Web Semantization

SWA-104

ABSTRACT

In this paper we developed the idea of web semantization, which can help to arch over the gap between the web of today and the Semantic web. We present a proof of concept that even today it is possible to develop a semantic search engine (in the scale of pages of Czech domain – *.cz) designed for software agents. The idea is supported by models of a web repository, automated annotation tools producing third party semantic annotations, semantic repository serving as a sample of semantized web and a proposal of an intelligent software agent.

Categories and Subject Descriptors

H.3.1 [**INFORMATION STORAGE AND RETRIEVAL**]: Content Analysis and Indexing; H.3.3 [**INFORMATION STORAGE AND RETRIEVAL**]: Information Search and Retrieval; I.2.4 [**ARTIFICIAL INTELLIGENCE**]: Knowledge Representation Formalisms and Methods

General Terms

Web Semantization

Keywords

Semantic Web, Web Content Mining, Linguistic Analysis

1. INTRODUCTION

In their Scientific American 2001 article [3], Tim Berners-Lee, James Hendler and Ora Lassila unveiled a nascent vision of the semantic web: a highly interconnected network of data that could be easily accessed and understood by a desktop or handheld machine. They painted a future of intelligent software agents that would "answer to a particular question without our having to search for information or pore through results" (quoted from [5]). Lee Feigenbaum, Ivan Herman, Tonya Hongsermeier, Eric Neumann and Susie Stephens in their Scientific American 2007 article [5] conclude that "Grand visions rarely progress exactly as planned, but the Semantic Web is indeed emerging and is making online information more useful as ever". L. Feigenbaum et al. support their claim with success of semantic web technology in drug discovery and health care (and several further applications). These are mainly corporate applications with data annotated by humans. Ben Adida when bridging clickable and Semantic Web with RDFa ([1]) assumes also human (assisted) activity by annotations of newly created web resources.

Nobody seems to care in the semantic web community about the content of the web of today or of pages published without annotations. The content of the web of today is too valuable to be lost for emerging semantic web applications by our opinion.

In this paper we would like to address the problem of semantization (enrichment) of current web content as an automated process of third party annotation for making at least a part of today's web accessible for machine processing and hence enabling it intelligent tools for searching and recommending things on the web (see [2]).

Our main idea is to fill a semantic repository with information that is automatically extracted from the web and make it available to software agents. We give a proof of concept that this idea is realizable in the size of several TB (terabytes) of textual part of the Czech web (domain *.cz).

Our web crawler (see Fig.1) downloads a part of the web to the web repository (Web Store). Resources with semantic content can be uploaded directly to the semantic repository (Semantic Store). Extractor 1 (classifier) extracts those parts of Web Store which are suitable for further semantic enrichment (we are able to enrich only a part of resources). More semantic content is created by several extractors and annotators in several phases. The emphasis of this paper is on the automation and the usage of such extracted/enriched data.

2. IDEA OF WEB SEMANTIZATION

First idea is the idea of a web repository. It develops as follows in details. Semantic enrichment is in fact a data mining task (although a special one) - to add to web documents a piece of knowledge, which is obvious for human perception not for a machine. That means to annotate data by concepts from an ontology which is the same as to map instances to ontology. Such a data mining task will be easier to solve when there is a sort of a repetition (modulo some similarity).

We decided to enrich only textual content for the present, no multimedia (this substantially reduces the size of infor-

SAC'09 March 8-12, 2009, Honolulu, Hawaii, U.S.A.

