Inputs**: Set of documents to be clustered and the number of clusters **Outputs**: Clusters of given documents
1. Start 2. Randomly generate cluster centers (n) 3. Repeat For each object computes Membership Weights (2) Recompute the new centers 4. Until no changes in the cluster are observed (for i = n) 5. End

Fig. 1. The Proposed algorithm for document clustering

The algorithm takes a set of documents and partitions them into clusters. The clusters can be used later for near – duplicate document detection. The reason behind this is that similar documents are grouped together, and that is a necessity for near – duplicate document detection.

Algorithm: Near Duplicate Detection (NDD) Algorithm
Inputs: Set of training document sets (D), Set of Testing Documents (TD)
Outputs: Near duplicate document detection

Training Phase:

1. Start
2. Calculate the Feature Sets
3. Derive Training Patterns
4. Build a Classifier
5. Compute Discriminative Function F (x)
6. End

Testing Phase:

1. Start
2. Initialize the value for i = 1;
3. Initialize the len to size of TD
4. While i>len

Calculate feature set of td of i^{th} and $i+1^{th}$ position of TD

Derive similarity vector V

If f(V)>=0 THEN

Detect td at $i+1^{th}$ position as near-duplicate and remove from TD

ELSE

Consider the document as non-near duplicate

END if

i = i + 1

5. End While
6. End

Fig. 2. Algorithm for Near Duplicate Detection

There are two phases in the algorithm, namely training phase and testing phase. Discussion on the need for the training phase is appropriate here. Why a threshold for similarity measure cannot give satisfactory results? Why, because the threshold is a relative value which cannot be fixed for a program. It is assumed and believed that only human experts can provide accurate information with respect to near-duplicate documents and make some ground-truth documents. These ground-truth documents can be used in the training phase. Of given documents, the training phase computes features sets that represent local characteristics of the documents. From the feature sets, training patterns are derived. These patterns are used by a classifier that learns and computes a discriminative function which is further used by the detection phase. In the detection phase, two documents are considered, and feature sets are computed. From the feature sets, similarity vector is created. Based on the discriminative function, the similarity vector is verified, and a decision is made whether the document is a near duplicate to the one to which it was compared.

4. Extensible Near Duplicate Detection Frame Work (XNDDF)

This section describes the proposed framework named eXtensible Near Duplicate Detection Framework (XNDDF). Our research in the future continues with this framework and a prototype application that demonstrates the proof of concepts of our research. The proposed framework is extensible and flexible. It has both offline operations and online operations. The inclusion of both offline and online operations provide greater opportunity to evaluate our research objective to explore near-duplicate document detection with fine-grained control over the variables. The XNDDF is said to be flexible as it provides placeholders for accommodating various kinds of clustering algorithms, different classifiers, and different near-duplicate detection algorithms. As seen in Fig.3, it is evident that the near- duplicate detection is taking place offline. This is the requirement of a search engine. However, for completeness and to have control over the experimental environment, we preferred to have offline operations such as clustering, learning classifier and near-duplicate document detection. The functionality of the framework is described here.

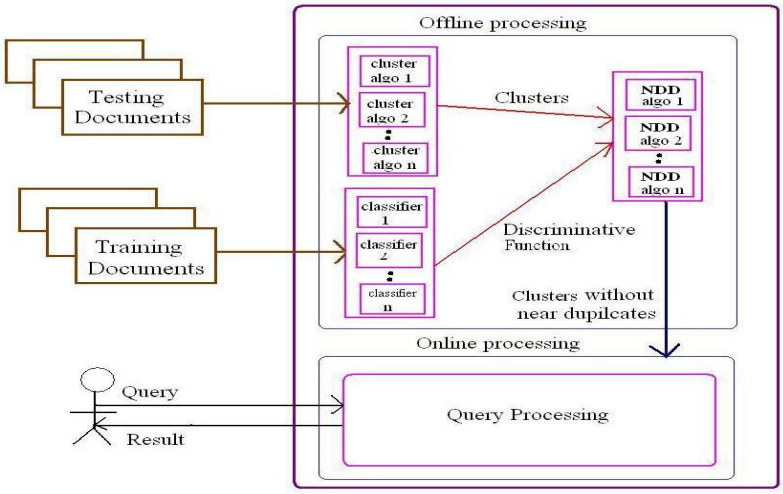

Fig. 3 –Overall functional diagram of XNDDF

First of all, the training document sets are given to the framework that uses one of the classifiers to compute discriminative function. The discriminative function is given to near-duplicate detection module which applies one of the algorithms to find near-duplicate documents. The algorithm operates on a particular cluster so as to reduce computational complexity. The clusters are provided by the clustering module which takes testing documents as input and generate clusters based on the similarity function as per the fuzzy C means algorithm described in Fig.1. The result of the offline process is the clusters of documents without near-duplicates. These documents are taken as input in the online processing which starts working when users online make queries. On making queries, only the online process starts, and the results of the query will have quality documents without near-duplicate documents.

