CRE: An Automatic Citation Record Extractor for Publication List Pages

Kai-Hsiang Yang[1], Shui-Shi Chen[2], Ming-Tai Hsieh[1], Hahn-Ming Lee[12], and Jan-Ming Ho[1]

[1] Institute of Information Science, Academia Sinica,Taipei, Taiwan
{khyang, mthsieh, hmlee, hoho}@iis.sinica.edu.tw
[2] Department of Computer Science and Information Engineering,
National Taiwan University of Science and Technology, Taipei, Taiwan
{M9515001,hmlee}@mail.ntust.edu.tw

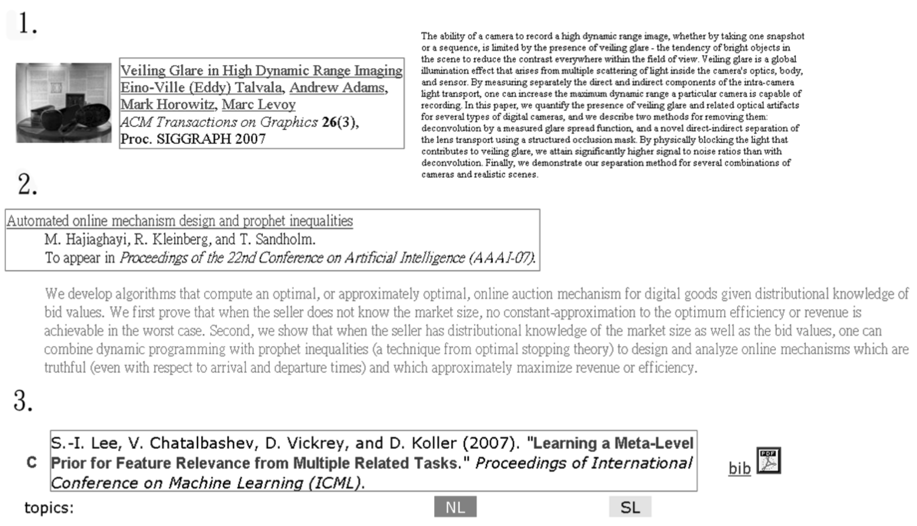
Abstract. Today a huge amount of researchers' publication list pages are available on the Web, which could be an important resource for many value-added applications, such as citation analysis and academic network analysis. How to automatically extract citation records from those publication list pages is still a challenging problem because many of those pages are crafted manually by researchers themselves, and the layouts of those pages could be quite different according to different researchers' affinities. In this paper, we propose a system, called the Citation Record Extractor (CRE), to automatically extract citation records from publication list pages. Our key idea is based on two observations form publication list pages. First, citation records are usually presented in one or more contiguous regions. Second, in the form of HTML structure, citation records are usually presented by using similar tag sequences and organized under a common parent node. Our system first identifies candidate common style patterns (CSPs) within pages in the DOM tree structure, and then filters out irreverent patterns by using a classifier which is based on the length distribution of citation records. Experimental results show that our method can perform well on real dataset with precision and recall at 80.4 % and 83.7% respectively, and provides more that 80% of F-measure for a majority of (around 90%) of the publication list pages in the real world.

1 Introduction

Researchers usually create their homepages on the Web for various reasons, such as describing their research and contributions, or providing material for their new courses. These Web pages often contain up-to-date information of the researchers, because some researchers often provide their new papers on their own publication list pages before they are formally published on journal magazines or conferences. Hence, it is possible to learn about the state-of-the-art knowledge and technologies from those researchers publication list pages.

Although publication information are really important, it is not easy to develop an automatic system to extract all publication records from publication list pages, because many publication list pages are crafted manually by researchers themselves, and the layouts of them could be quite different due to different researchers affinities. In this paper, we call a single publication record as a citation record.

Fig. 1 shows three examples of publication list pages with totally different presentation styles. For showing their position, each citation record is enclosed with a rectangle. We can easily observe that, in the first and second styles, there are some extra data like images or descriptions written in natural languages; in the third style, there are also some descriptive tags such as related topics for each citation record. Obviously, what we need is only the publication records enclosed with rectangles, while other data should be discarded. We also observe that a few publication list pages are created using a common interface, which provides a unified layout for Web presentation, and that will make this problem easier. Hence, we are most interested in the manually crafted personal publication list pages, since they contain various layouts and are, therefore, more practical and challenging.


1.