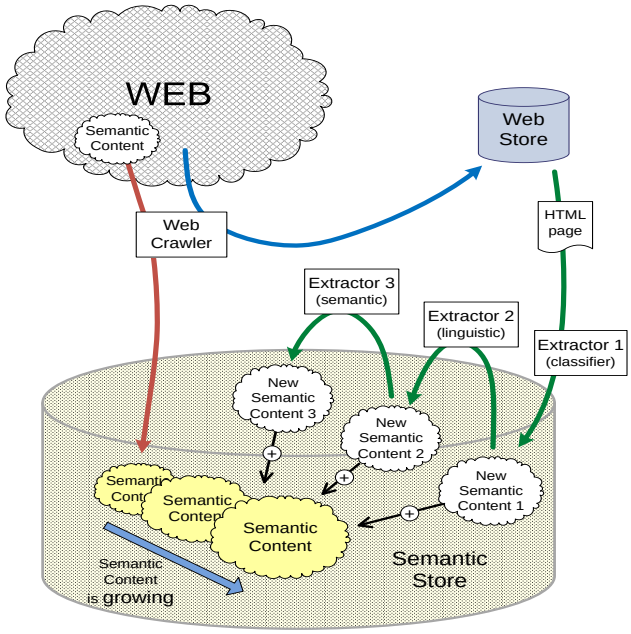

Figure 1: The process of semantization of the Web

mation we have to store). Especially we restrict to the pages with content consisting either dominantly of grammatical sentences (let us call them grammatical pages) and those containing large number of table cells (let us call them table pages). Of course this division need not be disjoint, and will not cover the whole web.

To be able to separate durable grammatical or table pages poses special requirements to search in our web repository.

Second idea is to split annotation process to two parts, one domain independent intermediate annotation and second domain dependent user directed annotation. Both should be automated, with some initial human assisted learning. This first part of learning could require assistance of a highly skilled system manager; the second (probably faster part) should be doable by an administrator with average computer literacy.

The first domain independent annotation will serve as an intermediate annotation supporting further processing. This will run once on the whole suitable data. Hence initial necessary human assistance in training can be done by a highly skilled expert (indeed it can be a long process).

The second domain dependent annotation part can consists of a large number of tasks with different domains and ontologies. This should be possible to be executed very fast and if possible with assistance of an average computer skilled administrator. We can afford this having intermediate annotation.

Domain independent intermediate annotation can be done with respect to general ontologies. First ontology is the general PDT tectogrammatical structure which captures semantic meaning of a grammatical sentence This is the task for computational linguistics; we make use of their achievements which were originally focused on machine translation. For structured survey pages we assume that their structure is not changing very often and many pages are generated by same tool, hence they are similar. So there is a chance

Type of annotation	Table pages	Grammatical pages
Intermediate general	Uses similarities	Does not use similarities
Domain dependent	Does not use similarities	Uses similarities

Table 1: Use of similarity in annotation approaches

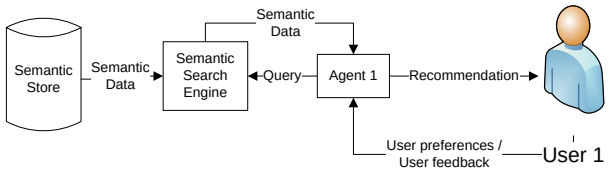

Figure 2: Process of querying Semantic Search Engine by an agent

to detect data regions and data records and possibly also attributes and values from detailed product pages. Here annotation tools will be also trained by humans – nevertheless only once for the annotation of the whole repository.

We have identified two types of web resources where this can be done. The first are "table pages", containing large number of cells with abbreviated description of a resource and corresponding detailed pages. Here we assume that content of cells is similar and content of detailed pages is also similar. Because of space limitations, we do not describe our approach for these pages.

The second are pages dominantly consisting of grammatical sentences. Thanks to the fact that we have a third party linguistic tool for the creation of tectogrammatical trees of sentences, this can be done without requirement of similarity repetition. This situation is illustrated in the Table 1.

Domain (task) dependent (on demand) annotation is concerning only pages previously annotated by general ontologies. This makes second annotation faster and easier. An assistance of a human is assumed here for each domain and new ontology. For grammatical pages repetitions make possible to learn a mapping from structured tectogrammatical instances to an ontology. This domain (task) dependent task can be avoided by a collaborative filtering method, assuming there is enough users' acquaintance.

Further important idea is to **design semantic repository**. It should contain some sort of uncertainty annotation besides above mentioned ontologies. The main reason is that annotation process is error prone and we can have in future different alternative annotation tools and aggregate results. This aspect is not further described in the paper.

Last, but also important idea is **to design at least one agent** which will give evidence that our semantization really improved general web search. Besides using annotated data it should also contain some user dependent preference search capabilities.

The process of a user searching for an article making use of an agent is represented in Figure 2. Our main focus is on the agent and on the extractors.

