ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/228963799

Bridging the gap: From multi document template detection to single document content extraction

Article · January 2008

CITATIONS
14

READS
47

1 author:

Thomas Gottron
Universität Koblenz-Landau

72 PUBLICATIONS **649** CITATIONS

SEE PROFILE

All content following this page was uploaded by Thomas Gottron on 06 June 2017.

The user has requested enhancement of the downloaded file. All in-text references underlined in blue are added to the original document and are linked to publications on ResearchGate, letting you access and read them immediately.

BRIDGING THE GAP: FROM MULTI DOCUMENT TEMPLATE DETECTION TO SINGLE DOCUMENT CONTENT EXTRACTION

Thomas Gottron
Institut für Informatik
Johannes Gutenberg-Universität Mainz
55099 Mainz, Germany
email: gottron@uni-mainz.de

ABSTRACT

Template Detection algorithms use collections of web documents to determine the structure of a common underlying template. Content Extraction algorithms instead operate on a single document and use heuristics to determine the main content. In this paper we propose a way to combine the reliability and theoretic underpinning of the first world with the single document based approach of the latter. Starting from a single initial document we use the set of hyperlinked web pages to build the required training set for Template Detection automatically. By clustering the documents in this set according to their underlying templates we clean the training set from documents based on different templates. We confirm the applicability of the approach by using an entropy based Template Detection algorithm to build a Content Extractor.

KEY WORDS
Content Extraction, Template Detection, Template Clustering, Web Mining.

1 Introduction

Web Content Management Systems (WCMS) are becoming a more and more common solution for maintaining web sites. Accordingly, more and more of the documents on the web are based on template structures. Templates allow to combine contents from different sources and to compile them into the documents intended for presentation to the web users. Among the different incorporated contents is the article which forms the main content of the document, but as well commercials, references to other documents and automatically generated contributions like the navigation menu or related links lists.

Several algorithms have been developed to automatically detect the template structures in order to identify and / or extract particular parts of a document such as the main content. These structure detection algorithms depend on training sets which usually require a manual or semi-automatic selection of documents which are based on the same template.

Content Extraction (CE) algorithms have a similar goal: to identify the main content within a single and isolated document. These approaches are usually based on heuristics and work for textual main contents only. If the main content is not a text or the assumptions about how to locate it do not hold, the algorithms might fail entirely. Their advantage instead is, that they do not need a set of training documents and therefore can be run out of the box.

Several applications can benefit from CE and Template Detection (TD) under different aspects: Web Mining and web based Information Retrieval applications use these techniques to pre-process the raw HTML data in order to reduce noise and improve accuracy, other applications use CE to rewrite web pages for presentation on small screen devices or access via screen readers for visually impaired users. Yet another field of application is to remove additional contents in order to speed up the download time via connections with narrow bandwidth.

This paper discusses a way to combine the advantages of the two worlds. Starting from a single initial document we collect all the web documents it references by hyperlinks. By clustering the collected documents according to their underlying templates, we create a clean training set suitable to detect the template structure of the initial document. We use this possibility to develop an entropy based CE algorithm. The extraction is achieved by eliminating the initial document's texts which are redundant in the training set, thus using the benefits of TD concepts.

We will proceed with a look at related works in the next section, before describing our solutions for collecting and clustering web documents to generate training sets in sections 3 to 5. Afterwards we outline the entropy based CE method in 6 and explain the experiment setup we used to evaluate our approach in section 7. After presenting the results we conclude the paper with a few remarks and options for future improvements.

2 Related Works

The problem of recognising template structures in web documents was first discussed by Bar-Yossef and Rajagopalan in [1], proposing a template recognition algorithm based on DOM tree segmentation and segment selection. Yang et al. proved in [2] the general problem of finding an unambiguous schema representing a template to be NP complete.

There is, however, a number of practical works about template recognition algorithms.

