2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology

Parsing Publication Lists on the Web

Kai-Hsiang Yang[1], Jan-Ming Ho[2]

[1] Department of Mathematics and Information Education,
National Taipei University of Education
Email:{khyang@tea.ntue.edu.tw}
[2] Institute of Information Science, Academia Sinica,Taipei, Taiwan
Email:{hoho@iis.sinica.edu.tw}

ABSTRACT

Researchers usually present their publication records (we call citation records in this paper) on publication lists on the Web, which could be an important data source for many applications to collect more publication records than from some digital libraries, such as DBLP. However, it is still not easy to design an algorithm to extract citation records from publication lists because of the diversity of page layouts and citation formats. In this paper, we propose an automatic approach to extract citation records from publication list pages by utilizing two properties. First, citation records are usually represented as nodes at the same level in the DOM tree. Second, citation records in the same page are presented by similar HTML tags. Extensive experiments are conducted to measure the effects of all parameters and system performance. Experiment results show that our approach performs stable and well (with 86.2% of F-measure on average).

Keywords

Web mining, citation extraction, data extraction.

1. INTRODUCTION

Citation records are essential to many applications such as the topic search, academic network analysis, venue ranking, citation analysis etc., so how to collect more citation records is definitively a critical issue. Today, the scientific publications on the Web have become an important resource for collecting citation records. Many researchers usually create their own publication lists on the Web for many reasons, such as describing their researches and contributions, or announcing their new papers before they are formally published on journals or conferences.

Parsing publication list pages can retrieve many up-to-date research results without any human intervention. To design an approach for this goal, finding and keeping track of researchers' publication list pages is the first step. In our previous work [12], we had proposed a system called "Publication List Web Page Finder" (PLF). With the help of PLF, we can easily collect many researchers' publication list pages. However most pages are crafted manually by researchers themselves and page layouts and citation formats are quite different. Some researchers like to add images or descriptions to show the importance of each record.

In this paper, we propose an approach to extract citation records from publication list pages based on the following properties. First, most citation records are represented as nodes at the same level in the DOM tree of pages. Second, most citation records in the same page are presented by similar formats, such as similar punctuation sequences, which are used in our approach to identify citation records. By using these two properties, our approach first analyzes the DOM tree and find out a tree level where nodes are most likely to represent citation records. To estimate whether a node is represented as a citation record, our previous work "BibPro" [13] is applied to calculate the probability, which was designed for parsing a citation record into several fields (e.g., author, title, venue, etc.). When a string of a node is given, BibPro can output the probability that the given string is a citation string, hence we can find out one tree level in the DOM tree where citation records exist.

Experiment results show that the system performance dependents on the structure and our approach can provide better results than the MDR system and performs stable (84.5% of F-measure on average), and a majority (around 90%) of publication list pages in our dataset can be correctly extracted with acceptable F-measure (from 75% to 90%).

2. RELATED WORK

This work is highly related to the information extraction (IE) systems that try to provide robust and flexible ways to assist in extracting interesting data from Web with less labor costs. Much research has been con-ducted to extract regular patterns from semi-structured web pages. Most of this research is related to wrapper generation. A wrapper is a program that extracts records from Web pages automatically, and basically there are two principal methods to generate wrappers: 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] and WIEN [6]. Because of the intensive labor work of previous two approaches, some automatic extraction approaches are therefore proposed. Chang et al. proposed IEPAD (Information Extraction based on Pattern Discovery) [9] 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 [7], 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) [8] which is based on partial tree alignment. NET [10] improves DEPTA so that it can handle nested tables. The systems proposed by Liu et al. mainly focus on mining data record formed by table and related tags such as <tr> or <td>. However, in our problem, there are a large amount of publications that are not presented by using table related tags, so their systems are not suitable for solving our problem. ViPER [11] is an enhanced version of MDR and DEPTA with the following improvement. First, they rank potential repetitive patterns with user's visual perception. Second, data records are aligned through multiple sequence alignment technique.