5. Prototype Implementation

This section describes the implementation details of the proposed framework. Since the implementation is in progress, we would like to describe some of the details in this paper. We implementation of clustering algorithm, classifier, and Near Duplicate Detection algorithm is in progress. Many sets of documents are used for testing the effectiveness of the proposed algorithms. Pre-processing of documents with respect to stop words, stemming and so on are completed. The work is under way with datasets such as Enron, RCV1, Reuters – 21578, and Keepmedia. Various similarity measures such as Euclidean and Extended Jaccard are used to ascertain the similarity between documents for clustering them. Our aim of clustering documents is to make use of the clusters to detect near-duplicate documents. Keeping this in mind reusable components is being implemented. Since the similarity measure plays an important role in the clustering process we are likely to experiment with multiple such measures to incorporate into the XNDDF.

We are planning to build the algorithms and framework components in such a way that they are flexible and extendable besides facilitating new algorithms. Since the similarity measures are used in multiple algorithms and the framework is intended to be flexible, we are building the prototype system that can allow users to choose from various combinations for comprehensive testing of the system that is aimed at detecting near duplicate documents. In the first phase we are focusing on proposed algorithms in this paper along with a classifier using both unsupervised and supervised approaches respectively. Since the near-duplicate detection assumes more importance in the real world and used in search engines in general, we incorporated both offline and online processing in the prototype system. We intend to describe our work done towards algorithms and the prototype system in future research papers. However, the main interface of the tool is as presented in Fig.4.

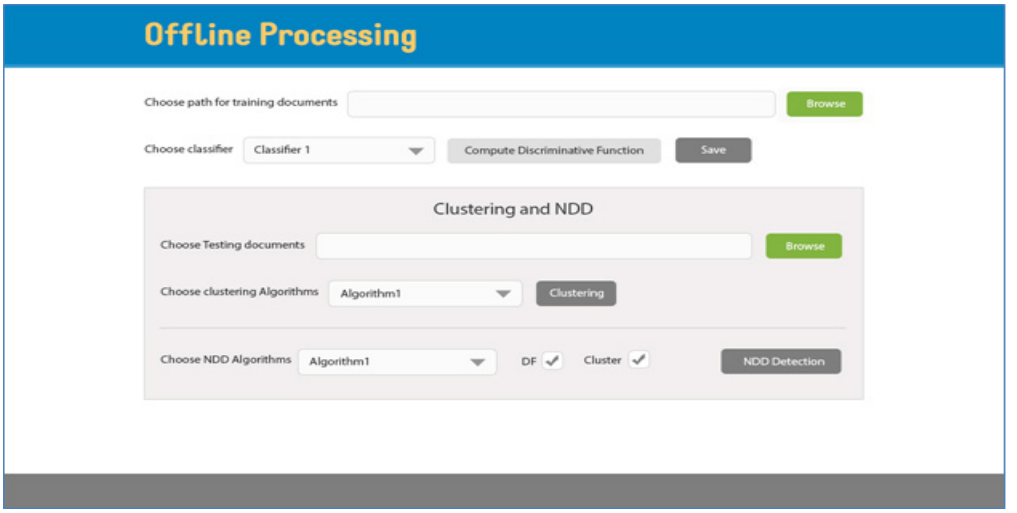

Fig. 4. UI of offline processing of the proposed tool

The tool which is under development has provision for underlying algorithms that can be dynamically incorporated into the tool through administrative functions to make the tool flexible and extensible. The tool facilitates selection of training and testing documents, selection of clustering, classifier and NDD algorithms prior to NDD detection.

6. Conclusions and Future Work

In this paper, we studied the problem of near-duplicate detection encountered by general search engines like Google, Yahoo and Bing and domain specific search engines like MedLinePlus and ArchiveGrid. Detecting duplicate documents is easier while detecting the near-duplicate documents is a difficult problem. Search engines may take many representations of near-duplicate documents. This will increase the computational cost due to an unnecessary process of such documents besides causing wastage of time to end users. Many techniques came into existence to solve this problem. Bag of words, similarity measures are used to detect near duplicate documents. In this paper, we proposed two algorithms. First one is a clustering algorithm that groups relevant documents. We believed that this step is essential to eliminate unnecessary comparison of all documents in near-duplicate detection. Fuzzy C Means clustering is used to achieve clustering. Second one is the near-duplicate detection algorithm which has two phases, namely training phase and testing phase. This algorithm exploits the local features of documents computed through weighted terms besides using discriminative function computed by the classifier. We proposed a framework named eXtensible Near Duplicate Detection Framework (XNDDF) which is flexible and future proof as it provides placeholders for clustering algorithms, classifiers, and near-duplicate detection algorithms. In the future, we implement XNDDF and our algorithms with an empirical study on a flexible framework with a prototype application.