Lin and Ho developed InfoDiscoverer [3] which is based on the idea, that – opposite to the main content – template generated contents appear more frequently. To find this redundant content, they disassemble the web pages in a training set into blocks of text content, calculate an entropy for each of those blocks and discard them if they have a too high entropy value. Debnath et al. used a similar assumption of redundant blocks in ContentExtractor [4]. They take into account not only words and text but also other features like the presence of image or script elements. The Site Style Tree approach of Yi, Liu and Li [5] is concentrating more on the visual impression single DOM tree elements are supposed to achieve. They look for identically formatted DOM sub-trees which frequently occur in the documents and therefore are declared to be produced by templates. A similar but slightly more flexible solution is to look not for equally formatted but for structurally similar sub-trees, as done by Reis et al. in [6]. They introduced the tree mapping algorithm RTDM to calculate a tree edit distance between two DOM trees. The tree edit distance is used as well to perform a cluster analysis in order to find clusters of different templates within the training set. Another approach to automatise building training sets was recently proposed by Chakrabarti, Kumar and Punera in [7]. They combine the site-level TD algorithm of [8] to produce training sets for a classifier which assigns a templateness score to each single node in the DOM tree.

One of the more prominent solutions for CE is the Crunch framework. It was introduced by Gupta et al. in [9] and is refined continuously. Avoiding a one-solution-fits-all approach, Crunch combines several heuristics to discover and remove e.g. link lists, text lists, blocks with a too high link ratio or commercial banners. The main objective of Crunch is to optimise HTML documents for presentation on small screen devices or to improve accessibility for screen readers. Debnath et al. developed the Feature Extractor algorithm and its extension the K-Feature Algorithm in [10]. The underlying idea is to segment a web document in blocks and analyse this blocks for the presence and prevalence of features like text, images, JavaScript etc. The extraction process is based on retrieving those blocks which correspond best to certain desired features, e.g. text for the classical main content. Finn et al. introduced the Body Text Extraction (BTE) algorithm in [11] as a pre-processor for their application classifying news articles on the web. Pinto et al. [12] extended the BTE approach to construct Document Slope Curves (DSC). In [13] we developed a way to measure, evaluate and compare CE algorithms based on the standard IR measures Precision, Recall and F1. Comparing different approaches, an adaptation of the DSC algorithm turned out to be the best performing general CE method.

Buttler [14] discusses several ways to measure the similarity between XML documents. Tree edit distances are stated to be probably the best but as well a computationally expensive similarity measure. Therefore Buttler proposes the path shingling approach which makes use of the shingling technique suggested by Broder et al. in [15].

3 Building Training Sets On-the-fly

To incorporate the CE algorithms' advantage of being able to operate on a single and isolated document into the TD algorithms, we have to build a set of suitable training documents automatically. We achieve this in two steps: First, to find a set of documents which are likely to be based on the same template as the initial document, and second, to filter out those documents which are not based on this template.

To solve the first problem we simply download the web documents referenced from the initial document via its hyperlinks. Recalling our initial remark about the increased use of WCMS serves this approach. Web pages created by a WCMS usually have a high number of intra-site links. Further, the documents originating from the same web site are a good source for building training sets as they are very likely based on the same or at least similar templates. To separate documents based on the same template as the initial document from those based on different templates or no template at all, we perform a cluster analysis. The clusters are formed based on the structural similarity of the documents, as the structure is imposed mainly by the underlying template. The cluster containing the initial document will then serve as training set for TD.

A similar approach of clustering documents to purify training sets is taken by Reis et al. in [6]. They use the top-down hierarchical tree matching algorithm RTDM to calculate the similarities and form clusters with a similarity threshold of 80%. Even though the algorithm has proven to perform quite well in practice, it still has a quadratic worst case complexity. Further, the paper does not mention any evaluation of how well this clustering approach works.

4 Measuring Structural Distances between Web Documents

Especially the above mentioned high computational cost for measuring the structural distance between documents by tree edit costs calls for faster alternatives. We will describe three approaches which are significantly faster in comparing documents.

4.1 Common Paths Distance

One alternative way to compare the structure of web documents is to look at the paths leading from the root node to the leaf nodes in the DOM tree. A path is denoted e.g. by concatenating the names of the elements it passes from root to leaf. For each document D it is then possible to represent it by the set $p(D)$ of paths it contains. A distance measure can be computed via the intersection of the two path sets of two documents D_1 and D_2:

$$d_{CP}(D_1, D_2) = 1 - \frac{|p(D_1) \cap p(D_2)|}{\max(|p(D_1)|, |p(D_2)|)} \quad (1)$$

Computing the paths for the documents can be done in linear time with respect to the nodes in the DOM tree. Using hashing, the intersection of the two resulting sets can be computed in expected linear time as well.

4.2 Common Path Shingles Distance

The idea of path shingles is not to compare complete paths but rather breaking them up in smaller pieces of equal length – the shingles. The advantage of this approach is that two paths which are differing only for a small part, but are quite similar for the rest, will have a large "agreement" on the shingles. The shingling can be realised in a way that it does not add any substantial cost to the computation compared to the CP distance.

