Semi-automatic Semantic Annotation of Slovak Texts*

Michal Laclavík[1], Marek Ciglan[1], Martin Šeleng[1], Stanislav Krajčí[2], Peter Vojtek[3], Ladislav Hluchý[1]

[1] Institute of Informatics, Slovak Academy of Sciences, Dúbravská cesta 9, Bratislava, 845 07, Slovakia
laclavik.ui@savba.sk
http://ikt.ui.sav.sk/

[2] Ústav informatiky, Prírodovedecká fakulta, UPJŠ, Park Angelinum 9, 040 01 Košice, Slovakia
stanislav.krajci@upjs.sk

[3] Ústav informatiky a softvérového inžinierstva, Fakulta informatiky a informačných technológií, Slovenská technická univerzita, Ilkovičova 3, 842 16 Bratislava
peter.vojtek@fiit.stuba.sk

Abstract. Automated annotation of the Web documents is a key challenge of the Semantic Web effort. Web documents are structured but their structure is understandable mainly for humans, which is the major problem of the Semantic Web. Many solutions for semi-automatic annotation exist based on neural networks, structure analysis or supervised learning techniques. Another possibility is to use pattern based methods for semantic annotation such as SemTag or C-PANKOW. Mentioned methods and solutions are applicable mainly in English and could not work well on highly inflective languages such as Slovak. We have developed the Ontea tool for semi automatic semantic annotation based on regular expression patterns, which together with tools for natural language identification, lemmatization or stemming of Slovak and specialized indexing mechanism provide promising results for semantic annotation of Slovak texts. Language identification is based on Markov processes, which enables to accommodate granularity of text modeling according to attributes of input text. Lemmatization can be done via existing lemmatizer or the simple lemmatizer based on finding same word suffix. The annotation method has been evaluated and success rate measured using recall, precision and F1-measures is over 70%. We can identify objects such as geographical locations: cities, villages, rivers; company names; or other application specific objects. Results can be used for further computer processing and for partial understanding of text by a machine.

* This work is supported by projects NAZOU SPVV 1025/2004, RAPORT APVT-51-024604, SEMCO-WS APVV-0391-06, VEGA 2/7098/27.

1 Introduction

Adding machine understandable information about documents content is one of main challenges of emerging semantic oriented systems. An ultimate goal is to allow machine based reasoning about content of vast quantity of documents produced by human experts and to allow automatic inference of new knowledge. One step towards this goal is to enable automatic and semi-automatic semantic annotation (ASA) of unstructured texts such as web pages, office documents that provides the means to transfer useful information from the documents to the ontology structures.

In this paper, we present combination of a traditional method from information retrieval domain and an annotation method from semantic web, which increase the relevance of automatic annotation. Namely, we have integrated lemmatization, full-text indexing and search mechanism with ASA based on regular expression matching. Indexes of full-text search engine are exploited to gather statistical information about words occurrences in the document collection to estimate the relevance of the ASA outputs.

We present ASA and full-text indexing, including the summarization of the state-of-the-art and description of our approaches (Sections 2, 3, 4); we continue by depicting the integration of the methods and present the experimental results. We conclude the paper with the summary and future work description.

2 Semantic annotation

Automated annotation of the Web documents is a key challenge of the Semantic Web effort. Web documents are structured but their structure is understandable only for humans, which is the major problem of the Semantic Web. Annotation solutions can be divided into manual and semi-automatic methods. This different strategy depends on a use of the annotation. There is number of annotation tools and approaches such as CREAM [15] or Magpie [16] which follow the idea to provide users with useful visual tools for manual annotation, web page navigation, reading semantic tags and browsing [18] or provide infrastructure and protocols for manual stamping documents with semantic tags such as Annotea[1], Rubby[2] or RDF annotation[3].

Semi-automatic solutions focus on creating semantic metadata for further computer processing, using semantic data in knowledge management [19] or in information extraction application. Semi-automatic approaches are based on natural language processing [11] [12], a document structure analysis [13] or

1 http://www.w3.org/2001/Annotea/

2 http://www.w3.org/TR/ruby/

3 http://ilrt.org/discovery/2001/04/annotations/

learning requiring training sets or supervision [14]. Moreover, other pattern-based semi-automatic solutions such as PANKOW and C-PANKOW [10] exist, using also Google API for automatic annotation. Other methods use a variety of pattern matching mechanisms. Another relevant automatic semantic annotation solution and the only one which runs on distributed architecture is SemTag [20]. SemTag uses Seeker [20] information retrieval platform to support annotation tasks. SemTag annotates web pages using Stanford TAP ontology [21].