3. INTERMEDIATE DOMAIN INDEPENDENT AUTOMATED ANNOTATION

Web information extraction is often able to extract valuable data. We would like to couple this process with con-

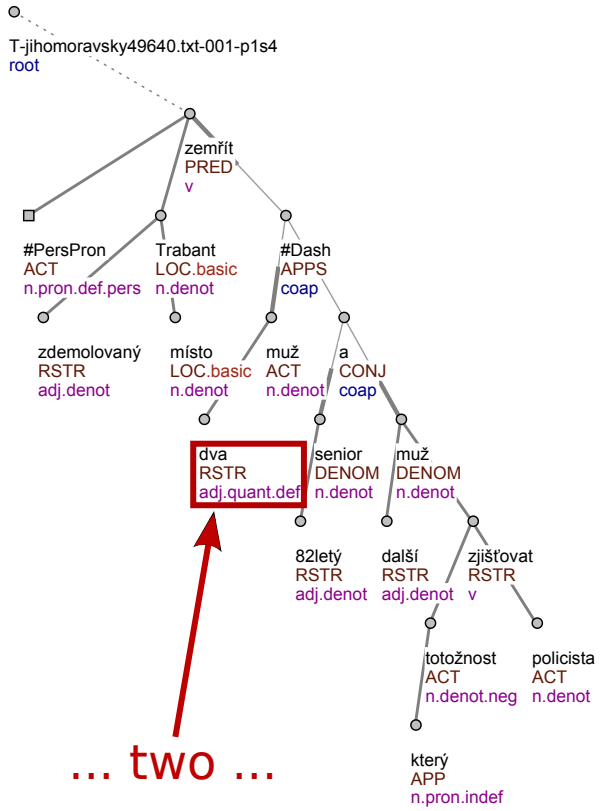

Figure 3: A tectogrammatical tree of sentence: "Two men died on the spot in the demolished Trabant..."

secutive semantic annotation (initially human trained; later it should operate automatically). In this chapter we would like to describe the details of our idea how to automatically annotate grammatical pages.

3.1 A method based on Czech linguistics

Our approach for Web information extraction of grammatical pages is based on Czech linguistics and NLP tools. We use a chain of linguistic analyzers ([6], [4], [7]) that processes the text presented on a web page and produces linguistic (syntactic) trees corresponding with particular sentences. These trees serve as a basis of our semantic extraction.

Unlike the usual approaches to the description of English syntax, the Czech syntactic descriptions are dependency-based, which means that every edge of a syntactic tree captures the relation of the dependency between a governor and its dependent node. Especially the tectogrammatical (deep syntactic) level of representation [8] is closer to the meaning of the sentence. The tectogrammatical trees (Example of such a tree is on the Figure 3) have a very convenient property of containing just the type of information we need for our purpose (extraction of semantic information), namely the information about inner participants of verbs - actor, patient, addressee etc. So far this tool does not work with the context of the sentence and hence does not exploit a possible frequent occurrence of similar sentences.

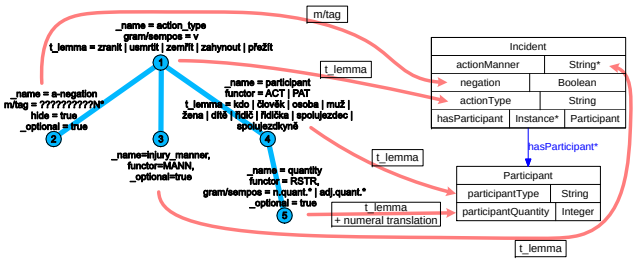

Figure 4: An example of extraction rule and its mapping to ontology

4. DOMAIN DEPENDENT AUTOMATED ANNOTATION

Second phase of our extraction-annotation process is domain dependent. It can make use of previous intermediate general (domain independent) annotation. Our goal is to make this process as fast and as easy as possible, e.g. to be trained very fast and precisely by any administrator with average computer skills.

4.1 Domain dependent annotation based on linguistics

Assume we have pages annotated by a linguistic annotator and we have a domain ontology. The extraction method we have used is based on extraction rules. An example of such an extraction rule is on Figure 4 (on the left side). These rules represent common structural patterns that occur in sentences (more precisely in corresponding trees) with the same or similar meaning. Mapping of the extraction rules to the concepts of the target ontology enables the semantic extraction. Example of such a mapping is demonstrated in Figure 4.

We experimented with obtaining extraction rules in two ways.

1. Rules and mappings were designed manually (like the rule on the Fig. 4).

2. Rules and mappings were learned using Inductive Logic Programming (ILP) methods (see following section).

Finally, having this mapping, we can extract instances of the target ontology from the answer corresponding to an extraction rule. This answer is given by the semantic repository, and the instances of the ontology are also stored there.