So, if $ps(D)$ provides the path shingles for a document D, the path shingle distance can be computed similarly as above by:

$$d_{CPS}(D_1, D_2) = 1 - \frac{|ps(D_1) \cap ps(D_2)|}{\max(|ps(D_1)|, |ps(D_2)|)} \quad (2)$$

4.3 Common Tag Sequence Shingles Distance

The shingling technique allows us to look at the structure of a document in yet another way. While comparing the entire sequence of the tags as they appear in the source codes of the documents is computationally too expensive[1], comparing the shingles $ts(D)$ of this sequence is feasible. The shingles maintain a certain context for each tag without having to look at the complete document. The distance can be computed analogous to the path shingle distance:

$$d_{CTSS}(D_1, D_2) = 1 - \frac{|ts(D_1) \cap ts(D_2)|}{\max(|ts(D_1)|, |ts(D_2)|)} \quad (3)$$

5 Template Clustering

The first indication for evaluating these three approaches was to look at the time they needed for computing the distance between documents – especially in comparison with RTDM. For this purpose we measured how long it took to compute a distance matrix for a set of documents. The time needed to calculate the distances between all combinations of two documents in the set is obviously increasing quadratic with the size of the document set. However, the graph in figure 1 shows the additionally impact of the quadratic complexity of comparing two documents using RTDM. CP, CPS and CTSS distance measures instead can be computed reasonably fast.

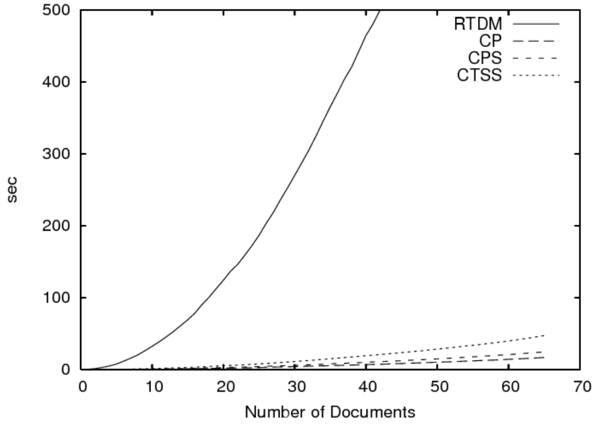

Figure 1. Time to compute a distance matrix for an increasing number of documents using the different distance measures.

Distance Measure	RTDM	CP	CPS	CTSS
I_{Dunn}	0.466	1.169	1.227	1.290

Table 1. Dunn index I_{Dunn} for all distance measures.

Very important is as well how suitable the different distance measures are for clustering documents according to their underlying templates. To evaluate this suitability we use a corpus of 500 documents taken from five different German news web sites. The documents from each web site are selected from different areas and are based on the same or at least similar templates[2]. To see how well the five groups of documents based on the same template are separated by the distance measures we use the Dunn index. The Dunn index sets in comparison the maximum distance d_{max} which occurs between items in the same group with the minimum distance d_{min} which occurs between items from different groups and is defined as $I_{Dunn} = \frac{d_{min}}{d_{max}}$. High values for I_{Dunn} correspond to a better partitioning of the documents with regard to their underlying templates.

The results in table 1 are quite surprising: compared to the other measures the tree edit distance algorithm is less suitable for the task of separating documents according to their templates. This insight in combination with the higher computational cost led to the decision to discard the RTDM based distance for incorporation in a cluster analysis.

To further determine a suitable distance threshold separating documents based on the same template and those based on different templates we looked at the distribution of distances within the distance matrices. Figure 2 shows the histograms for the distance distributions using a logarithmic scale. All three measures show large gaps (distances which never occur) between higher and lower distances. The CP distance even appears to have two gaps. Assum-

[1] Computing the longest (not necessarily continuous) subsequence has again quadratic time complexity.

[2] We deduced the common template from the visual impact of the documents and a brief analysis of the source code.

Table 2. Using the Rand index I_{Rand} to evaluate single linkage clustering with the distance thresholds chosen according to the gaps in the distance histograms.

Distance Measure	CP	CP	CTSS	CPS
Threshold	0.7	0.9	0.85	0.6
No of. Clusters	5	3	5	5
I_{Rand}	1.000	0.760	1.000	1.000

ing the distance distributions and their gaps to be typical we used the single linkage clustering method with clustering thresholds of 0.6 for CPS, a threshold of 0.85 for CTSS and with thresholds of 0.7 and 0.9 for the CP measure.