3. Method

Figure 1 shows the architecture of our proposed system. Our system can be divided into two components: (1) Citation Record Candidate Finder and (2) Citation Record Filter. The goal of the Citation Record Candidate Finder is to generate a DOM tree for a given publication list page and apply the citation parser, "BibPro", that we proposed to assign each node a probability score that the node represents a citation record. By calculating the probability scores, our approach can determine one level in the DOM tree where citation records exist.

Since not all nodes at the found level are citation records, the second component, Citation Record Filter, applies two filters to filter out irreverent nodes. In the following sections we will provide much more details of each component.

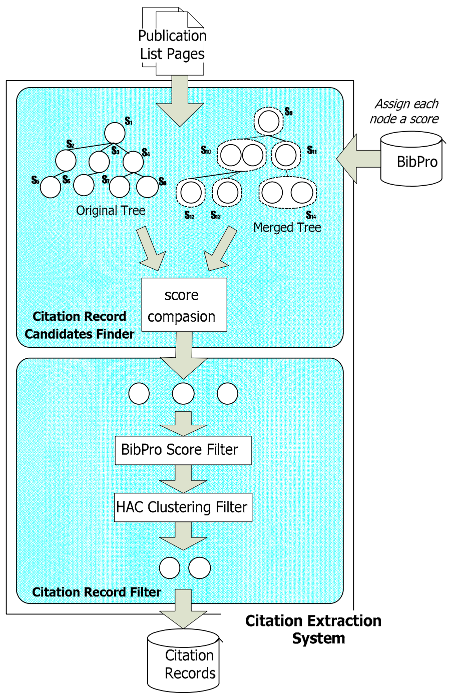

Figure 1. System architecture

3.1 BibPro

Bibro [13] is a template-based citation parser, and the key idea of BibPro is using the order of punctuation marks and reserved words in a citation string to represent its citation style. For a given citation string, BibPro encodes it as a protein sequence, which preserves citation style information. We collected many different styles of citation records from the Web and generated more than 4000 different citation styles in BibPro. When parsing a given citation string, BibPro will generate its sequence and then match it with all sequences of citation styles by using BLAST, which is a popular protein sequence alignment tool. Then BibPro assigns a score as the probability that the given string is a valid citation string.

3.2 Citation Record Candidate Finder

The goal of Citation Record Candidates Finder is to find out a tree level in the DOM tree as the citation candidates. BibPro is first applied to assign a score to each node of the DOM tree. Then for each level *i*, a score is calculated by the following formula:

$$LEVEL\_SCORE(i) = \sum_{j=1}^{N} SCORE(NODE_j) \times PRIORITY(NODE_j)$$

N is the total number of nodes at level *i*, $SCORE(NODE_i)$ is the score of each node and $PRIORITY(NODE_i)$ is a weight to show whether the node has similar styles as sibling nodes, where the weight is 2 if the node has a sibling node with similar sequence. Otherwise, its default value is 1.

By comparing the level scores, the level in the DOM tree that has highest score is chosen and all nodes at the level are regarded as citation record candidates.

3.3 Citation Record Filter

After finding citation record candidates, two filters are then applied to filter out nodes at the chosen level that are not citation records. The first filter is based on the scores assigned by BibPro, where we define a threshold to filter out citation record candidates whose scores are lower than the threshold.

The second filter is based on the Hierarchical Ascendant Classification (HAC) algorithm. The idea is that the nodes representing true citation records should have similar punctuation sequences and are supposed to form an independent cluster, while irreverent noises will be excluded. For given two nodes *r* and *s*, their similarity value is defined as the matching ratio between protein sequences of node *r* and node *s*. The pair-wise similarity between two nodes is defined as:

$$Similarity(r,s) = \frac{S_{r,s}}{L_r + L_s},$$

Where $\boldsymbol{S_{r,s}}$ is the similarity score between node ***r*** and ***s***, which is computed by global alignment of sequences in BibPro. $\boldsymbol{L_r}$ and $\boldsymbol{L_s}$ are the lengths of sequences of node *r* and *s*, respectively. To define the similarity between two clusters, we adopt the average linkage clustering, which means similarity between two clusters is defined as the average similarity value between all pairs of nodes in two clusters. We also define a similarity threshold to determine which nodes should be clustered together. After clustering all nodes, we choose the biggest group as the output citation records.

