# Comparison of Approaches for Information Extraction from the Web

Michal Toman, *Department of Computer Science and Engineering, Univerzitni 22, Plzen, Czech Republic. mtoman@kiv.zcu.cz*

***Abstract*— In this paper we compare two new methods for information extraction from the web pages. The first method is based on statistical analysis of web page content and the second one uses the XQueries for information extraction from semi-structured documents. We compare precision and recall rate of both automatic methods with manually created extracts. The paper includes a comparison of our extraction methods with other methods used for information extraction from web sources.**

## I. Introduction

The web environment evolves in a general source of information stored mainly in semi-structured HTML format. The current web contains data, which are intended for a human reader. The viewer is able to assign correct semantic information to the presented text. Each web document expresses its information in a different way, which is meaningful for humans but complicated for computer understanding of the content. This paper focuses mainly on Information Extraction (IE) from HTML pages where a vast majority of web content resides. Information extraction is a crucial factor in the computer-based text understanding. It is among the most important tasks in the current semantic web development. Under the term of information extraction we understand a transformation of selected parts of the source HTML pages into the XML format by using so-called *wrappers* [5].

Wrappers are classified according to the way how they process documents. One classification divides the methods into those that use the structure of the HTML format (e.g. W4F [7] and Lixto [1] and those that ignore the HTML tags-based hierarchy and treats the web page as a plain-text (e.g. Cameleon# [3]). Another classification is possible according to the amount of human supervision over the wrapper generation. The robustness increases and the time needed for wrapper training decreases with a transition from manual to automatic methods. On the opposite side, the extraction precision may decrease. Machine learning algorithms, such as decision trees, inductive learning and classification, are applied in fully automatic methods – e.g. SoftMealy [4], Stalker [6] and InfoDiscoverer [9].

Our original motivation for the development of an information extraction method came from the need to create large multilingual corpora. The aim of the method is to extract a useful plain-text and associated metadata, e.g., publication date, author, topic, and abstract, from the web page. In other words, we want to remove the non-informative content blocks such as advertisements, navigation menu, news categories, etc. We designed two different IE methods which are described in Section 2. Both yield satisfying results according to the experiments presented in Section 3.

The former method is intended mainly for text corpus making. The later one is suitable not only for the text extraction but also for general extraction of the data from the web pages. We implemented a prototype of an IE system to evaluate both methods and we subsequently compared the results with manually created extracts.

## II. Information extraction methods

### A. Statistical-based method NIT

The NIT (Node Information Threshold) method is an automatic statistical-based algorithm using the web page structure for information extraction. It transforms the source document into the hierarchical structure – DOM (Document Object Model) [8]. Each DOM node represents one web page element. The NIT method is based on detection of most *useful nodes* in the DOM tree. The useful nodes are included into the extraction result and they represent an ideal plain-text extract in the best case.

The main idea of the NIT algorithm can be described as follows:

Let us define the function $I(N)$ that evaluates the information measure of the node $N$ in the DOM hierarchy. For simplicity, the function is defined in our preliminary experiments as a number of rendered letters of the node $N$ divided by the total document length. The range of the function is <0, 1> in our case. To improve the results, we may modify the function using TF-IDF weighting or Bayesian classifier which classifies each word of the node into two different groups – useful content and non-informative content.

DOM tree is split to sets of sibling nodes having the same root. The siblings of the DOM tree are sorted in decreasing order such that $I(N_1) > I(N_2) > \ldots > I(N_k)$.

We determine the parameter $n < k$ that satisfies the formula (1).

Nodes $N_1$ , $N_2$ … $N_n$ are marked as useful and they are chosen into the extract.

$$\frac{1}{n}\sum_{i=1}^{n} I(N_i) > \tau > \frac{1}{n+1}\sum_{i=1}^{n+1} I(N_i) \qquad (1)$$

The value $\tau$ is an adjustable parameter. It is chosen on empirical grounds to maximize the F1-measure [10] of the extraction.

### *B. Template-based method XQT*

The main idea of XQT (XQuery Templates) method is based on XQuery transformation from HTML format into XML. XQT can be classified as semi-automatic that uses web page features. Its idea is slightly similar to Lixto [1] and W4F [7] systems. The Lixto system uses a proprietary language Elog. On the contrary, our XQT system uses XQuery [2] which is an industry standard.