Veiling Glare in High Dynamic Range Imaging
Eino-Ville (Eddy) Talvala, Andrew Adams,
Mark Horowitz, Marc Levoy
ACM Transactions on Graphics **26(3)**,
Proc. SIGGRAPH 2007

The ability of a camera to record a high dynamic range image, whether by taking one snapshot or a sequence, is limited by the presence of veiling glare - the tendency of bright objects in the scene to reduce the contrast everywhere within the field of view. Veiling glare is a global illumination effect that arises from multiple scattering of light inside the camera's optics, body, and sensor. By measuring separately the direct and indirect components of the intra-camera light transport, one can increase the maximum dynamic range a particular camera is capable of recording. In this paper, we quantify the presence of veiling glare and related optical artifacts for several types of digital cameras, and we describe two methods for removing them: deconvolution by a measured glare spread function, and a novel direct-indirect separation of the lens transport using a structured occlusion mask. By physically blocking the light that contributes to veiling glare, we attain significantly higher signal to noise ratios than with deconvolution. Finally, we demonstrate our separation method for several combinations of cameras and realistic scenes.

2.

Automated online mechanism design and prophet inequalities
M. Hajiaghayi, R. Kleinberg, and T. Sandholm.
To appear in *Proceedings of the 22nd Conference on Artificial Intelligence (AAAI-07).*

We develop algorithms that compute an optimal, or approximately optimal, online auction mechanism for digital goods given distributional knowledge of bid values. We first prove that when the seller does not know the market size, no constant-approximation to the optimum efficiency or revenue is achievable in the worst case. Second, we show that when the seller has distributional knowledge of the market size as well as the bid values, one can combine dynamic programming with prophet inequalities (a technique from optimal stopping theory) to design and analyze online mechanisms which are truthful (even with respect to arrival and departure times) and which approximately maximize revenue or efficiency.

3.

C S.-I. Lee, V. Chatalbashev, D. Vickrey, and D. Koller (2007). **"Learning a Meta-Level Prior for Feature Relevance from Multiple Related Tasks."** *Proceedings of International Conference on Machine Learning (ICML).* bib

topics: NL SL

Fig. 1. Examples of different citation presentation styles

In this paper, we propose a system, called the Citation Record Extractor (CRE), to automatically extract citation records from publication list pages. Our key idea is based on two observations. First, citation records are usually presented in one or more contiguous regions within a publication list page. Second, citation records are usually presented by using similar HTML tag sequences and organized under a common parent node. Based on these two observations, our system first identifies candidate common style patterns (referred to as CSPs hereafter) in the DOM tree structures of pages, and then filters out irreverent

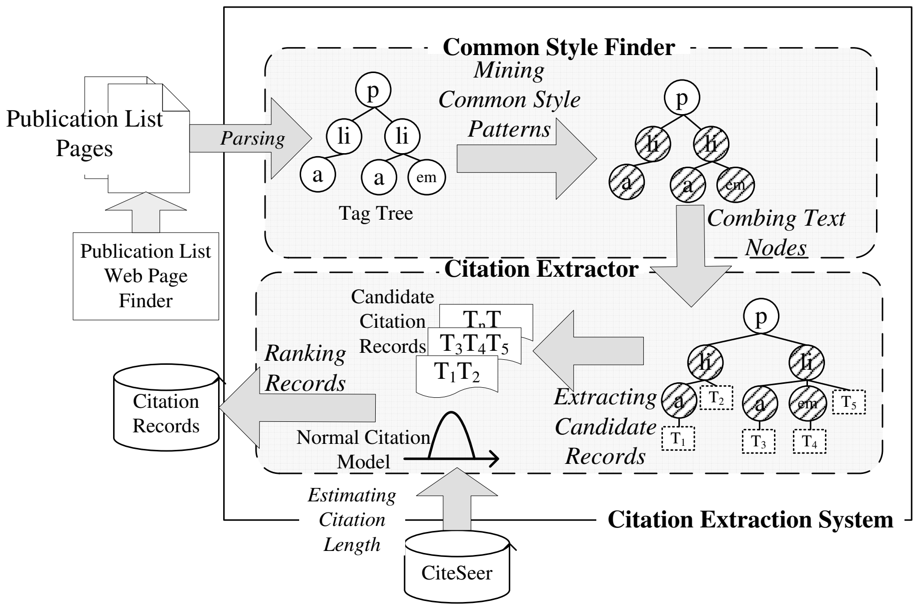

Common Style Finder
Mining Common Style Patterns
Publication List Pages
Parsing
Tag Tree
Combing Text Nodes
Publication List Web Page Finder
Citation Extractor
Candidate Citation Records
Ranking Records
Extracting Candidate Records
Citation Records
Normal Citation Model
Estimating Citation Length
Citation Extraction System
CiteSeer

Fig. 2. System architecture of CRE

patterns by using a classifier which is based on the length distribution of citation records. Experimental results show that our approach can perform well on datasets with precision and recall at 80.4% and 83.7% respectively; and, moreover, provides more that 80% of F-measure for a majority of (around 90%) of the publication list pages in the real world.