4.2 Using Inductive Logic Programming (ILP)

ILP [9] is a method for a generation of rules that describe some properties of data. ILP takes three kinds of input

- Positive examples E+ – objects that have the desired property.
- Negative examples E- – objects that do not have the desired property.
- Background knowledge – facts about the domain (so far we do not consider background knowledge in the form of rules).

ILP tries to generate such rules that all positive examples and no negative example satisfy them. It uses concepts from

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Ve zdemolovanem trabantu na miste zemreli dva
% muzi - 82lety senior a dalsi muz, jehoz ...
%
% Two men died on the spot in the demolished
% trabant - a senior 82 years old and another
% man, who's ...

%%%%%%%% Nodes %%%%%%%%
tree_root(node0_0). node(node0_0).
id(node0_0, t_jihomoravsky49640_txt_001_p1s4).

node(node0_1).
t_lemma(node0_1, zemrit).
functor(node0_1, pred).
gram_sempos(node0_1, v).

node(node0_2).
t_lemma(node0_2, x_perspron).
functor(node0_2, act).
gram_sempos(node0_2, n_pron_def_pers).
...

%%%%%%%% Edges %%%%%%%%
edge(node0_0, node0_1).
edge(node0_1, node0_2).
edge(node0_1, node0_3).
...
edge(node0_34, node0_35).

%%%%%%%% Injury %%%%%%%%
injured(t_jihomoravsky49640_txt_001_p1s4).
```

Figure 5: Sample from prolog representation of a sentence

the background knowledge in the body of the rules. The main advantage of ILP is that it can be trained to mine arbitrary data described in predicate logic - the process of learning is dependent only on the positive and the negative examples and on the amount of information we have about them. The more information we have, the more precise the learning will be.

Thanks to the fact that ILP learns from positive and negative examples, we can propose an assisted learning procedure for learning and also tuning the extractor (described in previous section). The process is such that the extractor gets pure textual or semantically annotated (by the linguistic extractor) data from a web page and presents it to the user. He or she then annotates the data or only corrects the data saying which are well and which are badly annotated. The extractor can be taught and tuned to the desired goal in this way. Of course the user has to understand the semantics of the annotations - so he or she has to be an expert in that way of understanding the target ontology. Yet this ontology can be quite simple in the case we are interested only in a specific kind of information e.g. number of people injured during a car accident like in our experiments (see next section) and motivation.

We discovered that we can use a straightforward transformation of linguistic trees (see an example on the Figure 3) to predicates of ILP (example of the transformation is in the Figure 5) and the learning procedure responds with significant results, which are presented in next section.

On the Figure 3 we can see the relationship between particular words of a sentence and nodes of tectogrammatical tree - the highlighted node of the tree corresponds with the word "two" in the sentence (also highlighted). This relationship allows propagation of information from the user (which annotates just the text of sentence) to the ILP learning procedure.

5. EXPERIMENTS

Our experiment was to enable our agent to access information from natural language text. We wanted to find out the number of injured persons during car accidents. We have used firemen reports; some of them were about car accidents, some were not. Each report was split into sentences and each sentence was linguistically analyzed and transformed into a tectogrammatical tree, as described in the previous two sections. These trees were transformed to a set of Prolog facts (see Figure 5).

Sentences, which talk about an injury during a car accident, were manually tagged by predicate `injured(X)`, where X is the ID of the sentence. Those sentences that do not talk about injured persons during a car accident were tagged as `:-injured(X)`, which represents a negative example. This tagging can be done even by a user unexperienced in linguistics and ILP, but he or she has to understand the semantics of information he is tagging (in usually means that he or she has to understand the target ontology). These tagged sentences were the input for ILP, we have used 22 sentences as positive examples and 13 sentences as negative examples. We used Progol [10] as ILP software. The rules ILP found are in following Figure 6.

The first four rules are overfitted to specific sentences. Only the last two represent generally applicable rules. But they do make sense - "zranit" means "to hurt" and "nehoda" means "an accident" in Czech. These two rules mean that