2.1 Ontea

One of pattern based solutions is also Ontea [19] developed in the NAZOU project. Ontea works on text, in particular domain described by domain ontology and uses regular expression patterns for semi-automatic semantic annotation. Ontea detects or creates ontology elements/individuals within the existing application/domain ontology model according to defined patterns. Ontea tool analyzes text using a regular expression patterns and detects equivalent semantic elements according to the defined domain ontology. Several cross application patterns are defined but in order to achieve good results, new patterns need to be defined for each application. In addition, Ontea creates a new ontology individual of a defined class and assigns detected ontology elements/individuals as properties of the defined ontology class.

3 Document indexing and search

Information retrieval is a process of identifying the text resources of interest from a large collection of documents that would satisfy the user needs. Full text search is a wildly used concept in today information systems for information retrieval. Full-text search engines usually operate over the index structure which keeps information about documents content. Indexes of the documents content are exploited because of the time efficiency of retrieving information from those structures.

3.1 Related work

Documents content indexing is a well established method for information retrieval which crossed the border of academic research and become a part of every day life. Full text search engines are used to find documents stored at users workstations (desktop search engines) as well as to locate resources in intranet and Internet. Main technological challenges addressed by document indexing and search solutions are: index Data Structures, performance (Maintenance, Lookup speed), transformation of words to their base form – usually done by stemming or lemmatization (this topic is discussed in detail in section 3.3 and 3.4 for Slovak), provide rich query mechanism (phrase queries, wildcard queries, proximity queries, range queries), stop words filtering, search

results ranking. A lot of document indexing solutions are available both under commercial and open source licenses with different level of features implementation. We mention several popular systems suitable for intranet and document repository indexing and searching:

Apache Lucene [1] is a search engine designed for high-performance search, supporting large number of query mechanisms. Another search engine is OpenFTS [2] (Open Source Full Text Search engine) using relational database PostgreSQL as backend for storing indexes, provides online indexing of data and relevance ranking. MnoGoSearch [3] is a search engine designed primary for indexing HTML content with HTML specific features such as META tags support, robots exclusion standard support.

3.2 Language identification using Markov processes

Before providing text operations such as lemmatization or stemming, we need to identify language of the text. This can be done using a variety of methods. In our annotation solution we have used the NALIT method which uses Markov processes [23, 24].

The categorization method used in the NALIT tool (Markov Processes based Categorization) [25] is based on method proposed by Dunning [24]. First a statistical model for each category is created in a learning phase. Each of these category models is constructed from pre-selected training text documents, every document represents a certain category in selected categorization. In our case categories represents documents in different languages. The NALIT tool allows to execute proper lematization algorithm as well as to use proper regular expression paterns which are also language dependent. In our experiments NALIT identified correct language of document in 100% cases. In other more general evaluation [25] NALIT was successful in more than 95%.

3.3 Words base forms

One of the driving factors for developing yet another indexing and search engine was the study of stemming and lemmatization methods for the Slovak language and subsequent integration of suitable methods with the search engine. Therefore, we describe in this subsection basic approaches to the words' base form acquisition. Different morphological variants of the natural languages words have in most cases the same or very similar semantic interpretations and can be considered as equivalent for the purpose of information retrieval systems. This means that different morphological forms can be represented by a single representative term. Queries can then produce more relevant responses and the dictionary size needed for representing a set of documents decrease. A smaller dictionary size results in a saving of storage space and processing time. Two main approaches to words' base form

acquisition are lemmatization and stemming. Lemmatization uses the dictionaries produced by human experts to retrieve the base form of a given word. Wordnet [4, 5] is one of sophisticated dictionaries for the English language that can be used for lemmatization. Stemming is a method, which algorithmically derives the stem of a given word; stems produced by stemming algorithms often do not belong to the given natural language, however they identify the class of words from natural language. Popular stemming algorithm for the English language is Porter algorithm [8, 9].

3.4 Lematization of Slovak – Tvaroslovník

One of the tools for lemmatization of Slovak texts is Tvaroslovnik [22]. Its algorithm works on finding the same longest word endings in dictionary words.