2 Related Work

Many researches had been conducted to extract regular patterns from semi-structured Web pages, and these researches are mainly related to wrapper generation. A wrapper is a program that extracts records from Web pages automatically, and there are several different ways to generate wrappers. First, a wrapper can be crafted based on the observation of pattern formats within Web pages. Such an approach is not scalable since it requires human labor and is thus time-consuming. Furthermore, the extraction rules must be modified once the format of the Web page changes. Systems proposed by Chawathe et al. [1] and Chidlovskii et al. [2] are based on such hand-coded extractors. The second approach is wrapper induction, which is based on machine learning techniques. This approach requires a set of manually labeled positive and negative examples to learn extraction rules, and the labeling process is also labor intensive. Proposed systems based on this approach include Stalker [3], Softmealy [5], WL [4] [6] and WIEN [7]. Because of the intensive labor work of previous two approaches, some automatic extraction methods are therefore proposed. Chang et al. proposed Information Extraction based on Pattern Discovery(IEPAD) [10] which encodes HTML tags as a binary string and applies the PAT tree and a sequence alignment tool to find maximal repeated patterns in the Web page. In [8], Liu et al. proposed MDR that takes advantage of the DOM tree structure to segment a Web page into several data regions composed of similar tag sequences and use heuristics to extract data records. They further proposed an improved system DEPTA (Data Extraction based on Partial Tree Alignment) [9] which is based on partial tree alignment. NET [11] improves DEPTA so that it can handle nested tables.

3 Proposed System

As mentioned before, our proposed system is designed according to the observations that citation records usually share similar presentation styles and are located at a contiguous region in a publication list page. For example, publication list pages usually contain more than one citation records and use a set of style tags like: <i>, <em>, etc. to present them. Our task is to extract citation records in a given Web page by mining its CSPs, while CSPs are mined via analyzing the regularity of tag sequences in the DOM tree structure.

Fig. 2 depicts the system architecture, which contains two modules, namely, a Common Style Finder and a Citation Extractor. The Common Style Finder extracts all possible CSPs by analyzing pairs of nodes (or combined nodes) at all levels of the DOM tree structure for the given Web page, and the Citation Extractor then filters out irreverent patterns caused by some repetitive items within Web pages by using a classifier. The classifier is built by establish a normal length distribution model for citation records, and can calculate the probability of a given string to be a citation record according to the model. Here, we use the online available CiteSeer database to build a normal distribution for the length of citation records, which is very useful to filter out patterns caused by some repetitive items in a Web page. In the following sections we will provide more details about each module.

Common Style Finder The Common Style Finder finds all possible CSPs formed by pairs of nodes or combined nodes. As shows in Fig. 3, citation records are usually arranged in parallel in a DOM tree, and therefore we can find the CSPs by analyzing their tag similarities. Since a citation record may be presented by more than one node, we have to concern whether a combination of adjacent nodes forms a citation record. Instead of searching all possible combinations of nodes, we only compare adjacent nodes at each level of the DOM tree. We define a pair of adjacent nodes (or a pair of adjacent combined nodes of equal size n, where n denotes the number of nodes to be combined) share a CSP if the similarity of their tag sequences is under a pre-defined threshold, where each of the two adjacent (combined) nodes constitutes a candidate citation record. The tag sequence of a node x is defined as the concatenation of all tags in the subtree rooted at x through a Depth First Search (DFS) traversal. For a combined node

y, its tag sequence is the concatenation of all tag sequences of the nodes that comprise y.

