Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 4, Issue. 6, June 2015, pg.962 – 965

RESEARCH ARTICLE

Finding Near Duplicates using Categorical Approach

Janhvi Japee[1], Prof. Sumita Menaria[2]
[1]PIET, Waghodiya
[2]PIET, Waghodiya
[1st] jini.janu@gmail.com, [2nd] sumitra.menaria@gmail.com

Abstract--- The huge data on internet have the existence of near duplicate web pages plays a very important role in search engine. The web mining faces huge problem due to the existence of near duplicates. The web page increases the storage space and serving cost. By introducing efficient methods to detect and remove such documents from the web not only decrease the computation time but also increase the relevancy to search result. The aim is to find near duplicates web pages using categorical data and apply clustering algorithm to find categorical data. And to finding near duplicates fingerprinting sim hash algorithm is used to finding near duplicates. By apply these algorithm improve of efficiency and accuracy of web pages.

Keywords--- Near duplicates, Web mining, Web pages, Categorical data

I. INTRODUCTION

Information on the Web is very huge in size. There is a need to use this big volume of information efficiently for effectively satisfying the information need of the user on the Web. Search engines become the major breakthrough on the web for retrieving the information. Search engines are critically important to help users find relevant information on the Web. Search engines in response to a user's query typically produces the list of documents ranked according to closest to the user's request. Filtering the search engines' results consumes the users' effort and time especially when a lot of near duplicate. Web search engines considerable problems due to duplicate and near duplicate web pages. These pages increase the space required to store the index, either decelerate or amplify the cost of serving results and so exasperate users. The identification of similar or near-duplicate document in a large collection is a significant problem with wide-spread applications.

II. PROBLEM STATEMENT

Search engine are most frequent used approach to find information across the web, the information that is search faces the problem of near duplicates, similar join, plagiarism etc. the near duplicates contents are the major issues that are face by search engines. To overcome the problems main techniques have been invented to find the near duplicates, but still the issues are not completely solve. There has to be some advance approach or techniques to efficient find and remove the near duplicates from the web.

III. RELATED WORK

The method based on shingles and the signature method when compared, the signature method in the presence of inverted index was more efficient. As a result, the above stated syntactic approaches carry out only a text based comparison. And these approaches did not involve the URLs or any link structure techniques in identification of near-duplicates.

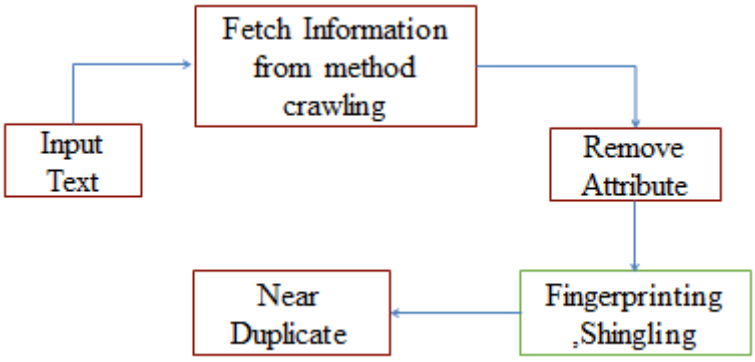

Fig 1. Existing System

IV. CACTUS (CATEGORICAL CLUSTERING USING SUMMARIES) ALGORITHM

Data summary (inter- & intra- attribute summary) is sufficient enough to find candidate clusters which can then be validated.

A three-phase clustering algorithm: Summarisation, Clustering, Validation

Summarisation Phase: Assumption: the inter- & intra- attribute summary of any pair of attributes fits easily into main memory.

Inter-attribute Summaries:

– Use a counter set to 0 initially for each pair (ai,aj) ϵ Di x Dj.
– Scan the dataset, increment the counter for each pair.
– After the scan, compute σ*D (ai,aj) and reset the counters of those whose σ* < E[σD (ai,aj)]. Store those values pairs.

Intra-attribute Summaries:

– Scan the dataset and find those tuples (T1,T2) of one domain such that T1.a is strongly connected with T1.b and T2.a is strongly connected with T2.b.
– Very fast operation, hence only compute them when needed

Clustering Phase: A two-step operation:

– Step 1. analyse each attribute to compute all cluster-projections on it
– Step 2. Synthesise candidate clusters on sets of attributes from the cluster-projections on individual attributes

Step1: Compute cluster-projections on attributes

– Step A. Find all cluster-projections on Ai of cluster over (Ai, Aj).
– Step B. Compute all the cluster-projections on Ai of cluster over {A1,…,An} by intersecting sets of cluster-projects from Step A.
– Step A is NP-Hard! Solution: use distinguishing sets.
• Distinguishing sets identify different cluster-projections.
• Construct distinguishing sets on Ai and extend w.r.t Aj some of the candidate distinguishing sets on Ai.
• Detailed steps are too long for this presentation, sorry!
– Step B: intersection of Cluster-projection
• Intersection joint S1ΠS2 = {s: there exist s1ϵS1 and s2ϵS2 such that s=s1Πs2 and |s|>1}
• Apply intersection joint to all sets of attribute values on Ai.

Step 2: Try to augment ck with a cluster projection ck+1 on attribute Ak+1. If new cluster <ci,ck+1> is a sub-cluster on (Ai,Ak+1), i ϵ {1,…,k}, then add ck+1 = <c1,…ck+1> to the final cluster.

Validation Phase: Use a required threshold to recognise false candidates which do not have enough support because some of the 2-clusters combined to form a candidate cluster may be due to different sets of tuples.