For evaluation of Tvaroslovnik we have used 8 documents containing Job offers in Slovak. We ran lemmatization on these documents and results are summarized in the table below.

words	lemmas	percentage
67	42	63%
82	50	61%
133	78	59%
114	71	62%
90	55	61%
82	51	62%
79	56	71%
148	97	66%
795	500	63%

Table 1. Tvaroslovnik lemmatization results

Words column represents word count in the documents. Lemmas column represents found words in other than lemma form, where lemmas were identified. Over half of content is in other than basic word forms even in partially formalized documents as job offers. In this article we evaluate the Ontea algorithm on company and location objects, which names are usually in basic form in job offer documents. Thus we do not compare algorithm success rate with and without lemmatization. Nevertheless preliminary results are promising as demonstrated also by the table above. For example objects related to education or job type are better identified in case of using lemmatization. Examples are:

- Text: „*práca s pokladňou*“ lemma: „*pokladňa*“
- Text: „*stredoškolské s maturitou*“ lema: „*maturita*“

Files used in the experiment as well as log output information can be found on the web[4].

4 `http://ikt.ui.sav.sk/archive/Tvaroslovnik/test_tvaroslovnik_ontea.zip`

3.5 RFTS

We have developed a tool for document indexing and document search, named RFTS (Rich full-text search). The motivation for implementing another search engine was to have an easily extendable and configurable document indexing tool to evaluate novel methods for information retrieval, documents statistical analysis and lemmatization and stemming methods for the Slovak language. Detailed information about documents content is stored in the index structure, including the positions of the word in the documents, phrase number within the document. The words in tool's dictionary are kept in the basic form; different stemming algorithms are used for documents in different languages. The tool exploits relational database to store all the information about documents and its contents. From engineering point of view, it is worth to mention that RFTS functionality in conjunction with Corporate Memory [7] (also developed within the project NAZOU [6]) can be accessed locally (using JAVA interfaces or command line tools) as well as remotely using RPC calls or Web Service interface. The remote access and Web Service interface allows easy integration of the RFTS indexing and search solution in other components and allows rapid prototyping of new tools that require full-text search or some form of statistical analysis of document collection.

4 Integrated annotation method

Ontea's method of automatic annotation based on regular expressions matching showed promising results for domain specific texts. However it suffers from frequent mismatching which creates imprecise instances of ontological concepts. We propose to overcome this obstacle by evaluating the relevance of candidate instances by the means of statistical analysis of the occurrence of the matched words in the document collection. Based on regular expression, Ontea identifies part of a text related to semantic context and match the subsequent sequence of characters to create an instance of the concept. Let us denote the sequence of words related to semantic context by C and word sequence identified as a candidate instance as I. We evaluate the relevance of the new instance by computing the ration of the close occurrence of C and I and occurrence of I:

$$\frac{close\_occurrence(C, I)}{occurrence(I)}$$

RFST indexing tool provides us with enough functionality to retrieve required statistical values computed from the whole collection of documents stored in RFTS index structures.

Let *COLL* be a collection of the documents $d_1, d_2, \ldots, d_n$: $COLL = d_1, d_2, \ldots, d_n$
Let d in *COLL*, *distance* N, and $w_1, w_2, \ldots, w_k$ be the words from natural language.
Function $dist(d, distance, w_1, w_2, \ldots, w_k)$, where $k \leq distance$, denotes the number of distinct word sequences of the length distance containing the words $w_1, w_2, \ldots, w_k$.

We compute the relevance of candidate instance as:

$$relevance(C, I, wordsdist) = \frac{\Sigma\, dist(d, wordsdist, C \cup I)}{\Sigma\, dist(d, (I), I)}$$

If the resulting relevance value exceeds defined threshold, the candidate word sequence I is considered to be a valid instance of the semantic concept related to sequence C. For the experimental evaluation of the approach, the threshold was set manually after inspecting the preliminary relevance values of the generated candidate instances.

The utilization of RFTS brings also an important benefit of treating different morphological word forms as a single class of equivalence represented by the word stem or lemma.

All the documents that are subject to semantic annotation by Ontea must be part of the document collection indexed by RFTS tool.

4.1 Ontea with Lucene

The Ontea annotation method can be also used with Lucene [1] information retrieval library. When connected with RFTS indexing, Ontea asks for relevance based on words distance. When connecting with Lucene, Ontea asks for percentage of occurrence of matched regular expression pattern to detected element represented by word. Example can be Google, Inc. matched by pattern for company search: [\\s]+([-A-Za-z0-9][ ]*[A-Za-z0-9]*),[ ]*Inc[\\.\\s]+, where relevance is computed as Google, Inc. occurrence divided by Google occurrence. RFTS indexing tool also supports this type of queries, however Lucene can achieve better performance. So far we did not compare those two methods of finding relevance of new created individual, but both are implemented.

