Computationally Effective Algorithm for Information Extraction and Online Review Mining

Boris Kraychev
Faculty of Mathematics and Informatics
Sofia University St. Kliment Ohridski
Sofia, Bulgaria
boris.kraychev@uni-sofia.bg

Ivan Koychev
Faculty of Mathematics and Informatics
Sofia University St. Kliment Ohridski
Sofia, Bulgaria
koychev@fmi.uni-sofia.bg

ABSTRACT

The World Wide Web provides continuous sources of information with similar semantic structure like news feeds, user reviews and user comments on various topics. These sources are essential for the goal of online opinion mining. The paper proposes a computationally efficient algorithm for structured information extraction from web pages. The algorithm relies on a combination of analysis of structured data and natural language processing of text content. It maps HTML pages containing news, reviews or user comments to a custom designed RSS feed like structure. Such information usually includes the textual opinions, and factual information like publication date, product price, author name and influence. Due to the real time nature of the data sources the computational complexity of such a solution should be linear or close to linear. The computational complexity of the proposed algorithm is linear. In comparison similar previously published approaches have complexity no smaller than $O(n^2)$. Further we conduct experiments with real world data that achieves extraction accuracy of 84% to 92% which is comparable to the recent results in this field. Finally the paper discuses the results of the experiment and shares gained experience that can be useful for applying the algorithm in other domains.

Categories and Subject Descriptors

H.3.3. Information Search and Retrieval, Retrieval models

General Terms

Algorithms, Performance.

Keywords

Information Extraction, Wrapper, Tree Matching

1. INTRODUCTION

The Web is the largest source of publicly available personal opinions, comments and reviews. The number of new pages published each day on the Internet grows exponentially and so does the quantity of the available information. With the expansion of the Web 2.0 and the mobile Internet devices people are encouraged to share their view on products, news articles, books and any kind of subject. It becomes practically impossible to manually gather the important information and monitor new trends in a timely manner without an automated support. The goal of this paper is to provide a solution for web page metadata extraction like author, publication date, main content, comments feed etc. with linear computational complexity. Programs that perform such Information Extraction are called *wrappers* [Zhai and Liu 2005b].

Each web page can be represented as a hierarchical structure of elements, which usually are around 5000 per page. Our aim is to simplify and align this structure to a much smaller structure containing the desired metadata, which is similar to an RSS feed, containing 3 to 5 elements per item. Our key approach is to exclude irrelevant elements by semantic estimation of the textual content. Our wrapper is equipped with natural language processing tools like Part Of Speech (POS) tagger, Dictionary of sentiment aware words and Thesaurus tool.

To our knowledge, the existing tree alignment algorithms used by wrappers have a computational complexity of $O(n^2)$ or higher: [Zhai and Liu, 2005b], [Ferrara and Baumgartner, 2011]. Tree alignment problems have been studied in compiler theory, where a similar problem of discovering similarity in program code structures has to be solved [Baxter et Al., 1998] with linear computational complexity. We apply similar approach in our Information Extraction wrapper and we define a hash function that compares linearly the candidate root elements from the web page to the desired root element of the metadata tree.

2. RELATED WORK

The problem of wrapper construction has been studied in systems like Stalker [10] which proposes symbolic method for information extraction. Unfortunately it strictly depends on the website syntax and needs an update after each modification of the HTML design. [14] proposes an algorithm for measurement of the distance between two trees. It has a computational complexity on the order of the product of the sizes of both trees. The method is dependent on the internal tree topology, which is not necessarily needed for content driven information extraction. [15] propose an algorithm for Web data extraction based on partial tree alignment. Their solution has a quadratic complexity and can be improved under certain constraints. [6] and later other authors like [3] work on syntactically based algorithms, which are also bound to the exact web page structure. Recent studies are exploring the automatic wrapper adaptation [4] and use of Markov Logic Networks [12].

An overview of recent applications of tree alignment techniques can be found in [13] and [5].