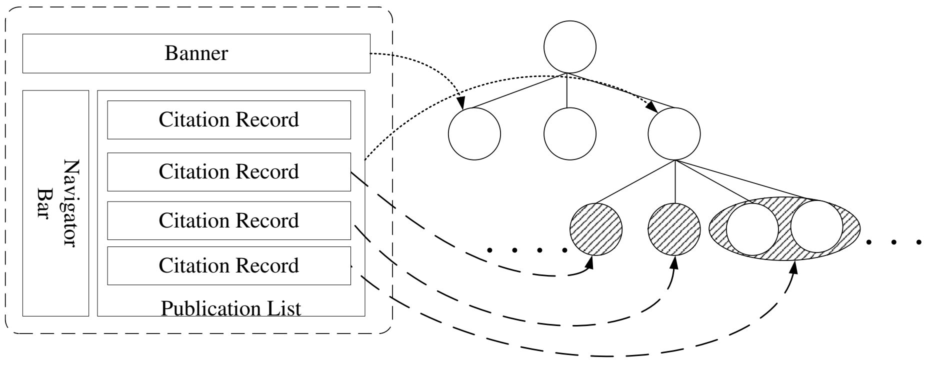


Fig. 3. DOM tree presentation of a Web page

Table 1 shows the algorithms for all modules. For the Common Style Finder, the similarity checking processes are performed for all adjacent nodes (or adjacent combined nodes with size from 2 to K) at the same level of the DOM tree. We set $K = 10$ empirically because it is unlikely that a citation record consists of more than ten tag nodes. In line 5-6, tag sequences of a pair of nodes (or combined nodes) are built via a function Encoding, which parses the DOM trees into tag sequences through a DFS traversal.

We use the edit-distance to calculate the similarity between two tag sequences. In this paper, similarity between two strings s_1 and s_2 is normalized between 0 and 1 as follows:

$$Similarity(s_1, s_2) = \frac{EditDistance(s_1, s_2)}{Max(|s_1|, |s_2|)} \tag{1}$$

Citation Extractor The goal of Citation Extractor is to extract all citation records by using the CSPs information. We define a record set as the union of several non-overlapping records extracted by one or a sequence of adjacent CSPs composed of equal number of nodes. There are two steps for extracting all candidate record sets. First, adjacent CSPs composed of equal number of nodes are merged, and the record set produced by a merged CSP is the union of non-overlapped records produced by each individual CSP. Second, text of each record is obtained by concatenating all text within the subtrees that form the CSP through a DFS traversal. For example, in Fig. 4, the CSP_1 produces two records ("$A_1T_1Y_1$", "$A_2T_2Y_2$"), CSP_2 produces ("$A_2T_2Y_2$", "$A_3T_3Y_3$") and CSP_3 produces ("$A_3T_3Y_3$", "$A_{4-1}A_{4-2}T_4Y_4$"), so the record set produced by $CSP_{1-3}(whichmeansCSP_1, CSP_2, andCSP_3)$ is the union of above records ("$A_1T_1Y_1$", "$A_2T_2Y_2$", "$A_3T_3Y_3$", "$A_{4-1}A_{4-2}T_4Y_4$").

Table 1. Common Style Finder and Citation Extractor algorithms

Commom Style Finder Algorithm

```
Procedure CommonStyleFinder(Node parent)
children =parent.children;
FOR i=1 to K
   FOR j=1 to candidates.length
      CNode1=Encoding(children (j to j+i-1));
      CNode2=Encoding(children (j+i to j+2*i-1));
      parent.SimilarityMatrix(i,j)=sim (CNode1, CNode2);
      j=j+2*i;
   END FOR
END FOR
FOR i=1 to children.length DO
     CommonStyleFinder(children(i));
END FOR
END Procedure
```

Citation Extractor Algorithm

```
Procedure CitationExtractor(Tagtree T)
CRSets[]=T.ALL_Record_Sets;
TCRSets[];
FOR i=0 to CRSets.length -1 DO
    OLPRSets[];
    FOR j=i+1 to CRSets.length DO
        IF CRSets [j] is overlap with CRSets [i] THEN
           OLPRSets.add(CRSets [j]);
        END IF
    END FOR
    IF OLPRSets!=empty THEN
        OLPRSets.add(CRSets [i]);
        TRSet=RankRecordSet(OLPRSets);
    ELSE
        TRSet = CRSets [i];
    END IF
    CRSets.remove(TRSet);
    CRSets.remove(RecordSets overlap with TRSet);
END FOR
Return TCRSets;
END Procedure
```