The applied single linkage clustering is a hierarchical clustering method. Hence, it has the advantage that it is not necessary to fix a-priori the number of clusters to be formed[3]. Instead it starts off with each document forming a cluster on its own. The clusters are iteratively merged, where the single linkage approach is always merging those two clusters for which the distance between two of the documents from the two clusters is minimal over all inter-cluster distances. If all distances between the clusters are above the given threshold the algorithm stops with the current cluster configuration as result.

To evaluate this configuration we use the Rand index, which is comparing how often a computed cluster configuration agrees or disagrees with a known reality. In our context an agreement corresponds to the cluster analysis either claiming correctly two documents to be based on the same template or to claiming correctly two of the documents having different underlying templates. A disagreement accordingly corresponds to either putting documents together in a cluster which have different underlying templates or to separate them in different clusters though they are based on the same template. So, if A and D are the number of agreements and of disagreements, the Rand index is defined as $I_{Rand} = \frac{A}{A+D}$.

The 0.9 threshold for CP turned out to be too low and resulted in three clusters only. Table 2 shows that beside this exception the gaps do really correspond to a separation of the actual clusters as the Rand index has a value of 1. So, choosing the distance threshold accordingly for a single linkage clustering results in perfect groups of documents which are based on the same template.

6 Entropy Based Template Detection

In section 2 we referred already to several works discussing TD algorithms. Some of them – namely InfoDiscoverer, ContentExtractor and to a certain degree as well the approach of Bar-Yossef and Rajagopalan – are segmenting the pages to find redundant pieces of information. We use a simplified version of those approaches to confirm that our idea of training set generation for TD is working and to complete the link to the CE world. Given the background of CE, we analyse only the texts appearing in the web documents of the training sets. We base the segmentation of a web document on its DOM tree representation: each text node in the DOM tree is considered a segment of the web page. Counting in how many documents the fragments appear we calculate the document frequency (DF) of the texts. If the DF for a text fragment found in the initial document is too high we accordingly remove it from the document. In this way all redundant texts are removed and the content remaining should be the main content – which corresponds to the functionality of a CE algorithm.

7 Evaluation

Our comparison of different CE algorithms in [13] used different criteria for comparing CE methods. For evaluation of the extraction performance we provided a gold standard for the main content of documents. This allows to use Information Retrieval measures like Recall, Precision and F1 to evaluate how close is the content extracted by a CE algorithm to the actual main content. Though we proposed several granularities for this evaluation, they turned out to deliver comparable results. Hence, we use only the approach modelling the main content as a sequence of words. As well a CE algorithm is extracting a sequence of words. Based on the longest common subsequence of these two sequences as a notion of extracted relevant content we can determine Recall, Precision and F1 for the extraction process. Precision P is the ratio of extracted relevant items (in our case words) to all extracted items, Recall R instead is the ratio of extracted relevant items to all relevant items. Based on those two concepts, the F1-measure is defined as $\frac{2 \cdot R \cdot P}{R+P}$ and combines both values in a single measure.

As base line for comparison we will use again the "plain" extraction method which means to extract all text contents from a web document. This pseudo-method corresponds to not using CE at all and allows to claim from any CE algorithm to perform better than this alternative. As in [13] we further found out, that an adaptation of the Document Slope Curve (DSC) approach of Pinto et al. [12] is the best performing general purpose CE algorithm, we will use as well this method for comparison with our approach. DSC is based on the heuristic that the main content is usually a longer text which contains few or no tags in the source code. Applying a windowing technique, DSC determines areas in a document which satisfy this characteristic of the main content.

As test data we used several web documents from three German and two Italian news web sites and outlined manually the text of their main content. Along with the documents themselves we downloaded all the linked web pages to provide our algorithm with the data needed for generating training documents. Altogether the test data comprised 746 documents with an outlined main content and 10.994 supplementary documents referenced by hyper-

[3] Something that would be very difficult to do as it would imply to estimate the number of templates found among the linked pages.

