Support Vector Mechanism Based Citation Parser

G. Guru Brahmam
M.Tech Software Engineering,
Vardhaman College of Engineering,
Hyderabad, India.

A. Bhanu Prasad
Associate Professor, Department of IT,
Vardhaman College of Engineering,
Hyderabad, India.

Abstract

The huge amount of researchers' publication list pages is available on the Web, which could be an important resource for many value-added applications, such as citation analysis and academic network analysis. Bibliographical references that appear in journal articles can provide valuable hints for subsequent information extraction. We describe our statistical machine learning algorithms for locating and parsing such references from HTML medical journal articles. Reference locating identifies the reference sections and then decomposes them into individual references. We formulate reference locating as a two-class classification problem based on text and geometric features. We implement and compare two reference parsing algorithms. One relies on sequence statistics and trains a Conditional Random Field. The other focuses on local feature statistics and trains a Support Vector Machine to classify each individual word, and then a search algorithm systematically corrects low confidence labels if the label sequence violates a set of predefined rules.

Keywords: Document Object Model (DOM), Support Vector Machine (SVM), Conditional Random Field (CRF).

1. Introduction

Citations play important roles for both the rhetorical structure and the semantic content of the articles, and as such, citation information has shown to benefit many text mining tasks including information retrieval, information extraction, summarization, and question answering. 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. Automatic metadata extraction from medical journals is key to the affordable creation of citations in MEDLINE®, the flagship database of the U.S. National Library of Medicine (NLM), containing over 17 million records and searched over 3 million times per day worldwide. Analyzing references, which are citations usually placed at the end of scientific publications, is an important pre-processing step for generating several MEDLINE bibliographic data items, e.g., identifying Comment-On/Comment-In articles (commentary article pairs), assigning MeSH (Medical Subject Heading) indexing terms through analyzing the MeSH terms already assigned to the cited articles, and many others. We therefore formulate reference location as a two-class classification. After rendering the HTML article in a Browser, geometric and text features are extracted from the zones in the HTML article, and an SVM classifier is used to classify these zones as either *reference zones* or *non-reference zones*. The third observation in the previous paragraph is a useful constraint which can expedite the process and increase its reliability.

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. CiteSeer is a well-known and successful citation indexing system developed at NEC Research Institute[13]. CiteSeer uses Web search engines and

