RASLAN 2011
Recent Advances in Slavonic Natural Language Processing

librix.eu

A. Horák, P. Rychlý (Eds.)

RASLAN 2011

Recent Advances in Slavonic Natural Language Processing

**Fifth Workshop on Recent Advances in Slavonic Natural Language Processing, RASLAN 2011
Karlova Studánka, Czech Republic,
December 2–4, 2011
Proceedings**

Tribun EU
2011

Proceedings Editors

Aleš Horák
Faculty of Informatics, Masaryk University
Department of Information Technologies
Botanická 68a
CZ-602 00 Brno, Czech Republic
Email: `hales@fi.muni.cz`

Pavel Rychlý
Faculty of Informatics, Masaryk University
Department of Information Technologies
Botanická 68a
CZ-602 00 Brno, Czech Republic
Email: `pary@fi.muni.cz`

ISBN 978-80-263-0077-9

Preface

This volume contains the Proceedings of the Fifth Workshop on Recent Advances in Slavonic Natural Language Processing (RASLAN 2011) held on December 2nd–4th 2011 in Karlova Studánka, Sporthotel Kurzovní, Jeseníky, Czech Republic.

The RASLAN Workshop is an event dedicated to the exchange of information between research teams working on the projects of computer processing of Slavonic languages and related areas going on in the NLP Centre at the Faculty of Informatics, Masaryk University, Brno. RASLAN is focused on theoretical as well as technical aspects of the project work, on presentations of verified methods together with descriptions of development trends. The workshop also serves as a place for discussion about new ideas. The intention is to have it as a forum for presentation and discussion of the latest developments in the field of language engineering, especially for undergraduates and postgraduates affiliated to the NLP Centre at FI MU. We also have to mention the cooperation with the Dept. of Computer Science FEI, VŠB Technical University Ostrava.

Topics of the Workshop cover a wide range of subfields from the area of artificial intelligence and natural language processing including (but not limited to):

* text corpora and tagging
* syntactic parsing
* sense disambiguation
* machine translation, computer lexicography
* semantic networks and ontologies
* semantic web
* knowledge representation
* logical analysis of natural language
* applied systems and software for NLP

RASLAN 2011 offers a rich program of presentations, short talks, technical papers and mainly discussions. A total of 15 papers were accepted, contributed altogether by 22 authors. Our thanks go to the Program Committee members and we would also like to express our appreciation to all the members of the Organizing Committee for their tireless efforts in organizing the Workshop and ensuring its smooth running. In particular, we would like to mention the work of Aleš Horák, Pavel Rychlý and Dana Hlaváčková. The TEXpertise of Adam Rambousek (based on LATEX macros prepared by Petr Sojka) resulted in the extremely speedy and efficient production of the volume which you are now holding in your hands. Last but not least, the cooperation of Tribun EU as both publisher and printer of these proceedings is gratefully acknowledged.

Brno, December 2011 Karel Pala

Table of Contents

Part I

Logic and Language

Analyzing Time-Related Clauses in Transparent Intensional Logic

Aleš Horák, Miloš Jakubíček, Vojtěch Kovář

Faculty of Informatics
Masaryk University
Botanická 68a, 602 00 Brno, Czech Republic
{hales, jak, xkovar3}@fi.muni.cz

Abstract. The Normal Translation Algorithm (NTA) for Transparent Intensional Logic (TIL) describes the key parts of standard translation from natural language sentences to logical formulae. NTA was specified for the Czech language as a representative of a free-word-order language with rich morphological system.
In this paper, we describe the implementation of the sentence building part of NTA within a module of logical analysis in the Czech language syntactic parser synt. We show the respective lexicons and data structures in clause processing with numerous examples concentrating on the temporal aspects of clauses.

Key words: Transparent Intensional Logic, TIL, Normal Translation Algorithm, NTA, logical analysis, temporal analysis, time-related clauses

1 Introduction

Logical representation of a natural language sentence forms a firm basis for semantic analysis, machine learning techniques and automated reasoning. In current practical systems all these fields build on propositional and first-order logic due to the advantages and simplicity of their processing [1]. However, it can be shown that first-order formalisms are not able to handle systematically the natural language phenomena like intensionality, belief attitudes, grammatical tenses and modalities (modal verbs and modal particles in natural language).

In the following text, we describe a system theoretically based on the formalism of the Transparent Intensional Logic (TIL [2]), an expressive logical system introduced by Pavel Tichý [3], which works with a complex hierarchy of types, system of possible worlds and times and an inductive system for derivation of new facts from a knowledge base in development.

The actual implementation of the system works together with the system for syntactic analysis of the Czech language named *synt*. The parsing mechanism in `synt` [4,5] is based on the meta-grammar formalism working with a grammar of about 250 meta-rules with contextual actions for various phrase and sentence level tests. The Normal Translation Algorithm (NTA [6]) as a standard way of logical analysis of natural language by means of the compositions of syntactic constituents such as verb phrase, noun phrase, clause and their modifiers was provided by a basic prototype implementation. Current works on NTA have supplemented the process with high-coverage lexicons and new logical tests and rules with growing coverage on common news paper texts.

2 Logical Analysis on the Clause Level

The central point of each clause (as a "simple" sentence) is formed by the verb phrase, which represents the predicative skeleton of the clause. Tichý comes with a classification of significant verbs into two groups according to the classification of their meaning:

1. *attributive verbs* express what *qualities* the attributed objects *have* or what the objects *are*. Typical examples are sentences with adjectival predicates, such as

   Zmíněné zařízení je číslicově řízené.
   (The mentioned device is numerically controlled.)
   $\lambda w \lambda t (\exists i)\Big([[\mathbf{číslicově}_{wt}, \mathbf{řízený}], i] \;\wedge\; [[\mathbf{zmíněný}, \mathbf{zařízení}]_{wt}, i]\Big) \ldots \pi$

   The appropriate analysis of the attributive verbs lies in a proposition that ascribes the alluded property to the subject.
2. *episodic verbs*, on the other hand, express *actions*, they tell what the subject *does*:

   Celková úmrtnost klesá.
   (The overall mortality rate decreases.)
   $\lambda w_1 \lambda t_2 (\exists x_3)(\exists i_4)\Big([\mathbf{Does}_{w_1 t_2}, i_4, [\mathbf{Imp}_{w_1}, x_3]] \;\wedge\; x_3 = \mathbf{klesat}_{w_1} \;\wedge\; [[\mathbf{celkový}, \mathbf{úmrtnost}]_{w_1 t_2}, i_4]\Big) \ldots \pi$

The main difference between attributive and episodic verbs consists in their time consumption — the attributive verbs do not take the time dimension into account, they just describe the *state* of the subject in the very one moment by saying that the subject has (or has not) a certain property. The analysis of episodic verbs is defined with the means called *events* and *episodes*, which are in detailed described in [7,8,6]. An example of the resulting analysis uses three functions — **Does** and **Imp**/**Perf** which describe the relation between the subject and the predicate in the sentence and display the verb's aspect.

In natural language, the time constructions are encoded with two principle means — the verb tense and time adverbial groups that serve as modifiers of the default verb meaning.

The contemporary Czech language has three verb tenses — the present tense, the past tense and the future tense. For the basic verb tense analysis, we can adapt the definitions specified for English in [7]. Unlike English the Czech verbs have the capability of expressing the verb aspect built directly into their grammatical category — every verb is exactly in either the imperfective or the perfective form. A special property that goes along with the perfective aspect is that these verbs do not form the present tense — they have only the past tense and the future tense forms.[1] The present tense of the perfective verbs is usually expressed by their imperfective counterparts.

The verb tenses that allow us to assert propositions with respect to the past or to the future can be looked at as mirror images of each other. In the analysis, we understand the past tense as a certain operation working over

1. the *underlying proposition* in the present tense form and
2. the *reference time span*
3. with regard to an *assertion moment*

The meaning of a past tense proposition is often connected not only with a certain reference time span (in the indefinite case the object Anytime) but also with a *frequency adverb* specifying how many times the proposition happened to be true. A frequency adverb is analysed as a $((o(o\tau))\pi)_\omega$-object, i.e. as a world-dependent operation that takes a proposition p to the class of time intervals that have the requested qualities regarding the chronology of p. For instance, the adverb 'dvakrát' (twice) takes every proposition p to a class of time intervals that have exactly two distinct intersections with the chronology of p. If the frequency adverb is not specified in the sentence, we assume that the frequency of the proposition is 'at least once' (object Onc) in its time span, which means that we do not limit the sentence's time span in that case.

Thus, if P denotes the past tense function a general schema of the logical analysis of a past tense sentence looks like

$$\mathsf{P}\Big(\text{<frequency adverb>}(\text{<proposition>}),\ \text{<reference time span>}\Big)$$

The simple past is best seen as a time-dependent relation between $(o(o\tau))$-objects and $(o\tau)$-objects which holds for those cases where the (past part of the) reference time span belongs to the acceptable classes of time moments as obtained by the frequency modification of the proposition's chronology. An example analysis of a sentence in the past tense is

Celní správa vyžadovala originální certifikát.
(Customs administration required an original certificate.)

$$\lambda w_1 \lambda t_2 \Bigg[\mathbf{P}_{t_2}, \Bigg[\mathbf{Onc}_{w_1}, \lambda w_3 \lambda t_4 (\exists x_5)(\exists i_6)(\exists i_7)\Big(\big[\mathbf{Does}_{w_3 t_4}, i_7, [\mathbf{Imp}_{w_3}, x_5]\big] \wedge$$
$$\wedge\ \big[[\mathbf{origin\acute{a}ln\acute{i}}, \mathbf{certifik\acute{a}t}]_{w_3 t_4}, i_6\big] \ \wedge\ x_5 = [\mathbf{vy\check{z}adovat}, i_6]_{w_3} \ \wedge$$
$$\wedge\ \big[[\mathbf{celn\acute{i}}, \mathbf{spr\acute{a}va}]_{w_3 t_4}, i_7\big]\Big)\Bigg], \mathbf{Anytime}\Bigg] \ldots \pi$$

The future tense may be now defined analogously to the simple past tense as $\mathsf{F}/(o(o(o\tau))(o\tau))\tau$.

[1] e.g. 'začal/začne' (he started/he will start), 'zabil/zabije' (he killed/he will kill) or 'udělal/udělá' (he did/he will do).

3 Clauses Combined in Complex Sentences

The logical analysis described in the previous section did not exceed the boundaries of a single clause. Now, we are going to show how we can combine the subconstructions corresponding to particular clauses in order to capture the appropriate subordinate and coordinate relations between the clauses in the resulting construction of the whole sentence.

An important feature of a connected (subordinate or coordinate) clause is the way (usually the conjunction) how this clause connects to the principal clause. We use the clause conjunction as a leading element for determining the kind of the clause (and thus the form of its analysis).

For each syntactic rule which connects a clause with a (possible complex) sentence, the clauses conjunctions are extracted and a lexicon of *sentence schemata* is searched for the corresponding logical analysis. A thorough list of possible conjunctions and the corresponding logical analyses can be found in [6, pp.106–113], but here we will rather demonstrate the whole process on the actual implementation of this part of NTA. Let us take an example sentence

Petr včera přišel, když Markéta telefonovala babičce.
(Petr came yesterday, when Markéta phoned her grandmother.)

The conjunction 'když' (when/if) can have two readings in a sentence: a) the temporal *when*, or b) the implication *if* (*if then* is translated as 'když pak'). The logical analysis offers ambiguously both readings. The analysis process first find the corresponding constructions for both the simple clauses:

Petr včera přišel (Petr came yesterday):
`time span TIL:` $\lambda t_1 \textbf{včera}_{tt_1} \ldots (o\tau)$
`frequency TIL:` $\textbf{Onc} \ldots ((o(o\tau))\pi)_\omega$
`verbal object TIL:` $x_1/(o(o\pi)(o\pi)) = \textbf{přijít}_w \ldots o$

`clause TIL:` $\lambda w_1 \lambda t_2 \Big[\textbf{P}_{t_2}, \Big[\textbf{Onc}_{w_1}, \lambda w_3 \lambda t_4 (\exists x_5) \Big($

$\big[\textbf{Does}_{w_3 t_4}, Petr, [\textbf{Perf}_{w_3}, x_5]\big] \;\wedge\; x_5 = \textbf{přijít}_{w_3} \Big) \Big], \lambda t_7 \textbf{včera}_{t_2 t_7} \Big] \ldots \pi$

and

Markéta telefonovala babičce (Markéta phoned her grandmother):
`time span TIL:` $\textbf{Anytime} \ldots (o\tau)$
`frequency TIL:` $\textbf{Onc} \ldots ((o(o\tau))\pi)_\omega$
`verbal object TIL:` $x_1/(o(o\pi)(o\pi)) = [\textbf{telefonovat}, i_2]_w \wedge$
$\wedge\, [\textbf{babička}_{wt}, i_2] \ldots o$

`clause TIL:` $\lambda w_1 \lambda t_2 \Big[\textbf{P}_{t_2}, \Big[\textbf{Onc}_{w_1}, \lambda w_3 \lambda t_4 (\exists x_5)(\exists i_6)(\exists i_7) \Big($

$\big[\textbf{Does}_{w_3 t_4}, Markéta, [\textbf{Perf}_{w_3}, x_5]\big] \;\wedge\; [\textbf{babička}_{w_3 t_4}, i_6] \;\wedge$

$$\wedge\, x_5 = \big[\textbf{telefonovat}, i_6\big]_{w_3}\Big)\Big], \textbf{Anytime}\Bigg] \ldots \pi$$

When these two clauses are joined together, the corresponding sentence schema is looked up in the lexicon:

```
sentence_rule_schema: schema = S1 'když' S2
reading #1: lwt(tense_temp(awt(#1),awt(#2)))
```

The first reading corresponds to the *when* translation of 'když', i.e. the sentence is analysed as adverbial temporal clause where we do not ascribe the generating of a time class to the conjunction, but rather use the clause's construction directly as a generator of a collection of the time moments where the clause's extension is True. The subordinate clause is used as a generator of the reference time span – a characteristic function of a class of time moments ($\lambda t_0[\ldots]$).

Thus the original main clauses *reference time span* $\lambda t_7 \textbf{včera}_{tt_7}$ is replaced with

$$\lambda t_1\Bigg(\textbf{včera}_{tt_2} \;\wedge\; \Bigg[\textbf{P}_t, \Bigg[\textbf{Onc}_w, \lambda w_5 \lambda t_6 (\exists x_7)(\exists i_8)(\exists i_9)\Big($$
$$\Big[\textbf{Does}_{w_5 t_6}, \mathit{Markéta}, \qquad \ldots \qquad \Big)\Big], \lambda t_7(t_7 = t_1)\Bigg]\Bigg]\Bigg) \ldots ((o\tau)\tau)$$

The resulting analysis of the whole sentence in the first (temporal) reading looks like

$$\lambda w_1 \lambda t_2 \Bigg[\textbf{P}_{t_2}, \Bigg[\textbf{Onc}_{w_1}, \lambda w_5 \lambda t_6 (\exists x_7)\Big(\big[\textbf{Does}_{w_5 t_6}, \mathit{Petr}, [\textbf{Perf}_{w_5}, x_7]\big] \;\wedge$$
$$\wedge\, x_7 = \textbf{přijít}_{w_5}\Big)\Bigg], \lambda t_9\Bigg(\textbf{včera}_{t_2 t_9} \;\wedge\; \Bigg[\textbf{P}_{t_2}, \Bigg[\textbf{Onc}_{w_1},$$
$$\lambda w_{13} \lambda t_{14} (\exists x_{15})(\exists i_{16})\Big(\big[\textbf{Does}_{w_{13} t_{14}}, \mathit{Markéta}, [\textbf{Perf}_{w_{13}}, x_{15}]\big] \wedge [\textbf{babička}_{w_{13} t_{14}}, i_{16}] \;\wedge$$
$$x_{15} = \big[\textbf{telefonovat}, i_{16}\big]_{w_{13}}\Big)\Bigg], \lambda t_{18}(t_{18} = t_9)\Bigg]\Bigg]\Bigg)\Bigg] \ldots \pi$$

The second reading of the sentence with 'když' as *if* (*then*) is a sort of *causal adverbial* clauses. In the logical analysis it represents a certain relation between the propositions denoted by the main and causal clauses. Specifically, *if–then* as a *conditional* clause determines the condition (real or unreal) under which the main clause's assertion holds.

In analogy with the analysis of the causal adverbial phrases, we let the clause's conjunction to denote the appropriate relation between the two propositions. The second reading of the example sentence is looked up as

```
reading #2: lwt([awt(0(když/(((o((ot)w)((ot)w))t)w))),#1,#2])
Schema trivialization TIL: když...(oππ)τω
```

The whole sentence with the (in this situation semantically improper) reading as a conditional statement is analysed as

$$\lambda w_1 \lambda t_2 \Bigg[\mathbf{když}_{w_1 t_2}, \lambda w_3 \lambda t_4 \Bigg[\mathbf{P}_{t_4}, \Bigg[\mathbf{Onc}_{w_3}, \lambda w_5 \lambda t_6 (\exists x_7) \Big($$

$$\big[\mathbf{Does}_{w_5 t_6}, \mathit{Petr}, [\mathbf{Perf}_{w_5}, x_7]\big] \;\wedge\; x_7 = \mathbf{přijít}_{w_5}\Big)\Bigg], \lambda t_9 \mathbf{včera}_{t_4 t_9}\Bigg],$$

$$\lambda w_{10} \lambda t_{11} \Bigg[\mathbf{P}_{t_{11}}, \Bigg[\mathbf{Onc}_{w_{10}}, \lambda w_{12} \lambda t_{13} (\exists x_{14})(\exists i_{15}) \Big($$

$$\big[\mathbf{Does}_{w_{12} t_{13}}, \mathit{Markéta}, [\mathbf{Perf}_{w_{12}}, x_{14}]\big] \;\wedge\; \big[\mathbf{babička}_{w_{12} t_{13}}, i_{15}\big] \;\wedge$$

$$\wedge\; x_{14} = [\mathbf{telefonovat}, i_{15}]_{w_{12}}\Big)\Bigg], \mathbf{Anytime}\Bigg]\Bigg] \ldots \pi$$

The conjunction of some causal clauses could be also translated into their logical equivalents (with the help of $\Rightarrow$, $\neg$, $\wedge$ or $\vee$). Since this process may bring some inaccuracies into the analysis, we prefer in the NTA, at least at the moment, to keep the "causal" functions in one to one relation to the actual NL expressions that denote them. However, nothing hinders us from describing the implications between propositions in all respects by means of rules of the inference mechanism.

4 Conclusions

In the paper, we have described the implementation of the complex part of the Normal Translation Algorithm in Transparent Intensional Logic that corresponds to building complex sentences. The resulting logical analysis allows to capture any natural combination of the temporal characteristics of the combined propositions.

We have shown the data structures and lexicons used for possibly ambiguous processing of the semantics of complex sentence based on the conjunctions involved. A special treatment is payed to temporal clauses where their characteristics allow to generate the intensional chronology of the original proposition.

The resulting logical analyses form a quality input for any complex knowledge representation and reasoning system.

Acknowledgements This work has been partly supported by the Ministry of Education of CR within the LINDAT-Clarin project LM2010013, by EC FP7 project ICT-248307 and by the Czech Science Foundation under the project P401/10/0792.

References

1. Harrison, J.: Handbook of practical logic and automated reasoning. Cambridge University Press (2009)
2. Duží, M., Jespersen, B., Materna, P.: Procedural Semantics for Hyperintensional Logic. Foundations and Applications of Transparent Intensional Logic. Volume 17 of Logic, Epistemology and the Unity of Science. Springer, Berlin (2010)
3. Tichý, P.: The Foundations of Frege's Logic. de Gruyter, Berlin, New York (1988)
4. Horák, A., Kadlec, V.: New Meta-grammar Constructs in Czech Language Parser synt. In: Lecture Notes in Artificial Intelligence, Proceedings of Text, Speech and Dialogue 2005, Karlovy Vary, Czech Republic, Springer-Verlag (2005) 85–92
5. Horák, A.: Computer Processing of Czech Syntax and Semantics. Librix.eu, Brno, Czech Republic (2008)
6. Horák, A.: The Normal Translation Algorithm in Transparent Intensional Logic for Czech. PhD thesis, Faculty of Informatics, Masaryk University, Brno (2002)
7. Tichý, P.: The semantics of episodic verbs. Theoretical Linguistics **7** (1980) 264–296
8. Tichý, P.: The logic of temporal discourse. Linguistics and Philosophy **3** (1980) 343–369

Mind Modeling using Transparent Intensional Logic

Andrej Gardoň

NLP Centre, Faculty of Informatics, Masaryk University
Brno, Czech Repulic
`xgardon@fi.muni.cz`

Abstract. In this paper the possibility of human mind modeling is discussed. According the recent knowledge from cognitive and neurology sciences we inspect the human brain principles. Position of natural language (NL) is outlined alongside its connection with human senses. As a result the Theory of three layers (TTL) is introduced. Arguments from the Incompleteness theory are used to support its own truthfulness and legitimacy of TTL. Moreover the boundary between processes that can be modeled using AI and those that are purely characteristic for living beings is sketched. Finally we briefly describe the GuessME! system that uses Transparent Intensional Logic (TIL) to extract the natural language meaning.

Key words: Theory of mind, Theory of three layers, Universal grammar, TIL, GuessME!

1 Introduction

Artificial intelligence field has recently been amazed by the IBM's Watson system [1]. Watson participation in Jeopardy! quiz showed how a set of binary digits can defeat human contestants with a great advance in final points. Has the AI reached the point of the artificial brain? Truly, Watson is a kind of nerd as he uses powerful search algorithms to compete human players. Some categories from the quiz revealed Watson actually does not understand NL [2]. Other projects like True Knowledge [3] and Aura [4] also tries to simulate processes taking place in human brain. As none of the systems focuses on the overall architecture of the brain they are all limited to the simulated area. Search through memorized knowledge is the case of Watson, strict domains of information are found in Aura and True Knowledge is limited to simple factual questions [2]. We focus on data available through observations, neurology studies and cognitive science experiments, to inspect the brain and propose the theory of three layers that describes the principles of human intellect. At first, importance of natural language followed by the nature of objects denoted by its sentences is figured out. Then the countability of the artificial mind is discussed. As a result we describe a GuessME! system based on Transparent Intensional Logic. This system acquires knowledge through simple game that simulates the evolution of human brain during life.

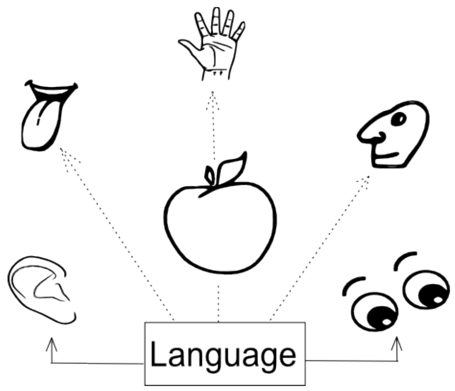


Fig. 1. Observing an apple by five senses accompanied by NL module providing communication channel through sensual and audible senses

2 Position of natural language

Natural language is a powerful tool to communicate not only our internal states but also anything else. Some people think NL is a cultural artifact that has to be learned like any other ability. But as Steven Pinker explains in his remarkable book [5] NL is rather an instinct and is developed spontaneously in childhood. One passage form the book describes a deaf boy born to deaf parents who learned the sing language in their late ages. According this fact they were unable to memorize all grammatical rules and they avoided complex signs. These parents had taught the boy their sign language and he spontaneously invented missing grammatical parts. Such phenomenon, reported as a movement from a pidgin language to creole one, outlines that NL is more likely to develop on DNA base rather than by cultural imprinting. Moreover we presume that it is Language itself that is innate and its actual nature depends on a culture and personal abilities. Story of the deaf boy shows how the imprinted rules of Language affect communications skills even if typical sources of information are limited. Observation of an apple by an average human delivers five codes (see Figure 1). Data from visual and audible senses seems to have an advantage. They are heavily used by NL as they provide rapid communication channel without visible effort. However NL does not use original information. Instead it is a set of generalized codes (words, sentences) that are easy to understand for a communication partner - saying the word "apple" is more efficient than painting its shape into sand. Natural selection identified vision and audition as the best candidates for this sort of reduction of actual information. This puts constraints on NL not Language itself as can be thought. Language is independent from the code used. One example is art. To find the nature of pictures by Pablo Picasso means to understand his Language developed over visual codes. Blind people miss essential sense for NL processing nevertheless they are able to acquire NL by introduction of new code provided by tactile sense. Consideration of the vision and audition as cornerstones of Language would put deaf-blind people into very sad life. Individuals with such handicap miss information from eyes and ears so their ability to interact with environment looks impossible. There is no way to teach those people NL but with special types of Language (usually based on touch) even mathematic can be learned by them [6]. We think every human, independently of available senses, disposes with a language based on the rules of Language (also called Universal grammar by Noam Chomsky [7]). Communication of such language depends on communicators. Deaf-blind baby born into environment that exclusively uses NL is comparable to misunderstood artist trying to sell own pictures. Independence of NL and Universal grammar can also be proved by Aphasia disorder caused by damage to certain parts in brain. When Wernicke's area undergoes damage patients are unable to understand words of NL [8]. They can formulate sentences but usually use random or invented words. Such behavior denotes existence of centralized system for mapping words of a language into corresponding codes provided by senses. People using sing language supports the theory by reporting similar troubles after the damage of the area [9]. Knowing the words one want to use in communication, the process proceeds to Broca's area responsible for grammatical structure of a sentence. Malformation in this area causes non-fluent sentences with disjointed words [10]. Complicated language elements including passive are misinterpreted [11] revealing the inability to process the syntax. Again sign language can be similarly limited by damage in the area. Consequently two subparts of language are distinguished - one for dictionary and one for actual grammar of the language. Whether one uses NL, sign language or any other form of language it seems all are processed by these parts. Aphasia studies showed another surprising fact. When people with damage to Broca's area had recovered they reported that they were aware what they wanted to say but could not express themselves [8].

3 Subject of Natural language and the theory of incompleteness

The aphasia disorder introduced in the previous section, reported not only the autonomous position of Language but also the independence of its subject. One deaf Mexican Indian lived for more than 20 years in total isolation as his parents were unable to sign. He had developed no language but was found obviously intelligent [12]. Language-less people are intelligent due to thoughts. However, this kind of thinking should not be confused with a sort of a Language. Artists use their language to express thoughts not vice versa. When Albert Einstein was cogitating on the theory of relativity could he depend only on NL or some formal logic system (FLS)? A FLS try to describe the nature of world and discover new facts about it. But the necessity of axioms and inference rules position it on the same level with NL (see Figure 2). Albert Einstein discovered his theory, according P.J. Laird and the book "The Computer and the mind" [13],

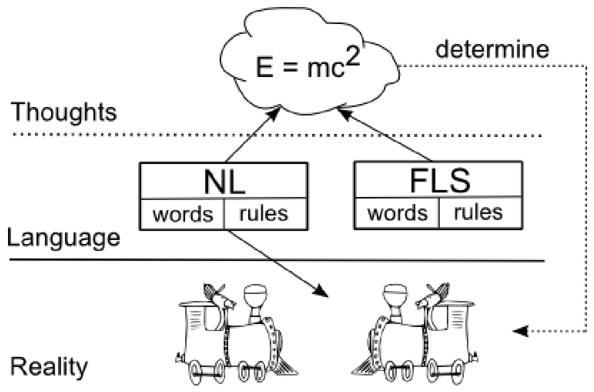


Fig. 2. Thoughts at the topmost level

thanks Calculation, Creation, Induction or Deduction. Abstractedly from the principle used he had had to realize the nature of the things. This could not happen at the level of language as no writer was able to synthesize such text. His observations, purely based on thoughts, were described by a language (recall famous examples of the Relativity theory with trains traveling at the speed of light) only after he had realized them. Thoughts seem to be generally available and the real art of science is the ability to realize them and create appropriate terms in a language for them. Darwinian Theory massively uses natural selection principle. Was this principle available only after people had named it? We consider thoughts as a foundation of all objects in reality. Esoteric nature of claims is supported by Kurt Gödel and the Theory of incompleteness. Proofs used by this theory are on edge for many people. Sentence *"This sentence is not provable"* is difficult to understand. Taking figure 2 into account, we position this sentence both into Language and Thoughts layer. Simply said it tries to associate non existing reality with thought. Science usually observes reality and describes it with a language (bottom-up processing). Einstein used top-down processing while he cogitated in thoughts, than described them by NL and confronted with reality. Gödel tried different processing. He created a structure combining all three layers and postulated equality between them. The structure proved the impossibility of its own existence and introduced unusual paradox. Anyone can think about such structure, describe it by language but fails to confront it with reality. Human senses detect material objects and consequently codes available to a language are mainly connected with Reality. Reasoning is dependent from the theory of truthfulness and humans tend to confront everything with outside world. This is the key of Gödel's theory. It shows the independence of Reality, Language and Thoughts. While Language can describe both Reality and Thoughts all three layers postulate a hierarchy as in figure 2. Theory of three layers (TTL) put Reality at the bottom. Position of Language in the middle is hard to prove. Basically, principles of Language have roots in Thoughts and no language can exists without those ideas. It is possible to describe these principles by NL (language is a set of words over an alphabet accompanied by rules to simplify codes and allow the communication) but their application is actually used. One can put the principles to account of natural selection but will always end up with a set of ideas (accidence, natural selection) that seems to be major. TTL presuppose Thoughts are the fundamental concept and consider Gödel's theory as a fact limiting what can be computed. Any language (NL, FLS, C++, etc.) is limited by Universal grammar set within Thoughts. Even neural networks are regulated by principles from Thoughts. Neuroplasticity is the ability of neural network to accommodate various tasks. Area usually processing touch can learn to treat visual information [14] or blind people can use tongue to see [15]. Although Thoughts at the top level of the hierarchy can be hardly proved by science (operating over Reality) several graduated scientists have confirmed the benefits resulting from positive tuning of human mind [16] [17]. Brain as a product of reality can inspect the surrounding world through senses, use a language to describe and communicate it. However, the character of the mind settled within the brain is not possible to specify. TTL puts the mind into the first layer and therefore excludes it from computational set of problems. Indubitable a sort of language in the brain of Einstein was incorporated while discovering the Theory of relativity but what coded the ability to realize key concepts for the theory (initial axioms)? How the universal program for the following story would look like?

