MJRET
Open Access
Multidisciplinary Journal of Research in Engineering and Technology
Journal homepage: www.mjret.in
ISSN:2348 - 6953

ELIMINATE DUPLICATE URLs USING MULTIPLE ALIGNMENT OF SEQUENCES

Jayashri Waman, Prof. Pankaj Agarkar

Department of Computer Engineering
Dr.D.Y.Patil School of Engineering Pune, Maharashtra, India.

Abstract: *Duplicate content means search engines have to waste time in crawling all the different duplicate versions of a page, and you're relying on them to do it in the way you want them to. Duplicate content is the substantive block either in domain or across domains that either partially or completely matches with other contents. Mostly, this is not deceptive in origin. Use of such duplicated data is a waste of resources which results in poor user experiences. We focus on removing links of duplicate contents by address of the website i.e. URL. We will convert URL into multiple alignments of sequences and perform the operations.Also URL tokenizer is used to understand the web protocol and top -level domain. This approach will help in a healthy way to remove same content from a set of web pages. So web crawlers can easily accept this approach and can make better indexing possible.The proposed method reduced number of duplicate URLs than the existing approach*

Keywords: *Uniform Resource Locator (URL).*

1. INTRODUCTION

The URLs which are having similar content are called as DUST (Duplicate URLs with Similar Text). Syntactically these URLs are different but having similar content. For example, in order to facilitate the user's navigation, many web sites define links or alternative paths to access a document. In addition, webmasters usually mirror content to balance web request load and ensure fault tolerance. Other common reasons for the occurrence of duplicate content are the use of parameters placed in distinct positions in the URLs and the use of parameters that have no impact on the page content, such as the session_id attribute, used to identify a user accessing the content.

Detecting DUST is an extremely important task for search engines since crawling this redundant content leads to waste of resources such as Internet bandwidth and disk storage.

The DUST creates disturbance in results of link analysis algorithms and also results in poor user experience due to duplicate results. To resolve these problems, several authors have proposed methods for detecting and removing DUST from search engines. Initially efforts were focused on comparing document content to remove DUST, which was again a resource consuming process. However more recent studies proposed methods that inspect only the URLs without fetching the corresponding page content.

These methods, known as URL-based de-duping, mine crawl logs and use clusters of URLs referring to (near) duplicate content to learn normalization rules that transform duplicate URLs into a unified canonical form. This information can be then used by a web crawler to avoid fetching DUST, including ones that are found for the first time during the crawling. The main challenge for these methods is to derive general rules with a reasonable cost from the available training sets. As observed in [6], many methods derive rules from pairs of duplicate URLs. Thus the quality of these rules is affected by the criterion used to select these pairs and the availability of specific examples in the training sets. To avoid processing large numbers of URLs, most of the methods employ techniques such as random sampling or by looking for DUST only within sites, preventing the generation of rules involving multiple DNS names. Because of these issues, current methods are very susceptible to noise and, in many cases, derive rules that are very specific. Thus, an ideal method should learn general rules from few training examples, taking maximum advantage, without sacrificing the detection of DUST across different sites.

People use search engines for searching information. But retrieved documents contains a large volume of duplicate documents. Hence there is need to improve the search results. Data filtering algorithms used by some of search engines which eliminate duplicate and partial duplicate documents to save time and effort.

In this paper survey present an up-to-date review of the existing literature in duplicate and near duplicate detection in 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. These documents are presented to the user for examination and evaluation. Web users have to go through the long list and inspect the titles, and snippets sequentially to recognize the required results. Filtering the search engines' results consumes the users' effort and time especially when a lot of near duplicate. [2]

2. RELATED WORK

- **DustBuster [3]**

  DustBuster is a system proposed for finding decides that change an offered URL to different URLs that are liable to have comparative substance. DustBuster mines clean viably from past creep logs or web server logs, without looking at page details. However to check these principles, the strategy uses testing procedure which brings couple of genuine pages. It can reduce crawling overhead by up to 26% and thus increase crawl efficiency. It can also reduce indexing overhead.

