The Use of Ontologies in Wrapper Induction

Marek Nekvasil

Department of Information and Knowledge Engineering, University of Economics,
Winston Churchill Sq. 4, 130 67, Prague 3, Czech Republic
nekvasim@vse.cz

Abstract. The purpose of this entry is to bring in an extension of ontologies so that they can be utilized in the process of automated information extraction from the web documents. Major part of it is dedicated to a proposition and derivation of an inference model for evaluation of the pattern matches and their combination. Further is proposed a simple naïve method of wrapper induction which is able to use the results of the first part.

Keywords: ontology, automatic annotation, information extraction, wrapper

1 Introduction

One of the simplest alternatives to the manual handling of web documents is the use of *wrappers*, sets of rules to identify the desired values in the document. These rules can be created either by hand or automatically, in which case we are talking about *wrapper induction* [3]. For automatic wrapper induction in principle some examples of the real data to be extracted are needed along with their respective context, which form basically an annotated document. The purpose of this entry is to propose a concept by which it should be possible to annotate the documents automatically with the use of *ontologies*. We will call any ontology that is designed to this use in information extraction an *extraction ontology*.

2 Extending an OWL ontology

An ontology written in a bare OWL language has very limited capabilities of describing the possible values of datatype properties (properties which's value is a literal) and therefore does not contain enough information to identify these inside a document. To remove this insufficiency we introduce an extension which will enable us to append a *pattern* of typical values to each datatype property.

Any general rule for which it is possible on any continuous part of a document (in terms of a string of characters or words) to determine to what extent it is satisfied we call a *pattern*. An example of a pattern can be a rule that evaluates whether a given part of a document is a number from a given range or a string from a given list.

From now on we will distinguish between two kinds of patterns. Foremost there will be simple patterns, or rather *atomic patterns* that will be formed by a simple rule, which can be directly evaluated on a part of a document. An example of atomic patterns can be the aforementioned patterns that match a number or a string.

Moreover there will be *composite patterns*, which we will define as such a combination of rules that can be evaluated on a given continuous part of a document as a whole. As such, the composite pattern will be always a combination of other

J. Pokorný, V. Snášel, K. Richta (Eds.): Dateso 2007, pp. 132–135, ISBN 80-7378-002-X.

patterns, both atomic and composite. The composite patterns can be hierarchically combined which can be of significant concern in some specific situations.
As it has sense to assign only one pattern per datatype property, more patterns will have to be joined via including them in some composite pattern. Every part of a document that matches a given pattern, i.e. the pattern rule evaluates positively on that part, will be considered a suspected *partial candidate* for the occurrence of the value of datatype property the pattern is assigned to. If that given pattern is the one that is assigned directly to the datatype property, every matching part of the document will be considered to be a suspected *candidate* for the occurrence of the value.

2.1 Atomic Patterns

While evaluating the match of atomic patterns we encounter the problem of deriving the certainty degree of marking the candidate. We take a pattern as an algorithm that can tell for every place in the document to what extent the rule is satisfied depending on its parameters. Here we have two distinct terms, the *degree of pattern match* which represents the certainty with which the pattern's algorithm marked the given place in the document, and the *certainty degree of marking the partial candidate for the value of a certain datatype property* which represents the certainty that the given place in the document really is the occurrence of the value, given sole by the observation of the single pattern and independently of any other patterns. We will denote marking the partial candidate as the *pattern evidence* and therefore the second term will be equivalent to a *degree of pattern evidence*.
The degree of pattern match and the degree of pattern evidence should intuitively correspond. If we denote the pattern match as $\boldsymbol{A}$ and the pattern evidence as $\boldsymbol{E}$ we can write down this inference rule:

$$\boldsymbol{A \rightarrow E} \quad \textbf{(1)}$$

We have chosen for our purposes a fuzzy logic inference model [2], but it should be possible to use any other. In fuzzy logic we can define $\boldsymbol{A}$ and $\boldsymbol{E}$ as propositions

- $\boldsymbol{A}$ – "*The pattern has marked the given place in the document*."
- $\boldsymbol{E}$ – "*The marked place is really a pattern evidence*"

and corresponding degrees as their truth values (i.e. degree of pattern match $a=val(\boldsymbol{A})$ and the degree of pattern evidence $e=val(\boldsymbol{E})$). We introduce also two universal parameters for every pattern, namely *precision* and *cover* and we define them:

$$p = val(\boldsymbol{A \rightarrow E}) - \textit{pattern precision} \quad \textbf{(2)}$$

$$c = val(\boldsymbol{E \rightarrow A}) - \textit{pattern cover} \quad \textbf{(3)}$$

While using these parameters we can derive on Łukasiewicz fuzzy logic this form of the inference rule (the detailed derivation is available in [6]):

$$\boldsymbol{((A \,\&\, (A \rightarrow E)) \vee \neg(E \rightarrow A)) \Rightarrow E} \quad \textbf{(4)}$$

If we take into account the prescriptions of p and c and do not overestimate the degree of pattern evidence e, we get:

$$e = max(a + p - 1,\ 1 - c) \quad \textbf{(5)}$$

With the use of parameter p we can set a top limit to the degree of a given pattern evidence. The pattern precision denotes in this context a certainty with what the high degree of pattern match leads to a high degree of pattern evidence. The c parameter sets the lowest possible value of the degree of pattern match.