There is a well in the sandy desert. The settlement of aborigines is two kilometers far away. Every morning a man has to take bottles to the well for refilling. One morning when the man is walking towards a well a horse is galloping next to him. How should he realize that by riding the horse the refilling is much more efficient?

TTL excludes the existence of such program. Any reasoning measurable by science takes place in the brain and therefore is a sort of language (inference mechanism of FLS is an application of Language's principles). Realization of axioms and initial rules must be done by mind set in uncountable Thoughts layer. Artificial intelligence can take advantages of efficiency, faultlessness but will always be left to the smartness of living beings.

4 Using TIL as a language

Although the previous section claimed the nature of the mind it also proved the possibility of NL understanding by a computer. The key task is a FLS that implicitly codes the sentence's information - subject, objects, time tense, relations, etc. With a natural language sentence Broca's and Wernicke's areas probably process the associations between different codes and ensure appropriate formulation of relations between them. The Transparent Intensional Logic [18] is capable of simulating these areas. A NL parser (e.g. SYNT [19]) transcribes a sentence into TIL construction that implicitly codes all information from the sentence (relations, attitudes, time tense, etc.). Such construction is computa-

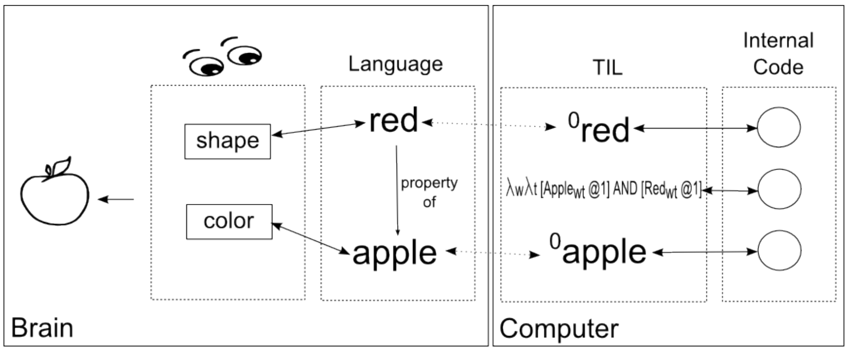


Fig. 3. Using TIL as a language for a computer

tionally feasible and can be processed by the Dolphin Nick system [20] - the TIL interpreter. Understanding of NL depends from senses therefore it is necessary to introduce a virtual code that maps on the TIL constructions. Figure 3 plots the situation. While words in human brain are associated to codes from senses, in a computer system virtual code is used for this purpose. Notice the possibility of visual information (in the form of camera output) incorporation. Generally any sense can replace the internal code but as artificial perception is under intensive research, virtual code is chosen. Using TIL one can teach the computer to understand facts and check consistency of theories. Thanks the TIL property of possible worlds, one can postulate axioms and rules for the theory of chemical reactions independently from the theory of physics. It is even possible to combine them in a new theory.

Taking TTL facts into account we designed a GuessME! system. It is inspired by Watson and Aura projects which try to create an artificial agent capable of communicating in NL. Neither project does it properly [2]. Either it ignores the meaning (Watson) or suppresses the NL itself (Aura). GuessME! is based on a simple game for two players. One player chooses an object or event which should be then guessed by a competitor. Beside Yes/No questions, players are allowed to use time (was it before...?) and structures having a set of simple words as the answer (what is the color of...?). By playing such game computer can obtain new facts. Actually the main purpose of the GuessME! is a simulation of the human brain evolution during life. No one is born intelligent and has to examine the surrounding world. In system like Watson huge libraries are developed to operate over. Usually the system is not aware of the facts meaning. It uses search algorithms or kind of inference mechanism to obtain answers. In GuessME! software we grant the independence of asking. Universal grammar sets the existence of properties, relations and events. By generating questions about objects' properties or relations, system can obtain new information. Books from first classes of elementary school are used to teach GuessME! initial knowledge. We hope such system will achieve higher levels of NL understanding than any other AI system developed till these days.

5 Conclusions

This paper sets the boundary between computers and living beings by consulting results from recent scientific experiments. Theory of three layers excludes the existence of a formal calculus for mind and therefore prohibits existence of software as intelligent as human. It positions the science in the Reality layer and enforces the incompleteness of any artificial project. Hereby it reveals the possibility of processing NL by a computer. As a result Transparent Intensional Logic is chosen to enable the extraction of NL meaning by a computer. Progressive method used in GuessME! system allows acquiring of knowledge in natural manner and introduces new method of AI system education.

Acknowledgments. This work has been partly supported by the Czech Science Foundation under the project P401/10/0792.

References

1. Ferrucci, D., Brown, E., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A.A., Lally, A., Murdock, J.W., Nyberg, E., Prager, J.: Building Watson : An Overview of the DeepQA Project. AI Magazine **31**(3) (2010) 59–79
2. Gardoň, A.: Towards An Intelligent Question-Answering System. In: Proceedings of the International Conference on Agents and Artificial Intelligence (ICAART 2012), Portugal (2012)
3. Tunstall-Pedoe, W.: True Knowledge: Open-Domain Question Answering Using Structured Knowledge and Inference. AI Magazine **31**(3) (2010)
4. Gunning, D., Chaudhri, V.K., Clark, P., Barker, K., Chaw, S.Y., Greaves, M., Grosof, B., Leung, A., McDonald, D.D., Mishra, S., Pacheco, J.: Project Halo Update-Progress Toward Digital Aristotle. AI Magazine **31**(3) (2010) 33–58
5. Pinker, S.: The Language Instinct : How the Mind Creates Language (Perennial Classics). Harper Perennial Modern Classics (2000)
6. Řezáčová, P.: Specifika edukačního procesu u jedinců s hluchoslepotou (Particularities of education process for deaf-blind people). PhD thesis, Masaryk university (2007)
7. Cook, V.J., Newson, M.: Chomsky's Universal Grammar: An Introduction. Wiley-Blackwell (2007)
8. Tanner, P.: Exploring the Psychology, Diagnosis, and Treatment of Neurogenic Communication Disorders. iUniverse.com (2010)
9. Hickok, G., Bellugi, U., Klima, E.S.: How Does The Human Brain Process Language? SCIENTIFIC AMERICAN **June** (2001) 46–53
10. Purves, D., Williams, S.M.: Neuroscience. NCBI bookshelf. Sinauer Associates (2001)
11. Grodzinsky, Y.: The neurology of syntax: Language use without Broca's area. Behavioral and Brain Sciences **23**(1) (2000) 1–21
12. Schaller, S., Sacks, O.W.: A man without words. University of California Press (1995)
13. Johnson-Laird, P.: The Computer and the Mind: An introduction to Cognitive Science. Harvard University Press (1989)
14. Laserson, J.: From Neural Networks to Deep Learning: zeroing in on the human brain. XRDS **18**(1) (2011) 29–34

15. Williams, M.D., Ray, C.T., Griffith, J., De L'Aune, W.: The Use of a Tactile-Vision Sensory Substitution System as an Augmentative Tool for Individuals with Visual Impairments. Journal Of Visual Impairment& Blindness **105**(1) (2011) 45–50
16. Dyer, D.W.W.: Change Your Thoughts - Change Your Life: Living the Wisdom of the Tao. Hay House (2009)
17. Lore, M.J.: Managing Thought: How Do Your Thoughts Rule Your World? Ferne Press (2008)
18. Tichý, P.: Collected Papers in Logic and Philosophy. Prague: Filosofia, Czech Academy of Sciences, and Dunedin: University of Otago Press (2004)
19. Horák, A.: Computer Processing of Czech Syntax and Semantics. 1 edn. Librix.eu, Brno, Czech Republic (2008)
20. Gardon, A., Horák, A.: Time Dimension in the Dolphin Nick Knowledge Base Using Transparent Intensional Logic. In: TSD. (2011) 323–330

Verbs as Predicates: Towards Inference in a Discourse

Zuzana Nevěřilová, Marek Grác

NLP Centre, Faculty of Informatics,
Masaryk University, Botanická 68a, 602 00 Brno, Czech Republic

Abstract. Generally, it is usual that communication partners mention only facts that are not believed to be known to both parties of the communication. This shared implicit knowledge is also known as common sense knowledge. Because of not mentioning everything important it is difficult (if not impossible) to analyze a discourse for computer programs without any background knowledge.
In this paper a relationship between logic and language is discussed. In NLP systems verbs are often seen as predicates and verb valencies are seen as arguments. According to this approach new sentences (propositions) can be inferred from the discourse.
Second, a system for inference from verb frames was created and a evaluation proposal is described. We have picked up 174 verbs occurring in Czech cooking recipes. For these verbs 232 inference rules were manually created. The inference process was tested on a corpus of 37 thousands tokens (2 400 sentences). As the result 253 new sentences were generated.

Key words: inference, common sense inference, corpus annotation, verb valency lexicon

1 Introduction

Much information in a discourse is not explicit. For example, the cookbook story "fry the onion till it looks glassy" actually means peel a fresh, uncooked onion, chop it, put grease into a cooking pot and heat it, put the onion into the pot and wait until the onion looks glassy. In natural language processing (NLP) systems we have to deal with this phenomenon to resolve stories such as: fry the onion till it looks glassy, reduce heat and cover. Where the heat comes from? What to cover?

Texts in natural languages usually contain "facts" (also known as common sense propositions or common sense facts) that are considered to be true in "normal" situations (also referred as stereotypical information [6]), e.g. fried onion looks glassy. From such facts some other propositions can be inferred, e.g. the glassy onion was fried in a cooking pot.

Henry Lieberman argues that common sense inference (CSI) differs from mathematical inference (MI). While MI operates with exact definitions, universally true statements and provides complete reasoning, CSI operates with vague definitions, therefore it infers contingent statements that are considered true until the opposite cannot be proved (also known as non-monotonic reasoning) [8]. In [13] a broader definition of logic is provided: "any precise notation for expressing statements that can be judged true or false". In the same context an inference rule is defined as "a truth-preserving transformation: when applied to a true statement, the result is guaranteed to be true".

In this paper we concentrate on inferring new propositions, obvious for humans, but unreachable for computer programs. The method is based on transformations on syntactic level and evaluation on semantic level. This means that the system works with syntactic units such as noun phrase (NP) and verb phrase (VP) and during the evaluation the meaning of the proposition is examined. As an example domain the Czech cooking recipes corpus was created and processed.

Within this domain we have constructed 232 inference rules for 174 verbs. The inference process was tested on a corpus of 37 thousands tokens (2 400 sentences).

2 Logics and inference

Mathematical inference takes place in several logics. Propositional (Boolean) logic is considered to be the basic logic. It uses well known reasoning patterns such as modus ponens or and-elimination [11, p. 239]. Propositional logic is useful since it is easy to implement, but for complex reasoning tasks it is not enough expressive. First-order logic introduces quantifiers, predicates and variables.

In knowledge representation mathematical logic (such as propositional or first-order logic) is not used directly. Usually knowledge is stored as objects in categories [11, p. 350] and the basic reasoning pattern is the inheritance. "Description logics provides a formal language for constructing and combining category definitions and efficient algorithms for deciding subset and superset relationships between categories." [11, p. 377]

Unlike MI principal inference tasks in DL are subsumption (checking if one category is a subset of another by comparing their definitions) and classification (checking if an object belongs to a category) [11, p. 381]. The DL is usually represented by a set of inference rules describing stereotypes ("normal" situations).

3 Verb Frames

Verb frames are closely related to the argument structure of sentences, but also to the lexical meaning of the VP itself and its dependents. "Argument structure is an interface between the semantics and syntax of predicators (which we may take to be verbs in the general case). Its function is to link lexical semantics to syntactic structures." [1]

Verbs mostly describe an action or state. Since the verb "is a hook upon which the rest of a sentence hangs" [12], it is often seen as a predicate (for example *to_fry*(x, y) means that x fries y). Verb valency refers to the number of arguments of a verbal predicate — the capacity of a verb to bind a certain number and type of syntactically dependent language units [9]. Syntactic valencies describe the syntactic properties (such as subject or object) of an argument. In Czech (as well as other Slavic languages) syntactic properties are expressed by the case and possibly a preposition (e.g. syntactic subject is in nominative).

VerbNet (English language) [12] and VerbaLex (Czech language) [4] are examples of current collections of verb frames and their arguments (in the frame lexicons often called slots). These collections capture the syntactic information (e.g. information about prepositions and cases of the arguments in VerbaLex) as well as semantics (reference to semantic roles and Princeton WordNet [2] (PWN) hypernym in VerbaLex, flat representation using predicates in VerbNet). In our work we proceeded from VerbaLex and its relation to PWN.

4 Inferring New Propositions

In this section we describe the whole process from preparing the data and the inference rules to applying the rules to corpus data and generating new sentences.

4.1 The Language of Cooking Recipes

The language of cooking recipes differs from the general language in the following attributes:

– use of imperative. In Czech cooking recipes most cooking recipes authors use first person plural instead of imperative (literally "we fry the onion…"). Sometimes, infinitive (literally "to fry the onion…") or imperative forms are used.
– frequent use of coordinations of NPs and of VPs
– use of adverbs describing duration and manner

4.2 Corpus Annotation

The annotation method was that of the BushBank project [3]. The corpus was annotated on several language levels: tokens (words and sentence boundary marks), morphology (lemma and morphologic tag for tokens), syntactic structures (NPs, VPs, coordinations and clauses), syntactic structures relations (dependencies). Annotation itself was done purely by automatic tools (desamb [10], SET [5]) and manual annotation was used for confirmation of syntactic structures and relations between them. It means that structures that were not identified by automatic tools could not be added by annotators.

This was done contrary to traditional requirements in which we tried to obtain completness of annotation. BushBank ideas put greater impact on *simplicity* of annotation (without definition of all border-line cases), *usability* (proved by this project of inference) and *rapid-development* (annotation itself was done in 40 (wo)man-hours). As we are working on concept, data were manually checked by just one annotator.

4.3 Creating Inference Rules

Verb valencies refer to the arguments of a verbal predicate (see 3. Therefore valencies play a critical role in inference. In this section we will outline the inference:

Let C_i be the input clause, I be the inference rule and C_o be the output clause. Then the inference is a function $I(C_i) = C_o$.

Both C_i and C_o are sets of NPs and a VP: $C_i = \{N_{i1}, N_{i2}, \ldots, N_{im}, V_i\}$ and $C_i = \{N_{o1}, N_{o2}, \ldots, N_{on}, V_o\}$. The inference rule describes:

- the type of the inference (see below)
- what NPs will participate the transformation
- what syntactic changes are needed for each transformed NP

Therefore the inference rule I is a tuple $(\mathbb{S}, t)$, where $\mathbb{S}$ is a set of syntactic transformation rules and t is the inference type (see below). Each transformation rule $S \subset \mathbb{S}$ defines a transformation for an input pair $(preposition_i, case_i)$ to an output pair $(preposition_o, case_o)$. Prepositions can be either none (direct case) or prepositions agreeing to a case. Case is marked be a number[1].

The system covers the following inference types t: effect, precondition, decomposition, equivalence. These relations are often used in discourse planning and are therefore present in common sense knowledge bases such as CyC [7]. The use of inference types is more general than just saying *a implies b*. Moreover, it is relatively straightforward for humans to describe inference rules of these types.

4.4 Generating New Propositions

In this section the whole process of inferring new propositions is described. First, the input text is parsed by the SET syntactic parser [5]. The output of this parsing consists of decomposition on sentences and clauses, identification of constituents of the clause and grammatical properties of NPs.

Second step consists of annotation of valencies. This step is necessary to eliminate NPs independent of the verb (e.g. parts of adverbial phrases). This process is so far done manually but with the use of verb valency lexicon VerbaLex [4]. Annotators had to detect if a NP is a valency of the verb according to VerbaLex. They could benefit from VerbaLex binding to PWN (e.g. they can easily detect what meaning of the verb to choose).

[1] 1 – nominative, 2 – genitive, 3 – dative, 4 – accusative, 6 – locative or 7 – instrumental

```
<title>'opéct' has precondition 'rozpálit pánev'</title>
<verbalex:inference type="precondition" verb="opéct">
   <verbalex:ruleset id="heat_pan" inferred_verb="rozpálit pánev"
      negation="False">
     <verbalex:rule case="c1" prep="" inferred_case="c1"
        inferred_prep="" />
     <verbalex:rule case="c4" prep="" inferred_case="c4"
        inferred_prep="na" />
   </verbalex:ruleset>
</verbalex:inference>
```

Fig. 1. Inference rule that says that *roast* (opéct) has precondition of *heating the pan* (rozpálit pánev)

Third step is the application of inference rules. Each NP identified as a verb valency and the VP are the input of inference rules (such as in Figure 1. If the grammatical agreement in case and preposition is fulfilled the NP is transformed according the particular syntactic transformation. The morphological parser majka [14] is used for generating the new NP (the grammatical number is the one property not affected by the transformation). Note that not all NPs contained in the inference rules has to be present. In such cases only those NPs that are present are transformed. This can sometimes lead to semantically incomplete sentences such as "we have the meat".

New sentences are generated as a sequence of new NPs and a VP. Since Czech language uses nearly free word order this method is correct. If the syntactic transformation is correct the method generates a syntactically correct phrase.

5 Evaluation Proposal

Since the project is a work-in-progress this section will only outline the evaluation steps. Evaluation will proceed on two levels: syntactic and semantic. At each level annotators will have to decide whether or not the new sentence is correct. On syntactic level, there is one possible source of ambiguity: mistaking nominative/accusative forms. This ambiguity is eliminated during the (manual) valency detection. Therefore a failure of the new sentence on this level points to either an error in the inference rule, or corpus annotation error, or morphological generator error.

On semantic level only syntactically correct (those with a good annotator agreement on syntactic level) sentences will appear. Here the annotators have to decide whether or not the sentence is "true". The truthfulness will be judged in the context between two clauses of the original text, first of them is the inference input clause. Here we expect low coverage but high precision.

We have chosen a well delimited domain of cooking recipes. Therefore we should avoid most problems with lexical ambiguity. However, we expect that problems with lexical ambiguity arise in case of highly polysemous verbs such as put, give or bring. In this case further evaluation of the results will be needed.

6 Conclusion and Future Work

In this paper we have discussed the relation between natural language and logic. We suppose that logic and inference are widely used within a discourse but we have to broaden the definition of inference. With the notion of inference as a transformation we did experiments on a cooking recipes corpus. The result of applying inference rules on a 37 000 size corpus is that 253 new sentences were generated. Since the paper is a preliminary research in this area the evaluation is not finished, but a evaluation method is proposed.

In future, we plan to evaluate in detail the "quality" of the new sentences and examine the failures on each level. Moreover we plan to add adverbial constituents to inference rules because in the case of cooking recipes we have observed that the adverbs strongly influence the semantic of the inferred proposition. Next step will consist of picking up other objects that are not mentioned in the discourse. In case of cooking recipes these objects will be pans, pots, spoons, grease etc.

Acknowledgements This work has been partly supported by the Ministry of Education of CR within the Center of basic research LC536 by the Czech Science Foundation under the project 407/07/0679.

References

1. Bresnan, J.: Lexicality and argument structure. In: Minimal Ideas. pp. 283–304. John Benjamins Publishing Company, Paris (Apr 1996)
2. Fellbaum, C.: WordNet: An Electronic Lexical Database (Language, Speech, and Communication). The MIT Press (May 1998), published: Hardcover
3. Grác, M.: Case study of BushBank concept. In: PACLIC 25 25th Pacific Asia Conference on Language, Information and Computation. p. 9 (2011), [release Dec 2011]
4. Hlaváčková, D., Horák, A.: Verbalex – new comprehensive lexicon of verb valencies for czech. In: Proceedings of the Slovko Conference (2005)
5. Kovář, V., Horák, A., Jakubíček, M.: Syntactic analysis using finite patterns: A new parsing system for czech. In: Human Language Technology. Challenges for Computer Science and Linguistics: 4th Language and Technology Conference, LTC 2009, Poznan, Poland, November 6-8, 2009. p. 161 (2011)
6. Lehmann, D.J.: Stereotypical reasoning: Logical properties. CoRR cs.AI/0203004 (2002)
7. Lenat, D.B.: CYC: a Large-Scale investment in knowledge infrastructure. Communications of the ACM 38(11), 33–38 (1995)
8. Lieberman, H.: MAS.964 Common Sense Reasoning for Interactive Applications. Massachusetts Institute of Technology: MIT OpenCourseWare (2002), [Retrieved on-line from `http://ocw.mit.edu`; accessed 11-January-2006]. License: Creative Commons BY-NC-SA

9. Lopatková, M.: Valency Lexicon of Czech Verbs: Towards Formal Description of Valency and Its Modeling in an Electronic Language Resource. Habilitation thesis, Charles University in Prague, Prague (2010)
10. Pavel Šmerk: Towards Morphological Disambiguation of Czech. Ph.d. thesis proposal, Faculty of Informatics, Masaryk University (2007)
11. Russell, S.J., Norvig, P., Candy, J.F., Malik, J.M., Edwards, D.D.: Artificial intelligence: a modern approach. Prentice-Hall, Inc., Upper Saddle River, NJ, USA (1996)
12. Schuler, K.K.: VerbNet: A Broad-Coverage, Comprehensive Verb Lexicon. Ph.D. thesis, Faculties of the University of Pennsylvania (2005)
13. Sowa, J.: Fads and fallacies about logic. IEEE Intelligent Systems p. 84–87 (2007)
14. Šmerk, P.: Fast morphological analysis of czech. In: Proceedings of the Raslan Workshop 2009. Masarykova univerzita (2009)

Part II

Syntax, Morphology and Lexicon

Czech Morphological Tagset Revisited

Miloš Jakubíček, Vojtěch Kovář, Pavel Šmerk

Natural Language Processing Centre
Faculty of Informatics, Masaryk University
Botanická 68a, 602 00 Brno, Czech Republic
{jak,xkovar3,xsmerk}@fi.muni.cz

Abstract. Lot of natural language processing is built on top of some solid morphological annotation. In this paper we present an update of the Czech morphological tagset as given by the analyzer Ajka that has been used for academic as well as commercial purposes for more than dozen years. The revision reacts on rather practical issues that we had to face during development of subsequent tools for NLP, parsers in the first place. We describe the reasoning behind each of the changes and include the full updated tagset reference manual. Finally we provide a comparison and mapping to the Universal tagset as produced by Google.

Key words: morphology, tag, tagset, annotation, Czech

1 Introduction

Morphology is usually the core of many NLP applications and we are confident that its usefulness heavily depends on the underlying tagset. Despite 20 years of intensive development of NLP applications, there are no conclusions on a widely accepted universal tagset standard across multiple languages, and mostly even within a single language. For Czech, two tagsets have been available since the 90's, provided by two leading NLP groups: one developed in the Institute of Formal and Applied Linguistics at the Charles University in Prague[1] and another one in the NLP Centre at the Masaryk University in Brno[2]. This paper presents a revision of the second tagset together with the underlying morphological database used by the analyzer Majka[3,4]. The main principles of the tagset are outlined in Section 2 and the tagset itself is provided in Appendix A. In Section 3 we describe the changes to the tagset and reasoning and motivation behind them. Finally we provide the current tagset reference together with basic disambiguation rules.

2 An Attributive Tagset for Czech

The main properties of the morphological tagset that is described in this paper are as follows:

- *attributive*
  A tag is a sequence of `xY` pairs, denoting that `Y` is the value of the attribute `x`.
- *non-positional*
  The position of the attribute-value pairs in the tag does not matter.

In Figure 1 we provide a sample annotation of the Czech sentence 'Máme zaměstnance , které občas vysíláme na služební cestu.' (*We have employees that we sometimes send on a business trip*). For explanation of the tags meaning, refer to the Appendix A.

Máme	(We) have	`k5eAaImIp1nP`
zaměstnance	employees	`k1gMnPc4`
,	,	`kIx,`
které	that	`k3yRgMnPc4`
občas	sometimes	`k6eAd1`
vysíláme	(we) send	`k5eAaImIp1nP`
na	on	`k7c4`
služební	business	`k2eAgFnSc4d1`
cestu	trip	`k1gFnSc4`
.	.	`kIx.`

Fig. 1. Example of the annotation using current tagset standard.

3 Changing Tagset

We are fully aware of the fact that doing changes to an existing and well-established tagset is an unpopular step that implicates compatibility issues with the old revision, might arouse confusion among current users and definitely should not be carried out without careful consideration of the overall impact. Having that in mind, we briefly outline the most important reasons that led us to take this decision:

- **usability**
  The tagset and Majka have been used extensively in many NLP applications for over dozen years and we can profit from that experience to make the annotation standard more useful in terms of its informativeness and benefit for particular applications (e. g. parsers).
- **consistency**
  Though every effort has been made to eliminate incosistencies in the original tagset as they might be confusing for the users, everyday usage of the tagset showed that one could still make improvements in this respect.
- **simplicity**
  Einstein's famous quote saying that 'everything should be made as simple as possible, but no simpler.' is in the case of any standardization even more

appropriate than otherwise and we took the opportunity to follow it more closely.

- **standardization**
  The current description of the tagset is not really up to date since it is quite often the case that different tools use differently evolved versions of the original tagset. This paper aims at creating a new standard that will be subject to further references and common development. We consider the old tagset description to be version 1.0, this new one to be version 2.0 and intend to continue versioning of the tagset in case of future changes.
- **decidability and disambiguability**
  For anything in the tagset there must be a clear deterministic procedure saying how a word should be annotated, the goal being a very short manual for disambiguation. This requirement moved us to the decision that the tagset should distinguish between two levels of annotation:
  - **restricted (poor) tagset**
    The poor tagset will be restricted to attributes that we expect to be annotated manually (in case of corpus annotation) with very high inter-annotator agreement. In other words it will contain only those attribute where anybody with basic linguistic understanding of the related grammatical notion will be, having the annotation manual available, capable to determine the attribute value.
  - **full (rich) tagset**
    The rich tagset is a superset of the poor one and will superseed it by containing also attributes that do not fulfill our strict requirements imposed on the poor tagset, e. g. attributes that are assigned to a very small set of word forms and can be automatically generated from the lexicon (hence do not require manual disambiguation).