5 Evaluation

In this chapter we discuss the algorithm evaluation and success rate.

To evaluate the performance of annotation, we used the standard recall, precision and F1 measures. Recall is defined as the ratio of correct positive predictions made by the system and the total number of positive examples. Precision is defined as the ratio of correct positive predictions made by the system and the total number of positive predictions made by the system:

$$Recall = \frac{Match}{Count} = \frac{Relevant\ retrieved}{All\ relevant}$$
$$Precision = \frac{Match}{Ontea} = \frac{Relevant\ retrieved}{All\ retrieved}$$

Recall and precision measures reflect the different aspects of annotation performance. Usually, if one of the two measures is increasing, the other will decrease. These measures were first used to measure IR (Information retrieval) system by Cleverdon [11]. To obtain a better measure to describe performance, we use the F1 measure (first introduced by van Rijsbergen [12]) which combines precision and recall measures, with equal importance, into a single parameter for optimization. F1 measure is weighted average of the precision and recall measures and is defined as follows:

$$F_1 = \frac{2 * Precision * Recall}{Precision + Recall}$$

5.1 Test set of documents

As reference test data, we used 500 job offers downloaded from web using wrapper which prepared us some structured data. This was converted to a defined ontology and manually checked and edited according to 500 html documents representing reference job offers. Ontea processed reference html documents using the reference ontology resulting in new ontology metadata consisting of 500 job offers, which were automatically compared with reference manually checked job offers ontology metadata.

5.2 Target ontological concepts for identification

In this test, Ontea used simple regular expressions matching from 1 to 4 words starting with a capital letter. This experiment is referred to as "Ontea" in next chapter. In the second case we used domain specific regular expressions which identified locations and company names in text of job offers and Ontea was also creating individuals in knowledge base, while in the first case Ontea did not create extra new property individuals and only searched for relevant individuals in knowledge base. This second case is referred to as "Ontea creation". The third case used previously described RTFS indexing tool to find out if it is feasible to create new individual using word occurrence functionality of RTFS this case is referred as "Ontea creation, RFTS"

So we did our experiments in 3 cases:

- Ontea: searching relevant concepts in knowledge base (KB) according to generic patterns
- Ontea creation: creating new individuals of concrete application specific objects found in text
- Ontea creation, RTFS: Similar as previous with the feedback of RTFS to get relevance computed above word occurrence. Individuals were created only when relevance was above defined threshold which was set up to 10%

We have used following regular expressions:

- Generic expression matching one or more words in text. This was used only to search concepts in KB.
  ([A-Z][-A-Za-z0-9]+[\\s]+ [-a-zA-Z]+)
- Identifying geographical location in text and if not found in KB individual was created
  Location:[\\s]*([A-Z][-a-zA-Z]+[ ]*[A-Za-z0-9]*) used for English
  [0-9]{3}[ ]*[0-9]{2}[ ]+([A-Z][^\\s,\\.]+[ ]*[^0-9\\s,\\.]*)[ ]*[0-9\\n,]+
  used for Slovak text where settlement name is usually next to ZIP code
- Identifying company in the text, this was used also with other abbreviations such as Ltd or a.s., s.r.o. for the Slovak language
  [\\s]+([-A-Za-z0-9][ ]*[A-Za-z0-9]*),[ ]*Inc[\\.\\s]+ for English
  [\\s]+([A-Z][^\\s,\\.]+[ ]*[^\\s,\\.]*[ ]*[^\\s,\\.]*)[,]*s\\.r\\.o\\.[\\s]+
  used for Slovak texts

5.3 Experimental results

Experimental results using precession, recall and F1-measures are in Table 1. In the table we compare our results with other semantic annotation approaches and we also list some advantages and disadvantages. The column "relevance" corresponds to F1-measures in case of Ontea, in case of other methods, it can corespondto other evaluation techniques. For example for C-PANKOW, relevance is referred to as recall.

The experimental results are summarized in Table 1. Rows relevant to our annotation approach are in grey color, showing success rate of three evaluation cases mentioned in the previous chapter. The row "Ontea creation, RTFS" case is the most important concerning evaluation where we combined indexing and annotation techniques. By using this combination we were able to eliminate some not correctly annotated results. For example by using [Cc]ompany[:\\s]*([A-Z][-A-Za-z0-9][ ]*[A-Za-z0-9]* regular expression in the second case we have created and identified companies such as This position or International company which were identified as not relevant in the third case with RTFS.