For each web site, it is necessary to create manually a template in a visual editor. XQT template determines an output structure and includes the queries generated by the editor. The queries transform parts of the web page into an output format. With this template based approach it is possible to process the downloaded websites in a batch. The structured data are loaded into XML database. XQT method is general in its approach and can be used to extract not only text (similar to NIT method) but also general data from semi-structured documents.

## III. EXPERIMENTS

We tested both methods on a corpus consisting of *Deutsche Welle* web pages. We have randomly chosen 200 articles and manually created extract for each of them.

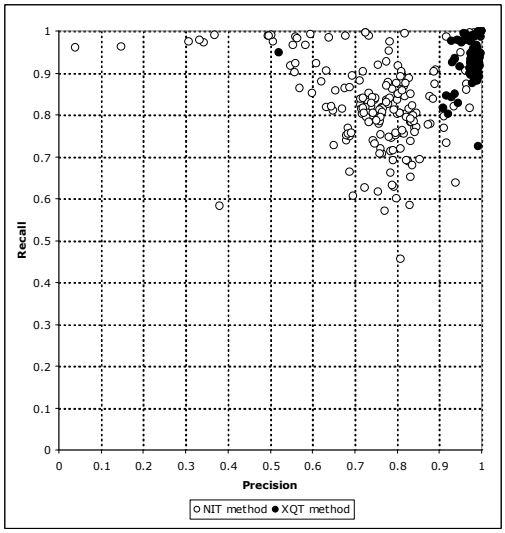


Fig. 1. Precision and recall rates for NIT ( $\tau$ = 0,018) and XQT method.

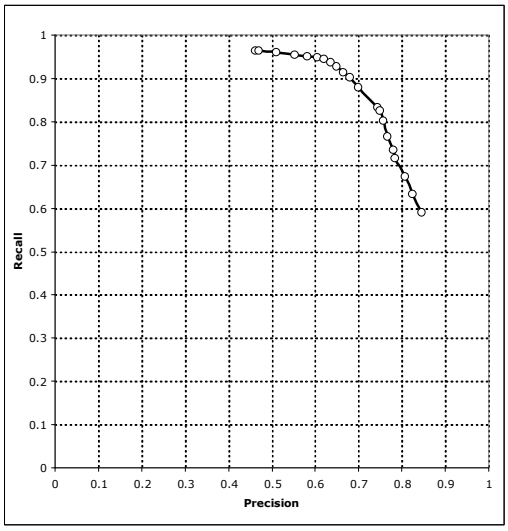


Fig. 2. Precision to recall dependency.

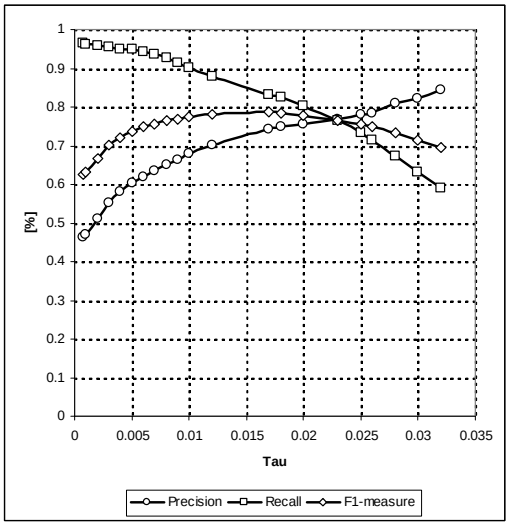


Fig. 3. F1-measure, precision and recall dependency on the adjustable parameter $\tau$ .

### *A. NIT method*

Despite of the fact that NIT method is fully automatic, it gives promising results in comparison with other methods e.g. [1], [9]. Only one parameter denoted $\tau$ is used to adjust the method. In the best case scenario (F1-measure 0.79), the average precision is 74 % and the average recall reaches 83 %.

Because of the simplicity of *I(N)* function used by the method (see Section 2.1), the result is partially dependent on the length and the structure of the article. However, the *I(N)* function can be replaced at any time with a more sophisticated one with no influence on the rest of the system.