```
injured(A) :- id(B,A), id(B,t_57770_txt_001_p5s2).
injured(A) :- id(B,A), id(B,t_60375_txt_001_p1s6).
injured(A) :- id(B,A), id(B,t_57870_txt_001_p8s2).
injured(A) :- id(B,A), id(B,t_57870_txt_001_p1s1).

injured(A) :- id(B,A), edge(B,C), edge(C,D),
        t_lemma(D,zranit).
injured(A) :- id(B,A), edge(B,C), edge(C,D),
        t_lemma(D,nehoda).
```

Figure 6: Rules mined by ILP

Table 2: Results on the test set

	A	¬A
P	11	1
¬P	4	22

the root element is connected either to a noun that means an accident or to a verb that means to hurt.

We tested these rules on the set of 15 positive and 23 negative examples. The result accuracy was 86.84%. Detailed results are in Table 5. P is the positive sentence from ILP, P is negative sentence, A is sentence tagged as positive and ¬A is sentence tagged as negative. 11 sentences that were tagged as positive by the user were also tagged as positive by ILP, 4 was tagged as negative by ILP. 22 sentences that were tagged as negative by the user were also tagged as negative by ILP, only 1 was tagged as positive by ILP.

In this case we learned a set of rules that identifies relevant sentences - roots of relevant tectogrammatical trees. We understand these results as a proof of concept that ILP can be used for finding other kinds of information present in nodes and structure of linguistic trees. Se we can for example extract number of injured people from relevant trees if we modify the training data.

6. CONCLUSIONS AND FURTHER WORK

In this paper we have developed the idea of web semantization, which can help to arch over the gap between Web of today and Semantic Web. We have presented a proof of concept that even today it is possible to develop a semantic search engine in the scale of *.cz pages. In particular contributions consists of two sorts of automated third party annotation of existing web resources, the idea of a semantic repository serving as a sample of semantized web with extension of the ontology model for additional annotation (e.g. uncertainty annotation) and a proposal of an intelligent agent.

Future work goes in two directions. First is in the integration of specific parts of the system, in the automation of the whole process and in the extensive experiments with a larger number of different users. Second is in improving special parts of the system - either making it more precise or making it more automatic (able to train by a less qualified user).

7. ACKNOWLEDGMENTS

This work was partially supported by Czech projects 1ET100300517, 1ET100300419 and MSM-0021620838.

8. ADDITIONAL AUTHORS

9. REFERENCES

[1] B. Adida. Bridging the clickable and semantic webs with rdfa. *ERCIM News - Special: The Future Web*, 72:24–25, January 2008.

[2] T. Berners-Lee. The web of things. *ERCIM News - Special: The Future Web*, 72:3, January 2008.

[3] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web, a new form of web content that is meaningful to computers will unleash a revolution of new possibilities. *Scientific American*, 284(5):34–43, May 2001.

[4] M. Collins, J. Hajič, E. Brill, L. Ramshaw, and C. Tillmann. A Statistical Parser of Czech. In *Proceedings of 37th ACL Conference*, pages 505–512, University of Maryland, College Park, USA, 1999.

[5] L. Feigenbaum, I. Herman, T. Hongsermeier, E. Neumann, and S. Stephens. The semantic web in action. *Scientific American*, 297:90–97, December 2007.

[6] J. Hajič. Morphological Tagging: Data vs. Dictionaries. In *Proceedings of the 6th Applied Natural Language Processing and the 1st NAACL Conference*, pages 94–101, Seattle, Washington, 2000.

[7] V. Klimeš. Transformation-based tectogrammatical analysis of czech. In *Proceedings of the 9th International Conference, TSD 2006*, number 4188 in Lecture Notes In Computer Science, pages 135–142. Springer-Verlag Berlin Heidelberg, 2006.

[8] M. Mikulová, A. Bémová, J. Hajič, E. Hajičová, J. Havelka, V. Kolářová, L. Kučová, M. Lopatková, P. Pajas, J. Panevová, M. Razímová, P. Sgall, J. Štěpánek, Z. Urešová, K. Veselá, and Z. Žabokrtský. Annotation on the tectogrammatical level in the prague dependency treebank. annotation manual. Technical Report 30, ÚFAL MFF UK, Prague, Czech Rep., 2006.

[9] S. Muggleton. Inductive logic programming. *New Generation Comput.*, 8(4):295–, 1991.

[10] S. Muggleton. Learning from positive data. In *Inductive Logic Programming Workshop*, pages 358–376, 1996.