Similarly Ontea creation identified also companies like Microsoft or Oracle which is correct and in combination with RTFS this was eliminated. Because of this issue, recall is decreasing while precession is increasing. Here it seems that RTFS case is not successful but the opposite is true because in many texts Microsoft is identified as products e.g. Microsoft Office so if we take more text to annotate it is better to not annotate Microsoft as company and decrease recall. If we annotate Microsoft as company in other texts, used in context of Microsoft Office we would decrease precision of annotation.

So it is very powerful to use presented annotation technique in combination with indexing in applications where precision needs to be high.

	Method	Rel. %	Prec. %	Recall %	Disadvantages	Advantages
Ontea	**regular expresions, search in knowledge base (KB)**	**71**	**64**	**83**	**high recall, lower precision**	**high success rate, generic solution, solves the duplicity problem, fast algorithm**
SemTag	disambiguity check, search in KB	high	high		works only for TAP KB and English	fast and generic solution
Ontea creation	**regular expressions (RE), creation of individuals in KB**	**83**	**90**	**76**	**application specific patterns are needed**	**supports Slovak**
Ontea creation RFTS, TS	**RE, creation of individuals in KB + RFTS**	**73**	**94**	**69**	**low recall**	**disambiguities are found and not annotated**
Wrapper	document structure	high	high		zero success with unknown structure	high success with known structure
PANKOW	pattern matching	59			low success rate	generic solution
C-PANKOW	POS tagging, and pattern matching Qtag library	74		74	suitable only for English, slow algorithm	generic solution
Hahn et al.	semantic and syntactic analysis	76			works only for English, not Slovak	
Evans	clustering	41			low success rate	
Human	manual annotation	high	high	high	problem with creation of individuals duplicities, inaccuracy	high recall and precision

Table 2. Annotation experimental results

6 Conclusion

By integrating information retrieval system based on lemmatization (Tvaroslovnik), full-text indexing and search (RFTS) and the semantic annotation tool (Ontea), we were able to improve the results of the automatic semantic annotation process for domain specific documents – increasing the precision of newly created instances at least by 4%. However, the recall of identified instances decreased. This is an advantageous trade-off as the ontological data precision is the primary goal of our work on automatic annotation.

We have also identified, but not proved yet, that using a large collection of experimental texts or documents "Ontea creation" (without RTFS indexing) precission will decrease and in combination with RTFS precision will still stay over 90%, which is very high for semi-automatic annotation solution.

Ontea algorithm disadvantage is a requirement to set up domain specific patterns. While annotation methods as C-PANKOW are more generic, Ontea is a simpler, faster solution with a better success rate, suitable for knowledge management, information extraction or knowledge acquisition applications, where large number of documents needs to be annotated. SemTag on the other hand is faster than Ontea but is not able to create new ontology metadata, only identify their existence in the knowledge base.

Main advantages of described method are: supporting the Slovak language, fast algorithm comparing to other methods, instance duplicity identification and very high precision.

7 Future work

The presented work is an intermediate result on our research on automatic and semi-automatic semantic annotation. Subsequent effort will be focused on studying and tuning instance relevance computation from the document collection. We plan to study the effect of increasing the distance parameter (distance larger than cardinality of C and I sequences), relevance computation based on C and I membership in a phrase in documents (instead of word distance concept), document preprocessing methods that would pre-format the selected terms in order to increase regular expressions matching precision. We will study the effects of extending the document collection (that form the basis for our statistical analysis) by text, which do not belong to the specific domain and we will examine the results obtained from document from different domains. We will also analyze the effects of document collection size on the instance relevance identification.

We would also like to evaluate Ontea with use of Lucene and evaluate better Ontea method on Slovak texts.