Below we list all changes to the tagset that have been performed, together with a detailed explanation of what was the motivation to conduct such a change. The description is structured according to the part-of-speech kinds, i. e. the `k` attribute. Different change types are marked by the related bullets as follows:

■ an attribute value has been removed
□ an attribute value has been added
● an attribute has been removed
○ an attribute has been added
★ disambiguation note

3.1 k1 – Substantives

★ *substantive-adjective collision*
For any word form that is tagged both as an substantive and adjective there must be serious corpus evidence that the word indeed falls into both of these categories, otherwise only substantive or adjective must be chosen. Under adjective usage we understand that the word describes a property of

some other word (e.g. *červený*); the substantial usage means the word can be used as such and does not directly express the property of something else (e.g. *vrátný, pohřešovaný*).

■□ *family gender*
The `gR` attribute value has been removed and all tags containing `gR` are transformed to `gM` with an additional subclassification attribute value `xF`. This is mainly to simplify further processing since these words behave syntactically as animate masculines.

■ *family number*
The `nR` attribute value has been removed as duplicate to the `gR` (see above).

3.2 k2 – Adjectives

■ *dual number*
The `nD` attribute value has been removed as dual should be handled just as a variant of plural. The same applies for pronouns and numerals.

★ *adjective-verb collision*
There are problems on the syntactic layer caused by the duality between a short form of an adjective and a verb in past participle (e.g. *pečen*). We introduced a new disambiguation rule saying that if the relative verb exists, it is always the verb. Also, the morphological database needs manual checking of all these dualities.

3.3 k3 – Pronouns

● *person*
The `p` attribute is to be assigned only for words forms of *já, ty, on, my, vy, oni*. In the other cases it has currently no usage and is rather confusing.

□ *gender*
The `g` attribute should be specified always except for derivatives of *se, si, kdo, co, někdo, něco, nikdo, nic, já, ty, my, vy*.

3.4 k4 – Numerals

■ *xG, xH*
The `xG` and `xH` attribute values have been removed as there were no adjectives with such tag in the morphological database.

★ *noun-numeral collision*
Syntactic agreement should be used to disambiguate between a noun and a numeral: if there are usages where agreement applies (e.g. *s tisíci psy*), it is a numeral. Otherwise (if the word is always followed by a genitive phrase) it is a noun.

3.5 k5 – Verbs

■ *biaspectual verbs*
The aB attribute value has been removed. Instead, both aI and aP tags will be used in the morphological database for relevant verbs. When tagging particular corpus occurences, the aspect should always be disambiguated.

3.6 k6 – Adverbs

■ *xM, xS*
The xM and xS attribute values have been removed as there are no data with them in the morphological database. The respective information is to be added into the rich tagset under the t attribute.

3.7 k9, k0 – Particles and Interjections

★ *revise ambiguity*
In the morphological database, there are lot of ambiguous words where one of the options is k9, k0. Sometimes this ambiguity is relevant (e.g. *spíš*) but in many other cases it just causes disambiguation problems and brings no added value. Namely, the conjunctions should not be handled as particles simultaneously as even humans are not able to agree on related disambiguation rules. All these ambiguities need to be manually gone through and disambiguated.

3.8 kY – Conditionals

●□ *class removed*
The whole class will be split between conjunctions (*aby, kdyby* and their derivatives) and particles (*by* and its derivatives) as this division better corresponds to their syntactic behaviour. zY attribute will be added to these words to mark the conditional. Person and number will not be determined for simplicity.

3.9 kA – Abbreviations

●□ *class removed*
There is no syntactic or semantic motivation for this part of speech. Rather than that, it causes a lot of problems in automatic syntactic analysis. The words will be divided into the other categories according to their syntactic and semantic properties. zA attribute will be added to these words to mark the abbreviation.

3.10 kI – Punctuation

○ *class added*
We have added a new kI attribute for all types of punctuation. The punctuation was not marked before in any way. An x subclassification attribute is to be specified as per the tagset reference in Appendix B.

3.11 Common attributes

- □ *frequency characteristics*
  We have added a new common `~` attribute which can be used to store unspecified frequency characteristics (e. g. relative frequency in a particular corpus, normalized to the scale from 0 to 9).
- ■ *derivative information*
  The `rD` attribute marking the derivation sequence of a word form has been removed. A separate derivative morphological database is currently being prepared, will be available in the future and respective tagset attributes will be added accordingly.
- ■ *stylistic subclassification*
  The `wA, wC, wE, wK, wO` attributes have been all removed as they are not used anymore in the morphological database.

3.12 Gender Problems

We were also concerned with the problem that grammatical gender was not specified for some numerals (e. g. *deset*). At first we were about to add all possible gender values (i. e. `M, I, F, N`) to the tags of these numerals. However a detailed corpus analysis has shown that in some cases where we expect a syntactic agreement in case, number and gender, there is no real agreement in gender – there are no examples that would distinguish one gender from another by a word form. This concerns adjectives, pronouns and numerals in genitive, dative, local and instrumental case of plural. Every noun phrase we have found in the corpus showed the agreement just in the case and number.

For example, an adjective phrase *s těmi deseti malými* (*with those ten small*) will remain the same no matter if we are talking about masculina (*s těmi deseti malými černoušky, s těmi deseti malými hrady*), feminina (*s těmi deseti malými ženami*) or neutra (*s těmi deseti malými městy*). According to our corpus research, all possible adjective, pronoun and numeral phrases behave in the same way.

Based on this observation, we decided (in contrast to our primary intention) to remove the gender attribute from all the adjectives, pronouns and numerals in the respective cases as it does not reflect the real behaviour.

3.13 Canonical ordering

While non-positionality is a handy property when the tagset is used by people, its automated processing resulted into preferring one particular ordering of attributes as a sort of industry standard. This ordering is now part of the tagset reference and it is recommended to be used by applications and their APIs.

4 Remaining issues

In this section, we outline some of the problems that have been discussed but so far we were not able to agree on a solution. Many of these problems relate to the extreme complexity of Czech morphology.

4.1 Numerals and Pronouns vs. Nouns and Adjectives

From one point of view, almost all numerals and pronouns behave in a very similar way as nouns and/or adjectives, with just marginal differences. However, the differences remain there and so far we were not able to agree on a way this similarity should be expressed on the morphological layer.

4.2 Gender of Numerals

It is the nature of the Czech language that many words of the same part of speech behave in a slightly different way. It is then debatable where it is meaningful to mark some properties (see problems with expressing grammatical gender above). Definitely we do not want to mark something that does not describe a real phenomenon. On the other hand, we want the formalism to be simple enough for people to remember it and to be able to manually tag a sentence.

One example for all is the gender of numerals: The numerals *jedna, dva* (*one, two*) do have gender and this is important in syntactic agreements. It is not completely clear if these two are the only ones (and thus the only two that should have gender marked) – this would require another corpus study to reveal the real behaviour and set some sensible rules for the gender assignment.

5 Mapping to the Google Universal Tagset

Together with this morphological tagset release, we decided to create a mapping to the universal tagset created by joint effort of Google Research and Carnegie Mellon University [5]. The mapping is given in Table 1.

Table 1. Mapping of the Czech tagset to the Google Universal Tagset

universal tag	description	attributive tags
VERB	verbs (all tenses and modes)	k5.*
NOUN	nouns (common and proper)	k1.*
PRON	pronouns	k3.*
ADJ	adjectives	k2.*, k4.*xO, k4.*xR
ADV	adverbs	k6.*
ADP	adpositions (prepositions and postpositions)	k7.*
CONJ	conjunctions	k8.*
DET	determiners	(none)
NUM	cardinal numbers	k4.*xC
PRT	particles or other function words	k9.*
X	other: foreign words, typos, abbreviations	k0
.	punctuation	kI

6 Conclusions

We have introduced some practically motivated changes to the attributive tagset for the Czech language. The newly defined tagset should become a new standard that all the tools will be compliant with. With this release, we will also start versioning of the tagset with the hope that we will avoid much confusion in the future. We have provided the mapping from the newly defined standard to the Google universal tagset that is hoped to be an interlingually compatible tagset.

We have also mentioned some remaining open problems and outlined the future research in the dark area of Czech morphology.

Acknowledgements This work has been partly supported by the Ministry of Education of CR within the Center of basic research LC536 and by the Czech Science Foundation under the projects P401/10/0792 and 407/07/0679.

References

1. Hana, J., Zeman, D., Hajič, J., Hanová, H., Hladká, B., Jeřábek, E.: Manual for Morphological Annotation PDT. Technical Report 27, Institute of Formal and Applied Linguistics, MFF UK, Prague, Czech Republic (2005)
2. Pala, K., Rychlý, P., Smrž, P.: DESAM – Annotated Corpus for Czech. In: Proceedings of SOFSEM '97, Springer-Verlag (1997) 523–530
3. Šmerk, P.: Fast Morphological Analysis of Czech. In: Proceedings of the RASLAN Workshop 2009, Brno (2009)
4. Šmerk, P.: Towards Computational Morphological Analysis of Czech. PhD thesis, Faculty of Informatics, Masaryk University, Brno (2010)
5. Petrov, S., Das, D., McDonald, R.: A universal part-of-speech tagset. Arxiv preprint ArXiv:1104.2086 (2011)

A Current Tagset (revision from 2006)

k1 – Substantives

x	Special paradigm
P	půl (half)

g	Rod
M	Animate masculine
I	Inanimate masculine
N	Neuter
F	Feminine
R	Family (surname)

n	Number
S	Singular
P	Plural
D	Dual
R	Family (surname)

c	Case
1–7	First–Seventh

w	Stylistic flag
A	Archaism
B	Poeticism
C	Only in corpora
E	Expressive
H	Conversational
K	Bookish
O	Regional
R	Rare
Z	Obsolete

z	Word Form Type
S	-s enclitic

k2 – Adjectives

e	Negation
A	Affirmation
N	Negation

g	Gender
M	Animate masculine
I	Inanimate masculine
N	Neuter
F	Feminine

n	Number
S	Singular
P	Plural
D	Dual

c	Case
1–7	First–Seventh

d	Degree
1	Positive
2	Comparative
3	Superlative

w	Stylistic Flag
A	Archaism
B	Poeticism
C	Only in corpora
E	Expressive
H	Conversational
K	Bookish
O	Regional
R	Rare
Z	Obsolete

z	Word Form Type
S	-s enclitic

k3 – Pronomina

x	Type (x)
P	Personal
O	Possessive
D	Demonstrative
T	Delimitative

y	Type (y)
F	Reflexive
Q	Interrogative
R	Relative
N	Negative
I	Indeterminate

p	Person
1	First
2	Second
3	Third
X	First, second or third

g	Gender
M	Animate masculine
I	Inanimate masculine
N	Neuter
F	Feminine

n	Number
S	Singular
P	Plural
D	Dual

c	Case
1–7	First–Seventh

w	Stylistic Flag
A	Archaism
B	Poeticism
C	Only in corpora
E	Expressive
H	Conversational
K	Bookish
O	Regional
R	Rare
Z	Obsolete

z	Word Form Type
S	-s enclitic

k4 – Numerals

x	Type (x)
C	Cardinal
O	Ordinal
R	Reproductive
G	Grammar
H	Grammar

y	Type (y)
N	Negative
I	Indeterminate

g	Gender
M	Animate masculine
I	Inanimate masculine
N	Neuter
F	Feminine

n	Number
S	Singular
P	Plural
D	Dual

c	Case
1–7	First–Seventh

w	Stylistic Flag
A	Archaism
B	Poeticism
C	Only in corpora
E	Expressive
H	Conversational
K	Bookish
O	Regional
R	Rare
Z	Obsolete

t	Grammar Terminal
A–F	Terminal A–F
I–R	Terminal I–R
S	Q@
T	O@
U	L@
V	jedno
W	sto
X	dvě
Y	stě
Z	tři/čtyři

z	Word Form Type
S	-s enclitic

k5 – Verbs

e	Negation
A	Affirmation
N	Negation

a	Aspect
P	Perfect
I	Imperfect
B	Biaspectual

m	Type (Mode)
F	Infinitive
I	Present indicative
R	Imperative
A	Active part. (past)
N	Passive part.
S	Adv. part. (present)
D	Adv. part. (past)
B	Future indicative

p	Person
1	First
2	Second
3	Third

g	Gender
M	Animate masculine
I	Inanimate masculine
N	Neuter
F	Feminine

n	Number
S	Singular
P	Plural

w	Stylistic Flag
A	Archaism
B	Poeticism
C	Only in corpora
E	Expressive
H	Conversational
K	Bookish
O	Regional
R	Rare
Z	Obsolete

z	Word Form Type
S	-s enclitic

k6 – Adverbs

e	Negation
A	Affirmation
N	Negation

x	Pron. Adv. Type (x)
D	Demonstrative
T	Delimitative
M	Modal
S	Status

y	Pron. Adv. Type (y)
Q	Interrogative
R	Relative
N	Negative
I	Indeterminate

d	Degree
1	Positive
2	Comparative
3	Superlative

w	Stylistic Flag
A	Archaism
B	Poeticism
C	Only in corpora
E	Expressive
H	Conversational
K	Bookish
O	Regional
R	Rare
Z	Obsolete

z	Word Form Type
S	-s enclitic

k7 – Preposition

c	Case
1	First
2	Second
3	Third
4	Fourth
5	Fifth
6	Sixth
7	Seventh

k8 – Conjunction

x	Type
C	Coordinate
S	Subordinate

z	Word Form Type
S	Word form with -s enclitic

k9 – Particle

z	Word Form Type
S	Word form with -s enclitic

k0 – Interjection

kA – Abbreviation

kY – by, aby, kdyby

m	Relation to the Verb Mode
C	conditional

p	Person
1	First
2	Second
3	Third

n	Number
S	Singular
P	Plural

w	Stylistic Flag
A	Archaism
B	Poeticism
C	Only in corpora
E	Expressive
H	Conversational
K	Bookish
O	Regional
R	Rare
Z	Obsolete

Notes

Tag	Note
wH	795
rD,rD	INF : ADJ-cí
rD,rD	INF : ADJ-ší
rD,rD,rD,rD	INF : ADJ-ý : SUBST-í : ADJ-n//-t
rD,rD,rD	INF : SUBST-í : ADJ-cí
rD,rD,rD,rD,rD,rD	INF : SUBST-í : ADJ-cí : SUBST-í : ADJ-ý : ADJ-n//-t
rD,rD,rD	INF : SUBST-í : ADJ-ší
rD,rD,rD,rD,rD,rD,rD	INF : SUBST-í : ADJ-ší : ADJ-ší : SUBST-í : ADJ-ý : ADJ-n//-t
rD,rD,rD,rD	INF : SUBST-í : ADJ-ý : ADJ-cí
rD,rD,rD,rD,rD,rD,rD	INF : SUBST-í : ADJ-ý : ADJ-cí : SUBST-í : ADJ-ý : ADJ-n//-t
rD,rD,rD,rD	INF : SUBST-í : ADJ-ý : ADJ-ší
rD,rD,rD,rD,rD,rD,rD	INF : SUBST-í : ADJ-ý : ADJ-ší : SUBST-í : ADJ-ý : ADJ-n//-t
rD,rD,rD,rD,rD	INF : SUBST-í : ADJ-ý : ADJ-n//-t : ADJ-cí
rD,rD,rD,rD,rD,rD	INF : SUBST-í : ADJ-ý : ADJ-n//-t : ADJ-cí : ADJ-cí
rD,rD,rD,rD,rD	INF : SUBST-í : ADJ-ý : ADJ-n//-t : ADJ-ší
rD,rD,rD,rD,rD,rD	INF : SUBST-í : ADJ-ý : ADJ-n//-t : ADJ-ší : ADJ-ší
rD,rD,rD,rD,rD,rD,rD,rD	INF : SUBST-í : ADJ-ý : ADJ-n//-t : SUBST-í : ADJ-ý : ADJ-n//-t : ADJ-cí
rD,rD,rD,rD,rD,rD,rD,rD	INF : SUBST-í : ADJ-ý : ADJ-n//-t : SUBST-í : ADJ-ý : ADJ-n//-t : ADJ-ší
rD,rD,rD,rD,rD,rD,rD,rD,rD	INF : SUBST-í : ADJ-ý : ADJ-n//-t : SUBST-í : ADJ-ý : ADJ-n//-t : ADJ-ší : ADJ-ší
rD,rD,rD,rD,rD,rD,rD	INF : SUBST-í : ADJ-ý : SUBST-í : ADJ-ý : ADJ-n//-t : ADJ-cí
rD,rD,rD,rD,rD,rD,rD	INF : SUBST-í : ADJ-ý : SUBST-í : ADJ-ý : ADJ-n//-t : ADJ-ší
_,hF	SUBST : FEMPOSS
_,hM	SUBST : MASKPOSS
_,_,hM,hF,_,hR	M : F : Mpřivl : Fpřivl : rodina : Rpřivl
wZ	Obsolete
wB	Poeticism
tQ	Expresses extent
tA	Expresses respect
tL	Expresses place
tT	Expresses time
tC	Expresses reason
tM	Expresses manner
tD	Modal adverb
tS	Status adverb
wR	Rare
hT	Represents thing
hP	Represents person

xC	Cardinal numeral
xO	Ordinal numeral
xR	Reproductive numeral
yQ	Interrogative
yR	Relative
xD	Demonstrative
yN	Negative
xT	Delimitative
yI	Indeterminate
xP	Personal pronomina
yF	Reflexive pronomina
xO	Possessive pronomina
xC	Coordinate conjunction
xS	Subordinate conjunction
c1	Preposition with first case
c2	Preposition with second case
c3	Preposition with third case
c4	Preposition with fourth case
c6	Preposition with sixth case
c7	Preposition with seventh case
aP	Perfect
aI	Imperfect
aB	Biaspectual
wH	Conversational
wN	Dialectal

B New Tagset

Common attributes

k	Part-of-speech
1	Substantives
2	Adjectives
3	Pronomina
4	Numerals
5	Verbs
6	Adverbs
7	Prepositiona
8	Conjunctions
9	Particles
0	Interjections
I	Punctuation

g	Gender (k1–k4)
M	Animate masculine
I	Inanimate masculine
F	Feminine
N	Neuter

c	Case (k1–k4, k7)
1–7	First–Seventh

n	Number (k1–k4)
S	Singular
P	Plural

e	Negation (k2, k5, k6)
A	Affirmation
N	Negation

d	Degree (k2, k6)
1	Positive
2	Comparative
3	Superlative

p	Person (k3, k5)
1–3	First–Third

w	Stylistic subclassification (k0–k9)
B	Poeticism
H	Conversational
N	Dialectal
R	Rare
Z	Obsolete

z	Common subclassification (k1–k9)
S	Contains *-s* enclitic
Y	Word form of *aby, kdyby, by*
A	Abbreviation

⋆ **~**	Statistical characteristics
0–9	E. g. frequency

k1 subclassification

⋆ **x**	Type
P	Word form of *půl*
F	Family surname

k3 subclassification

x	Type
P	Personal
O	Possessive
D	Demonstrative
T	Delimitative

y	Type
F	Reflexive
Q	Interrogative
R	Relative
N	Negative
I	Indeterminate

k4 subclassification

x	Type
C	Cardinal
O	Ordinal
R	Reproductive

y	Type
N	Negative
I	Indeterminate

k5 subclassification

m	Type (mode)
F	Infinitive
I	Present Indicative
R	Imperative
A	Active part. (past)
N	Passive part.
S	Adv. part. (present)
D	Adv. part. (past)
B	Futreu indicative

a	Aspect
P	Perfect
I	Imperfect

k6 subclassification

x	Type
D	Demonstrative
T	Delimitative

y	Type
Q	Interrogative
R	Relative
N	Negation
I	Indeterminate

⋆ **t**	type
S	Status
D	Modal
T	Expresses time
A	Expresses respect
C	Expresses reason
L	Expresses place
M	Expresses manner
Q	Expresses extent

k8 subclassification

x	Type
C	Coordinate
S	Subordinate

kI subclassification

x	punctuation list
.	.!?
,	,:;
"	";'„"
(	({[<
)	)}]>
~	~$%^&-_+=\\|/# etc.

Attribute-to-PoS assignment

k	attributes
1	gnczw~
2	egncdzw~
3	gncpxyzw~
4	gncxyzw~
5	eampgnzw~
6	edxytzw~
7	cw~
8	xzw~
9	zw~
0	w~
I	x~

Canonical ordering

kegncpamdxytzw~

Korean Parsing in an Extended Categorial Grammar

Juyeon Kang

NLP Centre, Faculty of Informatics,
Masaryk University, Botanická 68a, 602 00 Brno, Czech Republic
qkang@aurora.fi.muni.cz

Abstract. This paper gives an automatic morpho-syntactical analysis with the ACCG parser which use the Categorial Grammar, the Combinatory Logic in the framework of Cognitive and Applicative Grammar. We focus on the contribution of the parser to the analysis of morphological case system in Korean.

Key words: Categorial Grammar, Combinatory Logic, Cognitive and Applicative Grammar, Korean Parsing, Case system

1 Introduction

The theoretical point of view of this work is that all linguistic units of Natural Languages are operators and operands with functional types. We show how it is possible to build up formal semantic representations from morpho-syntactical configurations using Extended Categorial Grammar and the combinators of Combinatory Logic (CL) with functional types [4] (Curry and al., 1972). CL is a useful formalism for studying the grammatical and lexical meanings [8] (Desclés, 1999). All expressions of CL are applicative expressions where an operator is applied to an operand. CL is generated from abstract operators, called combinators, whose aim is to combine more elementary operators (for instance, linguistic units into complex operators).

We give an automatic syntactical analysis with the ACCG parser by focusing on the analysis of morphological case system in Korean within the framework of the Cognitive and Applicative Grammar (CAG) model.

2 Frameworks

Categorial Grammars [7] are systems of types (analogue to Church's functional types); the instances of types are linguistic units analyzed as operators and operands. The calculus on syntactical types (or Lambek calculus, van Benthem, 1988) is closely associated to applications of operators onto operands. It has already been studied how Combinatory Logic [4] can be used with success for a semantic and computational analysis of voices, for instance in accusative and ergative natural languages [8]. The linguistic units being operators with assigned types, they can be composed by different ways. Thus CL is an adequate and "natural" formalism to express applicative expressions built by the application of operators to operands, and different compositions between operators. Indeed, CL is a logic of operators with abstract operators, called "combinators", which are used to compose and to define complex operators from more elementary operators. In an applicative calculus, combinators are introduced or eliminated by rules in Gentzen's style (Fitch, 1974). CL analyzes new concepts introduced in theories (logical, mathematical, linguistic, computer science, biology and nano-structure... theories) by an equivalence law between a definiendum and a definiens. The definiendum is a new unit and the definiens is an applicative expression where a combinator X describes how different more elementary operators are combined together.

The explicit articulation between morpho-syntactic configurations (organized by concatenation) and semantic and cognitive representations is described inside a computational architecture with intermediary levels. This architecture is defined in the formal and linguistic model of Cognitive and Applicative Grammar [1].

2.1 Cognitive and Applicative Grammar (CAG)

CAG is analogue to a compiling program with 7 interrelated levels of representations. This model, presented as a "bottom-up" analysis, which allows relating semantic representations and linguistic observables by means of formal calculus abstract.

The 7 levels are:

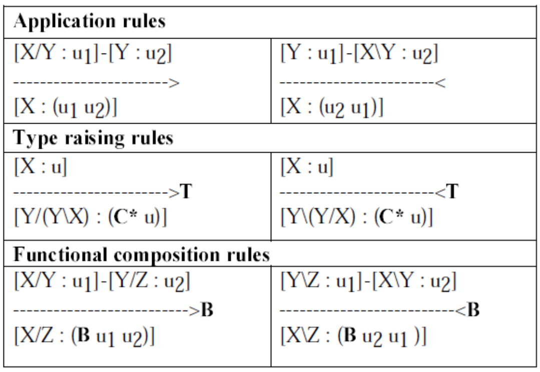
1. ***morpho-syntactical configuration level*** where the sentences are presented as concatenational strings (level (1));
2. ***operator-operand level*** is the result of an Extended Categorial Grammar analysis (ACCG parser); it is a set of applicative expressions associated to sentences of the level (1) (level (2));
3. ***analysis of diathesis and topicalisations*** in using combinators of Combinatory Logic (level (3)) [8];
4. ***analysis and representation of speaking acts*** for describing tenses, aspects (see below analyses), modalities and commitment operations (level (4));
5. ***formal representation of the meaning of lexical predicates by Semantic-Cognitive Schemes*** (SCS) (level (5));
6. ***integration of speaking conditions with SCS*** (level (6));
7. ***cognitive representation level*** (by diagrams or iconic representations) in relation with cognitive abilities of perception and action (level (7)).

This work concerns the levels (1), (2) and (3).

2.2 Applicative Combinatory Categorial Grammar

The Applicative Combinatory Categorial Grammar formalism is an extension of the Combinatory Categorial Grammar developed by Steedman. This ACCG formalism was originally developed by J-P. Desclés and I. Biskri (1995, 1996) for the analysis of coordination and subordination structure in French with the tools of Combinatory Logic by introducing canonical associations between some rules and the combinators.

We present here the rules[1] of the ACCG for the analysis of Korean sentences.



Application rules	
[X/Y : u_1]-[Y : u_2] ------------------------> [X : (u_1 u_2)]	[Y : u_1]-[X\Y : u_2] ------------------------< [X : (u_2 u_1)]
Type raising rules	
[X : u] ------------------------>**T** [Y/(Y\X) : (**C*** u)]	[X : u] ------------------------<**T** [Y\(Y/X) : (**C*** u)]
Functional composition rules	
[X/Y : u_1]-[Y/Z : u_2] -------------------------->**B** [X/Z : (**B** u_1 u_2)]	[Y\Z : u_1]-[X\Y : u_2] --------------------------<**B** [X\Z : (**B** u_2 u_1)]

Fig. 1. ACCG's rules

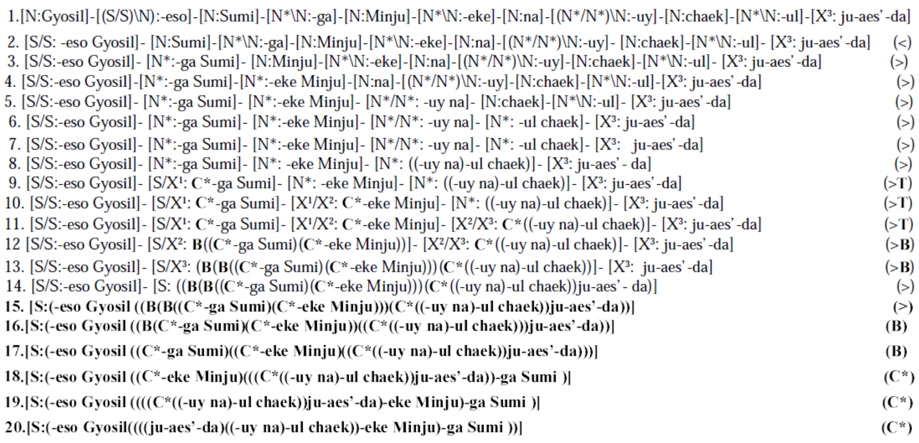
To the two classical basic types N(nominal) and S(sentence), we add a new basic type N* for the complete nominal phrases.

We use predefined notations to facilitate our categorial analysis.

$$X^0=S$$
$$X^1=(S\backslash N^*)$$
$$X^2=(S\backslash N^*)\backslash N^*$$
$$X^3=((S\backslash N^*)\backslash N^*)\backslash N^*$$

[1] **B** is a composition combinator. Its β-reduction is: Bfgx->f(gx). It is joined to the functional composition rule. This combinator allows us in particular to handle the free word order structure in the Korean sentence. **C*** is a type raising combinator joined to the type raising rule. Its β-reduction is: C*fg->gf This combinator transforms the operand (argument) to operator (function). It is used essentially to analyze nouns of the Korean as the operators.