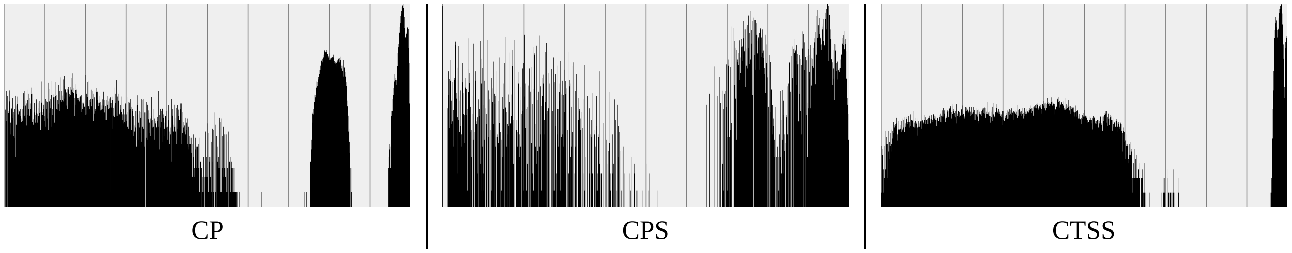

Figure 2. Distribution of distances for the fast distance measures (logarithmic scale)

links. The documents used in this tests were different from the ones used for fixing the distance thresholds for the single linkage clustering to avoid interferences between the experiments.

Clustering for each of the documents the linked pages using each of the three distance measures CP, CPS and CTSS, we determined training sets for the TD algorithm described above in 6. A text fragment was discarded as being redundant if it appeared in more than a third of the documents in the training set. The remaining texts were compared to our gold standard for the main content. The same procedure – without the detour of building training sets – was used for the DSC and plain approach.

Table 3 summarises the results of the experiments, showing the average F1 score obtained for the CE process on the web documents of the five different news web sites. The first thing to notice is that the performance of the entropy based algorithm is always above the alternative of not using CE at all. This demonstrates the applicability of the approach. It suggests as well, that the generated training sets are suitable for a TD task. Otherwise the results would be comparable to the plain method as no redundant template information would be found and accordingly no texts would be removed.

Further, all three underlying distance measures lead to the same quality in the results. For some test documents there are slight differences in performance, but the variations are not significant enough to derive a superiority of one of the approaches. This allows to favour the simple path based CP distance measure, as it is slightly faster compared to the other two distance measures.

Compared to DSC the results are not as clear. For the documents taken from the web sites of l'espresso and Telepolis the entropy based approach is delivering better F1 results, for the other three web sites the results are worse. Looking more closely at the performance in the Recall and Precision measures explains this observation. The entropy based algorithm always scores a very high, often even a perfect Recall value, i.e. the extractor almost certainly finds the main content. The Precision instead is not as good, which means that some other contents are not removed even though they ought to be. Thus, the Precision improvements compared to the plain series are usually not of the same magnitude as those of DSC. DSC however is never scoring a perfect Recall. This tradeoff of sacrificing Recall for improving Precision favours DSC in the light of the F1 measure.

An explanation for the poorer Precision performance of the entropy based CE method is the way the extractor is working. If a text is created by a WCMS in the context of a particular document – e.g. a list of related links – they might not appear in the same form on other, linked pages found in the training set. Accordingly these texts will not be discovered to be redundant and will not be removed from the document. The problem, however, is common to all entropy based TD methods and can appear even for perfect training sets.

Looking at the results, the entropy based algorithm might be a suitable solution for applications, which require a high recall and can easily handle additional data. A typical scenario for such an application is streamlining web pages for displaying them on small screen devices or preprocessing them for screen readers. Mutilating the main content by extracting too little of it is obviously counterproductive in these cases, while the user can easily ignore some additional contents, which were extracted wrongly.

Another drawback of the TD algorithms is the time they need. While most traditional CE algorithms process an HTML document within fractions of a second, any algorithm needing a training set will take far longer. Downloading the linked documents, calculating the distance matrix and applying the TD algorithm is very time consuming and renders this approach so far unsuitable for a decent on-the-fly CE.

8 Conclusions

In this paper we presented a way to bring together the two major benefits from the worlds of Template Detection and Content Extraction. Combining the possibility of working out of the box and without the need of explicitly providing a training set with the theoretic underpinning of TD algorithms allows to create a new class of CE algorithms. The aim was reached mainly by performing a cluster analysis on the set of referenced documents to find a subset of documents based on the same template. This subset was used as an automatically generated training set for a simple entropy based TD algorithm. We showed in our experiments that this approach scores very high Recall values and therefore might be a good solution for applications that require

CE Algorithm	Web site providing the documents				
	l’espresso	Heise Online	la Repubblica	Spiegel Online	Telepolis
Entropy (CP)	0.721	0.690	0.807	0.636	0.901
Entropy (CPS)	0.722	0.675	0.794	0.636	0.899
Entropy (CTSS)	0.722	0.671	0.791	0.637	0.902
DSC	0.690	0.808	0.906	0.938	0.854
Plain	0.523	0.501	0.677	0.548	0.861

Table 3. Performance of the Content Extraction methods

to retrieve the main content entirely.