- **Mining and learning URL Rewrite rules [4]**

  The proposed system uses a set of URLs divided into equivalence classes that are content based. The URLs having similar content are considered in the same class. The proposed system addresses the problem of mining URL set and learning URL rewrite rules which transforms all URLs of an equivalence class to the same canonical form. The proposed system is based on automatically generation of URL rewrite rules by mining a given collection of URLs with content-similarity information. It uses fixed set of delimiters; which are useful to study the effect of a more flexible tokenization. These rewrite rules can be useful to eradicate duplicates between URLs that are encountered for the initial time during web crawling, devoid of fetching their content. Such transformation rules can be expressed by a simple framework which is general enough to get the most ordinary URL rewrite patterns happening on the web. The proposed system, offer an algorithm for extracting and learning URL rephrase rules and show that under some assumptions, it is complete. The test results of proposed system shows reduction ratios of up to 60%. In future work, the system can be enhanced to capture a wider set of rules, while still being efficiently learnable.

- **URL based De-duping [5]**

  In this paper, authors proposed a set of methods to extract conventions from URLs and use these studied rules for de-duplication by just URL strings not including fetching the content clearly. The technique is made of extracting the crawl logs and using clusters of related pages to mine detailed rules from URLs which belongs to each cluster. It presents deep and basic tokenization of URLs to mine all possible tokens from URLs which are extracted by rule generation techniques for generating normalization rules. Thus, to reduce the number of pair-wise rules, the generated pair-wise rules are consumed by decision tree algorithm to generate precise generalized rules. Since preserving each rules for de-duplication is not enough due to the great number of exact rules, the proposed system presents a machine learning method to simplify the set of rules. These generalize set of rules minimizes the resource footprint to be functional at web-scale. The rule extraction methods are strong against website URL rules.

- **Pattern tree based URL normalization [6]**

  In this paper, a pattern tree-based approach is proposed to learning URL normalization rules. The pattern tree helps to leverage the statistical information from all the training samples to create the learning process further strong and reliable. First a pattern tree is constructed based on the training set, and then normalization rules are generated by identifying duplicate nodes on the pattern tree. The learning process is also accelerated as policy is dependent on pattern tree nodes. Thus, with the statistical information, the learning process is more robust to both the noise and incompleteness of the training samples. In addition, the pattern tree assist in selecting deployable rules by eradicating conflicts and redundancies. The computational cost is also low as rules are directly induced on patterns, rather than on every duplicate URL pair. A graph-based strategy has been proposed to select a subset of deployable rules for normalization. The system significantly reduces the running time of the learning process by around 50%.

As an improvement, the pattern tree construction algorithm should be further accelerated as it is the current computational bottleneck. Also the current algorithms fully trust the duplicate information labeled in the training set and auto-generated training sets may contain some fake duplicate information.

3. PROBLEMS IN EXISTING SYSTEMS

The existing systems are having some problems as given below:

I. To avoid processing large numbers of URLs, most of the existing methods employ techniques such as random sampling due to which these methods are very susceptible to noise.

II. Many existing methods derive rules from pairs of duplicate URLs. Thus the quality of these rules is affected by the criterion used to select these pairs and the availability of specific examples in the training sets.

III. Most existing methods generate very specific rules. So to increase detection of duplicate URLs more number of rules are required.

IV. Existing methods cannot detect DUST across different sites as candidate rules are derived from URL pairs within the dup-cluster.

V. Complexity of existing methods is proportional to the number of specific rules generated from all clusters, which can be infeasible in practice.

4. PROPOSED SYSTEM ARCHITECTURE

To overcome the limitations of existing methods of de-duping, this paper presents, a new method called as DUSTER, which obtains a smaller and more general set of normalization rules using multiple sequence alignment.

The proposed method is able to generate rules with an acceptable computational cost even when crawling in large scale scenarios. Also its complexity is proportional to the number of URLs to be aligned.