Let us analyze the following sentence including the five major cases:
(1) *Gyosil-eso, Sumi-ga Minju-ege na-ege chaek-eul ju-oess-da*
Class-LOC Sumi-NOM Minju-DAT me-GEN book-ACC give–PS-DC
In the class, Sumi gave my book to Minju.



```
1.[N:Gyosil]-[(S/S)\N):-eso]-[N:Sumi]-[N*\N:-ga]-[N:Minju]-[N*\N:-eke]-[N:na]-[(N*/N*)\N:-uy]-[N:chaek]-[N*\N:-ul]-[X^3: ju-aes'-da]
2. [S/S: -eso Gyosil]- [N:Sumi]-[N*\N:-ga]-[N:Minju]-[N*\N:-eke]-[N:na]-[(N*/N*)\N:-uy]- [N:chaek]-[N*\N:-ul]- [X^3: ju-aes' -da]   (<)
3. [S/S:-eso Gyosil]- [N*:-ga Sumi]- [N:Minju]-[N*\N:-eke]-[N:na]-[(N*/N*)\N:-uy]-[N:chaek]-[N*\N:-ul]- [X^3: ju-aes' -da]   (>)
4. [S/S:-eso Gyosil]-[N*:-ga Sumi]-[N*:-eke Minju]-[N:na]-[(N*/N*)\N:-uy]-[N:chaek]-[N*\N:-ul]-[X^3: ju-aes' -da]   (>)
5. [S/S:-eso Gyosil]- [N*:-ga Sumi]- [N*:-eke Minju]- [N*/N*: -uy na]- [N:chaek]-[N*\N:-ul]- [X^3: ju-aes' -da]   (>)
6. [S/S:-eso Gyosil]- [N*:-ga Sumi]- [N*:-eke Minju]- [N*/N*: -uy na]- [N*: -ul chaek]- [X^3: ju-aes' -da]   (>)
7. [S/S:-eso Gyosil]- [N*:-ga Sumi]- [N*:-eke Minju]- [N*/N*: -uy na]- [N*: -ul chaek]- [X^3:   ju-aes' -da]   (>)
8. [S/S:-eso Gyosil]- [N*:-ga Sumi]- [N*: -eke Minju]- [N*: ((-uy na)-ul chaek)]- [X^3: ju-aes' - da]   (>)
9. [S/X^1: C*-ga Sumi]- ... 
```

Table 1. Syntactic types of case markers

Case		**Examples**	**Syntactic types**
Nominative		*-i/ga, -eun/neun, -kkeseo, -eso*	$(S/X^1)\backslash N$
Accusative		*-eul/reul*	$(X^1/X^2)\backslash N$, $(X^2/X^3)\backslash N$
Dative		*-ege, -kke, -hante, -bogo, -deoreo*	$(X^1/X^2)\backslash N$
Genitive		*-uy*	$(N/N)\backslash N$
Adverbials	**Place**	*-e, -eso, -eul/reul*	$(X^1/X^1)\backslash N$
	Depart	*-eso, -eul/reul*	$(X^1/X^1)\backslash N$
	Direction	*-e, -lo/eulo*	$(X^1/X^1)\backslash N$
	Goal of action	*-e, -lo/eulo, -eul/reul*	$(X^1/X^1)\backslash N$
	Quality	*-e, -lo*	$(X^1/X^1)\backslash N$
		-eulo	$(X^2/X^2)\backslash N$
	Time	*-e*	$(X^1/X^1)\backslash N$, $(S/S)\backslash N$
		-eul/reul	$(X^1/X^1)\backslash N$
	Instrument	*-lo/eulo*	$(X^1/X^1)\backslash N$
	Situation	*-e, -lo/eulo*	$(X^1/X^1)\backslash N$
	Cause	*-e, -lo/eulo*	$(X^1/X^1)\backslash N$
Vocative		*-a/ya, -yeo*	$(S/X^1)\backslash N$

more semantic and cognitive analyses. This categorial parser, ACCG[2], is based on the Applicative Combinatory Categorial Grammar [1], which is an extended version of the Categorial Grammars [7,9]. It uses combinators of the CL.

We wrote the algorithms of the categorial calculi (on syntactic functional types) with an implementation of applicative combinatory categorial rules and meta-rules (Biskri, 1995; Kang, Desclés, 2008). We present the results of the sentence (2) obtained by the ACCG Parser (Figure 2).

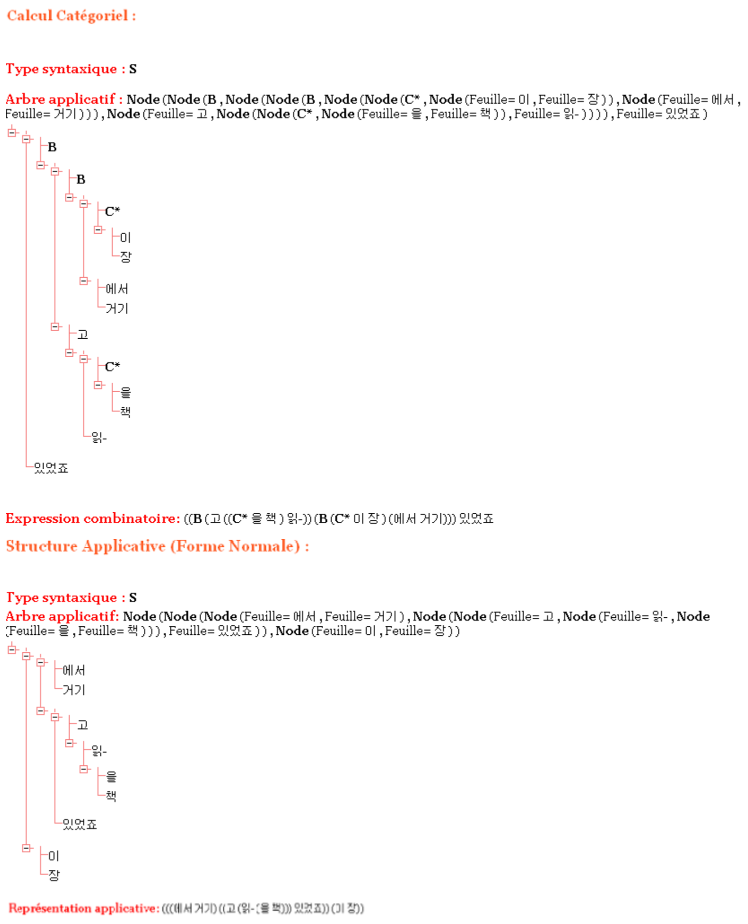
(2)장이 거기에서 책을 읽고 있었죠.

Jean-i	geogi-eso	chaek-eul	il-go	iss-eoss-jo
John-Nom	there-Place	book-Acc	read-Comp.	is-Ps-Nar.

John was reading a book there

The applicative representations marked in Figure 2 show the structures of operator/operand of the example sentences. This result is the applicative tree generated from the applicative representation. Comparing the results obtained by other categorical parsers, such as CCG Parser [3] and POSPAR [2], ACCG Parser offer linguistically correct analyses (See [6] for the comparison of the systems).

[2] The ACCG Parser runs to give a syntactic analysis in French, in English and in Korean, but in this paper, we focus on the results obtained in Korean.



Calcul Catégoriel :

Type syntaxique : **S**

Arbre applicatif : **Node** (**Node** (**B** , **Node** (**Node** (**B** , **Node** (**Node** (**C*** , **Node** (Feuille= 이 , Feuille= 장)) , **Node** (Feuille= 에서 , Feuille= 거기))) , **Node** (Feuille= 고 , **Node** (**Node** (**C*** , **Node** (Feuille= 을 , Feuille= 책)) , Feuille= 읽-)))) , Feuille= 있었죠)

```
├B
  ├B
    ├C*
      ├이
      └장
    ├에서
    └거기
  ├고
    ├C*
      ├을
      └책
    └읽-
└있었죠
```

Expression combinatoire: ((**B** (고 ((**C*** 을 책) 읽-)) (**B** (**C*** 이 장) (에서 거기))) 있었죠

Structure Applicative (Forme Normale) :

Type syntaxique : **S**

Arbre applicatif: **Node** (**Node** (**Node** (Feuille= 에서 , Feuille= 거기) , **Node** (**Node** (Feuille= 고 , **Node** (Feuille= 읽- , **Node** (Feuille= 을 , Feuille= 책))) , Feuille= 있었죠)) , **Node** (Feuille= 이 , Feuille= 장))

```
    ├에서
    └거기
      ├고
        ├읽-
          ├을
          └책
      └있었죠
  ├이
  └장
```

Représentation applicative: (((에서 거기) ((고 (읽- (을 책))) 있었죠)) (이 장))

Fig. 2. Result obtained by ACCG Parser

4 Conclusion and Future Work

As we have shown in this paper, an extended Categorial Grammar such as ACCG allows us to scope the difficult characteristics of the Korean language. In particular, we could consider the cases in Korean as operators which play an essential role in the Korean analysis and give the automatic morhpo-syntactical analysis with the ACCG parser. But the data was not enough to make a serious evaluation of this parser. So we plan to evaluate the system using more numerous texts and to find some possibilities of its application to other languages such as Czech.

Acknowledgements This work has been partly supported by the Ministry of Education of CR within the Center of basic research LC536 and by the Czech Science Foundation under the project P401/10/0792.

References

1. Biskri, I. and Desclés, J.-P. (2005). Applicative and Combinatory Categorial Grammar and Subordinate Constructions in French. *International Journal on Artificial Intelligence Tools*, Vol 14, No. 1& 2: 125-136.
2. Cha, J.W. , Lee, G. and Lee, J.H. (2002). Korean Combinatory Categorial Grammar and statistical parsing. Computers and the Humanities, 36(4):431-453.
3. Clark, S. and Curran, J. R. (2004). Parsing the WSJ using CCG and log-linear models. In *Proceedings of the 42nd Meeting of the ACL*, 104-111.
4. Curry, H.B. and Feys, R. (1958). *Combinatory Logic*. Volume 1. Amsterdam: North-Holland Publishing.
5. Desclés, J.-P. (2004). Combinatory Logic, Language and Cognitive Representations. In Weingartner P. (eds), *International Meeting on Alternative Logics*, Springer: 115-148.
6. Kang, J., Desclés, J.-P. (2011). ACCG : un système catégoriel d'analyse automatique de constructions syntaxiques. *ACFAS*. 9-13 mai.
7. Oehrle, R.T., Bach, E., Wheeler, D. (1988). *Categorial Grammars and Natural Languages Structures*. D. Reidel, Boston, MA.
8. Shaumyan, S. K. (1987). *A Semiotic Theory of Natural Languages*, Bloomington, Indiana Univ. Press.
9. Steedman, M. (2000). *The Syntactic Process*. MIT Press/Bradford Books, Cambridge, MA.

Extracting Phrases from PDT 2.0

Vašek Němčík

NLP Centre
Faculty of Informatics, Masaryk University
Brno, Czech Republic
xnemcik@fi.muni.cz

Abstract. The Prague Dependency Treebank (henceforth PDT) is a large collection of texts in Czech. It is renown for its respectable size and rich multi-layer annotation covering a wide range of complex phenomena. One the other hand, it can be argued that the complexity of the dataset may be a notable hindrance to using certain aspects of the data in a straightforward way. To overcome these problems, we present an export filter converting PDT into a more transparent data format, containing information about the most common phrase types. We believe that availability of the PDT data in this form will help encourage people unfamiliar with the underlying theory to use the corpus.

Key words: PDT, corpus, treebank, export, format, complex annotation, phrase, clause

1 Introduction

The Prague Dependency Treebank 2.0 (henceforth just PDT) is a large collection of Czech texts compiled at the Institute of Formal and Applied Linguistics at the Charles University in Prague. It was created within an open-ended project for manual annotation of substantial amount of Czech-language data with linguistically rich information ranging from morphology through syntax and semantics/pragmatics and beyond. [1]

PDT is a notable linguistic resource and is renown for its fair size and the underlying sophisticated linguistic theory, the FGD (Functional Generative Description), with a respectable tradition in the field of general linguistics. The corpus is organized in three layers, two of them tree-based, and covers a wide range of linguistic phenomena. The annotation principles are derived mainly from the concept of syntactic dependence.

The concept of syntactic dependence offers a straightforward theoretical base for building tree structures. However, in order to create fully connected syntactic trees for all sentences, it is necessary to spoil its theoretical purity by adding exceptions necessary to handle the irregularities and obscurities of everyday language. In such a situation, it is necessary to reach a certain trade-off between theoretical purity on one hand, and clarity, transparence and ease of use on the other.

In my opinion, the PDT is strongly inclined towards theoretical purity. The scope of the attributes and phenomena encompassed by the PDT is very broad, from part-of-speech and morphological tags to complex verb groups, coordinations, reconstructions of elliptical phrases, and semantic features not directly present, but inferrable from the data. In order to annotate the corpus correctly, the human annotators were provided with rather bulky annotation guidelines,[1] describing many important structural differences based on making minute distinctions. The resulting representation is theoretically sound, but for a person not thoroughly familiar with the underlying theory rather difficult to grasp. This is striking especially in sentences exhibiting a combination of several complex linguistic phenomena. Abundance of such sentences in the corpus lead us to create an export tool projecting the data to a more straightforward data format. We hope it helps computational linguists to use the data in practically oriented systems in a more efficient way.

Next section gives an overview of selected annotation principles that may cause confusion with new users of the corpus. Further, in Section 3, we present the output format of the proposed *pdt2vert* export tool and sketch on the conversion technicalities. Finally, we summarize the paper and comment on our future work.

2 Complex Annotation Drawbacks

This section gives an overview of selected features of the PDT data, which are rather irregular and may be a source of misunderstandings or confusion.

The core of the annotation, the analytical and tectogrammatical layer, is based on the concept of dependence. This concept has a long tradition, and the underlying relation can be straightforwardly defined and determined,[2] and exhibits theoretically appealing properties. To cover commonly occurring phenomena, the dependency concept was supplemented by the following constructions:

Coordination introduces a different type of edges into the PDT trees. These edges connect the coordination root (eg. a conjunction or a comma) to the coordinated elements. The dependency relation can be virtually reconstructed as "skipping" the coordination root – ie. the parent node of the coordination root having the coordination members as its dependent nodes. Coordinations are often recursive, which makes this reconstruction of node dependencies nontrivial.

Ellipsis concerns words that have been omitted from the sentence. It is possible to think of them as of nodes "skipped" when creating a branch of

[1] The annotation manual for analytical layer [2] has 301 pages, for tectogrammatical layer [3] 1215 pages.

[2] The syntactic dependence between elements A and B, the element A being dependent on the element B, specifies that B can occur without A, but A cannot occur without B. [4]

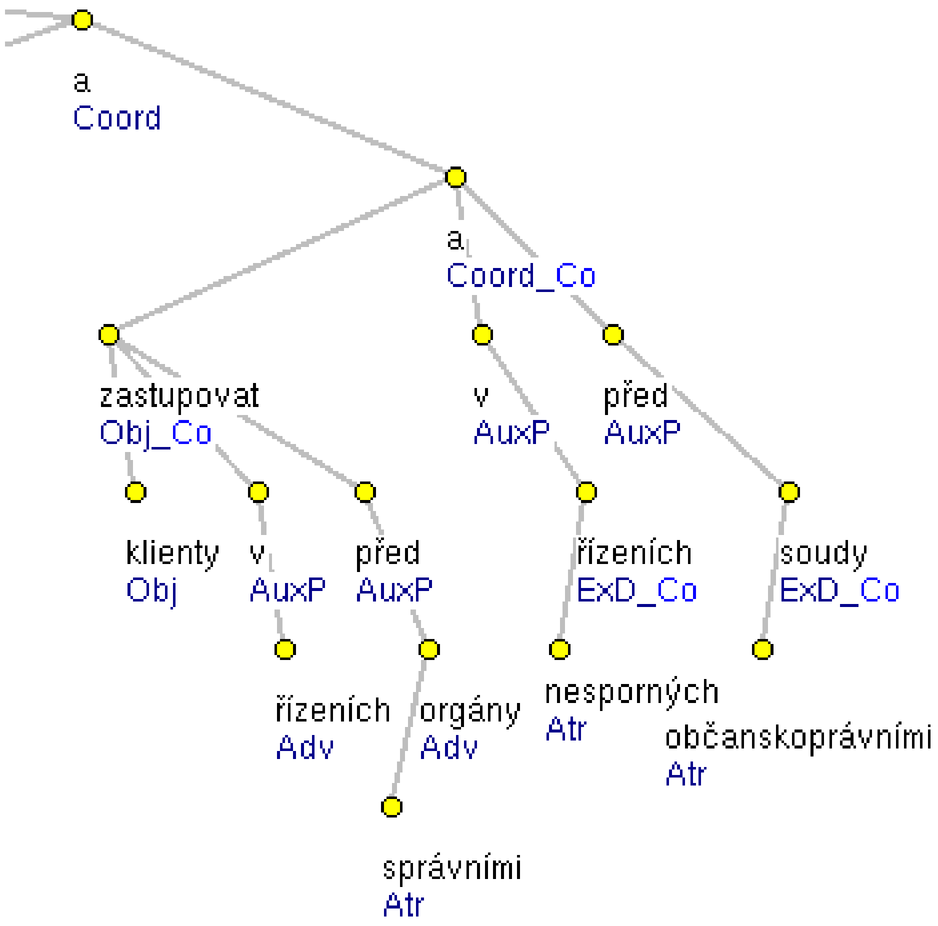


Fig. 1. A fragment of an analytical tree.

the dependency tree. Their descendants are placed below its parent with the analytical functor `ExD` (which unfortunately veils their real functors).

Shared attributes of coordinated nodes are represented by nodes directly dependent on the coordination root, but not marked as coordination members. In my opinion, this convention has the most far-reaching consequences. Most notably, phrases are not necessarily subtrees of the sentence tree, and after reconstructing the dependency relation from this notation, it does not form a tree (the root of such a shared subtree has several parents). When working with the data, this can come rather unexpected and counter-intuitive.

Conjunctions and prepositions are represented by nodes with analytical functors `AuxC` and AuxP respectively. These words are syntactically governors, but their function is mostly auxiliary. As a result, they are ignored (skipped) in many annotation rules.

The tectogrammatical layer is very complex and its annotation involves making many decisions. These are often based on rather fine distinctions and have various consequences – not only different tectogrammatical functors (for example `&Gen` vs. `&Unsp`), but also structural differences (such as ellipsis vs. shared attributes). In my opinion, with regard to the abundance of structural exceptions and irregularities, and the number of node attributes and values, this annotation level may be insightful for a human linguist, but is rather unsuitable for automatic processing.

Especially combinations of these phenomena may lead to a certain confusion based on false assumptions about the data structure. This can be illustrated by the example in Figure 1. It contains a part of an analytical tree, containing a multiple coordination (coordinating an infinite verb form and two prepositional phrases), and ellipsis, where the nodes signalling the coordination membership and ellipsis (with its `ExD_Co` functor) are below preposition nodes.

In order to make the PDT data more transparent, and minimize the interference of annotation of different phenomena, we have proposed an alternative export data format. We hope it will encourage computational linguists who would otherwise be discouraged by the complexity of the data, to use it in their applications.

The proposed export format is described in the following section.

3 The PDT2vert format

The *pdt2vert* is proposed to offer an alternative, more straightforward way of presenting the PDT data. We also believe this format is more convenient for automatic processing within typical NLP applications.

The output structure is linear, and is based on the so-called vertical format. The main principle of the vertical format is that each line contains either information about one token, or a structural tag. A token line typically contains the word surface form, lemma, morphological tag, etc. Structure tags are used to express the global data structure, mainly boundaries of sentences, paragraphs, and documents.

Table 1. Example of a sentence in the *pdt2vert* format.

```
<sentence id="ln94202-55-p5s2">
<clause id="t-ln94202-55-p5s2w3">
<markable id="ln94202-55-p5s2w1" type="np" mtag="NNFS7---A--">
Podmínkou podmínka              NNFS7---A--   Pnom
</markable>
však      však                  J^-------     AuxY
je        být                   VB-S--3P-AA-- Pred
</clause>
,         ,                     Z:-------     AuxX
<clause id="t-ln94202-55-p5s2w9">
aby       aby                   J,-------     AuxC
<markable id="ln94202-55-p5s2w8" type="np" mtag="NNFP1---A--">
tyto      tento                 PDFP1-----    Atr
činnosti  činnost_^(*3ý)        NNFP1---A--   Sb
</markable>
s         s-1                   RR-7-----     AuxP
<markable id="ln94202-55-p5s2w12" type="np" mtag="NNIS7---A--">
právním   právní                AAIS7--1A--   Atr
úkonem    úkon                  NNIS7---A--   Obj
obsaženým obsažený_^(*5áhnout) AAIS7--1A--   Atr
v         v-1                   RR-6-----     AuxP
<markable id="ln94202-55-p5s2w16" type="np" mtag="NNIS6---A--">
notářském notářský              AAIS6--1A--   Atr
zápise    zápis                 NNIS6---A--   Adv
</markable>
</markable>
nebo      nebo                  J^-------     Coord
s         s-1                   RR-7-----     AuxP
<markable id="ln94202-55-p5s2w19" type="np" mtag="NNFS7---A--">
přípravou příprava              NNFS7---A--   Obj
<markable id="ln94202-55-p5s2w21" type="np" mtag="NNIS2---A--">
tohoto    tento                 PDZS2-----    Atr
úkonu     úkon                  NNIS2---A--   Atr
</markable>
</markable>
</clause>
.         .                     Z:-------     AuxK
</sentence>
```

As demonstrated by Table 1, the *pdt2vert* format extends the scope of structures accounted for. It covers most notably noun phrases and clauses, optionally, singleton tags are inserted to represent zero subjects. The token attributes can contain the analytical functor (last column in Table 1), the functor of the corresponding tectogrammatical node (when applicable), the token identifier, or the identifier of the parent node. The choice and order of attributes can be customized.

The conversion of the data is not straightforward, the most difficult part being clause boundary detection, necessary for instance for detecting zero subjects and pruning noun phrases. The convertor is written in Perl, using the BTrEd interface supplied with the corpus.

4 Conclusions and Further Work

We have presented an export filter for converting PDT into a more transparent data format, which is perhaps more appealing to users not interested in the advanced features of the corpus.

As a next step, the PDT data in this format will be used for various evaluation tasks. As mentioned in [5] and [6], PDT in its original form has a certain bias when used to evaluate parsers. The PDT dependency trees are too detailed (eg. accounting for punctuation and non-word tokens) and a more compact representation would be needed to serve as a plausible evaluation standard. Further, this format will be used to evaluate algorithms for anaphora resolution and topic-focus articulation.

The generality of the output format makes it possible to create files that can be conveniently used to train or evaluate a wide range of linguistic models.

Acknowledgments This work has been partly supported by the Ministry of Education of CR within the Center of basic research LC536 by the Czech Science Foundation under the project 407/07/0679.

References

1. Hajič, J., et al.: The Prague Dependency Treebank 2.0. Developed at the Institute of Formal and Applied Linguistics, Charles University in Prague. (2005) `http://ufal.mff.cuni.cz/pdt2.0/`.
2. Hajič, J., Panevová, J., Buráňová, E., Urešová, Z., Bémová, A.: Anotace Pražského závislostního korpusu na analytické rovině: pokyny pro anotátory. Technical Report 28, ÚFAL MFF UK, Prague, Czech Republic (1999)
3. Mikulová, M., Bémová, A., Hajič, J., Hajičová, E., Havelka, J., Kolářová-Řezníčková, V., Kučová, L., Lopatková, M., Pajas, P., Panevová, J., Razímová, M., Sgall, P., Štěpánek, J., Urešová, Z., Veselá, K., Žabokrtský, Z.: Anotace Pražského závislostního korpusu na tektogramatické rovině: pokyny pro anotátory. Technical report, ÚFAL MFF UK, Prague, Czech Republic (2005)

4. Bußmann, H.: Lexikon der Sprachwissenschaft. Alfred Kröner Verlag, Stuttgart (2002)
5. Horák, A., Holan, T., Kadlec, V., Kovář, V.: Dependency and phrasal parsers of the czech language: A comparison. In: Proceedings of 10th International Conference on Text, Speech, and Dialogue (TSD 2007), Berlin, Heidelberg, Springer (2007) 76–84
6. Kovář, V., Jakubíček, M.: Prague dependency treebank annotation errors: A preliminary analysis. In: RASLAN 2009 : Recent Advances in Slavonic Natural Language Processing, Brno, Masaryk University (2009) 101–108

Extended VerbaLex Editor Interface Based on the DEB Platform

Dana Hlaváčková, Adam Rambousek

NLP Center
Faculty of Informatics, Masaryk University,
Brno, Czech Republic
{hlavack,xrambous}@fi.muni.cz

Abstract. The paper presents new editing interface for lexicon of verb valencies (VerbaLex). Previous method of editing text files in custom setup of VIM text editor will be replaced by web application based on DEB platform.
New interface will contain more strict control of the valencies format and structure. It should also be easier to learn for new editors. The data will be saved in XML format, thus allowing conversion to many output formats.

Key words: VerbaLex, verb valencies, DEB, dictionary writing systems

1 Introduction

The beginnings of building the verb valency frame dictionary at the Faculty of Informatics at Masaryk University (FI MU) dates back to 1997 [1]. Since then, the dictionary, denoted as Brief, has undergone a long development and has been used in various tools from semantic classification to syntactic analysis of Czech sentence [2].

VerbaLex is a large lexical database of Czech verb valency frames and has been under development at The Centre of Natural Language Processing at the Faculty of Informatics Masaryk University (FI MU) since 2005. The organization of lexical data in VerbaLex is derived from the WordNet structure. It has a form of synsets arranged in the hierarchy of word meanings (hyper-hyponymic relations). For this reason, the headwords in VerbaLex are formed by lemmata in synonymic relations followed by their sense numbers (standard Princeton WordNet notation).

Verbalex is based on three independent resources – electronic dictionaries of verb valency frames:

– BRIEF — a dictionary of 50,000 valency frames for 15,000 Czech verbs, which originated at FI MU in 1997 [11]
– VALLEX – a valency lexicon of Czech verbs based on the formalism of the Functional Generative Description (FGD) developed during the Prague Dependency Treebank (PDT) project [13]

SYNSET: BAVIT:1, ROZPTÝLIT:2, ROZPTYLOVAT:2
DEFINITION: poskytovat někomu zábavu/make (somebody) laugh

- *passive: yes*
- *meaning: I*
- *class: amuse-31.1-1*
- *impf: bavit:1 pf: rozptýlit:2 impf: rozptylovat:2*

frame: AG <person:1>$^{\text{obl}}_{\text{whoNom}}$ VERB PAT <person:1>$^{\text{obl}}_{\text{whatAccus}}$
ACT <act:2>$^{\text{opt}}_{\text{by doing whatInstr}}$

- *example: impf: bavil děti hrou (he amused the children by playing the game)*
- *attr: use: prim, reflexivity: obj_ak*

Fig. 1. An Example of a Verbalex Valency Frame

- Czech WordNet valency frames dictionary created during the Balkanet project [12] and containing 1,359 valency frames (incl. semantic roles) associated with 824 sets of synonyms (synsets)

The structure of valency information can be seen in Figure 1 and is briefly described below.

The current version of VerbaLex contains 6,360 synsets, 21,193 verb senses, 10,482 verb lemmata and 19,556 valency frames. The valency database is developed in TXT format and available in XML, PDF and HTML formats.

2 The DEB platform for dictionary writing systems

VerbaLex editor is built on the DEB development platform, which allows the system to use many components common to dictionary writing systems. DEB (Dictionary Editor and Browser, `http://deb.fi.muni.cz/`) is an open-source software platform for the development of applications for viewing, creating, editing and authoring of electronic and printed dictionaries. The DEB platform follows the client-server architecture. Most of the functionality is provided by the server side and client side offers graphical interfaces for users. The client applications communicate with the server using the standard web HTTP protocol.

The server part is built from small, reusable parts, called servlets, which allow a modular composition of all services. Each servlet provides different functionality such as database access, dictionary search, morphological analysis or a connection to corpora.

The overall design of the DEB platform focusses on modularity. The data stored in a DEB server can use any kind of structural database and combine the results in answers to user queries without the need to use specific query languages for each data source. The main data storage is currently provided by the Oracle Berkeley DB XML [5], which is an open-source native XML database providing XPath and XQuery access into a set of document containers. However, it is possible to switch to another database backend easily.

We have experienced issues with the database performance, so we compared several XML databases in series of benchmarks. Database systems working with XML data (both native XML databases and XML enabled relational databases) are already widespread and used in many areas. Their performance was benchmarked by many projects using several benchmarks. However, conclusions of previous publications [6,7,8] do not provide one definitive answer as for the choice of the best XML database. Generally, the results suggest that different XML benchmarks can show different weak and strong points of each database systems. When comparing the two classes of XML databases, i.e. relational databases with XML support and native XML databases, we can see that XML enabled relational databases process data manipulation queries more efficiently, and native XML databases are faster in navigational queries which rely on the document structure.