3. METHODOLOGY

Our idea of information extraction can be explained as follows: First of all we fetch the web page content and parse it to a tree-like hierarchical structure. A standard news article with comments contains more than 5000 nodes. At a second step we apply Data Type assignment to each node with the goal to identify interesting candidates for the desired metadata like personal names, dates, textual content, and sentiment aware textual content. Finally we apply the tree alignment algorithm, which compares nodes of the web page tree and the resulting metadata tree via hash function.

The limitations of the algorithm are related to the HTML validity and the web pages structure of the crawled data. This problem is discussed in more details in paragraph 3.4.

3.1 Data Collection

The web pages being analyzed are gathered by a crawler, forwarded to a general purpose HTML parser to obtain their hierarchical structure and then processed by a data-type analyzer to assign on all leaf nodes one of the following types:

- *Date statement* – e.g. Added by {Author} on 20/11/2011;
- *Comment/Review Candidate* - Statement with presence of sentiment aware words.
- *Main Text Candidate* - Longest text of the page, not being a link and containing morphologically complete sentences.
- *Name Entity* - Detected by regular expression and list of known personal names.
- *Extraneous* - Leafs with no particular data type assigned.

Once all leafs are assigned a type, we delete: all ones of type *Extraneous*, all nodes not leading to one of the remaining leafs; and all nodes having a single child.

On next step to each node a triple of numerical values is assigned. The triple elements are:

- Number of descendant nodes (and leafs)
- Number of words in descendant nodes
- Sentiment score

The values are then used as an input for the tree matching via hash function.

The value is calculated with a bag-of-words technique (lexicon of sentiment aware words) [7]. The sentiment aware lexicons can be obtained by the methods described by [7]. The value is used to discover user reviews and comments.

- *Factual score* - the factual score is calculated analogically to the sentiment score with automatically built lexicon of factual aware words. The applied method for lexicon construction is described by [8].
- *HTML tag* - the original HTML tag of the node.
- *Number of similar adjacent nodes* - the number of adjacent nodes with identical HTML tag and similar Sentiment and Factual scores. It is used to discover repeating structures on the page like user comments.

An optional technique that could be applied to achieve cleaner results is to double fetch the page and exclude content changing nodes as potentially enclosing advertizing blocks or irrelevant information like examples of unrelated reviews.

3.2 Alignment with the Wrapper Tree

Our goal is to automatically extract a Rich Site Summary (RSS) feed of online reviews or comments from a Web page. The desired output has simple tree structure that has to be mapped to the one obtained with the method described in previous section.

The conventional algorithms like Simple Tree Matcher and adapted center star [16] have computational complexity of $O(n^2)$ or higher. Our approach is to reduce significantly the computational complexity by applying a hash function that estimates the sub-trees under each node. The method has been applied in compiler design [1] and is pertinent for application to information extraction area too. Given that the hash function distributes an N-node tree into B buckets the complexity of the algorithm would be $O(N/B)$. Depending on the hash function design we could ignore the leaf order or the tree internal structure and compare trees only by the contained information leafs. For example the result of the function could be the sorted list of leaf data types in the underlying sub-tree of a given node. For example 'ACD' could be the hash value for all ancestors of exactly one *Author* leaf, one *Comment* leaf and one *Date* leaf, ignoring their exact order in the tree. Collisions are resolved by preference of the node with the closest height and first horizontal occurrence.

3.3 Computational Complexity of the Algorithm

The data type assignment to the leaf structure has linear complexity, because it is done independently for each leaf in a sequential order. The complexity of the text analysis itself could be non-linear, but this does not concern the current algorithm and by the other hand linear scoring algorithms have acceptable performance for text classification.

The tree reduction phase which cleans leafs with unnecessary data types and parent nodes not leading to meaningful leafs has also linear complexity and is done from the leafs towards the root node.