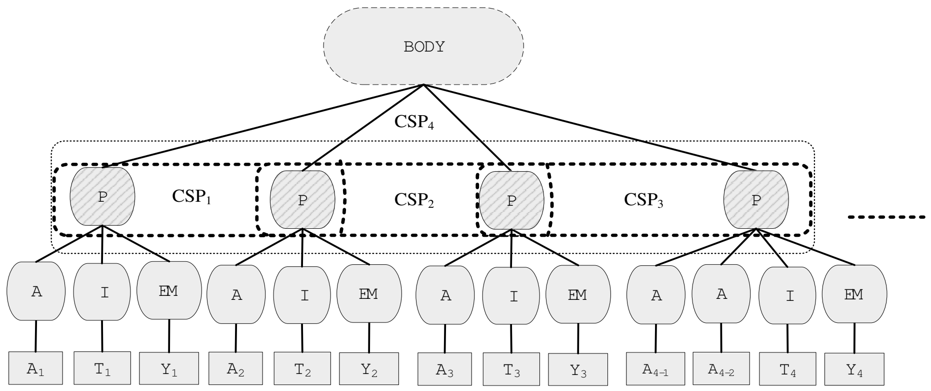


Fig. 4. An example of Common Style Patterns

In Fig. 4, the correct record set should be those extracted by CSP_{1-3}, but not $("A_1T_1Y_1A_2T_2Y_2", "A_3T_3Y_3A_{4-1}A_{4-2}T_4Y_4")$ extracted by CSP_4, which overlap with the record set extracted by CSP_{1-3}. In other words, the CSP_4 is not a correct combination for extracting citation records. To find the correct CSPs, for those overlapped record sets we design a scoring function to give each record set a score to represent the probability that the text in the record set is a citation record, and the record set with highest probability is then chosen as candidate citation record set. Suppose S is the text of a record set with word count $|S|$, the scoring function is defined as follows:

$$Score(S) = P(S \text{ is Citation} \mid |S|) \tag{2}$$

In order to model the distribution of word count in a citation record, we collected an online dataset which consists of 6,568 citation records from CiteSeer Web cite as our sampling dataset. Fig. 5 depicted the histogram of the dataset. Observing from the shape of histogram, we use a normal distribution model to fit it such that the probability model can be expressed by a probability density function:

$$P(l) = \frac{1}{\sqrt{2\pi\sigma^2}} exp\{-\frac{(l-\mu)^2}{2\sigma^2}\}, \tag{3}$$

where the l indicates the word count of a citation record, σ =6.09 and μ=25.60, respectively. We empirically define a threshold 0.01 to filter out strings in a record set with a probability less than the threshold.

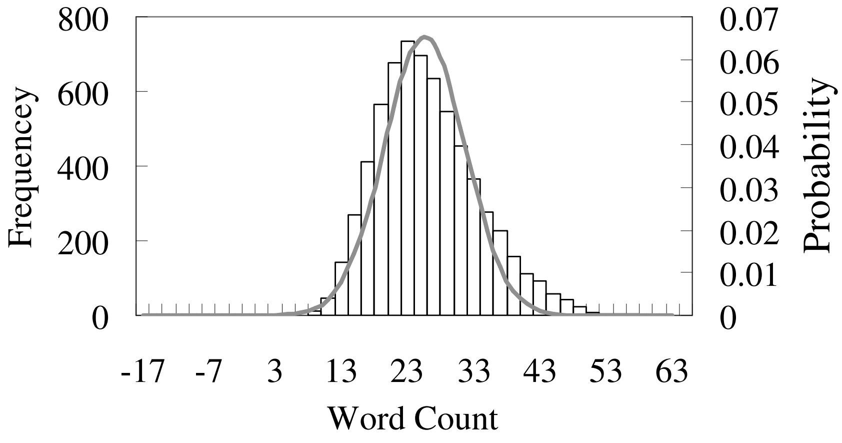


Fig. 5. Distribution of word count of a string

The main steps of Citation Extractor algorithm are shown in below part of Table1. In line 4, we extract all record sets of a given tag tree with CSPs labeled and store them as candidate citation record sets. From line 4 to 19, we find the overlapped record sets ranked by using our scoring function. Finally the citation records are reported without any text overlap.

4 Experiments

Data Sets We build two datasets for our experiments. Dataset (I) consists of 60 publication list pages collected randomly from computer science faculties of Pennsylvania, Berkley and Stanford universities. Dataset (I) is used as a training dataset for tuning the best similarity threshold of CSPs. For dataset (II), we collected the computer science faculties from the top 12 universities in America based on a Computer Science Ranking. We randomly select 240 researchers from the collected faculties, and only 187 of them have their personal publication list pages. Some researchers have their publications written in PDF format, and we do not handle such cases since our algorithm is involved in the HTML processing. We manually labeled answer sets of dataset (I) and dataset (II), while dataset (II) is used as our testing dataset, and there are no overlapped pages between dataset (I) and (II). In some cases, a researcher may have his/her publications distributed in several publication list pages. These pages are usually separated using different research topics or date of publishing. For such cases, we randomly select one of these publication pages to represent the researchers whole publications, since most of these publication list pages have similar layouts because they are created by the same author.