Because of the special focus on dictionary writing systems, we ran different test suites designated to both "raw speed" of the database and to specific requirements of knowledge and ontology systems. According to the results of the tests (see [9] for the details of the tests results), none of the available native XML databases can supersede the others for all kinds of operations needed for knowledge and ontology storage and manipulation. Berkeley DB XML cannot efficiently solve the queries involving multiple nodes and full-text queries. The eXist database contains the Lucene module for text search and supports many XML standards, so it can be recommended for deployment where these features are more important than the database performance. On the other hand the MonetDB database can be, according to its specific architecture, conveniently used for when working with very large amounts of XML data. For middle-size data collections, the Sedna database can provide the same performance as MonetDB, while offering richer set of features. The potential drawbacks of Sedna are the need to use special queries for the defined data indexes and the use of commercial tool for optimized full-text queries. However, the full-text queries without this optimization are already comparably fast.

During the testing of both database engines within the DEB platform, we found out that the MonetDB programming interface for the Ruby language used in the DEB platform is not stable enough and not developed actively at the moment. Because of that, MonetDB is not ready yet to be included in the platform. Fortunately, Ruby interface for Sedna is stable and maintained and better suited for DEB platform. That is why Sedna was chosen for the DEB database backend transition. It is now used for all new project and existing projects will be transfered later.

The user interface, that forms the most important part of each dictionary application, usually consists of a set of flexible forms that dynamically cooperate with the server. Most of the DEB client applications are developed using the Mozilla Development Platform [10]. The Firefox web browser is one of the many applications created using this platform. The Mozilla Cross Platform Engine provides a clear separation between application logic and definition, presentation and language-specific texts. Furthermore, it imposes nearly no limits on the computer operating system of the users when accessing the dictionary data – the DEB applications run on MS Windows, Linux or Mac OS.

Thanks to the enhanced features of new HTML standards and their support in modern web browsers, many of the dictionary writing systems can be implemented as web applications.

The main assets of the DEB development platform can be characterized by the following points:

– All the data are stored on the server and a considerable part of the functionality is also implemented on the server, while the client application can be very lightweight.
– Very good tools for team cooperation; data modifications are immediately seen by all the users. The server also provides authentication and authorization tools.
– Server may offer different interfaces using the same data structure. These interfaces can be reused by many client applications.
– Homogeneity of the data structure and presentation. If an administrator commits a change in the data presentation, this change will automatically appear in every instance of the client software.
– Integration with external applications.

3 Current DEB applications

The DEB development platform provides a basis for many different kinds of lexicographic applications. The list of real dictionary systems that was developed on the DEB platform currently contains the following applications:

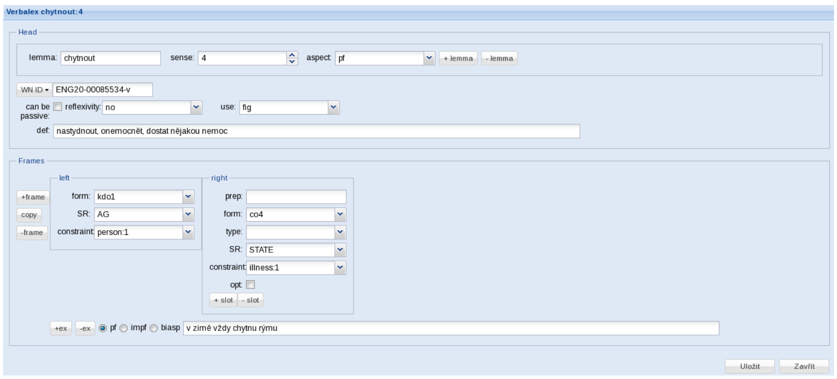
– DEBDict, a general multiple-dictionary browser
– DEBVisDic, wordnet editor and browser
– TeDi, multilingual terminological dictionary of art terms
– TeA, multilingual terminological dictionary of agriculture terms
– Cornetto, editor and browser of Dutch lexical-semantic database
– Global Wordnet Grid, publicly accessible multilingual wordnet dictionary
– PRALED, complex application for building new Czech lexical database
– KYOTO, backend for wordnet and ontology storage in EU-FP7 project
– PDEV (CPA), Pattern Dictionary of English Verbs, tightly connected with corpora
– Family Names in UK, web editor for Comprehensive Dictionary of English Surnames

The first two applications are widely used with hundreds of users all over the world and with participation in various national and multilingual research projects. In the following paragraphs, we will provide more details about DEB-Dict and DEBVisDic as well as PDEV and the Dictionary of English Surnames, which are the most interesting (besides PRALED) from the lexicographic point of view. The whole next section will then be devoted to PRALED.

4 New VerbaLex interface

New editing interface was designed to be easily understandable by new editors and also to lower the possibility of editing error (for example, typo in the value from the list). Because of that most of the valency elements allow only the selection from predefined set of values. Usually, editor writes just the verb sense definition and frame examples.

Main difference from the old data format is removal of optional elements, for example it was possible to define combination of several semantic roles for one frame. This feature caused problems in automatic processing and conversions of the lexicon data. In the new format, each frame can contain only one semantic role and frames are duplicated with role changes, if needed. New web editor can be seen in figures 2 and 3.


Verbalex chytnout:4

Head

lemma: chytnout sense: 4 aspect: pf

WN ID: ENG20-00085534-v

can be passive: ☐ reflexivity: no use: fig

def: nastydnout, onemocnět, dostat nějakou nemoc

Frames

left
form: kdo1
SR: AG
constraint: person:1

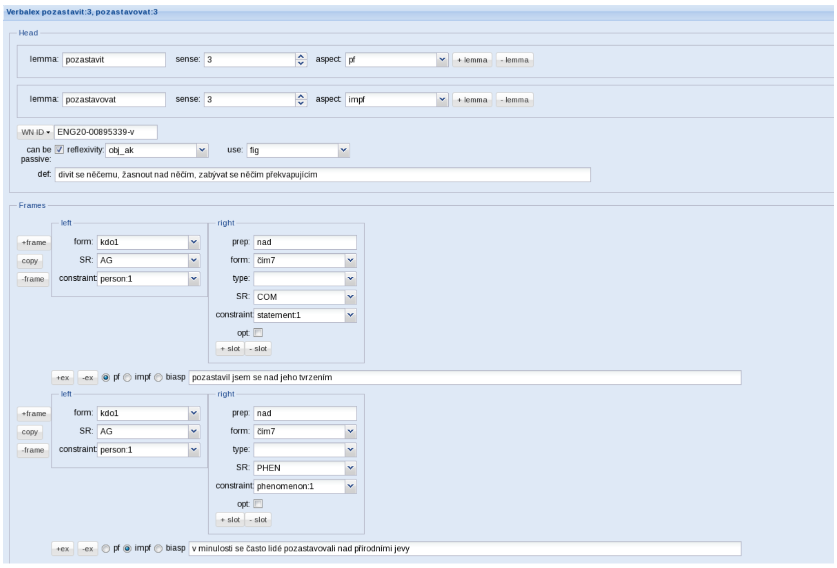
right
prep:
form: co4
type:
SR: STATE
constraint: illness:1
opt: ☐

[✓ pf] impf biasp v zimě vždy chytnu rýmu

Fig. 2. Example of new VerbaLex editor interface.

5 Conclusions

New editing interface is already used to add new verbs and missing verb senses to VerbaLex lexicon. Over a hundred of verb senses was edited, the interface and XML format for the data was tweaked during the testing phase. Existing data will be converted to the new XML format and checked during the process.



Verbalex pozastavit:3, pozastavovat:3

Head

lemma: pozastavit sense: 3 aspect: pf

lemma: pozastavovat sense: 3 aspect: impf

WN ID: ENG20-00895339-v

can be passive: ✓ reflexivity: obj_ak use: fig

def: divit se něčemu, žasnout nad něčím, zabývat se něčím překvapujícím

Frames

left — form: kdo1 SR: AG constraint: person:1

right — prep: nad form: čím7 type: SR: COM constraint: statement:1 opt: ☐

[✓ pf] impf biasp: pozastavil jsem se nad jeho tvrzením

left — form: kdo1 SR: AG constraint: person:1

right — prep: nad form: čím7 type: SR: PHEN constraint: phenomenon:1 opt: ☐

pf [✓ impf] biasp: v minulosti se často lidé pozastavovali nad přírodními jevy

Fig. 3. Example of new VerbaLex editor interface.

Acknowledgements This work has been partly supported by the Ministry of Education of CR within the Center of basic research LC536 and LINDAT-Clarin project LM2010013 and by the Czech Science Foundation under the project P401/10/0792.

References

1. Pala, K., Sevecek, P.: Valence Českých sloves (Valencies of Czech Verbs). In: Proceedings of Works of Philosophical Faculty at the University of Brno, Brno, Masaryk University (1997) 41–54
2. Smrz, P., Horak, A.: Determining type of TIL construction with verb valency analyser. In: Proceedings of SOFSEM'98, Berlin, Springer-Verlag (1998) 429–436
3. : Balkanet project website, `http://www.ceid.upatras.gr/Balkanet/`. (2002)
4. Stranakova-Lopatkova, M., Zabokrtsky, Z.: Valency dictionary of czech verbs: Complex tectogrammatical annotation. In M. Gonzalez Rodriguez, C.P.S.A., ed.: LREC2002, Proceedings. Volume III., ELRA (2002) 949–956
5. Chaudhri, A.B., Rashid, A., Zicari, R., eds.: XML Data Management: Native XML and XML-Enabled Database Systems. Addison Wesley Professional (2003)
6. Böhme, T., Rahm, E.: Multi-user evaluation of XML data management systems with XMach-1. Efficiency and Effectiveness of XML Tools and Techniques and Data Integration over the Web (2008) 148–159
7. Nambiar, U., Lacroix, Z., Bressan, S., Lee, M., Li, Y.: Efficient XML data management: an analysis. E-Commerce and Web Technologies (2002) 261–266
8. Lu, H., Yu, J., Wang, G., Zheng, S., Jiang, H., Yu, G., Zhou, A.: What makes the differences: benchmarking XML database implementations. ACM Transactions on Internet Technology (TOIT) **5**(1) (2005) 154–194

9. Bukatovič, M., Horák, A., Rambousek, A.: Which XML storage for knowledge and ontology systems? In: Knowledge-Based and Intelligent Information and Engineering Systems, Springer (2010) 432–441
10. Feldt, K.: Programming Firefox: Building Rich Internet Applications with XUL. O'Reilly (2007)
11. Pala, K., Ševeček, P.: Valence českých sloves, In: Proceedings of Works of Philosophical Faculty at the University of Brno, MU, Brno (1997) 41–54.
12. BalkaNet project website (2001–2004) http://www.ceid.upatras.gr/Balkanet.
13. Žabokrtský, Z.: Valency Lexicon of Czech Verbs. Ph.D. thesis, MFF UK, Prague (2005)

Part III

Text Corpora

Building Corpora of Technical Texts
Approaches and Tools

Petr Sojka, Martin Líška, and Michal Růžička

Faculty of Informatics, Masaryk University
Botanická 68a, 602 00 Brno, Czech Republic
`sojka@fi.muni.cz`, `martin.liski@mail.muni.cz`, `mruzicka@mail.muni.cz`

Abstract. Building corpora of technical texts in Science, Technology, Engineering, and Mathematics (STEM) domain has its specific needs, especially the handling of mathematical formulae. In particular, there is no widely accepted format to represent and handle math.
We present an approach based on multiple representations of mathematical formulae that has been used for math retrieval, similarity and clustering of mathematical corpus. We provide an overview of our toolset, summarize our experiments to date and propose further research directions and approaches.

Key words: mathematical corpora; information retrieval of mathematics; representation of mathematical formulae; math search and indexing normalization of MathML; canonicalization

1 Introduction

Leading research in empirical linguistics builds on the large (e.g. web-scale) corpora such as those created by Google (Google Books Corpus, Google Scholar) or by the Sketch Engine (TenTen Corpora). Such corpora allow for natural language processing (NLP) of a new quality level to solve such tasks as more relevant information retrieval, document clustering, classification and similarity, thesauri and ontology building, better word sense disambiguation, machine translation and many others. However, in these research mainstream activities, minority languages or domain specifics are neglected. Such a neglected 'language' is the language of mathematics – typical in Science, Technology, Engineering, and Mathematics (STEM) documents.

Mainstream NLP workflow for building corpora starts with tokenization, which is usually not aware of mathematical formulae or equations. Math is usually supported neither by optical character recognition (OCR) tools, nor by applications that generate PDF or (X)HTML. The use and representation of math on the web is far from settled. As a consequence, no mainstream tools support this niche market of 'the Queen of sciences'.

In previous projects that involved building Digital Mathematics Libraries (DML) such as DML-CZ [1] and EuDML [2], we had to deal with the fact that NLP corpora tools were unable to handle corpora of math texts, let alone build them. We therefore devised some tools for adequate support of mathematical formulae in NLP and information retrieval (IR) tasks. Proper semantic and math-aware representation is a necessary prerequisite for efficient and effective NLP processing of STEM corpora.

This task involved as the first step the design of math formulae representation (Section 2). Then, to build mathematical corpora, we had to preprocess and normalize heterogenious inputs (Section 3) into this new representation. It was also necessary to design ways of math retrieval (index and search are crucial, cf. Section 4). Our aim is to support math-aware document clustering, similarity and disambiguation (Section 5). We summarize our findings in the Section 6.

2 Math Representations

Mathematicians and other authors of STEM documents encode quantities and relations using formulae and equations in compact, often two-dimensional, notation. These objects have to be represented in unique way in the global STEM document handling system.

There are numerous ways of notating the same mathematical object, that has evolved in some geographical location or language. This is an example of different notations for a *binomial coefficient*:

$$\binom{n}{r} = \frac{n!}{r!(n-r)!} = {}_nC_r = {}^nC_r = C(n,r)$$

When searching STEM documents, it should be possible to find the same objects within a corpus, and assign them the same representation even though authors have used different notation. As is the case with text handling, where words with the same meaning are treated and indexed the same, formulae require the same treatment.

The matter is complicated by the fact that there are different formats for handling mathematics: TEX, MathML, OpenMath, etc.

2.1 TEX

Authors prefer the compact and logical notation of **TEX**. The American Mathematical Society (AMS) extended standard plain TEX and **LATEX** notation with AMS packages (so called **AMSLATEX**) for commutative diagrams, aligned equations, etc. The namespace of AMSLATEX macros is nowadays the de facto standard for the typesetting of mathematical documents, and this namespace is also supported in ithe metadata of DML-CZ (only this namespace is allowed and supported there, e.g. by conversion to MathML).

TEX math notation is in such demand that even for **Word** there is plug-in by Design Science that allows entering the formulae in TEX notation. This is much quicker, and more convenient than choosing the symbols from numerous menus and symbol tables. Nevertheless, using TEX notation for indexing purposes is a disaster, as an example of **LaTeXSearch** application by Springer (`http://latexsearch.com`) shows. Authors are so creative in macroexpansion use or TEX language formatting that different notations cannot be coped with by simple string similarity. A formulae structure and other types of similarity has to be used in formulae representation for similarity computation. For this, the tree structure of XML (MathML) is better, as it is understood by the majority of math-aware software developers.

2.2 MathML

In the world of applications and software interfaces, MathML usually wins, as it is supported by W3C and AMS. TEX's macro namespace extensibility is a nightmare to support by software without the full TEX macroexpansion complex engine, and here MathML clearly wins. MathML DTD allows easy formulae validation and processing with XML tools. There are even recently developed portable tools like **MathJax**, a JavaScript library that displays mathematics in web browsers, supporting both LATEX and MathML markup as it attempts to convert LATEX on-the-fly into appropriate markup language–HTML or MathML.

2.3 Set of M-terms

A mathematical document contains mathematical formulae, which are integral to the content of the document. As mentioned in the previous sections, these formulae are usually represented in TEX if authored by humans, or in MathML (presentation or mixed content-presentation) if produced by machines.

To be able to search for such structural information using a fulltext indexing approach as in the Math Indexer and Searcher (**MIaS**) system [3,4], a convenient representation needs to be selected. This representation needs to be a trade-off between the TEX powered authors' part of the world and machine-friendly, preserve-as-most-information-as-possible, structural and semantic notation such as Content MathML. In MIaS system we opted for Presentation MathML as it stands, we find, exactly halfway. It is relatively easy to obtain by converting the author's TEX markup and it still holds the necessary structural information for machine processing. It is still easily extensible by Content MathML trees capturing the formulae semantics.

Such mathematical markup needs to be preprocessed before the indexing. This is mainly to accommodate the best user search experience as possible. For each formula in the text, the system produces several representations which are stored in the index and are searchable in the same way as regular textual terms. These are called *M-terms*.

M-terms are translated from XML to a linear string form. In this form they are stored by the indexing core. This representation omits any XML markup that would be redundant in such a form, such as start and end tags, and replaces it with brackets to prevent ambiguity. Also most of the attributes setting the visual behaviour of the expression can be left out, since it does not hold any information pertaining to the meaning of the formula. This representation can be further compacted by substituting tag names for single characters to decrease storage space requirements.

For example, simple expression $a^2 + b$ in its XML form

```
<math>
  <mrow>
    <msup><mi>a</mi><mn>2</mn></msup>
    <mo>+</mo>
    <mi>b</mi>
  </mrow>
</math>
```

is translated to the linear form `mrow(msup(mi(a)mn(2))mo(+)mi(b))` and based on a custom tag name dictionary, where `mrow = R`; `msup = J`; `mi = I`; `mn = N` and `mo = O`. This is further compacted to `R(J(I(a)N(2))O(+)I(b))`. A set of sub-M-terms is generated for each input formula. It consists of subformula-weight pairs. For this particular expression, it is:

```
{
(mi(a),0.08166666),
(mn(2),0.08166666),
(msup(mi(a)mn(2)),0.11666667),
(mo(+),0.11666667),
(mi(b),0.11666667),
(mrow(mi(b)mo(+)msup(mi(a)mn(2))),0.16666667),
(msup(mi(1)mn(2)),0.093333334),
(mrow(mi(1)mo(+)msup(mi(2)mn(2))),0.13333334),
(msup(mi(a)mn(¶)),0.058333334),
(mrow(mi(b)mo(+)msup(mi(a)mn(¶))),0.083333336),
(msup(mi(1)mn(¶)),0.046666667),
(mrow(mi(1)mo(+)msup(mi(2)mn(¶))),0.06666667)
}
```

These formulae are derived from the original one and their level of similarity is expressed by the weight factor.

This representation not only grabs the structural similarity of mathematical formulae, it also copes with different variable names, and with mathematical properties of operators (commutativity). As such, representation of formulae by an M-term set with weights is directly useable for indexing or for document similarity computations.

To provide these and other uses of this representation, we have set up a RESTful web service, where for each input formula one can get a set of M-terms as they would be indexed in the MIaS system. An example of use can be found here:

```
http://aura.fi.muni.cz:8085/mias4gensim/mathprocess?mterm=<math><mrow>
<mi>a</mi><mo>+</mo><mi>b</mi></mrow></math>
```

3 Mathematical Corpora

3.1 Normalization

When building mathematical corpora using MathML as a language for mathematical formulae preservation, it emerges that it is very useful to process and normalize MathML that is being stored. It is necessary as one mathematical formula can be encoded in MathML in different forms – using different sequences of characters in the source code – but its meaning is the same.

For example, the formula $x^2 + y^2$ can be encoded in MathML in the form:

```
<math xmlns="http://www.w3.org/1998/Math/MathML">
    <msup>
        <mi>x</mi>
        <mn>2</mn>
    </msup>
    <mo>+</mo>
    <msup>
        <mi>y</mi>
        <mn>2</mn>
    </msup>
</math>
```

But some other author can use this form:

```
<math xmlns="http://www.w3.org/1998/Math/MathML">
  <mrow>
    <msup>
        <mi>x</mi><mn>2</mn></msup>
    <mo>+</mo>
    <msup>
        <mi>y</mi><mn>2</mn></msup>
  </mrow>
</math>
```

To be able to find documents that contains our formula in any of these codings we need one normalized form that will be stored in the index. Subsequently, any query for this formula in any coding has to be transformed to the normalized form at the beginning.

Moreover, examples of documents from the real world (PubMed Central digital library workflow) show that validation of MathML source codes is not enough. Elbow et al. [5] demonstrate a well-known fact that current authors' main target is print output – consequently one can find MathML fragment `<mml:mn>7</mml:mn><mml:mn>5</mml:mn>` as source code of the number '75' for example. These anomalies have to be sorted out before publishing and indexing in a repository.

For the semantically same formalae there exist infinitely many ways of representing them in MathML. For NLP handling it would be convenient to have one *canonical* representation of a formulae.

3.2 Canonicalization

Proper MathML normalization (canonicalization) is not easy given that MathML is a very complex markup language. Some existing tools we have tested fail when run over a set of MathML test documents [6] that were designed to cover a wide range of MathML features.

Our approach to MathML normalization has so far involved a trial use of **UMCL** (Universal Maths Conversion Library; `http://inova.ufr-info-p6.jussieu.fr/maths/umcl`). [7,8] The main purpose of the UMCL tool set is to enable transcription of the MathML formulae to Braille national codes. Related to this task is also the need for MathML formulae unification. UMCL transformation of the MathML to Canonical MathML is carried out using a set of XSL stylesheets [9].

With minor modifications, the UMCL MathML transformation was used in the WebMIaS interface [10] (see Section 4) that can be used to search over our MREC corpus (see Section 3.3). This showed benefits of formulae normalization in practice – search form $x^2 + y^2$ formula using the first form of MathML code from the previous section found no results. However, for the second form of MathML – the form that is the result of UMCL XSL transformation from the first form – there were 36,817 hits in MREC corpus version 2011.4.

Unfortunately, the MathML canonicalization module of the UMCL tool set is not as powerful as we thought at the beginning. Using the W3C MathML Test Suite mentioned in the previous section, some weak points in UMCL normalization process have been identified. Among other things, there are problems with MathML tags like 'mphantom', 'mfenced', 'mglyphe', 'mmultiscripts', 'mover' and 'mstyle' that are not properly converted. Furthermore, attributes of MathML elements are not reported in the UMCL canonicalized MathML.

These problems were consulted with UMCL developers but no fast and clear solution seems to be available. Due to these problems, UMCL in the current version does not seem to be directly applicable to MREC corpus and further research in this area is definitely necessary.

3.3 Corpus MREC

To provide a test platform for mathematical search tools, we are building a corpus of mathematical texts. We call this corpus MREC.

MREC is based on arXMLiv [11] – a project of Michael Kohlhase's group at Jacobs University Bremen. arXMLiv documents came from `arXiv.org` but have been translated to XML by arXMLiv project. These documents cover different STEM areas – Physics, Mathematics, Computer Science, Quantitative Biology, Quantitative Finance and Statistics.

However, MREC is not an exact copy of the arXMLiv content. MREC contains just a subset of the arXMLiv – arXMLiv puts transformed documents into several classes – successful, complete with errors and incomplete, depending on the results of the transformations. MREC contains papers from conversion classes, successful and complete with errors (missing macros) – see Table 1. We have collected 439,423 documents in well-formed XHTML, containing mathematical formulae in valid MathML.

Table 1. Documents collected from arXMLiv

arXMLiv transformation result class	Quantity
successful (no problem)	65,874
successful (warning)	291,879
complete with errors (missing macros)	81,670
All documents	439,423

Moreover, there were several modifications of the files that from our point of view were necessary in order to make the documents well-formed and valid. These modifications include removing unnecessary attributes, namespace proxies, 'div' elements nested in 'span' elements and so on.

MREC consists of well-formed XHTML documents. MathML is used for representation of mathematical formulae.

Although MREC is under constant development, it is necessary for both archive and comparison purposes to produce a stable release versions. For this reason, there are several version of MREC corpora available at `http://nlp.fi.muni.cz/projekty/eudml/MREC/`.

The first public version of MREC, version 2011.3.324, consists of 324,060 documents. The resulting corpus size was 53 GB uncompressed, 6.7 GB compressed. Documents contained 112,055,559 formulae in total, of which 2,129,261,646 mathematical expressions were indexed. The resulting index size was approximately 45 GB.

The newer version of MREC, version 2011.4.439, consists of 439,423 scientific documents containing 158,106,118 mathematical formulae. 2,910,314,146 expressions were indexed and the resulting size of the index is 63 GB. The sizes of uncompressed and compressed corpora are 124 GB and 15 GB, respectively.

4 Math Retrieval

Searching functionality is nowadays a key form of getting orientated in the vast amount of information "out there" and obtaining the information we seek. Just as websites providing special content such as images and videos enable searching for these tokens, portals providing mathematical content such as EuDML [12] should also be able to search for the formulae.

In our view, the optimal way of doing so is to provide a simple Google-like interface where one can pose mathematical and textual query tokens one alongside the other. Search results returned to a textual query can then be finely constrained by adding a formula to the query and, in fact, vice-versa. We present this approach in the WebMIaS interface [10].

For example, by posting a simple query $x^2 + y^2$ in our web interface, the system returns 36,817 results. Addition of one more keyword *Euclid* reduces the number of results to only 97 – all of them contain this textual term. Conversely, searching only for *Euclid* returns 848 results and by adding $x^2 + y^2$ expression, we get the same 97 matches (MREC 2011.4.439).

To implement math-aware IR system in addition to the web-interface it was necessary to create an index to be consulted during query evaluation. We use our M-term representation for this, as described in detail in [3,4]. We have evaluated the system's speed. As is shown in Table 2, the performance of the MIaS system scales linearly. This gives feasible response times even for our billions of indexed subformulae.

Table 2. Indexing scalability test results (run on 448 GiB RAM, eight 8-core 64bit processors Intel Xeon$^{\text{TM}}$ X7560 2.26 GHz driven machine).

# Docs	Input formulae	Indexed formulae	run-time [ms]	CPU time [ms]
10,000	3,406,068	64,008,762	2,145,063	2,102,770
50,000	18,037,842	333,716,261	11,382,709	10,871,500
100,000	36,328,126	670,335,243	23,066,679	21,992,100
200,000	72,030,095	1,326,514,082	46,143,472	44,006,180
300,000	108,786,856	2,005,488,153	71,865,018	66,998,550
350,000	125,974,221	2,318,482,748	83,199,724	77,886,160
439,423	158,106,118	2,910,314,146	104,829,757	97,393,301

5 Further Research Directions in Math Similarity, Clustering and Disambiguation

In mathematics, Mathematical Subject Classification (MSC) is used by most journals today, being supported and developed by both Mathematical Reviews (MR) and Zentralblatt Math (ZMath). Our research so far [13] has shown that machine-learned classification and similarity tasks are tractable to be supported by DMLs. However, previous research paid very little attention to the representation of mathematics. Either textual tokens alone were used, or the formulae were split into variables, constants and operators, and used in a 'bag of words' for documents. Such representation is insufficient given that it does not convey the structure of formulae, and neither does it pay attention to semantically similar formulae (e.g. written in different variable names, sorted differently as $a + b$ vs. $b + a$, etc.).

We are currently using the **Gensim** [14] system to evaluate the possibility of using M-terms instead of the usual tokenization and comparing the effects this new representation has on similarity and clustering improvements over non math-aware representations. We believe that M-term representation will significantly improve the quality of document similarity metrics computed by Gensim.

Further improvements could be achieved by employing cutting edge results on semantic disambiguation. Symbol f might play the rôle (have meaning) of a variable, functional, (linear) function, and potentially a dozen other meanings. To have a greater relevance to searching and better document clustering, even mathematical formulae should be disambiguated at this level, as authors are usually reluctant to do so in the ($\LaTeX$) sources or in Content MathML. Our representation method is easily inclusive with respect to these refinements – one just needs to add a notation and weighting for similarity of new terms representing Content MathML (semantics).

There were attempts to bring NLP approaches to math corpora handling recently [15,16]. The most consistent problem remains the high degree of ambiguity in mathematical formulae and nonexistence of tagged disambiguated math data.

There is a promising approach to distinguishing roles of words (math tokens) which depends on the contexts of use in a corpus called *LDA-frames* [17]. It uses statistics to distinguish different roles based on different structural patterns of word usage in corpora. We are considering the possibility of using a fuzzy version of Formal Concept Analysis (FCA) [18] to identify the rôles of math tokens in formulae, and in combination with LDA-frames to disambiguate them.