Finally the hash comparison of the resulting tree with the wrapper output tree is also in the order of N e.g. has linear complexity, because our hash function depends on the number of the tree's leafs and is well in the order of N. The complexity of the algorithm phases is estimated as follows:

- Data Type Assignment - $O(N)$
- Tree reduction - $O(N)$
- Tree alignment - $O(N/B)$

Since B is on the order of N it follows that the overall computational complexity of the algorithm is linear. We exclude the assessment of the HTML parsing complexity, because it is not in the subject of our research and many conventional solutions are available under various licensing policies.

3.4 Limitations and Solutions

The ideal input for our Wrapper are valid XHTML pages or HTML pages with clean tree-like hierarchy of the elements. Definitely there are many exceptions on the Web, but they tend to use older version of HTML and these web sites are losing their reputation. There are many techniques to cope with the problem of erroneous HTML structure and this is done successfully in most of the freely available HTML parsers.

The next set of limitations is related to the hash function. The detection and matching to repetitive sub-tree clones has to be resolved by adding additional signs in the hash function's result, for example the addition of parentheses is one possible solution.

Not all web sites have identical structure of main text and a list of comments or reviews, but without losing our linear complexity we could add more possible matching trees to the tree alignment phase. This would increase the complexity up to $O(M \times N)$ for M being the number of matching trees which is smaller than a given constant and could be ignored. The set of matching trees could describe a collection of blog page patterns and social media page patterns, although depending on the data type assignment quality this could be unnecessary.

4. EXPERIMENT

The purpose of our experiment is to demonstrate the extraction performance of the algorithm with real world data.

We chose the Open Directory Project as a source of websites which are potential candidates for our Wrapper. The list of included websites in the project can be downloaded and used as a seed for further web crawling, parsing and indexing. The websites are also classified in sections and languages. At the time of the experiment the directory contained approximately 4.9 millions of websites.

4.1 Experiment Design

Our wrapper was applied to two test sets of websites. The first result estimation was based on 100 manually selected websites in English language, split equally to negative and positive classes. The second set of experiments was designed to estimate larger number of websites, extracted from particular sections from the seed list.

Given the number of seed websites and the projected number of web pages to crawl, we chose Apache Hadoop as a computational platform.

4.2 Results

The first set of experiments was the classification of 100 websites where the presence of successfully extracted feed from a web page classifies the whole domain name as a positive result. The final classification can be found in the following table 1:

Table 1. Confusion matrix for the first set of the experiment

	Actual positive	**Actual negative**	**Total**
Classified Positive	81	5	86
Classified Negative	11	3	14
Total	92	8	100

The second set of experiments was targeted at categories of websites which are supposed fit the model of main story (review or article) and readers' comments. We chose the News section from the Open Directory Project (www.dmoz.org/News) which contained 7947 sites at the time of the experiment. The websites in this section are supposed to present news articles with their author and date of publication on a dedicated page, eventually containing repetitive user comments. Our algorithm found positive pages in 6853 sites or 86.23%. Manual verification on 100 randomly selected websites among the test set showed the results in table 2.

Table 2. Confusion matrix for the second set of the experiment over randomly selected 100 sample websites

	Actual positive	**Actual negative**	**Total**
Classified Positive	43	2	45
Classified Negative	7	48	55
Total	50	50	100

5. DISCUSSION

The proposed algorithm shows and accuracy of 91% in the manually selected test set, while the accuracy drops down on the randomly constructed test set to 84%. Which is comparable to the information extraction accuracy achieved in more structured data by [11].This could be due to at least two reasons. *First* of all the manually constructed test set was chosen by authors' preferences and willingly or not, the authors may have given preference to some particular style of web pages, which could be easier to parse or analyze. The *second* reason that could explain the difference is the fact that the websites present in the Open Directory Project may not follow the modern trends and some of them have kept their archaic (in the terms of the Web) HTML design which is outdated and unsuitable for modern parsers. Other observation is that although there is a large number of websites declared to deliver 'Breaking News', few of them are proposing Socialized functions like commenting and sharing reader's opinions. This poses more difficulties to the pattern recognition process.