Before conducting experiments, we need to determine which kinds of publications should be regarded as citation records in our data sets. There are miscellaneous kinds of information that may appear in personal publication list pages, including non-publication information such as e-mail, office, education, etc., and publications. There are also many different kinds of publications, such as journal articles, conference papers, books, technical reports, and so forth. According to Bibtex format, title, author and year are three basic elements that almost all kinds of publications possess. Therefore, we define a rule for determining whether a particular citation record shall be counted as a correct answer during the labeling process. Generally, a title is always required for a citation record. Besides, at least two out of three elements (name, year, origin) must appear in a citation record, where the origin is a multi-option element that can be a journal, a publisher, a school or an institution.

Evaluation Metrics Instead of using exact string matching, we apply an approximating string matching scheme as our evaluation metrics, since the answer set is created through human labeling and there might be some slight differences between automatically extracted citation records and manually labeled ones. We therefore adapt an approximated string matching technique by using longest common subsequence (LCS) [12]. For a set of sequences, the LCS is the longest subsequence that appears in all sequences, and the LCS can be noncontiguous. The idea is that if a LCS of an extracted citation record and the answer string covers most portion of the corresponding answer string, this citation record is regarded as being extracted correctly.

Assume X is an extracted citation record, Y is the manually labeled answer string corresponding to X, and Z is the LCS of X and Y, where $|X|$ indicates the number of characters of X. A citation record is defined to be extracted correctly if both of the following constraints are satisfied:

$$|Z| \geq 0.95 \times |Y| \; and \; \frac{|X| - |Y|}{|Y|} \leq 0.5 \tag{4}$$

In the first constraint, we allow a small range of difference between an extracted citation record and its corresponding answer string. That is, Z should cover at least 95 percent of the characters of answer string Y. The second constraint avoids the situation that even the first constraint is satisfied, the extracted citation record X is actually much longer than the answer string Y, meaning that X may cover a large portion of unrelated text on the publication list page. For each publication list page i, assume $|C|$ is the number of correct citation records, $|E|$ is the number of extracted citation records and $|CE|$ is the number of correctly extracted citation records. The precision *(Pi)* and recall *(Ri)* values for i are expressed as follows:

$$P(i) = \frac{|CE|}{|E|} \; , \; R(i) = \frac{|CE|}{|C|} . \tag{5}$$

The overall precision P and recall R are averages of all Pi and Ri of publication list pages:

$$P = \frac{1}{n} \sum_{i=1}^{n} P_i \; , \; R = \frac{1}{n} \sum_{i=1}^{n} R_i . \tag{6}$$

4.1 Experimental Results

In our experiments, we use an open source tool (JTidy) to process Web pages, which is a java program to clean up malformed HTML tags, and provides some DOM APIs for us to build a DOM tree structure for a web page.

We had compared our system with the MDR system [8] proposed by Liu et al. A threshold is also needed to measure edit distance similarity both for the MDR and CRE, so we incrementally test the performances of MDR and CRE on our dataset (I) under different threshold values as shown in Fig. 6. We can easily see that, when T=0.3 results in best performance for CRE; while when T=0 has best result for MDR. Hence, we use these settings for each system in the following experiments.

For further analyzing our system performance, we classify the dataset (I) and (II) into ten subsets according to the coverage rate of citation records, which is defined as the ratio of the number of characters of all citation records to the whole text of a Web page. The reason for doing such classification is that if a publication list page contains only few citation records or too much noise, it would have much more difficulty to correctly extract citation records. Fig. 7(a)