heuristics to crawl the Web and download PDF and PostScript articles. After converting to text, CiteSeer uses heuristics to locate the reference section, and then parses each reference to extract fields such as title, author, year of publication, and so on. Similar systems include ISI Web of Knowledge [26] and Google Scholar [27]. We focus on HTML articles. By rendering the HTML articles in a Web browser (e.g. Microsoft Internet Explorer), geometric information (locations and sizes of zones) can be extracted, and these are important features for reliably locating bibliographical references. There does not appear to be work reported on specifically locating references appearing in HTML articles. A related problem, which has been carefully studied recently by several researchers, is *mining data records from Web pages*. Data records are a list of similarly structured items, e.g., a list of products on sale. Liu et al. exploit the Web page structure and mostly depend on string matching of HTML tag sequences to detect data records [14]. Zhai and Liu extended this work, and used visual information and tree matching to detect data records, and then designed a partial tree alignment algorithm to align data records, and extract information from each one [23]. Reis et al. assumed that certain Web page groups share a common format and layout characteristics, and designed a tree matching algorithm to extract news content from news pages [19]. These data record mining algorithms have been used to extract consumer product reviews, news, Internet forum postings, and several other applications. These algorithms are mostly based on HTML DOM (Document Object Model) tree and HTML tags. The duplication of similar DOM tree structures is the primary cue for locating and aligning data records and for extracting information from them. In our reference locating problem, the text is a much more reliable feature compared to HTML tags. We therefore formulate the reference locating as a two-class classification based on geometric and text features. Reference parsing, on the other hand, has received far more attention. Existing reference parsing methods can be generally divided into two categories: *rule based* methods and those based on *machine learning*. Rule based methods usually rely on a set of rules based on a domain expert's observation. Chowdhury [4] and Ding et al. [7] have used *template mining* techniques. Templates are manually crafted to summarize the recognizable patterns formed by either the data and/or text surrounding the data. A set of rules is usually associated with the templates, and when text matches to the templates, the data are extracted according to the rules. Day et al. [5, 6] extended the template mining approach, and used INFOMAP, a hierarchical framework, for knowledge (template) representation. Huang et al. used a gene sequence alignment tool, BLAST (Basic Local Alignment Search Tool), to extract citation metadata [10]. Journal publishers usually require authors to strictly follow predefined citation styles, and careful editorial checking and correction are usually conducted before publishing. Therefore, for a small set of journals, rule-based methods can be very successful. On the other hand, rule-based methods require domain experts to design the rules and maintain them over time. This approach also prevents adaptability and it is difficult to tune the system due to the rigidity of the rules. As mentioned, for MEDLINE data, over 5,200 journals from hundreds of publishers need to be processed. Hence, automatic reference parsing through rule-based methods poses a challenge due to the large variation of citation styles. In contrast, machine learning approaches exhibit good adaptability by automatically learning the knowledge from training samples, and have therefore attracted a great deal of interest. Parmentier and Belaïd developed a *concept network* to hierarchically represent and recognize structured data from bibliographic citations [16]. Besagni et al. took a bottom-up approach based on Part-of-Speech (PoS) tagging [2]. In this approach, basic tags, which are easily recognized, are first grouped into homogeneous classes. Confusing tokens are then classified by either a set of PoS correction rules or a structure model generated from correctly detected records. Hidden Markov Model (HMM), a successful machine learning tool for information extraction from sequences, has also been studied for parsing references, e.g., Takasu applied HMM for parsing erroneous references [22]. Conditional Random Field, another popular sequence model, is recently reported to achieve better performance compared to HMM [18]. We have therefore included CRF as one of our reference parsing methods. Another frequently adopted machine learning method for information extraction is the Support Vector Machine (SVM) classifier. Han et al. took a two-stage approach for metadata extraction from the header part of research papers [9]. Okada et al. combined SVM and HMM for bibliographic component extraction [17]. We have implemented a reference parsing algorithm, which uses the SVM to classify each individual word. Intuitively, adjacent words in a reference usually are more likely to belong to the same entity. To exploit this important local dependency, we

use not only the features extracted from the word itself, but also those extracted from its neighbours.

2.1 Single word classification using SVM

From each word, 15 features are extracted. The first 14 are the same binary features listed in Table 3. The 15th feature is the normalized position, i.e., the position of the word normalized by the total number of the words in the reference. Intuitively, we expect adjacent words in a reference to usually have a higher probability of belonging to the same entity. In order to utilize these local contextual dependencies, the features used for the classification are extracted from not only the word itself, but also from its neighbours. As done for reference locating, we adopted LibSVM with RBF kernel function for this single word classification. Similarly, the two parameters, (penalty parameter of the errors) and $C\gamma$(RBF parameter), were also selected through exhaustive grid-search using cross-validation on training samples.