4. EXPERIMENTS

In our experiments, we created two datasets by developing programs to collect researchers' publication list pages from the Web. Dataset(I) consists of 60 publication list pages collected randomly from the computer science faculties of Pennsylvania, Berkley and Stanford universities, and which is used as the training dataset for tuning the similarity thresholds in our approach. For dataset(II), we first collected the computer science faculties of top 12 universities in America based on Computer Science Ranking, and then randomly selected 240 researchers from those collected faculties. After manually checking, only 187 of them have publication list pages. Some publication lists with the PDF formats are ignored since our algorithm is involved with the HTML processing. We manually label the answer sets for dataset(I) and dataset(II), while dataset(II) is used for our experiment evaluation, and there are no overlapped pages between dataset(I) and dataset(II).

To measure the system performance, we define the evaluation metric as follows. 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 (P_i) and recall (R_i) values for page *i* are expressed as follows:

$$P_i = \frac{|CE|}{|E|} \text{ and } R_i = \frac{|\mathrm{CE}|}{|\mathrm{C}|},$$

The overall precision *P* and recall *R* are averages of all P_i and R_i of publication list pages:

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

The F-measure, as a combination of precision *P* and recall *R*, is measured with the following formula:

$$F - measure = \frac{2 \cdot P \cdot R}{P + R}$$

4.1 Dataset preprocessing

The first step that we have to do is to verify and fix some error tags within Web pages, which might result in generating wrong DOM tree and low system performance. We apply an open source program, JTidy, to clean malformed HTML tags, and evaluate system performance. Table 1 shows that JTidy increases the F-measure performance on dataset(I). Hence we used the JTidy-processed dataset in our following experiments.

Table 1: Performance on raw and processed dataset(I) (without merging procedure and filtering)

	Average Precision	Average Recall	F-Measure
Raw Data	80.7%	86.08%	83.30%
JTidy-processed	82.53%	89.46%	85.85%

4.2 Tuning threshold for filtering

The goal of this experiment is to study the effect of the filtering threshold on system performance. The filtering threshold is set from 0 to 30 with an increment of 5. Figure 22 shows the performance results, which includes the precision, recall, and F-measure. It is obvious that using threshold 0 provides the highest recall value (90.32%), while using threshold 20 reaches the highest precision (88.3%). However, threshold 15 produces the best F-measure (87.05%). We set the filtering threshold to 15 in the following experiments.

4.3 Tuning threshold for HAC

In this section, we study the effect of HAC thresholds. We set the clustering thresholds from 0.4 to 1.3 with an increment of 0.1 in the HAC process, and the performance results are shown in Figure 3.

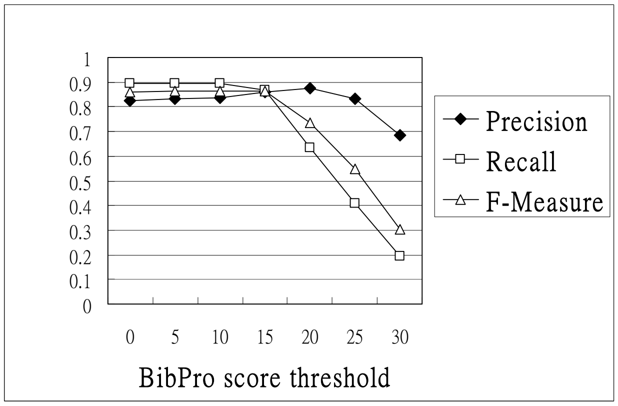

Figure 2. Performance for different thresholds

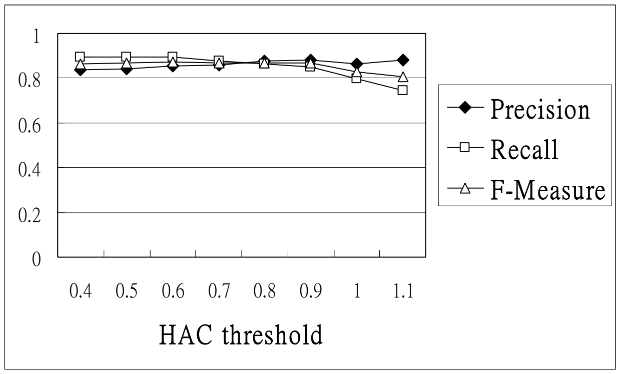

Figure 3. Performance for different thresholds in HAC

In Figure 3, we can easy to see that the recall value decreases when threshold increases. Highest recall value reaches 89.45% at threshold 0.4, while highest precision reaches 88.24% when threshold falls upon 0.9. Threshold 0.6 produces both highest recall and F-measure (87.4%), so it is chosen as our setting for HAC threshold.