4.1. Multiple sequence alignment:

It is a tool to identify similarities and differences among strings/sequences. These similarities and differences can be explored to determine fixed and mutable substrings in URLs, which helps to derive normalization rules. As multiple sequence alignment methods find patterns involving all the available strings, the method is able to find more general rules and avoids problems related to pairwise rule generation, and the problem related to finding rules across sites. Thus, a full multi-sequence alignment of duplicate URLs, which is performed before rules are generated, can make the learning process more robust and less susceptible to noise.

4.2. Phases of DUSTER:

The DUSTER method is divided in two main phases as mentioned below,

Phase 1: Candidate rules generation

In this phase first, the multi-sequence alignment algorithm, align all the URLs in the dup-clusters and obtains consensus sequences for each dup-cluster. Then the candidate rules are generated from these sequences. For large clusters, a heuristic is used to ensure the efficiency of the method.

Phase 2: Validating candidate rules:

In this phase the candidate rules get filtered out according to their performance in a validation set. Figure 1 depicts the framework of DUSTER algorithm.

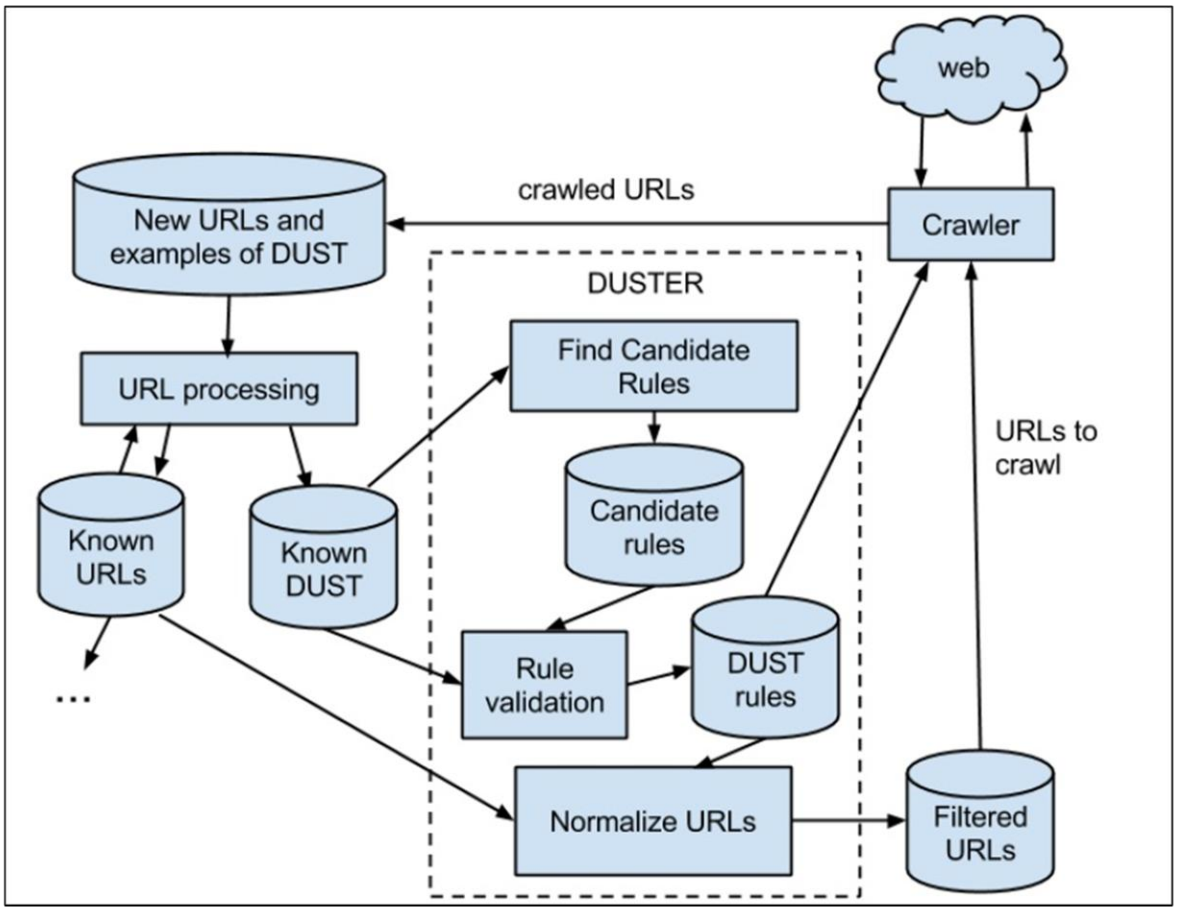

Fig.1: DUSTER Framework

As shown in above figure, when a new set of URLs are crawled, first it is merged with the already known URLs to form a new set of known URLs. These URLs are then processed to generate known URLs set and known DUST set. Also during crawling, the crawler is also able to identify examples of DUST by following canonical tags. As a result, a new set of known DUST is also available. Further this set can be enriched by processes such as those based on content signature, and manual inspection.

Once the final set of known DUST is available, DUSTER can use it to find and validate rules. It does it by splitting it in training and validating sets. The resulting rules are then used to normalize the known URLs which generates, a new reduced set of URLs to be crawled.

5. EXPECTED RESULT

The primary goal of this paper is to present an efficient method over existing methods to find DUST in URL set. The existing methods are compared with proposed method, based on the number of detected rules, the valid rules selected for detection and the performance of these methods in DUST detection with low false positive rate. The algorithm is tested on different URL sets to study the result.