Adjustable parameter $\tau$ (threshold) changes the probability of nodes to be included into the extract. If the

threshold is low, less valuable nodes (having smaller *I(N)*) are accepted. In such a case, the recall rise and precision rate falls. On the contrary, in case the threshold is higher, highly informative nodes are chosen, so the precision rate rises and the recall falls (see Figure 2 and 3).

Extremely fine-grained article division may cause recall to decrease, because the node is considered non-informative. Another pitfall of this method is the inclusion of the footer into extract, which decreases the precision especially for shorter articles. The method is considered adequate for unsupervised fully automatic large text corpora creation.

### B. *XQT method*

XQT method gives significantly better results than NIT method according to our expectation but there is a need to create a template. The precision rate exceeded 95 % while keeping recall rate over 90 %, which is comparable with the manual extraction. Pages including numbered lists, embedded objects, and tables are difficult to process because of the discontinuity of the content.

We performed preliminary experiments to compare the method with other approaches. We focused mainly on Lixto [1] and InfoDiscoverer [9] methods because of their similarity to our system. We reach 100 % precision rate when extracting single features (e.g., topic, price and author). The rate is the same as in Lixto approach. For the extraction of the informative text content we achieve an average precision rate 98.1 %, which is slightly better than InfoDiscoverer. However, the InfoDiscoverer approach has better results in the recall rate, which is 93.4% for our method.

## IV. Conclusion

Two new methods for text extraction from web pages, XQT and NIT, were presented along with their taxonomical classification. Our experiments prove that the methods are feasible to extract useful content from web pages.

The XQT method produces precise results and it is able to structure output data. It is suitable for text corpora building with an emphasis on high precision. Its results are comparable with manually created extracts.

TABLE I
PRECISION RATE, RECALL RATE AND F1-MEASURE OF OUR METHODS XQT AND NIT

| Method | Precision [%] | Recall [%] | F1-measure |
|---|---|---|---|
| NIT | 74.36 | 83.32 | 0.786 |
| XQT | 98.07 | 93.36 | 0.957 |
| XQT (single features) | 100.00 | 100.00 | 1.000 |

The NIT method gives results reaching 80 % precision and the same recall rate. Its usage is extremely straightforward because the setup depends only on a single parameter. Thus, the method is appropriate for large corpus building from web sources.

We plan to extend our work in several areas. We aim to replace the current function *I(N)* with a more sophisticated one. It will include TF-IDF scoring algorithm and Bayesian classifier as proposed in Section 2.1. Such function should improve precision and recall rate of the NIT method. Currently we are extending the template editor to cover more usage scenarios.

## Acknowledgment

This research was supported in part by National Research Programme II, project 2C06009 (COT-SEWing).

## References

[1] Baumgartner R., Flesca, S., Gottlob, G. Visual Web Information Extraction with Lixto, *The VLDB Journal*, pp 119-128, 2001.
[2] Chamberlin D., XQuery: A query language for XML. http://www.w3.org/
[3] Firat A., Madnick S., Yahaya A., Kuan W., Bressan S., Information Aggregation using Cameleon# Web Wrapper, *6th ICECWT*, 2005, ISBN 978-3-540-28467-3.
[4] Hsu, C., Dung, M., Generating Finite State Transducers for Semi-Structured Data Extraction form the Web. *Information systéme 23*, 1998, pp 521-538.
[5] Laender A.,Ribeiro-Neto B., Silva A., Teixeira J., A Brief Survey of Web Data Extraction Tools, *SIGMOD Record 31*, 2002, ISSN 0163-5808.
[6] Muslea, I., Minton, S., Knoblock, C. Hierarchical wrapper induction for semistructured information sources. *AAMA 4*, pp 93-104, 2001.
[7] Sahuguet A. Azavant F. Building Intelligent Web Applications using Lightweighted wrappers. *Data and Knowledge Eng. 36*, 2001, pp 283-316.
[8] WWW Consortium *W3C*. The Document Object Model. http://www.w3.org/DOM
[9] Shinan-Hua, L., Jan-Ming, H., Discovering Informative Content Blocks from Web Documents, *SIGKDD 2002*, ISBN 1-58113-567-X.
[10] Van Rijsbergen, C. J., Information Retrieval, 2nd edition, 1979, London.