Employing more sophisticated web document segmentation methods and more detailed algorithms for detecting redundant information might improve the Precision score obtained by the approach. Already finetuning the document frequency threshold used for the CE might lead to improvements. Another point for future works will be to speed up the extraction process. A possibility could be to limit the size of the generated training set, as the time complexity for computing the distance matrix underlying the clustering is of quadratic order. A further improvement could be to reduce the necessity to download all referenced documents by selecting only those links which look promising. A different and quite applied extension will be to build in a fallback solution if no suitable training documents for TD are found during the cluster analysis.

References

[1] Z. Bar-Yossef and S. Rajagopalan. Template detection via data mining and its applications. In *Proc. 11th Int. Conf. on WWW*, pages 580–591, New York, NY, USA, 2002. ACM Press.

[2] G. Yang, I. V. Ramakrishnan, and M. Kifer. On the complexity of schema inference from web pages in the presence of nullable data attributes. In *Proc. 12th Int. Conf. on Information and Knowledge Management*, pages 224–231, New York, NY, USA, 2003. ACM Press.

[3] S.-H. Lin and J.-M. Ho. Discovering informative content blocks from web documents. In *Proc. 8th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pages 588–593, New York, NY, USA, 2002. ACM Press.

[4] S. Debnath, P. Mitra, and C. L. Giles. Automatic extraction of informative blocks from webpages. In *Proc. 2005 ACM Symp. on Applied Computing*, pages 1722–1726, New York, NY, USA, 2005. ACM Press.

[5] L. Yi, B. Liu, and X. Li. Eliminating noisy information in web pages for data mining. In *Proc. 9th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pages 296–305, New York, NY, USA, 2003. ACM Press.

[6] D. C. Reis, P. B. Golgher, A. S. Silva, and A. F. Laender. Automatic web news extraction using tree edit distance. In *Proc. 13th Int. Conf. on WWW*, pages 502–511, New York, NY, USA, 2004. ACM Press.

[7] D. Chakrabarti, R. Kumar, and K. Punera. Page-level template detection via isotonic smoothing. In *Proc.16th Int. Conf. on WWW*, pages 61–70, New York, NY, USA, 2007. ACM Press.

[8] D. Gibson, K. Punera, and A. Tomkins. The volume and evolution of web page templates. In *Special Interest Tracks and Posters, 14th Int. Conf. on WWW*, pages 830–839, New York, NY, USA, 2005. ACM Press.

[9] S. Gupta, G. Kaiser, D. Neistadt, and P. Grimm. DOM-based content extraction of HTML documents. In *Proc. 12th Int. Conf. on WWW*, pages 207–214, New York, NY, USA, 2003. ACM Press.

[10] S. Debnath, P. Mitra, and C. L. Giles. Identifying content blocks from web documents. In *Foundations of Intelligent Systems*, LNCS, pages 285–293, 2005.

[11] A. Finn, N. Kushmerick, and B. Smyth. Fact or fiction: Content classification for digital libraries. In *DELOS Workshop: Personalisation and Recommender Systems in Digital Libraries*, 2001.

[12] D. Pinto, M. Branstein, R. Coleman, W. B. Croft, M. King, W. Li, and X. Wei. Quasm: a system for question answering using semi-structured data. In *Proc. 2nd ACM/IEEE-CS joint Conf. on Digital libraries*, pages 46–55, New York, NY, USA, 2002. ACM Press.

[13] T. Gottron. Evaluating content extraction on HTML documents. In *Proc. 2nd Int. Conf. on Internet Technologies and Applications*, pages 123–132, September 2007.

[14] D. Buttler. A short survey of document structure similarity algorithms. In *Proc. Int. Conf. on Internet Computing*, pages 3–9. CSREA Press, 2004.

[15] A. Z. Broder, S. C. Glassman, M. S. Manasse, and G. Zweig. Syntactic clustering of the web. *Computer Networks*, 29(8-13):1157–1166, 1997.

View publication stats