6 Summary and Conclusions

In this paper, we have identified and described the problems we have faced when building nontrivial corpora of STEM documents MREC. We have suggested M-term representation for math-aware indexing and similarity computations. We have reported current results in imath-aware indexing and isearching. We have discusses future research directions towards fully fledged math-aware corpora processing like math-aware document similarity or disambiguation of math symbols in formulae.

Acknowledgements This work has been partially supported by the Ministry of Education of CR within the Center of Basic Research LC536 and by the European Union through its Competitiveness and Innovation Programme (Information and Communications Technologies Policy Support Programme, “Open access to scientific information”, Grant Agreement No. 250503).

References

1. Bartošek, M., Lhoták, M., Rákosník, J., Sojka, P., Šárfy, M.: DML-CZ: The Objectives and the First Steps. In: Borwein, J., Rocha, E.M., Rodrigues, J.F., eds.: CMDE 2006: Communicating Mathematics in the Digital Era. A. K. Peters, MA, USA (2008) 69–79.

2. Borbinha, J., Bouche, T., Nowiński, A., Sojka, P.: Project EuDML—A First Year Demonstration. In: Davenport, J.H., Farmer, W.M., Urban, J., Rabe, F., eds.: Intelligent Computer Mathematics. Proceedings of 18th Symposium, Calculemus 2011, and 10th International Conference, MKM 2011. Volume 6824 of Lecture Notes in Artificial Intelligence, LNAI., Berlin, Germany, Springer-Verlag (2011) 281–284 `http://dx.doi.org/10.1007/978-3-642-22673-1_21`.
3. Sojka, P., Líška, M.: Indexing and Searching Mathematics in Digital Libraries – Architecture, Design and Scalability Issues. In: Davenport, J.H., Farmer, W.M., Urban, J., Rabe, F., eds.: Intelligent Computer Mathematics. Proceedings of 18th Symposium, Calculemus 2011, and 10th International Conference, MKM 2011. Volume 6824 of Lecture Notes in Artificial Intelligence, LNAI., Berlin, Germany, Springer-Verlag (2011) 228–243 `http://dx.doi.org/10.1007/978-3-642-22673-1_16`.
4. Sojka, P., Líška, M.: The Art of Mathematics Retrieval. In: Proceedings of the ACM Conference on Document Engineering, DocEng 2011, Mountain View, CA, Association of Computing Machinery (2011) 57–60 `http://doi.acm.org/10.1145/2034691.2034703`.
5. Elbow, A., Krick, B., Kelly, L.: PMC Tagging Guidelines: A case study in normalization. In: Proceedings of the Journal Article Tag Suite Conference 2011, National Center for Biotechnology Information (2011) `http://www.ncbi.nlm.nih.gov/books/NBK62090/#elbow-S8`.
6. W3C: MathML Test Suite (2010) `http://www.w3.org/Math/testsuite/`.
7. Archambault, D., Stöger, B., Batušić, M., Fahrengruber, C., Miesenberger, K.: A software model to support collaborative mathematical work between Braille and sighted users. In: Proceedings of the ASSETS 2007 Conference (9th International ACM SIGACCESS Conference on Computers and Accessibility), ACM (2007) 115–122 `http://portal.acm.org/ft_gateway.cfm?id=1296864&type=pdf`.
8. Archambault, D., Berger, F., Moço, V.: Overview of the "Universal Maths Conversion Library". In: Pruski, A., Knops, H., eds.: Assistive Technology: From Virtuality to Reality: Proceedings of 8th European Conference for the Advancement of Assistive Technology in Europe AAATE 2005, Lille, France, Amsterdam, The Netherlands, IOS Press (2005) 256–260.
9. Archambault, D., Moço, V.: Canonical MathML to Simplify Conversion of MathML to Braille Mathematical Notations. In: Miesenberger, K., Klaus, J., Zagler, W., Karshmer, A., eds.: Computers Helping People with Special Needs. Volume 4061 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2006) 1191–1198 `http://dx.doi.org/10.1007/11788713_172`.
10. Líška, M., Sojka, P., Růžička, M., Mravec, P.: Web Interface and Collection for Mathematical Retrieval. In: Sojka, P., Bouche, T., eds.: Proceedings of DML 2011, Bertinoro, Italy, Masaryk University (2011) 77–84 `http://www.fi.muni.cz/~sojka/dml-2011-program.html`.
11. Stamerjohanns, H., Kohlhase, M., Ginev, D., David, C., Miller, B.: Transforming Large Collections of Scientific Publications to XML. Mathematics in Computer Science **3** (2010) 299–307 `http://dx.doi.org/10.1007/s11786-010-0024-7`.
12. Sylwestrzak, W., Borbinha, J., Bouche, T., Nowiński, A., Sojka, P.: EuDML—Towards the European Digital Mathematics Library. In: Sojka, P., ed.: Proceedings of DML 2010, Paris, France, Masaryk University (2010) 11–24 `http://dml.cz/dmlcz/702569`.
13. Řehůřek, R., Sojka, P.: Automated Classification and Categorization of Mathematical Knowledge. In: Autexier, S., Campbell, J., Rubio, J., Sorge, V., Suzuki, M., Wiedijk, F., eds.: Intelligent Computer Mathematics—Proceedings of 7th International Con-

ference on Mathematical Knowledge Management MKM 2008. Volume 5144 of Lecture Notes in Computer Science LNCS/LNAI., Berlin, Heidelberg, Springer-Verlag (2008) 543–557.

14. Řehůřek, R., Sojka, P.: Software Framework for Topic Modelling with Large Corpora. In: Proceedings of LREC 2010 workshop New Challenges for NLP Frameworks, Valletta, Malta, ELRA (2010) 45–50 `http://is.muni.cz/publication/884893/en`, software available at `http://nlp.fi.muni.cz/projekty/gensim`.
15. Anca, Ş.: Natural Language and Mathematics Processing for Applicable Theorem Search. Master's thesis, Jacobs University, Bremen (2009) `https://svn.eecs.jacobs-university.de/svn/eecs/archive/msc-2009/aanca.pdf`.
16. Grigore, M., Wolska, M., Kohlhase, M.: Towards context-based disambiguation of mathematical expressions. Math-for-Industry Lecture Note Series **22** (2009) 262–271.
17. Materna, J.: LDA-Frames: an Unsupervised Approach to Generating Semantic Frames. In: Proceedings of CICLING 2012, Springer-Verlag (2012) 12 pages, submitted.
18. Bělohlávek, R.: Concept lattices and order in fuzzy logic. Annals of Pure and Applied Logic **128**(1–3) (2004) 277–298.

Corpus-based Disambiguation for Machine Translation

Vít Baisa

Masaryk University
Botanická 68a, 602 00 Brno
`xbaisa@fi.muni.cz`

Abstract. This paper deals with problem of choosing a proper translation for polysemous words. We describe an original method for partial word sense disambiguation of such words using word sketches extracted from large-scale corpora and using simple English-Czech dictionary. Each word is translated from English to Czech and a word sketch for the word is compared with all word sketches of its appropriate Czech equivalents. These comparisons serve for choosing a proper translation of the word: given a context containing one of collocates from the English word sketch, result data can serve directly in the process of machine translation of the English word and at the same time it can be considered as a partial disambiguation of that word. Moreover, the results may be used for clustering word sketches according to distinct meanings of their headwords.

Key words: word sense disambiguation, word sketch, machine translation, collocations

1 Introduction

Word sense disambiguation (WSD) is one of the most challenging and demanding tasks in natural language processing (NLP). People are able to disambiguate with help of general context of a discourse (what has been said, environment in which the communication occurs, mood of a speaker, common sense etc.) but for current WSD systems, availability of such information is extremely limited and in majority of cases, only narrow textual context is exploited. We use word sketches [1] for capturing most usual context (collocations) together with translations of English words for representation of their meanings.

Selection of a proper translation is based upon a simple presumption: that the most frequent context of an English word (and its meaning) is similar to a translated context of a Czech equivalent of the English word. In other words: the meaning represented by the English and the Czech word has the same or similar contexts in English and Czech languages respectively.

If a polysemous English word is to be translated with its proper Czech equivalent, we can see how its Czech equivalents act, what are their contexts within a Czech corpus and use these information for desired translational disambiguation.

2 Word Sketches and Context

A word sketch is "one-page, automatic, corpus-derived summary of a word's grammatical and collocational behaviour" [1]. It is in fact formalised and generalised context: given a word, its word sketch consists of most usual words which appear in grammatical relations in contexts of the given word.

Word sketches are segmented into various relations which are specified by *word sketch grammar* (WSG) rules. These rules are special CQL (corpus query language) formulas strongly depending on a language. Table 1 shows abridged word sketch for English word *key*.

Table 1. Abridged word sketch for English word *key*.

a_modifier	**object_of**	**n_modifier**	**modifies**	**modifier**
cryptographic	steal	cursor	element	together
primary	turn	ignition	stakeholder	chiefly
programmable	remove	shift	point	generally
minor	bend	backspace	area	forward
golden	obtain	activation	aspect	increasingly
lost	define	hash	principle	however
F11	enter	F	figure	perhaps

Columns contain words from various grammatical relations: *minor* is adjective modifier (**a_modifier**) of headword *key*, *key* **modifies** *point* etc.

2.1 New Word Sketch Grammars

Word sketches are derived automatically from morphologically tagged corpora. In our case, English corpus ukWaC [2] with more than 10^9 words and Czech corpus CZES with more than 350 millions words were used.

Since we needed to compare English and Czech word sketches, we were forced to develop two new word sketch grammars which define equivalent relations for both languages. Original grammars for Czech and English had only one relation in common (**a_modifier**).

We have developed rules for 26 relations for both English and Czech: a few rules were taken over from existing grammars almost unchanged. A few other relations were incompatible and therefore had to be omitted and several rules were developed from scratch. WSG for English use The Penn Treebank tagset [3] and Czech WSG use tagset of morphological analyser AJKA [4].

The first example on Figure 1 defines relation **coord** which is symmetric. The rule looks for triplets of lemmas where the second lemma is either "a" (and) or "nebo" (or). The end of the rule means that the first and the second lemmas must have the same PoS tag (`k` stands for PoS in AJKA's tagset) and must be in the same cases (`c` means case in AJKA's tagset).

```
1:[] [word = "a" | word = "nebo"] 2:[] & 1.k=2.k & 1.c=2.c
```

Fig. 1. Example of simple rule for symmetric relation **coord**.

The second example on Figure 2 defines dual relation **a_modifier/modifies** which means that order of two lemmas is important: the first lemma is *adjective modifier* of the second one and the second lemma *modifies* the first one.

The rule says that the first lemma must be adjective (`JJ.?` is regular expression matching all adjectives in a corpus) but not a noun (`NN.?.?`).

```
2:"JJ.?" "NN.?.?"{0,2} 1:"NN.?.?"
```

Fig. 2. Example of simplified rule for relations **a_modifier** and **modifies**.

Both Czech and English corpora were compiled with these grammars and new word sketches were obtained for further processing.

3 Dictionary and Meaning

We may consider an English-Czech dictionary as a source of meanings for English words: for a given English word the dictionary contains its Czech equivalents with distinct meanings. Some of them may be mutually synonymous but we consider them as distinct meanings.

It is worth comparing statistics derived from English-Czech dictionary used in our experiment [5] with WordNet 3 statistics [6] (see Table 2).

Table 2. Comparing statistics for the used dictionary and WordNet.

	WordNet	GNU-FDL
lemmas	155,287	101,918
polysemous words	26,896	26,132
avg. polysemy all	1.37	1.56

The numbers are remarkably similar and they speak in favour of our presumption about representation of meanings by dictionary equivalents.

4 Background of the method

Word sense disambiguation should link an occurrence of a polysemous word to its meaning. The meaning can be represented for instance by a synset in

WordNet. We use different representation: a connection of an English word with its Czech translation. E.g. *link* has at least four meanings represented as *link–odkaz*, *link–vztah*, *link–propojení*, *link–článek* etc. The second words are Czech equivalents of English word *link*.

This representation allows only partial discrimination of meanings because both an English word and a Czech word may be polysemous and they can share more than one meaning. For instance *key* and its Czech equivalent *klíč* share at least two meanings: a key both for locking and for coding but they share the same representation within our approach: *key–klíč*.

The process tries to find as many as possible collocates for a given English word which could help to disambiguate the word in a context. The results can serve also for clustering of word sketches since they contain collocates for all meanings of a headword. It is the case of word sketch in Table 1 on page 82: **a_modifier**s *minor* and *cryptographic* belong to two distinct meanings of headword *key*.

The process itself (looking for candidate collocates for English word *e*) may be outlined as follows:

```
1. Get a word sketch for $e$.
2. Translate $e$ into Czech ($c^1, c^2, \ldots$) equivalents. Get word sketches for them.
3. For each pair $e$–$c^1$, $e$–$c^2$, ...:
      For each shared relation in the word sketches:
         Compute links: an English lemma $a$ from English relation $r$ and a
         Czech lemma $b$ from Czech relation $r$ make a link iff we can translate
         $a$ to $b$ using the dictionary.
4. Compute unique links: unique link is exclusive for some pair $e$–$c^i$. In other
   words, it is not included in any pair $e$–$c^j$ where $j \neq i$.
```

Unique links are very important for choosing a proper translation. Let us consider collocation *small key*. *Key*'s another Czech equivalent (besides *klíč*) is *klávesa* (a key on a keyboard). Since appropriate word sketches contain both *malý klíč* and *malý klávesa* and lemma *malý* makes a link with lemma *small* within relation `a_modifier`, the link is not unique and cannot serve for the actual disambiguation. Obviously, out of its context, it is impossible to decide whether to translate *small key* as *malý klíč* or *malý klávesa* (we are considering lemmas not in correct word forms).

5 Results

We processed all one-word lemmas from corpus ukWaC which were covered by the dictionary. Best results of the process are summarized in Table 3. These numbers deserve brief explanation.

It may seem strange that adjective *raw* has more meanings than such highly polysemous verbs as *get*, *take* etc. It is probably caused by the dictionary – it has insufficient amount of equivalents for these verbs but many equivalents for *raw*.

Table 3. Results for words and PoS.

	lemma	**PoS**	**count**
Maximal polysemy	raw	adjective	25
Most links	keep	verb	359
Most unique links	part	noun	86

The highest number of links for the lemma *keep* means that English and particular Czech word sketches are rich enough to provide so many links.

86 unique links of noun *part* mean that we are able to choose a proper translation of *part* in case of its 86 top-frequent collocates.

Table 4 shows abridged results clustered by relations. EN stands for average length of English word sketch relation, CS for average length of appropriate Czech relation. AL stands for average number of links per relation in the first column and UL for average number of unique links. The last two columns AL% and UL% are percentual expression of coverage of English relation by common and unique links, respectively. The highest numbers in columns are bold, the lowest are typeset in italics.

Table 4. Results clustered by relations.

Relation	**EN**	**CS**	**AL**	**UL**	**AL%**	**UL%**
`be_adj`	38.29	22.67	6.96	4.12	18.17	10.76
`n_modifier`	45.53	32.05	3.76	*2.76*	*8.26*	*6.06*
`subj_be`	39.29	31.10	5.17	3.61	13.16	9.19
`a_modifier`	43.61	**38.45**	9.71	5.65	22.26	12.96
`has_obj`	**48.39**	33.50	8.40	4.56	17.36	9.42
`prec_prep`	29.99	20.48	**15.97**	5.66	53.25	18.87
`modifies`	45.02	36.91	7.07	4.90	15.70	10.88
`gen_2`	39.37	33.77	8.89	5.56	22.58	14.12
`possessed`	32.40	26.75	5.00	3.64	15.43	11.23
`gen_1`	40.32	35.76	5.52	3.58	13.70	8.88
`coord`	39.28	34.41	5.90	3.77	15.02	9.60
`post_prep`	*29.00*	19.78	15.61	5.51	**53.83**	**19.00**
`modifier`	39.17	34.39	15.08	5.97	38.50	15.24
`and_other2`	33.82	*13.62*	*3.57*	2.79	10.55	8.25
`is_obj_of`	43.40	32.66	11.68	**7.46**	26.91	17.19

Results from Table 4 can be interpreted in this way:

1. The highest average amount of items in relation **has_obj** agrees with the ability of verbs to have many collocates.
2. The higher a number for a relation in AL column is, the better are appropriate rules (defining the relation) since they catch more words across both languages. But it definitely depends also on used corpus.

3. The higher a number for a relation in UL column is, the better is the relation for disambiguation. For instance, we are able to use almost 1/5 of relation `prec_prep` for choosing proper translations.

Table 5. Summarized results.

# of retrieved words	44,249
# of retrieved polysemous words	19,316
avg # of Czech eq. per word	2.06
avg # of Czech eq. per polys. word	4.74
avg # of links per word	168.17
avg # of unique links per word	98.73
avg # of links per polysemous word	386.5
avg # of unique links per polys. word	225.84

Table 5 shows overall results for the experiment. 44,249 lemmas from the English corpus were found in the dictionary. Almost 20,000 were polysemous and these are we focus on. The most important number from the Table 5 is on the last line: average number of unique links per polysemous word. It means that we are able to use about 226 collocates for choosing a proper translation of a polysemous word.

6 Comments, Conclusion and Future work

The problem concerning insufficient discrimination of various meanings by connecting English words with their Czech counterparts could be solved by adding other languages. Using triplets, quadruples, ... instead of pairs might narrow a number of shared meanings. E.g. *line–linie–Linie* vs. *line–linie–Kurs* for English, Czech and German.

We are not aware of any similar work except [7]. Experiment dealing with word sketch clustering using the only relation (**a_modifier**) is described in [8].

Critical issue is developing of new grammar rules with higher coverage and precision. And ther are two other ways how to increase recall. The first consists in using even bigger corpora for richer word sketches and the second in involving better dictionary. Our dictionary is maintained by volunteers and does not reach a quality of other, commercial dictionaries.

All these suggestions are subjects of future work. But even the described simple approach and current results seem to be promising.

Acknowledgement This work has been partly supported by the Ministry of Education of CR within the Center of basic research LC536 and by EC FP7 project ICT-248307.

References

1. Kilgarriff, A., Rychly, P., Smrz, P., Tugwell, D.: Itri-04-08 the sketch engine. Information Technology **105** (2004) 116
2. Ferraresi, A., Zanchetta, E., Baroni, M., Bernardini, S.: Introducing and evaluating ukwac, a very large web-derived corpus of english. In: Proceedings of the 4th Web as Corpus Workshop (WAC-4) Can we beat Google. (2008) 47–54
3. Marcus, M., Marcinkiewicz, M., Santorini, B.: Building a large annotated corpus of english: The penn treebank. Computational linguistics **19**(2) (1993) 313–330
4. Sedláček, R., Smrž, P.: A new czech morphological analyser ajka. In: Text, Speech and Dialogue, Springer (2001) 100–107
5. Svoboda, M.: Gnu/fdl english-czech dictionary (2001)
6. Generated: Wordnet statistics (2011)
7. Dyvik, H.: Translations as semantic mirrors: from parallel corpus to wordnet. Language and computers **49**(1) (2004) 311–326
8. Baisa, V.: Towards disambiguation of word sketches. In: Text, Speech and Dialogue, Springer-Verlag (2010) 37–42

Building a 50M Corpus of Tajik Language

Gulshan Dovudov[1], Jan Pomikálek[2], Vít Suchomel[1], Pavel Šmerk[1]

[1] Natural Language Processing Centre
Faculty of Informatics, Masaryk University
Botanická 68a, 602 00 Brno, Czech Republic
{xdovudov,xsuchom2,xsmerk}@fi.muni.cz

[2] Lexical Computing Ltd.
73 Freshfield Road, Brighton, UK
jan.pomikalek@sketchengine.co.uk

Abstract. Paper presents by far the largest available computer corpus of Tajik Language of the size of more than 50 million words. To obtain the texts for the corpus two different approaches were used. The paper brings a description of both of them, discusses their advantages and disadvantages and shows some statistics of the two respective partial corpora. Then the paper characterizes the resulting joined corpus and finally discusses some possible future improvements.

1 Introduction

Tajik language is a variant of the Persian language spoken mainly in Tajikistan and also in some few other parts of Central Asia. Unlike Iranian Persian, Tajik is written in the Cyrillic alphabet.

Since the Tajik language internet society (and consequently the potential market) is rather small and Tajikistan itself is ranked among developing countries, there are almost no resources for computational processing of Tajik at all. Namely the computer corpora of Tajik language are either small or even still only in the stage of planning or development. The University of Helsinky offers a very small corpus of 87 654 words.[1] Megerdoomian and Parvaz [3,2] mention a test corpus of approximately 500 000 words, and the biggest and the only freely available corpus is offered within the Leipzig Corpora Collection project [4] and consists of 100 000 Tajik sentences which equals to almost 1.8 million words.[2] Iranian linguist Hamid Hassani is said to be preparing a 1

[1] http://www.ling.helsinki.fi/uhlcs/readme-all/README-indo-european-lgs.html

[2] Unfortunately, the encoding and/or transliteration vary greatly: more than 5 % of sentences are in Latin script, almost 10 % of sentences seem to use Russian characters instead of Tajik specific characters (e.g. kh instead of Tajik h, which sound/letter does not exist in Russian) and more than 1 % of sentences uses other than Russian substitutes to Tajik specific characters (e.g. Belarussian short u instead of proper Cyrillic u with macron) — and only the last case is easy to repair automatically.

million words Tajik corpus[3] and Tajik Academy of Sciences prepares a corpus of 10 million words[4]. Unfortunately, at least by now the latter is not a corpus of contemporary Tajik, but rather a collection of works—and moreover mainly a poetry—of a few notable Tajik writers (one of them is even from the 13th century).

In this paper we present a newly built corpus of contemporary Tajik language of more than 50 million words. All texts were taken from the internet. We used two different approaches to obtain the data for the corpus and we describe these methods and their results in the following two sections. In the Section 4 we present the resulting corpus and finally we discuss some possible future improvements.

2 Semi-automatically Crawled Part

The first part of the corpus was collected by crawling several news portals in Tajik language.[5] If the articles of a particular portal were numbered, then we tried to download all possible numbers, otherwise we get a list of articles either directly from the portal, or from the Google cache. Each single portal was processed separately to get the maximum of relevant (meta)information, i.e. correct headings, publication date, rubric etc.

Table 1. Statistics of the semi-automatically crawled part of the corpus.

source	**docs**	**pars**	**words**	**w/doc**	**tokens**	**MB**
ozodi.org	54301	343057	12037886	222	14073673	170
gazeta.tj archive	480	163572	5006650	10431	6032214	67
bbc.co.uk	8032	147073	3694769	460	4270087	51
jumhuriyat.tj	7598	98102	3485592	459	4134782	50
khovar.tj	15420	62280	2323445	151	2835757	36
tojnews.org	8377	64583	2254917	269	2740760	33
millat.tj	2373	44710	2004863	845	2361816	27
gazeta.tj	1608	31962	1130957	703	1357889	15
gazeta.tj library	130	98698	1053262	8102	1358584	15
pressa.tj	2165	20526	642394	297	782917	9
news.tj	1396	7759	269267	193	325164	4
muhabbatvaoila.tj	503	9957	241905	481	297964	3
all	**102383**	**1092279**	**34145907**	**334**	**40571607**	**480**

[3] `http://www.tajikistan.orexca.com/tajik_language.shtml`
`http://en.wikipedia.org/wiki/Hamid_Hassani`

[4] `http://www.termcom.tj/index.php?menu=bases&page=index3&lang=eng` (in Russian)

[5] Paradoxically, the two largest Tajik news portals are not located in Tajikistan, but in Czech Republic (ozodi.org, Tajik version of Radio Free Europe/Radio Liberty) and United Kingdom (bbc.co.uk, Tajik version of BBC).

In the Table 1 we present some statistics of the obtained data. Column docs contains a number of documents downloaded from the given source, pars is a number of paragraphs (including headings), words is a number of tokens which contain only characters from Tajik alphabet, w/doc is a words / document ratio (i.e. average length of possibly continuous texts), tokens is a number of all tokens (words, interpunction etc.) and MB is the size in megabytes of the data in vertical corpus format (i.e. plain text). The table is sorted by number of words. From the electronic library on `gazeta.tj` we choose only prose and omit all more structured texts as poetry, drama or e.g. computer manual. The articles in `gazeta.tj` archive are joined in one file on a weekly basis and that is why the words / document ratio is so high.

On allmost all websites, alongside the articles in Tajik there were also many articles in Russian. Because both alphabets, Tajik and Russian, contain characters which do not occur in the other alphabet, it is easy to distinguish between the two languages and discard the Russian articles even without any further language analysis.

3 Automatically Crawled Part

SpiderLing, a web crawler for text corpora [5], was used to automatically download documents in Tajik from the web. We started the crawl using 2570 seed URLs (from 475 distinct domains) collected with Corpus Factory [1].

The crawler downloaded 9.5 GB of HTML data in ca. 300,000 documents over three days. That is not much compared to crawling documents in other languages by the same tool. For example the newly built web corpus of Czech, which has roughly only two times more native speakers compared to Tajik, has more than 5 billion words, of course not in three days. We conclude the available online resources in Tajik are very scarce indeed. An overview of internet top level domains of URLs of the documents obtained can be found in Table 2.

Table 2. Number of documents by internet top level domain

TLD	docs downloaded	docs accepted in the corpus
tj	55.0 %	51.7 %
com	23.0 %	28.1 %
uk	8.9 %	7.2 %
org	6.8 %	7.7 %
ru	2.6 %	1.4 %
ir	1.6 %	2.4 %
other	2.0 %	1.5 %

Since Russian is widely used in government and business in Tajikistan (and other language texts may appear), 33 % of the downloaded HTML pages were removed by the SpiderLing's inbuilt language filter [5]. The obtained data was tokenized and deduplicated using Onion[6] with moderately strict settings[7]. Some statistics of the automatically crawled part of the corpus are in the Table 3 (only the ten most productive sources of data are detailed).

Table 3. Statistics of the automatically crawled part of the corpus.

source	docs	pars	words	w/doc	tokens	MB
*.wordpress.com	3385	68228	4007147	1184	4890924	55
bbc.co.uk	5238	97897	2556034	488	2939833	37
ozodi.org	6847	83955	2257307	330	2691184	33
khovar.tj	10716	30275	1801314	168	2202621	29
gazeta.tj	1655	8122	1110378	671	1334125	15
millat.tj	1381	20843	1108658	803	1299469	15
*.blogspot.com	800	18272	974842	1219	1208834	13
ruzgor.tj	1473	17575	943143	640	1084123	11
firdavsi.com	588	22433	857906	1459	1029023	11
pressa.tj	2312	13530	655840	284	792631	10
			...			
all	**61523**	**612178**	**28841537**	**469**	**34680994**	**405**

4 Corpus of Tajik Language

The two partial corpora were joined together and the result was deduplicated using Onion. We obtained a corpus of more than 50 million words, i.e. corpus positions which consists solely of Tajik characters, and more than 60 million tokens, i.e. words, interpunction, numbers etc. Detailed numbers follow in the Table 4.

It was rather surprising for us, that the fully automated crawling yielded even smaller data than the semi-automated approach. It has to be said, that at least 25 % of semi-automatically crawled data were inaccessible to the general crawler, as it cannot extract texts from RAR-compressed archives (gazeta.tj archive and library) and because it does not seem to exist any link to bigger part of older BBC articles although they remained on the server (we exploited Google cache to get the links). It is higly probable that also the other sites contain articles unreachable by any link chain and thus inaccessible for the general crawler. But even if we discount these data, the automated crawling did not outperform the semi-automated one in such an extent that we expected and which is common for many other languages. As we remarked in the Section 3, we attribute it to the scarceness of online texts in Tajik language. It means that we probably reach or almost reach the overall potential of internet resources,

[6] `http://code.google.com/p/onion/`

[7] removing paragraphs with more than 50 % of duplicate 7-tuples of words

Table 4. Statistics of the resulting corpus.

source	docs	pars	words	w/doc	tokens	MB
ozodi.org	56181	378790	12964584	231	15198491	183
gazeta.tj archive	480	163572	5006650	10431	6032214	67
bbc.co.uk	8188	147564	3706224	453	4283839	51
*.wordpress.com	3181	58604	3616710	1137	4416611	50
jumhuriyat.tj	7599	98141	3489592	459	4139709	50
khovar.tj	16554	66129	2502796	151	3048777	38
tojnews.org	8426	64770	2259311	268	2746100	33
millat.tj	2687	47823	2172612	809	2557686	29
gazeta.tj	1894	32537	1170516	618	1403929	16
gazeta.tj library	130	98698	1053262	8102	1358584	15
			...			
all	**134329**	**1430896**	**51768804**	**385**	**61943879**	**721**

i.e. even if we somehow get all Tajik online texts, the corpus might be bigger by half, but surely not for example ten times or even three times.