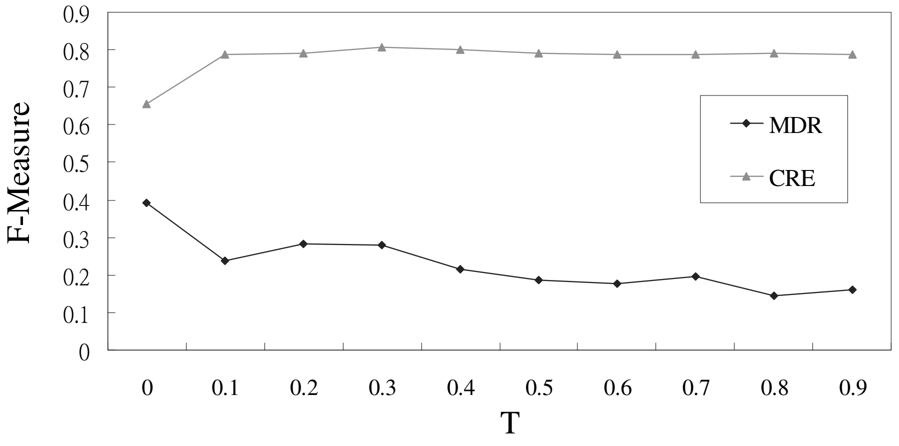


Fig. 6. Performances under different thresholds

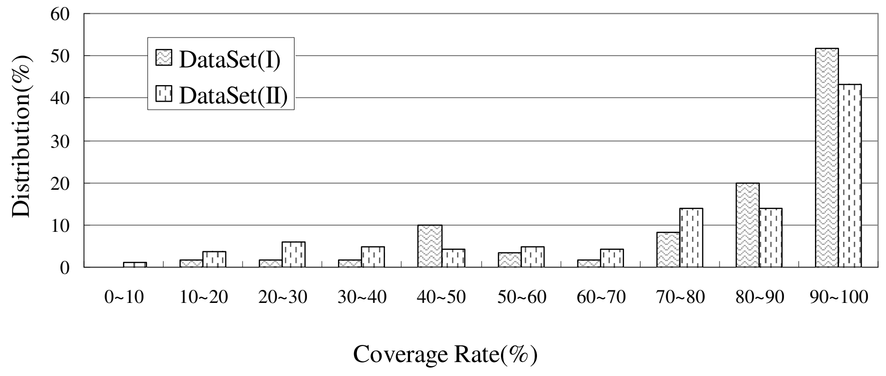
shows the distribution of each subset in dataset (I) and (II). We can easily see that more than 50% of pages in dataset (I) and more than 40 % of pages in dataset (II) have very high coverage rate (more than 90%), and around 90% of pages in dataset (I) and (II) have the coverage rate more than 40%.

Fig. 7(b) shows the average performance of CRE and MDR only for each subset of dataset (II), because dataset (I) is only used for training thresholds. In Fig. 7(b), it is clear to see that our proposed system performs well and stable (around 75% to 90%) when the coverage rate is higher than 30%, while it provides poor performance (around 40% for coverage rate from 10% to 30%) when the coverage rate is less than 30%. Generally, in such cases when the coverage rate is less than 30%, it is really hard to determine the CSPs because there are fewer citation records in those Web pages.

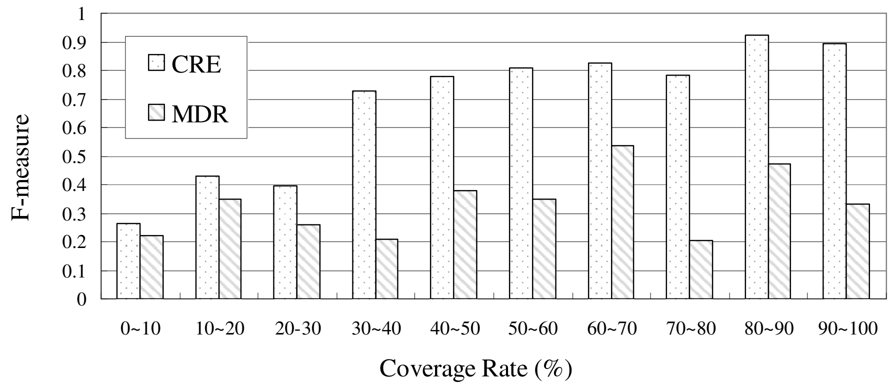
Compared with the MDR, our system definitely reaches a better performance. The main reason for this is that most citation records are not organized within tables that MDR system mainly focuses on. Besides, since we aim to solve domain-specific problem, say, citation extraction, it benefits a lot from the length distribution of citation records obtained from Citeseer database, and we believe that greatly boosts the accuracy of determining record boundaries; while MDR applies general heuristics of how people present data records on Web pages.