2.2 Search algorithm for finding optimal label sequence

Due to the high accuracy of single word classification, most references can already be correctly parsed. For those that do not pass the global rule test, nearly all of them are close to the correct label sequence with only a few words mislabelled. The goal is then to identify and correct those mislabelled words. We present a systematic search algorithm guaranteed to find a label sequence that is valid (obeys the global rules) and is most-likely (has the highest probability). The key to finding the most-likely and valid label sequence is then to search possible *label sequence modifications* in the ascending order of their costs. The search stops at the first label sequence, which obeys the global rules. Because there are $1-NM$ possible modifications, it is computationally prohibitive to calculate costs for all possible modifications and then sort them. We present an algorithm which enumerates sequence modifications in ascending order of their costs. We first calculate the costs for all) possible *single-token modifications* (only one word's label is modified) and sort them in ascending order. This is not computationally expensive. We arrange these) single-token modifications in the middle line of Figure 2 (marked with a dashed bounding box) in ascending order of their costs. <1> indicates the single-token modification with the minimum cost, and so on. It is easy to see that the first and second sequence modifications must be the first two single-token modifications. In each subsequent column we list all possible *multi-token modifications*, which are all possible combinations of the previous single-token modification and all other previous single- and multi- token modifications. For example, in Column 3, the previous single-token modification is <2>, and there is only one other modification, i.e., <1>, so there is only one multi-token modification, i.e., <2,1>. Let us assume that <1> and <2> are the modifications to the same word, so the modification <2,1> is meaningless. We mark it with a dashed circle and abandon it. In Column 4, the previous single-token modification is <3>, and all other possible previous modifications are <1> and <2>, so we have two multi-token modifications, as shown in Column 4, <3,1> and <3,2>. Let us assume the cost of <3,1> is less than that of <4>, but the cost of <3,2> is greater than that of <4>, and therefore, we place <3,1> on top of <4> and <3,2> below <4>. Similarly, we create Columns 5, 6, and so on. In this example, <1>, <2>, <5> are assumed to be single-token modifications of the same word, and <3> and <4> are single-token modifications of the other two words. Meaningless multi-token modifications are marked with dashed circles. For each column, let us call the modifications above the single-token modification the *upper column*, and the modifications below the single-token modification the *lower column*. Although the modifications in each column are ordered, the modifications in the lower column may have higher cost than the modifications in the later columns. However, a key observation is that the modifications in an upper column must be smaller than those in the lower column and the later columns. This is the key for creating new columns dynamically and enumerating all modifications in ascending order of their costs. The algorithm is shown below

Algorithm

1. Calculate costs for all single-token modifications, and sort them in ascending order. N(M-1)
2. Test the first single-token modification. If it obeys the rules, go to the end, otherwise continue.
3. Test the second single-token modification. If it obeys the rules, go to the end, otherwise continue.
4. Create Column 3, and save all modifications into an ascending ordered list.
5. Repeat for K=3, 4… N (M-1)-1:
    a. Repeat:
    i. Pop up and test the first modification from the ordered list.
    ii. If it obeys the rules, go to the end, otherwise continue.
    b. Until single-word modification <K> is tested.
    c. Create Column K+1, and save the modifications into the ordered list.
6. Finish testing remaining ordered list.
7. End

It is clear that the algorithm is still an exhaustive search, but it searches from the label sequence generated by single word classification, which, in our case, is close to the correct solution. Most searches, therefore, terminate very quickly. Because the search is conducted in the ascending order of costs, it is guaranteed to find the most-likely modification that obeys the rules. In an actual implementation, it is of course better to set a limit on the maximum number of modifications to be tested to avoid lengthy computation. In our implementation, the search terminates after 10,000 modifications have been tested. In practical systems, if the search does not terminate when the limit is reached, this is an indication that the parsing may not be accurate.

3. Analysis

This has been clearly demonstrated by the single word classification experiments. Regardless of the number of training samples, the accuracies are significantly improved if combining the features extracted from the immediate left and right neighbors (45 features). Combining features from an additional two adjacent neighbors (75 features), on the other hand, achieves only slight accuracy improvements. This is in agreement with many studies of statistical sequence models, where usually only the first-order correlation is modeled, and the first-order Markov Chain is the underlying graphic model. Global rule correction is effective. We believe that the global rule correction is a good practical heuristic to correct minor errors. When it fails, it also serves as a good indicator for low confidence parsing. The article title contains the most heterogeneous text, and therefore is the most difficult entity to extract. Both CRF-parsing and SVM-parsing achieve the lowest accuracy in Title chunk identification. On the other hand, both algorithms achieve high accuracy (around 99%) for entities having distinctive features, such as Number, Volume, Year and Pagination. Comparing Tables 5 and 7, when training with 600 references, CRF-parsing and SVM-parsing essentially achieve the same overall performance: about 99% accuracy at word level and above 97% accuracy at chunk level. SVM-parsing missed only 3 Publication Years. SVM is a sophisticated classifier, which is expected to achieve better performance on entities having distinctive features. On the other hand, CRF achieves 1% higher accuracy on Title chunk identification. Titles contain heterogeneous text, i.e., having indistinctive features. It is likely that CRF, by modeling the entire sequence, has better chance to label them correctly. The performance may be further improved, if the advantages of SVM (sophisticated local classifier) and CRF (powerful sequence model) can be combined. Most references in our collection are citations to journal papers (Examples (a)~(g) and (k) in Table 1). There are few errors in this kind of "standard" references; even organizational authors (Examples (k) in Table 1) can usually be successfully labeled. Only a very small percentage of references are citations to reports and books (Examples (h)~(j) in Table 1), and our current algorithm finds it difficult to label their Unknown (<U>) entities. For the edited books especially, the long word sequence of the editors sometimes confuses the algorithms. Further research is warranted to solve this problem.