The new corpus is not freely available for a download at the moment, but eventual interested researchers can be given an access to it through the Sketch Engine[8].

5 Future Work

The Table 5 shows statistics of the texts which were new in the automatically crawled part compared to the semi-automatically crawled data. The numbers indicate that there is a room for an extension of the semi-automated part. We will prepare specialized scripts for the most productive portals to download the data in a some more controlled way.

It is worth clarifying the case of ozodi.org. The general crawler tries to get all reasonable texts on the page, which, on the news portals, may include the readers' comments. On the other hand, because the comments may contain a substandard language features, they were omitted during the semi-automated crawling. Thus the 1880 documents from ozodi.org were not some newly added ones, but they were results of the deduplication which discarded the article itself and leaved only the comments as it processes corpus by single paragraphs. This is also one of the reasons why we prefer the semi-automated crawling when it is possible: we want to tag these comments to allow a creation of subcorpora of a (presumably standard) language of articles as well as of a language of comments.

Another problem with the comments—but not only with them—is a common absence of Tajik-specific characters. The language model for the general

[8] `http://www.sketchengine.co.uk/`

Table 5. The contribution of automatically crawled part

source	docs	pars	words	w/doc	tokens	MB
*.wordpress.com	3181	58604	3616710	1137	4416611	50
ozodi.org	1880	35733	926698	493	1124818	13
*.blogspot.com	752	15462	841228	1119	1049364	12
ruzgor.tj	1399	13281	709124	507	814561	8
firdavsi.com	472	19624	684496	1450	820569	9
khatlonpress.tj	870	7983	541402	622	628142	6
kemyaesaadat.com	439	12008	523194	1192	631801	7
ucoz.ru	742	9068	494783	667	596656	7
nahzat.tj	2435	5478	465748	191	545995	7
ozodagon.com	1962	6920	450334	230	542911	7
			...			
all	**31946**	**338617**	**17622897**	**552**	**21372272**	**242**

crawler was trained using Tajik Wikipedia[9] of our corpus to make the crawler download texts in language which looks like the language of Tajik Wikipedia. Unfortunately, in many Wikipedia articles the Tajic-specific characters are replaced by some other characters. The unambiguous replacements were trivially repaired in the whole corpus, but e.g. Cyrillic kh can sometimes stand either for Tajic-specific h or also for kh itself. On the one hand we plan to tag such texts to allow a creation of subcorpora with or without them, on the other hand we want to develop a program which would be able to repair them. We will also train the language model with another sets of texts to see how it will affect the crawled data.

Acknowledgements This work has been partly supported by Erasmus Mundus Action II lot 9: Partnerships with Third Country higher education institutions and scholarships for mobility, and by the Ministry of Education of CR within the Center of basic research LC536.

References

1. Kilgarriff, A., Reddy, S., Pomikálek, J., PVS, A.: A Corpus Factory for Many Languages. In: Proceedings of the Seventh International Conference on Language Resources and Evaluation, LREC 2010. Valleta, Malta (2010)
2. Megerdoomian, K.: Language Engineering for Lesser-Studied Languages, chap. Low-density Language Strategies for Persian and Armenian, pp. 291–312. IOS Press, Amsterdam (2009)

[9] The use of Wikipedia to train the language model is a part of default settings or a default scenario of the process of building corpora for new languages without any other utilizable resources. Of course we have better Tajik texts at hand, but the automatically crawled part of our corpus had also to act as a test of a general suitability of our technologies for the case of building corpora for low-density languages.

3. Megerdoomian, K., Parvaz, D.: Low-density Language Bootstrapping: The Case of Tajiki Persian. In: Proceedings of the Sixth International Conference on Language Resources and Evaluation, LREC 2008. Marrakech, Morocco (2008)
4. Quasthoff, U., Richter, M., Biemann, C.: Corpus Portal for Search in Monolingual Corpora. In: Proceedings of the Fifth International Conference on Language Resources and Evaluation, LREC 2006. Genoa (2006)
5. Suchomel, V., Pomikálek, J.: Practical Web Crawling for Text Corpora. In: Proceedings of the Fifth Workshop on Recent Advances in Slavonic Natural Language Processing, RASLAN 2011. Masaryk University, Brno (2011)

Practical Web Crawling for Text Corpora
Work in Progress

Vít Suchomel[1] and Jan Pomikálek[1,2]

[1] Natural Language Processing Centre
Faculty of Informatics, Masaryk University, Brno
{xsuchom2, xpomikal}@fi.muni.cz
nlp.fi.muni.cz

[2] Lexical Computing Ltd.
jan.pomikalek@sketchengine.co.uk

Abstract. SpiderLing—a web spider for linguistics—is new software for creating text corpora from the web, which we present in this article. Many documents on the web only contain material which is not useful for text corpora, such as lists of links, lists of products, and other kind of text not comprised of full sentences. In fact such pages represent the vast majority of the web. Therefore, by doing unrestricted web crawls, we typically download a lot of data which gets filtered out during post-processing. This makes the process of web corpus collection inefficient. The aim of our work is to focus the crawling on the text rich parts of the web and maximize the number of words in the final corpus per downloaded megabyte. We present our preliminary results from creating Web corpora of texts in Czech and Tajik.

Key words: Crawler, web crawling, corpus, web corpus, text corpus

1 Introduction

Text corpora have a wide range of applications in linguistics. The source of data for corpora which has become very popular in the recent years is the web. A web crawler is a computer program traversing web pages and downloading documents. Due to the enormous size of the web an important measure of the quality of a crawler is its speed – the number of bytes downloaded per time unit. However, often it also matters which bytes we are downloading. In the context of web corpora, an even more important measure of quality is the number of words in the final corpus retrieved per time unit. Therefore, in this context the quality of a crawler depends on the implemented traversing algorithm.

We have experimented with using third party software for obtaining text documents from the web. Following the example of other researchers [1], we have used Heritrix crawler[1] and downloaded documents for the language in

[1] http://crawler.archive.org/

interest by restricting the crawl to national web domains of the countries where the language is widely used (e.g. `.cz` for Czech). Though we managed to compile corpora of up to 3 billion words in this way, we were not satisfied with the fact that we need to keep the crawler running for several weeks and download terabytes of data in order to retrieve a reasonable amount of text. It turned out that most downloaded documents are discarded during post-processing since they contain only material with little or no running text.

In order to reduce the amount of unwanted downloaded content, we decided to create a custom web crawler which actively looks for text rich resources and avoids websites containing only material not suitable for text corpora. Our hope was that by avoiding the unwanted content we can not only save bandwidth but also shorten the time required for building a web corpus of a given size.

2 Analysis of previous work

We were interested to know how much data we download in vain when using Heritrix and if the sources which should be avoided can be easily identified. In order to get that information we analyzed the data of a billion word corpus of European Portuguese downloaded from the `.pt` domain with Heritrix. For each downloaded web page we computed its yield rate as

$$yield\ rate = \frac{final\ data}{downloaded\ data}$$

where *final data* is the number of bytes in the text which the page contributed to the final corpus and *downloaded data* is simply the size of the page in bytes (i.e. the number of bytes which had to be downloaded). Many web pages have a zero yield rate, mostly because they get rejected by a language classifier or they only contain junk or they only contain text duplicate to previously retrieved text.

We grouped the data by web domains and computed a yield rate for each domain as the average yield rate of the contained web pages. We visualized this on a scatterplot which is displayed in Fig. 1. Each domain is represented by a single point in the graph.

It can be seen that the differences among domains are enormous. For example, each of the points in the lower right corner of the graph represents a domain from which we downloaded more than 1 GB of data, but it only yielded around 1 kB of text. At the same time, there are domains which yielded more than 100 MB of text (an amount higher by 5 orders of magnitude) from a similar amount of downloaded data. These domains are positioned in the upper right corner of the graph.

Next, we selected a set of yield rate thresholds and computed for each threshold the number of domains with a higher yield rate and the sum of downloaded and final data in these domains. The results can be found in Table 1.

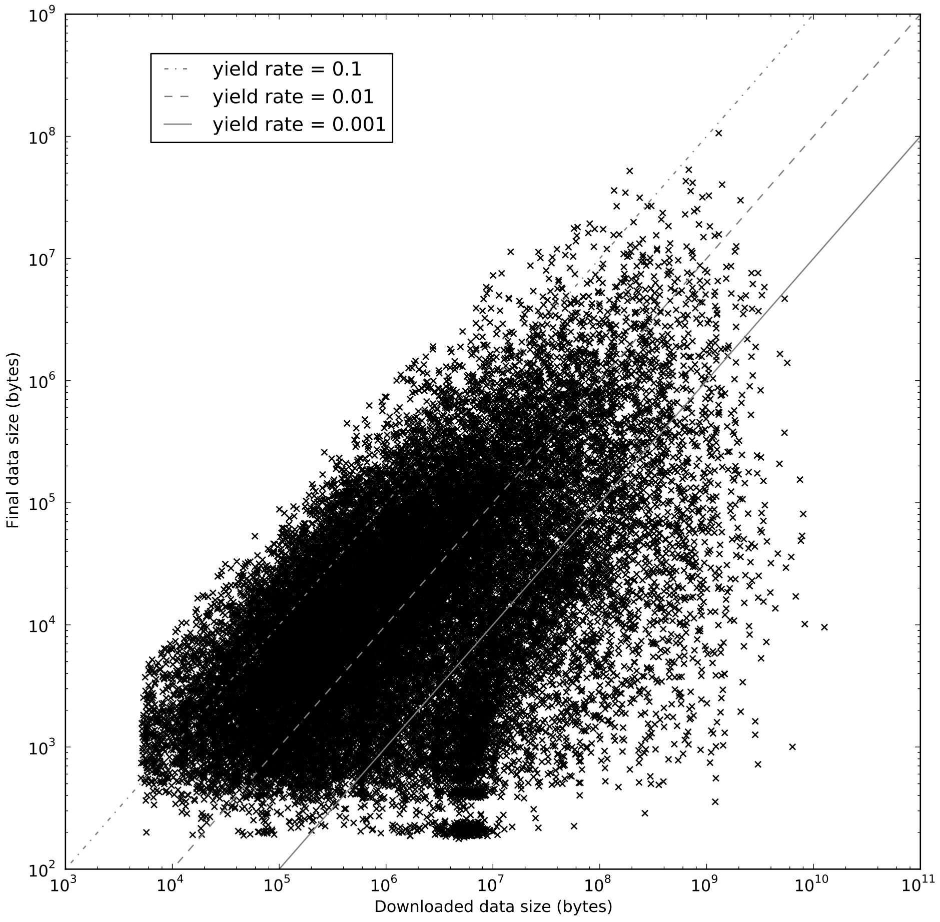


Fig. 1. Web domains yield rate for a Heritrix crawl on `.pt`.

Table 1. Sums of downloaded and final data size for all domains above given the yield rate threshold.

yield rate threshold	domains	total downl data	total final data
none	51645	1288.87 GB	4.91 GB
0	31024	1181.56 GB	4.91 GB
0.0001	29580	705.07 GB	4.90 GB
0.0002	28710	619.44 GB	4.89 GB
0.0004	27460	513.86 GB	4.86 GB
0.0008	25956	407.30 GB	4.80 GB
0.0016	24380	307.27 GB	4.68 GB
0.0032	22325	214.18 GB	4.47 GB
0.0064	19463	142.38 GB	4.13 GB
0.0128	15624	85.69 GB	3.62 GB
0.0256	11277	45.05 GB	2.91 GB
0.0512	7003	18.61 GB	1.98 GB
0.1024	3577	5.45 GB	1.06 GB
0.2048	1346	1.76 GB	0.54 GB
0.4096	313	0.21 GB	0.1 GB

It is easy to see that as the yield rate threshold increases the size of the downloaded data drops quickly whereas there is only a fairly small loss in the final data. This suggests that by avoiding the domains with low yield rate a web crawler could save a lot of bandwidth (and time) without making the final corpus significantly smaller. For instance if only domains with a yield rate above 0.0128 were crawled, the amount of downloaded data would be reduced from 1289 GB to 87 GB (to less than 7%) while the size of the final data would only drop from 4.81 GB to 3.62 GB (73.7%). This is of course only a hypothetical situation, since in practice one would need to download at least several pages from each domain in order to estimate its yield rate. Nevertheless, it is clear that there is a lot of room for making the crawling for web corpora much more efficient.

One could argue that a segmentation by domains is too coarse-grained since a single domain may contain multiple websites with both high and low yield rates. While we fully agree, we believe that identifying more fine-grained sets of web pages, such as websites, introduces further complications and we leave that for future work.

3 SpiderLing

We came to the conclusion that the easiest way of implementing our very specific requirements on web crawling is to create a custom crawler from scratch. We selected Python as our programming language to support rapid development. In order to make debugging easier we avoided a multi-threaded design. Instead, we use asynchronous communication for downloading data from multiple servers at the same time – a simple solution which scales up well (we can keep up to 5000 simultaneously open connections without any problems).

3.1 Improving yield rate

Our primary aim is to identify high-yielding domains and to avoid low-yielding ones. At the same time we want to make sure that we do not download all the data only from a few top-yielding domains so that we achieve a reasonable diversity of the obtained texts.

We collect information about the current yield rate of each domain as we are crawling the web. If the yield rate drops below a certain threshold we blacklist the domain and do not download any further data from it. We define a minimum amount of data which must be retrieved from each domain before it can be blacklisted. Currently the used limit is 8 web pages or 512 kB, whichever is a higher amount of data. The yield rate threshold is dynamic and increases as more pages are downloaded from the domain. This ensures that sooner or later all domains get blacklisted, which prevents overrepresentation of data from a single domain. Nevertheless, low-yielding domains are blacklisted sooner and thus the average yield rate increases.

The yield rate threshold for a domain is computed using the following function:

$$t(n) = 0.01 \cdot \left(\log_{10}(n) - 1\right)$$

where n is the number of documents downloaded from the domain. The function is based partly on the authors' intuition and partly on the results of initial experiments. Table 2 contains a list of thresholds for various numbers of downloaded documents.

Table 2. The yield rate threshold as a function of the number of downloaded documents.

# of documents	yr threshold
10	0.00
100	0.01
1000	0.02
10000	0.03

We experimented with various parameters of the yield rate threshold function. Fig. 2 shows how the average yield rate changes in time with different yield rate threshold functions. All these experiments have been performed with Czech as the target language. It can be seen that stricter threshold functions result in higher average yield rate. However, too high thresholds have a negative impact on the crawling speed (see section 3.5). It is therefore necessary to make a reasonable compromise.

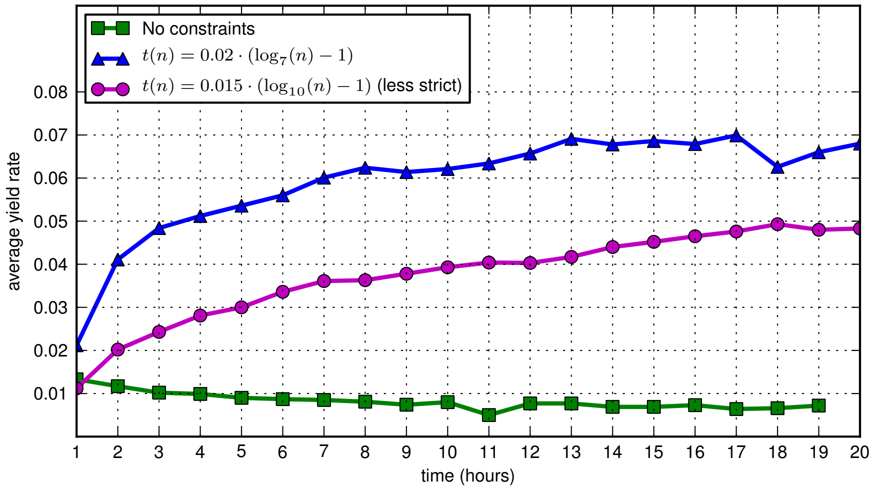


Fig. 2. Average yield rate in time for various yield rate threshold functions.

3.2 Removing junk and duplicates

We use jusText[2] [3]—a heuristic based boilerplate removal tool—to remove content such as navigation links, advertisements, headers and footers from downloaded web pages. Only paragraphs containing full sentences are preserved.

Duplicate documents are removed at two levels: (i) original form (text + HTML), and (ii) clean text as produced by jusText. Two correspondent checksums are computed for each web page and stored in memory. Documents with previously seen checksums are discarded. As a post-processing step, we also remove near-duplicates using onion[3].

We currently do not filter unwanted web content such as link farms and machine generated texts. This may be a subject to further research. Note though that some of such content (e.g. excerpts of Wikipedia articles on link farms) is already reduced in our current processing pipeline as a positive side effect of de-duplication.

3.3 Character encoding and language detection

We need to detect the character encoding of each downloaded document in order to be able to display its text correctly and/or to unify the encoding of all documents, e.g. by converting to UTF-8. Though for most web pages the character encoding may be determined from the meta tags or from HTTP headers, the information is not always available and not always correct and

[2] `http://code.google.com/p/justext/`

[3] `http://code.google.com/p/onion/`

in general cannot be relied upon. Therefore, we detect the encoding from the contained text by using chared[4].

Language detection is another problem which has to be addressed since we are typically building a web corpus for a particular language and want to avoid any texts in other languages. Unfortunately, it is difficult to detect a language of a text without identifying its character encoding first and vice versa. This is a typical chicken and egg problem. We use a simple trick here. We perform the character encoding detection first, assuming the input is in our target language. If the assumption is not correct it is very likely that the input gets rejected by the language filter in the next step anyway and thus it does not matter if the encoding detections fails.

Language filtering is performed at two levels. (i) jusText—our boilerplate removal tool—uses a list of the most frequent words in the language for identifying paragraphs containing grammatical text. The rationale is that the most frequent words are typically function words and a grammatical text should contain a certain proportion of these words. This filtering has a positive side effect of rejecting texts in other languages. (ii) We build a histogram of triples of characters on a sample text in the target language and compare the histogram of the text of each downloaded document with the model. This is done using the Trigram class created by Douglas Bagnall[5]. We use a similarity threshold of 0.5.

Both applied language filtering methods tend to accept texts in similar languages (e.g. Slovak texts when the target language is Czech). Nevertheless, this has not been a major problem so far. When creating a Czech corpus, only a small amount of Slovak has been included and we managed to identify and remove these texts during post-processing.

3.4 Starting URLs

A large set of starting URLs is needed for the crawler to quickly start retrieving the data from many domains in parallel. We use Corpus Factory [2] for getting a list of starting URLs for the target language. The tool compiles a list of medium frequency words in the language by using texts from Wikipedia. These words are then randomly combined into tuples of 3 to 5 and each tuple is used as a query to a search engine (we currently use bing[6]). As a result we get a list of URLs which are likely to contain documents in our target language.

3.5 Crawling speed

The maximum crawling speed we have achieved in our experiment was ca. 12 MB/s. However, we observe that the speed tends to decrease as the crawling progresses. Since we typically start from a large set of seed URLs, we have

[4] `http://code.google.com/p/chared/`

[5] `http://code.activestate.com/recipes/326576-language-detection-using-character-trigrams/`

[6] `http://www.bing.com/`

enough distinct domains to download from in parallel at the beginning and thus the initial crawling speed is good. However, as the crawling continues currently processed domains get blacklisted faster than new high-yielding domains are discovered. This reduces the number of domains available for download and thus limits the crawling speed.

Fig. 3 and Fig. 4 show how the crawling speed changes in time. The speed is measured as the amount of raw HTML and the amount of clean text retrieved per time unit. The data originates from two web crawls – for Czech and Tajik. For Tajik, the available online resources are very scarce which affects the crawling speed significantly.

3.6 Crawling constraints

A web crawler should abide by the Robots Exclusion Standard[7]. SpiderLing uses a third party Robot Exclusion Rules Parser[8] which implements the up-to-date (2008) version of the standard better than the Python built-in library. The used parser also supports several non-standard but frequently used robots.txt directives.

A crawler should not overuse web servers by querying too often. Our crawler implements both per web domain and per IP address limits and by default makes a maximum of 12 queries per minute and 10 queries per second respectively.

3.7 Checkpoints

SpiderLing supports periodical saving of all important in-memory data (visited domains, queued URLs, document checksums) to file system. It is possible to resume crawling from a saved state in case of a failure (e.g. due to a bug in the code or a server dropout). The state is saved in a human readable form and allows manual inspection for debugging purposes.

4 Results

4.1 Yield rate

By applying yield rate thresholds on domains we managed to reduce downloading data which is of no use for text corpora and increased the overall average yield rate. Fig. 5 contains the same kind of scatterplot as displayed in Fig. 1, this time on the data downloaded by SpiderLing with Czech as a target language. This is a significant improvement over the previous graph. For low-yielding domains only up to 1 MB of data is downloaded and high amounts of data are only retrieved from high-yielding sources. Table 3 contains a summary of the results of this web crawler run.

[7] `http://www.robotstxt.org/`

[8] `http://nikitathespider.com/python/rerp/`

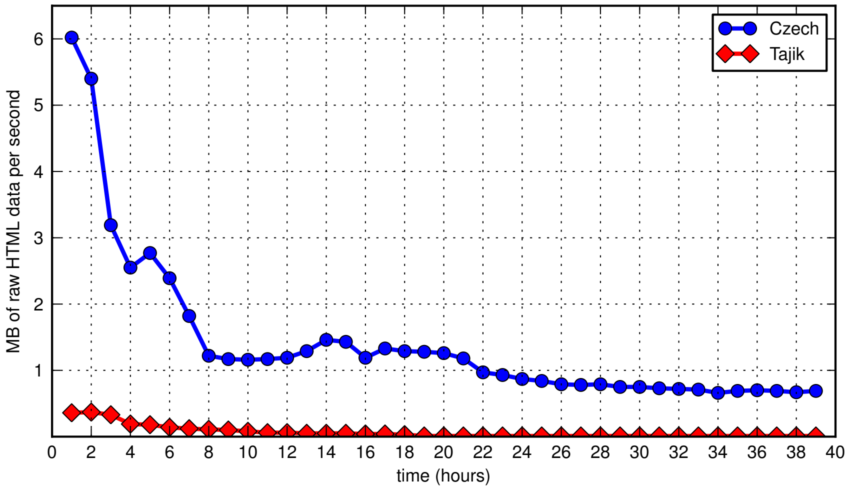


Fig. 3. Download speed in time in terms of downloaded raw HTML data.

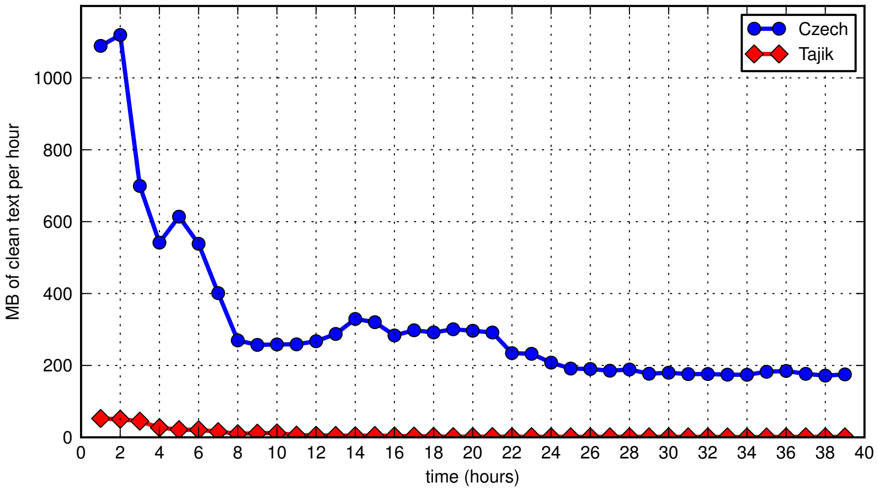


Fig. 4. Download speed in time in terms of downloaded clean texts.

Table 3. Results of crawling Czech web with SpiderLing.

downloaded documents	15,525,554
downloaded data size	515,580 MB
final data size	30,522 MB
yield rate	5.92 %

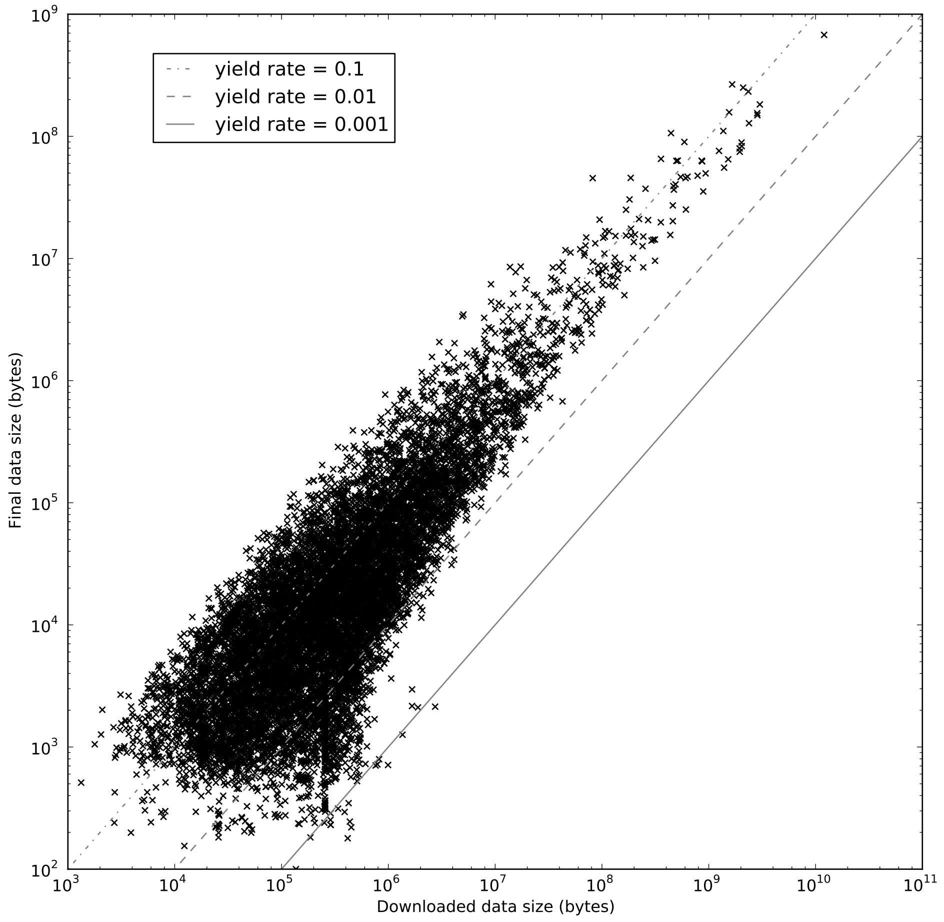


Fig. 5. Web domains yield rate for a SpiderLing crawl on Czech web.

4.2 Created corpora

So far we have used SpiderLing to create two corpora. During the development of the crawler we downloaded a total of ca. 4 TB Czech web pages in a number of web crawler runs. This amounts to ca. 5 billion tokens after all post-processing steps, including de-duplication with onion. We merged the corpus with a ca. 2 billion word Czech web corpus we have collected previously by using Heritrix. Since the two corpora overlapped to a high extent, the size of the final Czech web corpus after de-duplication is 5.8 billion tokens.

As a next exercise we ran SpiderLing on Tajik, partly to support the work of a visiting fellow researcher and partly to find out how the crawler will deal with scarce online resources. We started the crawl from 2570 seed URLs (from 475 distinct domains) collected with Corpus Factory. Over 3 days the crawler downloaded 9.5 GB of HTML data which yielded a 35 million tokens corpus after all post-processing.

5 Future work

The primary goal of our future work is to test the crawler on other languages and create further large web corpora. We believe that crawling speed might be less of a problem for languages where vast online text resources are available (e.g. English or Spanish). Nevertheless, we also want to invest more effort into optimizing the crawling constraints so that a higher crawling speed can be achieved even for scarcer resources.