References

1. Hatcher E., Gospodnetić O., Lucene in Action, Manning (12 Jan 2005), ISBN: 1932394281
2. OpenFTS – http://openfts.sourceforge.net
3. mnoGoSearch – http://www.mnogosearch.org
4. Miller, G.:WordNet: An On-line Lexical Database, Special Issue, International Journal of Lexicography, Vol. 3, Num. 4, 1990
5. WordNet: http://wordnet.princeton.edu/
6. Návrat, P., Bieliková, M., Rozinajová, V.: Methods and Tools for Acquiring and Presenting Information and Knowledge in the Web. In: CompSysTech 2005, B. Rachev, A. Smrikarov (Eds.), Varna, Bulgaria, June 2005. pp. IIIB.7.1-IIIB.7.6.
7. M. Ciglan, M. Babik, M. Laclavik, I. Budinska, and L. Hluchy. Corporate memory: A framework for supporting tools for acquisition, organization and maintenance of information and knowledge. In J. Zendulka, editor, Proc. of 9th Int. Conf. on Information Systems Implementation and Modelling (ISIM 2006), pages 185--192. MARQ, Ostrava, 2006.
8. Jones, K. S., Willet, P: Readings in Information Retrieval, San Francisco: Morgan Kaufmann,1997, ISBN 1-55860-454-4.
9. Van Rijsbergen, C. J., Robertson, S. E., Porter, M. F. New models in probabilistic information retrieval. British Library Research and Development Report, no. 5587, British Library, London, 1980.
10. Cimiano P., Ladwig G., Staab S.: Gimme' the context: context-driven automatic semantic annotation with c-pankow. In WWW '05, pages 332-341, NY, USA, 2005. ACM Press. ISBN 1-59593-046-9.
11. Madche A., Staab S.: Ontology learning for the semantic web. IEEE Intelligent Syst., 16(2):72-79, 2001
12. Charniak E., Berland M.: Finding parts in very large corpora. In Proceedings of the 37th Annual Meeting of the ACL, pages 57-64, 1999.
13. Glover E., Tsioutsiouliklis K., Lawrence S., Pennock D., Flake G.: Using web structure for classifying and describing web pages. In Proc. of the 11th WWW Conference, pages 562-569. ACM Press, 2002.
14. Reeve L., Hyoil Han: Survey of semantic annotation platforms. In SAC '05, pages 1634-1638, NY, USA, 2005. ACM Press. ISBN 1-58113-964-0. doi: doi.acm.org/10.1145/1066677.1067049
15. Handschuh S., Staab S.: Authoring and annotation of web pages in cream. In WWW '02, pages 462-473, NY, USA, 2002. ACM Press. ISBN 1-58113-449-5. doi: http://doi.acm.org/10.1145/511446.511506.
16. Domingue J., Dzbor M.: Magpie: supporting browsing and navigation on the semantic web. In IUI '04, pages 191-197, New York, NY, USA, 2004. ACM Press. ISBN 1-58113-815-6.

17. Uren V. et al.: Semantic annotation for knowledge management: Requirements and a survey of the state of the art. Journal of Web Semantics: Science, Services and Agents on the WWW, 4(1):14-28, 2005.
18. Uren V. et al.: Browsing for information by highlighting automatically generated annotations: a user study and evaluation. In K-CAP '05, pages 75-82, NY, USA, 2005b. ACM Press. ISBN 1-59593-163-5
19. Michal Laclavik, Martin Seleng, Emil Gatial, Zoltan Balogh, Ladislav Hluchy: Ontology based Text Annotation – OnTeA; Information Modelling and Knowledge Bases XVIII. IOS Press, Amsterdam, Marie Duzi, Hannu Jaakkola, Yasushi Kiyoki, Hannu Kangassalo (Eds.), Frontiers in Artificial Intelligence and Applications, Vol. 154, February 2007, pp.311-315. ISBN 978-1-58603-710-9, ISSN 0922-6389.
20. S Dill, N Eiron, et al.: A Case for Automated Large-Scale Semantic Annotation; Journal of Web Semantics, 2003
21. R.Guha and R. McCool. Tap: Towards a web of data. `http://tap.stanford.edu/`.
22. Stanislav Krajči, Róbert Novotný: Hľadanie základného tvaru slovenského slova na základe spoločného konca slov; In: 1st Workshop on Intelligent and Knowledge oriented Technologies – WIKT 2006 Proceedings; Michal Laclavik, Ivana Budinska, Ladislav Hluchy (Eds.); November 28 – 29, 2006; Bratislava, Slovakia; March 2007; ISBN 978-80-969202-5-9; March 2007
23. William J. Teahan (2000), Text classification and segmentation using minimum cross entropy, In: RIAO-00, 6th International Conference "Recherche d'Information Assistee par Ordinateur", Paris, FR.
24. Ted Dunning (1994), Statistical identification of language. Techical Report MCCS-94-273, Computing Research Lab (CRL), New Mexico State University.
25. Peter Vojtek, Mária Bieliková (2007), Comparing Natural Language Identification Methods based on Markov Processes. In: Slovko – International Seminar on Computer Treatment of Slavic and East European Languages, Bratislava. Accepted.