4. Conclusion

We have presented approaches for locating and parsing references in HTML medical journal articles. We formulate reference locating as a two-class classification, and have demonstrated that text and geometry are very reliable for locating references, and an SVM classifier based on these features can achieve near 100% accuracy. The first order correlation between reference words is important contextual information, and must be used in reference parsing algorithms. We implemented and compared two reference parsing algorithms. CRF-parsing focuses on modelling the word sequence with Conditional Random Fields, and SVM-parsing concentrates on local single word classification. The overall performance of these two approaches is about the same: above 97% accuracy at chunk level.

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. Borgho®, and P. Chevalier: Towards sophisticated wrapping of Web-based information repositories. The 5th International RIAO Conference, Mon-treal, 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 exible learning system for wrapping tablesand lists in HTML documents. The 11th International World Wide Web conference (2002)

[5] C.-N. Hsu and M.-T. Dung: Generating nite-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 ACMSIGKDD 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 °at 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)
[15] A.K. McCallum, *MALLET: A Machine Learning for Language Toolkit*, http://mallet.cs.umass.edu. 2002.
[16] F. Parmentier and A. Belaïd, "Logical structure recognition of scientific bibliographic references," *Proc. ICDAR*, pp. 1072-1076, 1997.
[17] T. Okada, A. Takasu and J. Adachi, "Bibliographic component extraction using support vector machines and hidden Markov models," *Proc. ECDL*, pp. 501-512, 2004.
[18] F. Peng and A. McCallum, "Accurate information extraction from research papers using conditional random fields," *Proc. of Human Language Technology Conference*, pp. 329-336, 2004.
[19] D.C. Reis, P.B. Golgher, A. S. Silva, A. F. Laender, "Automatic Web news extraction using tree edit distance," *Proc. WWW*, pp. 502-511, 2004.
[20] F. Sebastiani, "Machine learning in automated text categorization", *ACM Computing Surveys*, vol. 34, no. 1, pp. 1-47, 2002.
[21] C. Sutton and A. McCallum, "An introduction to conditional random fields for relational learning," book chapter in *Introduction to Statistical Relational Learning*. Edited by L. Getoor and B. Taskar. MIT Press. 2006.
[22] A. Takasu, "Bibliographic attribute extraction from erroneous references based on a statistical model," *Proc. JCDL*, pp. 49-60, 2003.
[23] Y. Zhai, B. Liu, "Structure data extraction from the Web based on partial tree alignment," *IEEE Tran. Knowledge and Data Engineering*, vol. 18, no. 12, pp. 1614-1628, 2006.
[24] J. Zou, D. Le, G.R. Thoma, "Structure and Content Analysis for HTML Medical Articles: A Hidden Markov Model Approach," *Proc. DocEng*, pp. 119-201, 2007.
[25] J. Zou, D. Le, G.R. Thoma, "Extracting a sparsely-located named entity from online HTML medical articles using support vector machine," *SPIE Proc. Document Recognition and Retrieval (SPIE-DR&R)*, pp. 68150P (1-10), 2008.