4.4 Comparison with other approaches

In this section we show the system performance of combining both filtering and HAC. Meanwhile, we compared our system with the MDR (proposed by Liu et al. [7]), which is an information extraction system designed to extract repeated patterns from Web documents. For our approach, the best thresholds of filtering and HAC tuned by using dataset(I). In MDR, a threshold is required to measure the edit distance similarity, so we tune the threshold by using dataset(I) as shown in Figure 4, and choose the best one as the threshold for MDR.

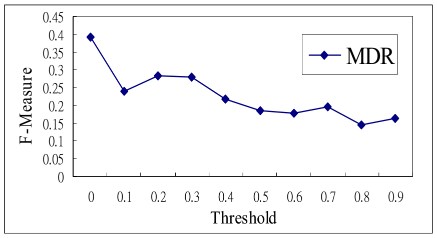

Figure 4. Tuning threshold of MDR using dataset(I)

Table 2. System performance comparison on dataset(II)

	Average Precision	Average Recall	F-Measure
F+H	83.33%	85.69%	84.5 %
MDR	25.29%	17.19%	20.46%

Table 2 shows the system performance of our approach and the MDR system. (F and H stands for filtering and HAC respectively). From Table 2, we can easily see that our approach can provide 84.5% F-measure performance, while the MDR system does not perform well on this experiment. The main reason is that the MDR system is designed for extracting general web records which are organized with tables, while most citation records are not organized with tables. From this comparison we can realize that extracting citation records from web pages is not a trivial problem.

5. Conclusion

Our motivation is to develop an automatic approach to extract all citation records from researchers' publication list pages. This task is interesting and still challenging, because many publication list pages are crafted manually by researchers themselves, and the page layouts and the citation formats are quite different depending on the different researchers' affinities. In this paper, we propose an approach that is based on two properties of citation records. First, most citation records are represented by nodes at the same level in the DOM tree of pages, and second, most citation records in the same publication list page are presented with similar formats, such as similar punctuation sequences. Moreover, experiment results reveal that system performance depends on the structure of the publication list pages, and our approach can provide better results than MDR system, and performs stable (84.5% of F-measure on average), and a majority (around 90%) of publication list pages in our dataset can be correctly extracted with acceptable F-measure (from 75% to 90%). We believe that more effort in this research area would be worthwhile.

6. Acknowledgements

This work was supported in part by the National Science Council of Taiwan under grants NSC 98-2221-E-001-010-MY3.

7. 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 (Agents-99)*, 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] N. Kushmerick "Wrapper induction: efficiency and expressiveness Artificial Intelligence", *Artificial Intelligence*, 118(1-2):15-68, 2000.

[7] B. Liu, R. Grossman, and Y. Zhai "Mining data records in Web pages", *the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD-2003)*, pp. 601-606, 2003.

[8] Y. Zhai and B. Liu "Web Data extraction based on partial tree alignment", *the 14th International Conference on World Wide Web (WWW)*, pp. 76-85, Japan, 2005.

[9] C.-H Chang and S.-C Lui "IEPAD: information extraction based on pattern discovery", *the 10th International Conference on World Wide Web (WWW)*, pp. 223-231, Hong-Kong, 2001.

[10] Y. Zhai and B. Liu "NET - A system for extracting Web data from flat and nested data records", *6th International Conference on Web Information Systems Engineering (WISE-05)*, 2005.

[11] K. Simon, and G. Lausen "ViPER: augmenting automatic information extraction with visual perceptions", *the 14th ACM international conference on Information and knowledge management*, pp. 381-388, 2005.

[12] 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)*, Nov. 2007.

[13] 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),* March 25-28 2008.