References

1. T.K. Arunprasath, Dr. C. Kumar Charlie Paul. (2014). FOCUS: ADAPTING TO CRAWL INTERNET FORUMS. IEEE. 3(1), p83-87
2. Ronny Lempel Yahoo Labs, Haifa. (2013). Introduction to Search Engine Technology. -. 0 (0), p1-31.
3. M.Maheswari, N.Tharminie. (2014). Crawler with Search Engine based Simple Web Application System for Forum Mining. IEEE. 16 (2), p79-82.
4. Nilakshi Joshi and Jayant Gadge. (2013). Near Duplicate Web Page Detection using NDupDet Algorithm. IEEE. 61 (4), p56-59.
5. Roland Schäfer, Adrien Barbaresi and Felix Bildhauer. (0). FocusedWeb Corpus Crawling. -. 0 (0), p1-7.
6. Ricardo Baeza-Yates" B. Barla Cambazoglu. (2014). Scalability and Efficiency Challenges in Large-Scale Web Search Engines, p1-90.
7. Tao Yang. (2013). Topic: Duplicate Detection and Similarity Computing. -. 0 (0), p1-9.
8. Balachander Krishnamurthy, New York, NY; Greg Minshall, Los Altos, (2014). United States Patent. -. 0 (0), p1-16.
9. J.Santhiya and M.Preethi. (2014). MULTI KEYWORD WEB CRAWLING USING ONTOLOGY IN WEB FORUMS. -. 2 (2), p98-102.