V. FINGERPRITING SIMHASH ALGORITHM

SimHash is a dimensionality reduction technique. It maps high-dimensional vectors to small-sized fingerprints are applied to web-pages as follows: we first convert a web-page into a set of features, each feature tagged with its weight. Features are computed using standard IR techniques like tokenization, case folding, stop-word removal, stemming and phrase detection. A set of weighted features constitutes a high-dimensional vector, with one dimension per unique feature in all documents taken together. With simhash, we can transform such a high-dimensional vector into an f-bit fingerprint where f is small, say 64.Given a set of features extracted from a document and their corresponding weights, we use simhash to generate an f-bit fingerprint as follows. We maintain an f-dimensional vector V, each of whose dimensions is initialized to zero. A feature is hashed into an f-bit hash value.

These f bits (unique to the feature) increment/decrement the f components of the vector by the weight of that feature as follows: if the i-th bit of the hash value is 1, the i-th component of V is incremented by the weight of that feature; if the i-th bit of the hash value is 0, the i-th component of V is decremented by the weight of that feature. When all features have been processed, some components of V are positive while others are negative. The signs of components determine the corresponding bits of the final fingerprint.

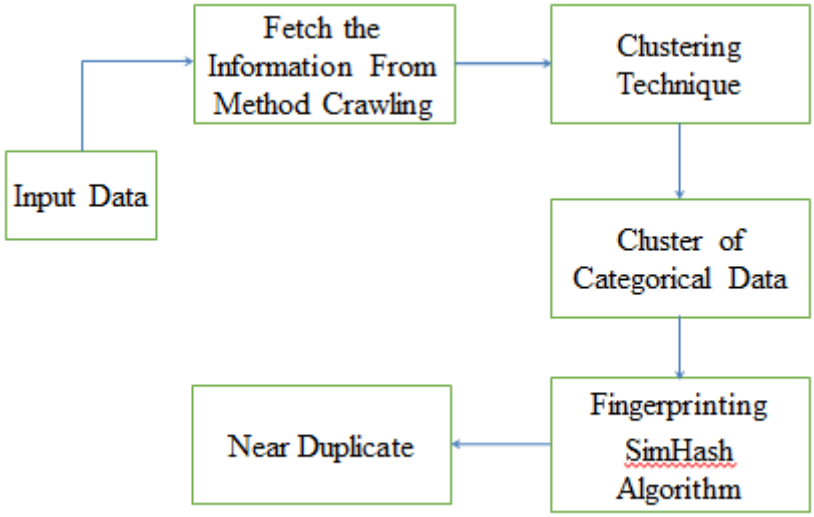

Fig 2. Proposed System

Implementation Methodology:

Step1. Input data from user.
Step2. Fetch the information from wen crawling.
Step3. Apply clustering technique for categorical data in which CACTUS algorithm is used.
Step4. Use of CACTUS algorithm we get data in the form of categories they made one Cluster.
Step5. Apply SimHash algorithm.
Step6. Finding near duplicates using SimHash algorithm.
Step7. Find the near duplicates with the method is SimHash algorithm.

VI. CONCLUSION

Search engine are most frequent used approach to find information across the web, the information that is search faces the problem of near duplicates, similar join, plagiarism etc. The near duplicates contents are the major issues that are face by search engines. To overcome the problems main techniques have been invented to find the near duplicates, but still the issues are not completely solved. There has to be some advance approach or techniques to efficient find and remove the near duplicates from the web. The future implement the proposed work and increase the efficiency and accuracy.

ACKNOWLEDGEMENT

For all the efforts behind the paper, I first & foremost would like to express my sincere appreciation to the staff of Dept. Information Technology for their extended help & suggestions at every stage of this paper. It is with a great sense of gratitude that I acknowledge the support, time to time suggestions and highly indebted to my guide Prof. Sumitra Menaria. Finally, I pay sincere thanks to all those who indirectly and directly helped me towards the successful completion of the paper.

REFERENCES

[1] Rasia Naseem, Sheena Anees, Muneer.K, Syed Farook.K, *Near Duplicate Web Page Detection With Analytic Feature Weighting* , IEEE International Conference Advances in Computing and Communiations 2013, pp. 324-327.
[2] Shine N. Das, Midhun Mathew, Pramod K.Vijayaraghavan, *An Efficient Approach For Finding Near Duplicate Web Pages Using Minimum Weight Overlapping Method*, IEEE International Conference on Information Technology 2012, pp. 121-126.
[3] Gurmeet Singh Manku, Arvind Jain, Anish Das Sarma, *Detecting Near Duplicates for Web Crawling*, ACM 2007, pp. 141-149.
[4] V.A.Narayana, P. Premchand, Dr. A. Govardhan, *A Novel and Efficient Approach For Near Duplicate Page Detection in Web Crawling*, IEEE International Advance Computing Conference IACC 2009, pp. 1492-1496.
[5] YuJuan Cao, ZhenDong Niu, WeiQiang Wang, Kun Zhao, *The Study on Detecting Near-Duplicate Web Pages*, , IEEE 2008 pp. 95-100.
[6] J. Prasanna Kumar, P. Govindarajulu, *Near Duplicate Web Page Detection : An Efficient Approach Using Clustering, Sentence Feature and Fingerprinting*, International Journal of Computational Intelligent Systems. Vol. 6, No. 1, January 2013, pp. 1-13.
[7] Martin Theobald, Jonathan, Siddharth Andreas Paepcke, *SpotSigs : Robust and Efficient Near Duplicate Detection in Large Web Collections*, ACM 2008.
[8] Dennis Fetterly, Mark Manasse, Marc Najork, *On the Evolution of Clusters of Near-Duplicate Web Pages*, IEEE 2003.
[9] V.A.Narayana, P. Premchand, Dr. A. Govardhan, *Near Duplicate Web Pages Detection : A Comparative Study of Two Contrary Approaches*, pp. 769-776.
[10] Arun K Pujari, *Data Mining Technique,* BOOK.