5 Discussion

Even through our experimental results show that the CRE can extract citation records presented with various layouts of publication list pages correctly and efficiently, there are still few pages that report poor performance mainly caused by two reasons. First, some errors exist in the DOM tree structures, which are



(a) Coverage rate distribution in datasets



(b) System performance in each subset of dataset(II)

Fig. 7. Coverage rate distribution in datasets and System performance in each subset of dataset(II)

caused by misusing HTML tags. Second, sometimes citation records are distributed in several sub-trees that do not share a common parent node, so that our algorithm can not find a single node in the DOM tree to represent exactly a citation record. In our future work we will focus on extracting those special case citation records.

6 Conclusion

In this paper, we present an automatically system to extract citation records from the researchers publication list pages on the Web. Our system applies the DOM tree structure and edit-distance techniques to identify candidate common style patterns (CSPs) within pages, and then filters out irreverent patterns by using a classifier which is based on the length distribution of citation records. Our experiments prove that calculating the tag sequence similarity is an applicable way to extract citation records accompanied with some citation-specific knowledge. Our system can perform well and stable (more that 80% of F-measure) for a majority (around 90%) of publication list pages.

In the future, we plan to provide an academic search system by integrating the CRE system with two of our previous works: (1) the PLF [13], which is a system to automatically retrieve researchers' publication lists on the Web, and (2) the BibPro [14], which is a citation parser system to extract metadata from the citation records extracted by the proposed CRE system. Implementation of such a system will facilitate and reduce the effort of academic searching, and due to the up-to-date characteristic of publication lists on the Web for most researchers, reliable and up-to-date query results which reflect to current trend can also be guaranteed. We believe that more research in these areas would definitely be worthwhile.

Acknowledgments. This work was supported in part by the National Science Council of Taiwan under grants NSC 96-2628-E-011-064-MY3, NSC 96-2221-E-011-084-MY3, NSC 96-2217-E-011-002 and NSC 95-2221-E-001-021-MY3.

References

1. S. Chawathe, H. Garcia-Molina and J. Hammer: The TSIMMIS project: integration of heterogeneous information sources. Journal of Intelligent Information Systems 8(2):117-132 (1997)
2. B. Chidlovskii, U. Borghoff, and P. Chevalier: Towards sophisticated wrapping of Web-based information repositories. The 5th International RIAO Conference, Montreal, Quebec, Canada, pp. 123-135 (1997)
3. I. Muslea, S. Minton and C. Knoblock: A hierarchical approach to wrapper induction. The third annual conference on Autonomous Agents pp. 190-197 (1999)
4. W. Cohen, M. Hurst and L. Jensen: A flexible learning system for wrapping tables and lists in HTML documents. The 11th International World Wide Web conference (2002)

5. C.-N. Hsu and M.-T. Dung: Generating finite-state transducers for semi-structured data extraction from the Web. Information Systems 23(8), pp. 521-538 (1998)
6. D.Pinto, A. McCallum, X. Wei and W. Bruce Croft: Table Extraction Using Conditional Random Fields. The 26th ACM SIGIR (2003)
7. N. Kushmerick: Wrapper induction: efficiency and expressiveness Artificial Intelligence. Artificial Intelligence 118(1-2):15-68 (2000)
8. B. Liu, R. Grossman, and Y. Zhai Mining: data records in Web pages. The ACM SIGKDD International Conference on Knowledge Discovery & Data Mining pp. 601-606 (2003)
9. Y. Zhai and B. Liu: Web Data Extraction Based on Partial Tree Alignment. The 14th International Conference on World Wide Web pp. 76-85 (2005)
10. C.-H Chang and S.-C Lui: IEPAD: Information extraction based on pattern discovery. The 10th International Conference on World Wide Web pp. 223-231 (2001)
11. Y. Zhai and B. Liu: NET - A system for extracting Web data from flat and nested data records. The 6th International Conference on Web Information Systems Engineering (2005)
12. D.S. Hirschberg: A linear space algorithm for computing maximal common subsequences. Communications of the ACM V.18, 6, pp. 341-343 (1975)
13. K.-H. Yang, J.-M. Chung and J.-M. Ho: PLF: A Publication List Web Page Finder for Researchers. The 2007 IEEE/WIC/ACM International Conference on Web Intelligence (WI-2007) (2007)
14. C.-C. Chen, K.-H. Yang and J.-M. Ho: BibPro: A Citation Parser Based on Sequence Alignment Techniques. The IEEE 22nd International Conference on Advanced Information Networking and Applications (AINA-08) (2008)