10. P.Senthil, S.Pothumani, and T.Nalini. (2013). Automated Path Ascend Locomotion – A Prescribed Topical Crawl Method. -. 3 (2), p359-363
11. Crista Lopes, UCI. (0). Crawling the Web. -. 0 (0), p1-25.
12. Kyle Williams, C. Lee Gilesyz. (0). Near Duplicate Detection in an Academic Digital Library.0 (0), p1-4.
13. Kyle Williams, Lichi Liy, Madian Khabsaz, Jian Wuy, Patrick C. Shihy and C. Lee Gilesyz. (0). A Web Service for Scholarly Big Data Information Extraction. -. 0 (0), p1-8.
14. Qing Zhang, David Y. Wang, Geoffrey M. Voelker. (0). DSpin: Detecting Automatically Spun Content on the Web. -. 0 (0), p1-16.
15. Raveena. S& Nandini.V. (2014). A Novel Algorithm to Reduce Page Rank For Near Replica's Using Document Level Features And Supervised Learning. -. 0 (0), p37-44.
16. T.Nagajothi and M.S.Thanabal. (2014). Automation of URL Discovery and Flattering Mechanism in Live Forum Threads. -. 2 (1), p535-541.
17. P.Jananipriya, Dr.P.Vivekanandan, Mrs.A.Anitha. (2014). MULTI KEYWORD WEB CRAWLING USING ONTOLOGY IN WEB FORUMS. -. 3 (1), p457-462
18. Anshumali Srivastava and Ping Li. (2014). Asymmetric LSH (ALSH) for Sublinear Time Maximum Inner Product Search (MIPS). , p1-9.
19. Oumair Naseer, Ayesha Naseer, Atif Ali Khan, Humza Naseer. (2013). Using Page Size for Controlling Duplicate Query Results in Semantic Web. -. 4 (2), p49-56.
20. Phuc-Tran Ho and Sung-Ryul Kim. (2014). Fingerprint-Based Near-Duplicate Document Detection with Applications to SNS Spam Detection. -. 0 (0), p1-9.
21. Erwin van Eyk and Wilco van Leeuwen. (2014). Performance of near-duplicate detection algorithms for Crawljax. -. 0 (0), p1-68.
22. Rahul Mahajan, Dr. S.K. Gupta, Mr. Rajeev Bedi. (2013) Challenges and Design Issues in Search Engine and Web Crawler, p42-44.
23. Yeye He, Dong Xin and Venkatesh Ganti. (0). Crawling Deep Web Entity Pages. -. 0 (0), p1-10.
24. Efficient Near -Public ate Document Detection using FPGAs. (0). Xiluo, walidnajjar, and vagelis hristidis. -. (0), p1-8.
25. K.Sandhya, M.Aruna. (2013). DISCOVERY OF URL PROTOTYPES INTENDED FOR WEB PAGE DEDUPLICATION. (6), p1301-1306.
26. Aarti Singh and Rajeev Soni. (2014). Intelligent Mechanism for Redundant Result Removal from Search Results. -. 3 (1), p10-18.
27. Ashish Kumar Ray, Mr. Ajay Kushwaha. (2014). Quality based Web information extraction approach using NLP and Text Mining. -. 3 (6), p7250-7254.
28. Gaudence Uwamahoro and Zhang Zuping. (2013). Efficient Algorithm for Near Duplicate Documents Detection. -. 10 (2), p12-18.
29. M.VijayaLakshmi and Mr.P.Senthil Kumar. (2014). URL Mining Using Web Crawler in Online Based Content Retrieval - 2 (2), p1-7.
30. Broder, A. Z. (2000). Identifying and filtering near-duplicate documents. In: Proceedings of the 11th annual symposium on combinatorial pattern matching, p1–10.
31. Yang, H., & Callan, J. (2005). Near-duplicate detection for eRulemaking. In: Proceedings of the national conference on digital government research, p.78–86.
32. Sood, S. & Loguinov, D. (2011). Probabilistic near-duplicate detection using simhash. In: Proceedings of the 20th ACM international conference on information and knowledge management, p. 1117–1126.
33. Jiang, Q., & Sun, M. (2011). Semi-supervised SimHash for efficient document similarity search. In: Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies, Vol. 1, p. 93–101.
34. Bag of words. (2012). <http://en.wikipedia.org/wiki/bag_of_words_model>.
35. Arasu, A., Ganti, V., & Kaushik, R. (2006). Efficient exact set-similarity joins. In: Proceedings of the 32nd international conference on very large databases, p. 918–929.
36. Fagin, R., Kumar, R., & Sivakumar, D. (2003). Efficient similarity search and classification via rank aggregation. In: Proceedings of the ACM SIGMODinternational conference on management of data, p. 301–312.
37. Gong, C., Huang, Y., Cheng, X., & Bai, S. (2008). Detecting near-duplicates in large scaleshort text databases. In: Proceedings of the 12th Pacific-Asia conference on advances in knowledge discovery and data mining, p. 877–883.
38. Huffman, S., Lehman, A., Stolboushkin, A., Wong-Toi, H., Yang, F., & Roehrig, H. (2007). Multiple-signal duplicate detection for search evaluation. In: Proceedings of the 30th annual international ACM SIGIR conference on research and development in IR, p. 223–230.
39. Qiu, J. & Zeng, Q. (2010). Detection and optimized disposal of near-duplicate pages. In: Proceedings of the second international conference on future computer and Communication, Vol. 2, p. 604–607.
40. Han, L., Finin, T., McNamee, P., Joshi, A., & Yesha, Y. (2012). Improving word similarity by augmenting PMI with estimates of the word polysemy. IEEE Transactions on Knowledge and Data Engineering. 10.1109/TKDE.2012.3
41. Kim, J., & Lee, H. (2012). Efficient exact similarity searches using multiple token orderings. In: Proceedings of the IEEE 28th international conference on data engineering, p. 822–833.
42. Bayardo, R. J., Ma, Y., & Srikant, R. (2007). Scaling up all pair's similarity searches. In: Proceedings of the 16th international conference on WWW, p. 131–140.
43. Huang, L., Wang, L., & Li, X. (2008). Achieving both high precision and high recall in near-duplicate detection. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, p. 63–72.
44. Matjaž Juršič, Nada Lavrač, FUZZY CLUSTERING OF DOCUMENTS, Solvenia, 2008, p1-4.
45. Manning, C. D., Raghavan, P., & Schutze, H. (2008). Introduction to information retrieval. Cambridge University Press.
46. Li, C., Wang, B., & Yang, X. (2007). Vgram: Improving performance of approximate queries on string collections using variable-length grams. In: Proceedings of the 33rd international conference on very large data bases, p. 303–314.
47. Theobald, M., Siddharth, J., & Paepcke, A. (2008). Spotsigs: Robust and efficient near duplicate detection in large web collections. In: Proceedings of the 31st annual international ACM SIGIR conference on research and development in information retrieval, p. 563–570.
48. Wang, J. -H. & Chang, H. -C. (2009). Exploiting sentence-level features for near-duplicate document detection. In: Proceedings of the 5th Asia information retrieval symposium on information retrieval technology, p. 205–217.