6. CONCLUSION & FUTURE SCOPE

In this paper, authors presented a new method called as DUSTER, to address the DUST related problems. This method detects, distinct URLs that correspond to pages with duplicate or partial duplicate content. The method learns normalization rules that are very precise in converting distinct URLs which refer the same content to a common canonical form, which makes it easy to detect DUST. DUSTER uses an algorithm based on a full multi-sequence alignment of training URLs with duplicate content. It analyzes the alignments obtained and generates accurate and general normalization rules. The authors evaluated the method on two different sample sets and found a reduction in the number of duplicate URLs, generating gain of 82% and 140.74%.

As future work, the proposed system will be improved for scalability and precision. Also the approach which we have presented, prioritizes head traffic and we would like to explore the feasibility of rule generation for the rest of tail traffic. Clusters which are the ground truth for generating rules may include false positives due to the approximate similarity measures. We would like to explore ways of handling these in a robust fashion. Generalization is performed separately for source and target and we would like to explore the feasibility of generalizing both in an iterative fashion.

ACKNOWLEGEMENT

We thank our guide Prof. Pankaj Agarkar ME coordinator of Computer Engineering for his inspiration and guidance and also thank the Head of Computer Engineering Department Prof. Soumitra S. Das.

REFERENCES

[1]. "Url normalization for de-duplication of web pages" by A. Agarwal, H. S. Koppula, K. P. Leela, K. P. Chitrapura, S. Garg, P. K. GM.

[2]. "Near duplicate document detection survey" by B. S. Alsulami, M. F. Abulkhair, and F. E. Eassa.

[3]. "Do not crawl in the dust: Different urls with similar text" by Z. Bar-Yossef, I. Keidar, and U. Schonfeld.*

[4]. "De-duping urls via rewrite rules" by A. Dasgupta, R. Kumar, and A. Sasturkar.

[5]. "Learning url patterns for webpage deduplication" by H. S. Koppula, K. P. Leela, A. Agarwal, K. P. Chitrapura, S. Garg, and A. Sasturkar.

[6]. "A pattern tree-based approach to learning url normalization rules" by T. Lei, R. Cai, J.-M. Yang, Y. Ke, X. Fan, and L. Zhang.

[7]. "SpotSigs: robust and efficient near duplicate detection in large web collections" by M. Theobald, J. Siddharth, and A. Paepcke.