2.2 Composite patterns

Would we like to combine the evidences of multiple patterns it will be a task in form of a set operation. For this purpose just two set operations come on force, namely union and intersection, which in combination with the complement operation can form any other set operation possible. The determination of the degree of composite pattern evidence itself is then a trivial matter. The degree of composite pattern match will be determined by simply assembling the degrees of evidences of the partial patterns with the appropriate logical operation, thus there will be conjoint patterns and disjoint patters (and possibly negating patterns). From the pattern match defined in this way we get the composite pattern evidence by the same way as we did in case of atomic patterns.
It is interesting to discuss the meaning of precision and cover parameters of the partial patterns in contrast with the use of the different kinds of composite patterns. In case of a disjoint composite pattern the high value of p implies that the partial pattern is a sufficient condition and in case of conjoint composite patterns the high value of c means that the partial pattern forms a necessary condition.

2.3 Designing the patterns

We will denote the patterns in the extraction ontology as XML elements from a special namespace nested in the elements of the datatype properties. The extent of the patterns can vary from distinguishing time values, named entities to patterns that evaluate the context or format of the document. A few basic patterns are proposed in [6], however many others are possible. While designing a new pattern it is needed to keep in mind the way it evaluates and think carefully the possibilities of its combination of other existing patterns.

3 A simple wrapper induction method

By applying the rules of patterns on the content of a document we get a set of evidences along with their certainty degrees for every datatype property in the extraction ontology with a pattern assigned. If we rely on tabular structure of data we can try to separate the evidences in a few segments according to the resemblance of their XPath. We can purge the sets of evidences if we realize that the precision attribute specifies the mean ratio of evidences that are marked correctly by the pattern. Therefore up to $1-p$ of evidences supplied by this pattern can be false and hence we can remove that much of the worst segments. If the data are stored in the tabular structure the relevant parts of text are generally contained in the same structure of elements that is not changing throughout the segment. On the level of XPath expression this will show up as a single changing index in the absolute path by omitting which we get a set of elements that would ideally all contain the value of the respective property.
The cover parameter is the mean rate of the evidences that the pattern identifies to the total real number of occurrences of the respective property. While the generalized XPath expression should identify all possible occurrences of the property we can

calculate the proportion of evidences of the pattern to this "complete" set and difference from the parameter *c* represent the error caused by generalizing the paths. Based on the number of evidences in segments and the respective absolute XPath we can assign the corresponding segments of different properties and form the instances of extracted class.
To sum up this approach is just a simple method and has many limitations. Besides that this method can extract only properties with cardinality 1 (the tabular structure) it is also limited in its tolerance to the irregularities in the structure of the document, on the other hand to the irregularities in the extracted values it is rather resistant.

4 Conclusion and future work

The proposed method of pattern notation allows hierarchical combining of partial patterns and is open to the possibility of designing additional patterns according to one's need. Similar approach is taken by [4] and [5], however unlike them we do not design proprietary formats of ontologies but try to start from OWL standard.
The limitation of the proposed wrapper induction method is the fact that it relies on the tabular structure of extracted data but the extraction is completely automatic and with proper setting of the attributes allows the estimation of extraction error.
To propose a way of automatic learning of the patterns or at least of their parameters could be an interesting subject of future work

Acknowledgement
The research leading to this paper was supported by the European Commission under contract FP6-027026, Knowledge Space of semantic inference for automatic annotation and retrieval of multimedia content, K-Space.

Reference

1. Anton T.: XPath-Wrapper Induction by generalizing tree traversal patterns, in: Antoniou, G., van Harmelen, F.: *A Semantic Web Primer*, Cambridge MA.: MIT Press, 2004, ISBN 0-262-01210-3
2. Hájek P.: *Metamathematics of fuzzy logic*, Dordrecht: Kluwer, 1998, ISBN: 0-792-35238-6
3. Kushmerick, N.: *Wrapper induction for information extraction*, PhD thesis, University of Washington, 1997
4. Labský M., Svátek V.: *On the Design and Exploitation of Presentation Ontologies for Information Extraction*, ESWC'06 Workshop on Mastering the Gap: From Information Extraction to Semantic Representation, Budva, Montenegro, 2006
5. Muslea, I., Minton, S., Knoblock, C.: *A Hierarchical Approach to Wrapper Induction*, *3rd Conference on Autonomous Agents*, 1999, http://www.isi.edu/~muslea/papers.html
6. Nekvasil M., *Využití ontologií při indukci wrapperů*, diplomová práce, VŠE, Praha 2006