The actual classification accuracy is strictly based on natural language processing methods like sentiment analysis, text mining of date fields and named entities (usernames, authors' names). The quality of this preliminary analysis is crucial for the final results and therefore has an impact over the assessment of the tree matching algorithm.

Other key point in the experiment is the design of the hash function. It determines the sensitivity of the tree alignment and its precision in the tree matching structure. Similar problems have been studied by [1] for detection of sub-tree clones in hierarchical structures.

6. CONCLUSIONS

The proposed method for information extraction relies on hash function design, which allows comparing and aligning similar trees with a linear computational complexity. The accuracy of the solution was demonstrated over a set of real world web pages. A key point in the research is the application of methods for sentiment analysis and text classification in order to pre-process the HTML data and assign high-level data types to the content.

The experiment was focused on the extraction of articles and product reviews, together with their user comments from web pages. This is, according to our persuasion, one of the most wide spread models of online opinion sharing. Further work may include the extraction of additional information like indications of the author's influence and extraction of complete text from individual pages.

7. ACKNOWLEDGMENTS

This research is supported by the SmartBook project, subsidized by the Bulgarian National Science Fund, under Grant D002-111/15.12.2008.

8. REFERENCES

[1] Baxter I.D., Yahin A., Moura L., Sant'Anna M., Bier L. 1998. Clone Detection Using Abstract Syntax Trees. *Proc. of ICSM'98*.

[2] Ciravegna F. 2000. Learning to Tag for Information Extraction from Text. *Proc. of the ECAI-2000*.

[3] Crescenzi V., Mecca G. 2004. Automatic Information Extraction from Large Websites. *J. ACM*, Vol. 51, Nr. 5 New York, NY, USA: ACM (2004) p. 731—779.

[4] Ferrara E., Baumgartner R. 2011. Combinations of Intelligent Methods and Applications – Springer.

[5] Ferrara E., Fiumara G., Baumgartner R. 2010. "Web Data Extraction, Applications and Techniques: A Survey", *Technical Report*.

[6] Freitag D. 1998. Information Extraction From HTML: Application of a General Learning Approach. *Proc. of the 15th National Conference on Artificial Intelligence* (AAAI-98).

[7] Godbole N., Srinivasaiah M., Skiena S. 2007. Large-scale Sentiment Analysis for News and Blogs, ICWSM.

[8] Grefenslette G., Qu Y., Evans and D.A.,Shanahan J. G. 2006. Validating the Coverage of Lexical Resources for Affect Analysis and Automatically Classifying New Words along Semantic Axes, Springer.

[9] Hassan A., Radev D. 2010. Identifying Text Polarity Using Random Walks, *Proceedings of the Association for Computational Linguistics.*

[10] Muslea I., Minton S., Knoblock C. A. 1999. A Hierarchical Approach to Wrapper Induction. *Proc. of the Intl. Conf. on Autonomous Agents (AGENTS'99)*, pp. 190–197.

[11] Peng F., McCallum A. 2004. Accurate Information Extraction from research papers using conditional random fields, *HLT-NAACL04*, p. 329-336.

[12] Satpal S., Bhadra S., Sundararajan S., Rastogi R., Sen P. 2011. Proceedings of the 20th international conference companion on World Wide Web, ACM New York, NY, USA.

[13] Tekli J., Chbeir R., Yetongnon K. 2009. "An overview of XML similarity: Background, current trends and future directions", *Computer science review, vol. 3, no. 3, pp.151-173.*

[14] Yang W. 1991. Identifying Syntactic Differences between Two Programs. Software Practice Experiment, 21(7), pp. 739–755.

[15] Zhai Y., Liu B. 2005. Extracting Web Data Using Instance-Based Learning. Proc. of 6th Intl. Conf. on Web Information Systems Engineering (WISE'05), pp. 318–331.

[16] Zhai Y., Liu B. 2005 Web Data Extraction based on Partial Tree Alignment. Proc. of the 14th Intl. World Wide Web Conference (WWW'05), pp. 76–85.