Other plans for the future include analyzing the topics and genres of the downloaded texts and eventually balancing the downloaded content in this respect.

6 Conclusion

We presented SpiderLing, a web crawler for text corpora. We have shown that the crawler can effectively avoid web data not suitable for text corpora and significantly improve the yield rate of the downloaded content. The crawler has already been successfully applied for creating a major part of a large (5.8 billion tokens) Czech web corpus. We also managed to create a 35 million tokens web corpus of Tajik by using SpiderLing. Though this is only a smallish corpus, we consider it a promising achievement since online Tajik texts are scarce.

Acknowledgements The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 248307 (PRESEMT project), from the Ministry of Education of CR within the Center of basic research LC536 and from the Czech Science Foundation under the project P401/10/0792.

References

1. M. Baroni, S. Bernardini, A. Ferraresi, and E. Zanchetta. The wacky wide web: A collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226, 2009.
2. A. Kilgarriff, S. Reddy, J. Pomikálek, and A. PVS. A corpus factory for many languages. *Proc. LREC, Malta*, 2010.
3. J. Pomikálek. *Removing Boilerplate and Duplicate Content from Web Corpora*. PhD thesis, Masaryk University, Brno, 2011.

Part IV

Language Modelling

A Bayesian Approach to Query Language Identification

Jiří Materna[1,2] and Juraj Hreško[2]

[1] Centre for Natural Language Processing
Faculty of Informatics, Masaryk University
Botanická 68a, 602 00, Brno, Czech Republic
`xmaterna@fi.muni.cz`
[2] Seznam.cz, a.s.
Radlická 608/2, 150 00, Praha 5, Czech Republic
`{jiri.materna,juraj.hresko}@firma.seznam.cz`

Abstract. In this paper we present a Bayesian approach to language identification of queries sent to an information retrieval system. The aim of the work is to identify both the language of a query as a whole and the language of particular words in the query. The method is evaluated on a test set of manually labelled queries. The evaluation shows that our method performs better than the Google Language Detect API and an implementation of the *n*-gram method on our testing set of queries.

Key words: language identification, query language, information retrieval

1 Introduction

Query language identification is one of the crucial issues that need to be solved in information retrieval systems such as Seznam.cz or Google. Based on the user's location and the query language, the system has to decide how to select resulting web pages, often written in various languages. Apart from identification of the language of a query as a whole, it may also be important to identify the language of particular terms of the query. This information is important, for instance, for correct morphological analysis of a given term.

Language identification of full documents is a well-explored area, however, queries are usually very short and rarely in the form of grammatically correct sentences. This makes the problem of query language identification more complicated. Moreover, since the query language identification is not so common problem as the language identification of a document, it is also less-explored. It turns out that some of the algorithms successfully applied to the language detection of text documents are not convenient for query language detection [5].

The task of language identification, sometimes called language detection or language recognition, is an instance of classification problems. The existing solutions can be divided into three main categories: approaches based on analysis of character *n*-grams [3,2], approaches that use dictionaries [9], and non-lexical approaches that use, for example, phoneme transcriptions [1] or an information about compression rate [6].

In the n-gram based approach, the idea is to compute relative frequencies of character n-grams for each language in the training dataset, and then use these statistics in order to detect language of previously unseen documents. The n-grams represent features in a vector space and similarity metrics such as the cosine measure can be used to find the most similar n-gram statistics to a processed document among statistics of training corpora of known languages.

Other algorithms, also based on the n-gram model, use Markov processes to determine the language of a text [8]. The idea behind the method is to detect language via the probabilities of observing certain character sequences. The probability of seeing particular character is dependent on a limited number of previous characters. The sequence of characters forms states of a Markov model.

The second widely-used approach to language detection is based on dictionaries of words rather than sequences of characters. In the dictionary method, for each language is created a set of language-specific words, where each word is associated with a relevance score. During the classification process, the processed document is compared against the trained dictionaries, and the language with highest scores wins. In comparison to the n-gram model, the dictionary based model requires tokenization and much more training examples, but for the language detection of short texts or queries is more appropriate.

2 Proposed Method

In our work we used the dictionary approach in a Bayesian framework. The training dataset consists of all documents indexed by the Seznam search engine[1] enriched with the information about language of particular documents. Permitted languages are $L = \{cz, en, sk, de, pl, fr, und\}$, where und represents an undefined language[2]. The language detection of web documents is performed using an n-gram classifier whose description is out of the scope of this paper. In order to model the probability $P(w|L)$ of word w being generated by language L we use the relative frequency of w in the corpus of language L smoothed by the Good-Turing Frequency Estimator [4].

For a given query $Q = \{w_1, w_2, \dots, w_N\}$, the goal of the classifier is to identify probabilities $P(L_Q|w_1, w_2, \dots, w_N)$ and $P(L_{w_i}|w_1, w_2, \dots, w_N)$ for each $i \in \{1, 2, \dots, N\}$, where L_Q stands for the language of the query as a whole and L_{w_i} stands for the language of word w_i. The probabilities are modelled using the Bayesian network shown in figure 1. The only observed variables are words $w_1, w_2, \dots, w_N$. To be able to infer the required probabilities we need to define prior probabilities $P(L_Q)$ for all languages and conditional probabilities $P(L_{w_i}|L_Q)$ for all combination of L_{w_i} and L_Q.

[1] http://search.seznam.cz

[2] A language that is not included in our set of supported languages.

The prior probabilities $P(L_Q)$ have been set according to the query language distribution in the search log of the Seznam search engine to the values given by table 1.

Table 1. Prior probabilities of query languages.

Language	cz	en	sk	de	pl	fr	und
Prior probability	61.9 %	19.1 %	3.0 %	1.7 %	0.7 %	0.6 %	13 %

Conditional probabilities $P(L_{w_i}|L_Q)$ of word language L_{w_i} being present in a query of language L_Q is hard to obtain. In order to get correct values we would need a great training corpus of queries with annotated both the language of query and languages of particular words in the query. We have avoided such demanding manual work by approximating the values using the following formula:

$$P(L_{w_i}|L_Q) = \begin{cases} \frac{9}{10} & \text{if } L_{w_i} = L_Q \\ \frac{1}{10} \times \frac{1}{|L|-1} & \text{else,} \end{cases} \tag{1}$$

where $|L|$ stands for the number of languages. In our case $|L| = 7$.

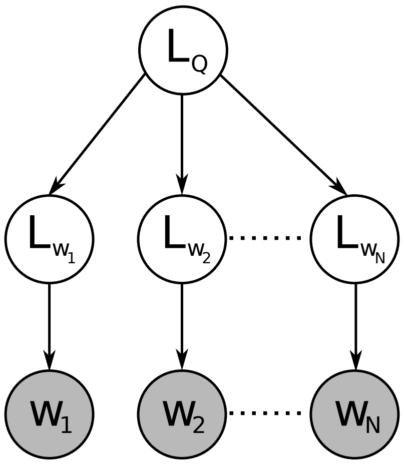


Fig. 1. Graphical model for query language identification.

From the Bayesian graphical model, we can express the probability of L_Q given query $Q = \{w_1, w_2, \ldots, w_N\}$ and the probability of L_{w_i} given query $Q = \{w_1, w_2, \ldots, w_N\}$ by

$$P(L_Q|w_1, w_2, \ldots, w_N) = \frac{P(L_Q)\prod_{i\in<1\ldots N>} P(w_i|L_Q)}{\sum_{L'_Q} P(L'_Q)\prod_{i\in<1\ldots N>} P(w_i|L'_Q)} \tag{2}$$

and

$$P(L_{w_i}|w_1, w_2, \ldots, w_N) = \sum_{L_Q} P(L_{w_i}|L_Q, w_i) P(L_Q|w_1, w_2, \ldots, w_N) \tag{3}$$

respectively, where

$$P(w_i|L_Q) = \sum_{L_w} P(w_i|L_w, L_Q)P(L_w|L_Q) = \sum_{L_w} P(w_i|L_w)P(L_w|L_Q) \tag{4}$$

and

$$P(L_{w_i}|L_Q, w_i) = \frac{P(w_i|L_{w_i})P(L_{w_i}|L_Q)P(L_Q)}{\sum_{L'_{w_i}} P(w_i|L'_{w_i})P(L'_{w_i}|L_Q)P(L_Q)} \tag{5}$$

3 Evaluation

To prove usability of our approach, we compared its results with two other implementations of language detection algorithms. The first one implements an *n*-gram based method. It takes into consideration 1 – 5 letter grams and is the result of the bachelor thesis by [7]. Web interface for the algorithm allows to detect more languages than our one, so we had to reduce the set of detectable languages to be fair. The other tool used to evaluate our method is the Google Language Detect API[3].

Both of these methods are intended to detect languages of documents, so they are not expected to perform so good on shorter examples like queries. We used small testing set of manually classified examples for query language detection. Number of examples was 300. The language distribution was chosen to represent distribution in real data and is listed in table 2. The Undefined language is completely missing because our reference classifiers do not support it. All queries came from Seznam.cz query log and were chosen randomly.

Table 2. Language distribution in the query test set.

Language	cz	en	sk	de	pl	fr
Examples %	65.7 %	18.0 %	6.0 %	5.3 %	2.7 %	2.3 %

In addition to the full test set, we also use its subsets for the evaluation purposes. Two of them were language based – Czech queries (197 samples) and English queries (54 samples). The rest of subsets were based on count of tokens of query[4]. The results are shown in table 3.

[3] http://code.google.com/apis/ajax/playground/#language_detect

[4] URLs in queries were split by stops, e.g. "www.wetter.de" was split into three tokens.

Table 3. Language identification accuracy on various test sets.

Set/Method	Bayesian	Google API	*n*-gram
All languages	91.67 %	61.33 %	51.67 %
Czech	91.37 %	50.76 %	46.70 %
English	92.59 %	75.93 %	52.26 %
1 token	79.31 %	36.21 %	39.66 %
2 tokens	95.80 %	61.54 %	47.55 %
3 or more tokens	93.00 %	76.00 %	64.00 %

All samples have been manually labelled with correct language, and after it classified using all three classifiers. Performance is expressed using the accuracy, i.e. number of correctly classified samples divided by total number of examples in the test set.

As we can see from the resulting scores, the hardest problem is the language identification for one-token queries. On two or more tokens all methods performed better. Worse performance of our Bayesian approach on "3 or more tokens" than on "2 tokens queries" category can be explained by the type of some of these queries. They more likely contain URLs with common words like *tchibo*, *mobile* or *ebay*, and country specific domain names.

Apart from the detection of query language, we also proposed a method for language detection of particular words in the query. To have at least some notion about the word language identification accuracy, we create a small test set consisting of two parts. The first part (150 queries) has been chosen randomly as in previous experiment and the other one (also 150 queries) has been taken from the suspected set of queries, created as a result of processing full query set when only queries with at least two languages were chosen. Before all, we picked a threshold 0.9 that defines the minimum probability of the word language for a word to be considered as comming from this language instead of being the same as the detected query language. This has to be done because our approach had some problems with combinations of some languages, especially the Slavic ones (Czech, Slovak, Polish).

With this adjustment we reached accuracy of 73.33% on our testing set. Most errors have been caused by the presence of URLs in queries and by the occurrence of very common words from one language in a query of another language in which the word occurs too, but with smaller frequency (e.g. "ou" in French means "or" and in Czech it is an abbreviation for "municipal office").

The performance on this task has not been compared to the other implementations as in the previous task because neither the *n*-gram implementation nor the Google Language Detect API does not support such functionality.

4 Conclusions

We have presented a method for automatic language identification of a query in a fulltext information retrieval system. The method supports detection of both language of a query as a whole and particular words in the query. In contrast to the most of available language detection tools, our method uses full Bayesian approach and is able to correctly classify even short text like queries. The method has been compared to the Google Language Detect API and the *n*-gram based tool by Josef Toman. Both tools have been outperformed in all test by the implementation of our Bayesian approach.

The method for identification of language of words also performs well, but to use it in a practical application, it needs some modifications. One possible approach is to learn the word language matrix on some relevant data instead of using just the simple function.

Acknowledgements This work has been partly supported by the Ministry of Education of CR within the Center of basic research LC536.

References

1. Kay Berkling, Takayuki Arai, and Etienne Barnard. Analysis of phoneme-based features for language identification. In *Proceedings of ICASSP-94*, pages 289–292, 1994.
2. William B. Cavnar and John M. Trenkle. N-gram based text categorization. In *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175, 1994.
3. Ted Dunning. Statistical identification of language. In *Technical Report MCCS-94-273*, volume 94. Computing Research Laboratory, 1994.
4. Irving John Good. The population frequencies of species and the estimation of population parameters. In *Biometrika*, volume 40, pages 237–264, 1953.
5. Thomas Gottron and Nedim Lipka. A comparison of language identification approaches on short, query-style texts. volume 5993, pages 611–614. Springer, 2010.
6. W. J. Teahan. Text classification and segmentation using minimum cross-entropy. In *RIAO'00*, volume 2, pages 943–961.
7. Josef Toman. Statistical language recognition, 2006.
8. Peter Vojtek and Mária Bieliková. Comparing natural language identification methods based on markov processes. In *Slovko, International Seminar on Computer Treatment of Slavic and East European Languages*, pages 271–282, 2007.
9. Radim Řehůřek and Milan Kolkus. Language identification on the web: Extending the dictionary method. In *Proceedings of Computational Linguistics and Intelligent Text Processing 10th International Conference CICLing 2009*, volume 5449, pages 357–368. Springer, 2009.

A Framework for Authorship Identification in the Internet Environment

Jan Rygl, Aleš Horák

NLP Centre
Faculty of Informatics
Masaryk University
Botanická 68a, 602 00 Brno, Czech Republic
{xrygl,hales}@fi.muni.cz

Abstract. Misuse of anonymous online communication for illegal purposes has become a major concern [2,12]. In this paper, we present a framework named *ART* (*Authorship Recognition Tool*), that is designed to minimize manual procedures and maximize the efficiency of authorship identification based on the content of Internet electronic documents. The framework covers the phases of document retrieval and database document management. *ART* provides implementations of efficient authorship identification algorithm and authorship similarity algorithm including the possibility to obtain extra data for learning and tests. The framework also determines whether or not different author's identities are interlinked.
The authorship is analysed by machine learning and natural language processing methods. Technical information such as IP address is considered only as an optional attribute for the machine learning because it can be easily forged or devalued if the author communicates from public places or through proxy servers. The decision which algorithm to use for determining the authorship of an anonymous document depends on the documents' language.

Key words: authorship identification, authorship similarity

1 Introduction

Recent extremist actions in the "civilized" parts of the world call for more efficient techniques for crime prevention. Relatively new medium for the communication of extremist groups and even individuals (when expressing their thoughts to the public) is the public Internet. This medium offers to the authors the possibility to publish very fast to a large audience with techniques available to remain anonymous. One way to defend the public against this anonymity is based on computational linguistic methods of forensic authorship analysis [1,9].

In the following text, we describe the design and implementation of a new framework, named *ART* (*Authorship Recognition Tool*), for effective solution of authorship identification in the online environment. The main task of the *ART* framework can be defined as:

Problem 1. Let us have an anonymous document *D*. The task is to identify the author of the document and find all other documents of the same author in the public Internet.

What most of currently prevailing techniques have in common is that they work with electronic communication from the crime, terrorism and extremism environment [5,10,11,12]. These studies are aimed on the identification methods, hence online messages are collected manually [10,11].

In this work a new approach is presented. The proposed framework is unique because it does not focus on various techniques of determining the authorship of documents, but it analyses the problem and offers solutions to many technical difficulties related to the authorship attribution in the Internet environment.

2 Framework for the Online Authorship Identification

The given problem, as we have specified in the introduction, is very complex, therefore it is necessary to decompose it into smaller, clearly defined tasks. Firstly, the scenario of a fully functional system is described. It works in 4 phases (as shown in Table 1):

1. In the first phase, the set of possible authors is restricted according to manually annotated domains' themes – if an unknown document from a Czech extremist website is analysed, only authors from this and other Czech extremist websites are taken into account. This step is important because the accuracy of authorship identification decreases with growing number of potential authors. Filtering out improbable authors increases the success rate of authorship detection algorithms at the cost of manual category tagging of selected web domains.
2. After narrowing the set of possible authors, each author's profile is compared to the unknown document by machine learning (comparison of several algorithms is available in [7]) and natural language processing methods (e.g. delta score [3], punctuation statistics [4,6]). The unknown document is associated with the author whose profile is the most similar to the document.
3. In the next phase, all web domains related to the theme selected in the first step are analysed. Their content is downloaded to the database that contains pairs of preprocessed documents and an author identity (a pseudonym or technical parameters can serve as the identity). Acquired documents are divided into groups according to their authorship. After the document clustering, groups are unified if their similarity exceeds a pre-set limit (their author published under more identities).
4. In the last phase, all documents of the guessed author are returned and new information are stored into the database.

Table 1. Process of the Authorship Identification

action	description
`Extract document's theme`	*(manually)*
`Extract domains' themes`	*(manually)*
`Select domains with related themes`	*section 3*
`Compare known authors to the document`	*section 4*
`Select the most similar author`	
`Analyze documents from selected domains`	*section 5*
`Cluster documents according to their authorship`	*section 6*
`Return the cluster of the selected author`	

Each phase is described in more detail in the following sections, including the semi-automatic document downloading from Internet.

3 Restricting only to Domains Related to the Document's Topic

Efforts to minimize the number of possible authors arise from the fact that current algorithms do not achieve a high success rate for difficult problems which include:

– High number of possible authors: Hundreds to thousands authors make detection unreliable. A baseline is defined as $\frac{1}{|authors|}$, therefore, improving the baseline significantly is not equal to achieving a high accuracy.
– Comparing different styles of documents: Letters, discussion posts, blogs and articles use different key words (addressing, signature), stylistics (formal, informal), etc. Mixing such documents has negative impact on algorithms using word and *n*-gram frequencies.
– Comparing documents of different topics: For example, difference between political articles and personal correspondence is significant, therefore two political articles from different authors can be more similar to each other than an e-mail and a political article, both written by one author.
– Documents do not contain enough text: This particularly applies to discussions and e-mails which can contain only several sentences.

The disadvantage of filtering domains out is that the unknown document's author may not be among the authors intended to compare. But the unknown document can never be compared to all authors – despite the fact that people publish in the Internet under their real identities (advertisements, school assignments, social networks), it is impossible to access all these data (to enable web domains' parsing requires automatic approach and the total number of authors in the database would still be too high).

4 Authorship Identification Algorithms

This section describes the text preprocessing in the *ART* framework. The language of documents is detected and the text is tokenized, morphologically and syntactically annotated and disambiguated. These processes are language dependent, therefore, modules for each language are needed.

The main goal of the proposed framework is to support collecting data from Internet and to solve technical difficulties typical for the online environment, hence there is no restriction on the authorship identification algorithms.

Since the time of Mosteller and Wallace [8], a substantial amount of new research was done in the topic of authorship identification taking advantage of new findings in areas such as machine learning, information retrieval, and natural language processing. Current studies recommend to use machine learning methods that are effective for large data. The new contribution of the *ART* framework is that the process of collecting data can be accelerated and machine learning methods can work with more data to achieve better accuracy.

5 Retrieving Documents from the Internet

An intelligent exploitation of the documents retrieved from Internet needs a description of the format of the stored document meta-data. Since it is tedious to manually extract the structure of web pages to be able to download information about documents and authors, we propose a new, semi-automatic approach. It consists of the following 4 steps:

1. Firstly, a domain from which documents are going to be extracted is selected and visited by an operator. The operator manually registers to the domain using data describing her institution. Then it is necessary to submit a small number of documents $d_1, \ldots, d_k$ (e.g. discussion posts, blogs) while logged as the registered user.
2. In the second step, a crawler[1] is used to download web pages $P_1, P_2, \ldots$ in the domain until all pages containing information about documents $d_1, \ldots, d_k$ are found: $P_{d_1}, \ldots, P_{d_k}$.
3. In the third step, the HTML tree structure of the selected pages is detected by a HTML parser. Then minimal sequences of HTML tags are extracted to describe each attribute of the documents $d_1, \ldots, d_k$ using local search heuristics. It is important that each sequence describes the same information in all downloaded web pages, e.g. the title sequence defines a path to information about the title for every document. The structure of the domain is stored into the database as generated sequences of HTML tags.
4. Finally, all documents from the domain are downloaded by the crawler and processed. With the knowledge of the web page's structure, only data relevant for the authorship identification are collected and saved in the database. The algorithm is summarized in Table 2.

[1] a tool for web page analysis and download including the page links

In any future attempt to retrieve documents from already processed domain, $P_{d_1}, \ldots, P_{d_k}$ are downloaded again and their content is compared to the saved data in the database $(d_1, \ldots, d_k)$. If the content differs, either documents were edited (which is unlikely because documents were created by the operator), or the structure of pages in the domain was changed. Therefore, in this case all 4 steps are executed again. Otherwise, only new pages are processed.

Table 2. Domain structure identification

action	example

`Select domain`
`and register as author`
`and submit article`
	$D = www.domain.com$
	(Name, E-mail) $\rightarrow D$
	(Title, Text, Name) $\rightarrow D$

`Download domain texts`	*documents* $t_1, \ldots, t_n \in D$
`Search inserted document`	$t_k =$ *(Title, E-mail, Name)*
`Extract structure of document`	$Title_k$: *body/div[@content]/h3* $Text_k$: *body/div[@content]/p* $Author_k$: *head/title*
`Process downloaded documents`	$Title_k$: *Introduction post* $Text_k$: *Text about web page's topic...* $Author_k$: *NLP Center*

6 Document Clustering According to the Authorship

Clustering of anonymous documents according to the authorship is very difficult. There even do not exist any recommended metrics for measuring the quality of a particular clustering in the authorship identification problem. We conducted some experiments but the results' accuracy was low. Although similarity of two documents can be compared with relatively high accuracy, for creation of large clusters many comparisons are made and even marginal errors decrease total accuracy significantly.

Therefore anonymous documents are not clustered and only documents signed by authors are put together. In order to adapt to data from an online environment for which identification of authors are not unique, an operation merging two clusters is allowed. It is very important to process data from different domains because the author's accounts may vary. Either it is a cosmetic change of identity (e.g. size of letters, leave out one word), or the author uses a completely different pseudonym.

Whenever a new author is inserted into the database, he or she is compared to each known author. If two authors' documents differ only marginally, their identities are connected. On the contrary, authors with same identities from different web domains are not linked automatically, their similarity has to exceed a specified limit that is more tolerant than in the case of two different identities. This is necessary because many pseudonyms and names overlap.

Despite the fact that the operation is time consuming, it is affordable because it is sufficient to apply it only to authors of documents with a similar theme and each author is processed only once.

7 Experimental results

Six evaluations were conducted to test the hypothesis that accuracy is improved by filtering out improbable authors. *ART* framework was used to automatically detect the structure of an extremist website *WM*[2] and to download all discussion posts (messages are mostly long 1 to 5 sentences).

Experiments are divided into two parts: In the part A, only documents of authors who wrote at least four documents are selected as test documents. At minimum three documents are used to create authors characteristics to which unknown document is compared. In the part B, author characteristic are generated from an arbitrary number of documents, therefore, the authorship recognition problem B is more difficult. Each evaluation is examined in Table 3 for part A and in Table 4 for part B. Each part consists of 3 scenarios:

- In the first case, documents (messages) were selected according to their extremist and racism theme. In the part A, authors who wrote less than three comments were filtered out because it is difficult to extract author's characteristics from very short texts. This scenario achieved the highest accuracy (22% and 6% documents were assigned to their author correctly in parts A and B).
- In the second scenario, data used in the first case were extended by small number of documents extracted from another website *I*[3]. Test documents remained same. The accuracy was lowered significantly to 3.1 and 0.7%.
- In the last scenario, data were further extended by big number of documents extracted from the website *I*. The accuracy was still 3.1 and 0.7%.

Results indicate that adding documents of different topic can substantially decrease the accuracy. If comparisons are not made under the same conditions (documents have different lengths, key words, levels of formality, ...), authorship recognition algorithms perform worse. Furthermore, despite the fact that increasing number of authors' documents by adding messages of the same theme decrease the accuracy to a lesser extent than adding messages of different topic the performance decrease is still significant.

8 Conclusions

The *ART* framework extends previous works for authorship identification. The process of determining authorship is unchanged, but the manual documents obtaining is

[2] An extremist website `http://www.white-media.info/` was selected because it was mentioned recently in newspapers.

[3] Authors of discussion posts from the news server `http://idnes.cz` are added to the group of possible authors.

Table 3. Filtering domain experiment

Scenario 1A: Authors wrote at least 3 documents about the selected theme			
	Filtered topic	Other Topic	Total
Number of learn documents	144	–	144
Number of potential authors	8	–	8
Number of test documents	32	–	32
Accuracy	21.9%		

Scenario 2A: All documents are about the selected theme			
	Filtered topic	Other Topic	Total
Number of learn documents	189	98	287
Number of potential authors	8	58	66
Number of test documents	32	–	32
Accuracy	3.1%		

Scenario 3A: All documents are about the selected theme			
	Filtered topic	Other Topic	Total
Number of learn documents	189	946	1135
Number of potential authors	8	228	236
Number of test documents	32	–	32
Accuracy	3.1%		

replaced by more effective intelligent semi-automatic data retrieval, that has a positive impact on the quality of machine learning models used to detect an authorship. The proposed framework improves previous systems, because documents can be downloaded and processed from the web domain and regular updates are supported, therefore the authors' database is actual. In case of change of a web domain structure, the system is able to adapt to the change and retrieve documents without manual intervention.

Acknowledgments This work has been partly supported by the Ministry of the Interior of CR within the project VF20102014003 and by the Ministry of Education of CR within the Center of basic research LC536

References

1. A. Abbasi and H. Chen. Applying authorship analysis to extremist-group web forum messages. *Intelligent Systems, IEEE*, 20(5):67–75, 2005.
2. Ahmed Abbasi and Hsinchun Chen. Applying authorship analysis to extremist-group web forum messages. *IEEE Intelligent Systems*, 20(5):67–75, 2005.
3. John Burrows. Delta': a measure of stylistic authorship 1. *Literary and Linguistic Computing*, 17(3):267–287, 2002.
4. Chaski,C. E. Who's At The Keyboard? Authorship Attribution in Digital Evidence Investigations. *International Journal of Digital Evidence*, 4(1):1–13, 2005.

Table 4. Filtering domain experiment

Scenario 1B: All documents are about the selected theme			
	Filtered topic	Other Topic	Total
Number of learn documents	189	–	189
Number of potential authors	37	–	37
Number of test documents	134	–	134
Accuracy	6%		

Scenario 2B: Small number of documents from another domain is added			
	Filtered topic	Other Topic	Total
Number of learn documents	189	98	287
Number of potential authors	37	58	95
Number of test documents	134	–	134
Accuracy	0.71%		

Scenario 3B: Many documents from another domain are added			
	Filtered topic	Other Topic	Total
Number of learn documents	189	946	1135
Number of potential authors	37	228	365
Number of test documents	134	–	134
Accuracy	0.7%		

5. Hsinchun Chen. Exploring extremism and terrorism on the web: the dark web project. In *Proceedings of the 2007 Pacific Asia conference on Intelligence and security informatics*, PAISI'07, pages 1–20, Berlin, Heidelberg, 2007. Springer-Verlag.
6. Jakubíček, Miloš - Horák, Aleš. Punctuation Detection with Full Syntactic Parsing. *Research in Computing Science, Special issue: Natural Language Processing and its Applications*, 46:335–343, 2010.
7. Koppel, Moshe - Schler, Jonathan - Argamon, Shlomo. Computational methods in authorship attribution. *J. Am. Soc. Inf. Sci. Technol.*, 60:9–26, January 2009.
8. F. Mosteller and D. L. Wallace. *Inference and Disputed Authorship: The Federalist*. Addison-Wesley, 1964.
9. E. Stamatatos. A survey of modern authorship attribution methods. *Journal of the American Society for information Science and Technology*, 60(3):538–556, 2009.
10. Sresha Yadav and Smita Jha. A framework for authorship identification of questioned documents: Forensic and linguistic convergence by. *MJAL*, 3(1):1–7, 2001.
11. Rong Zheng, Jiexun Li, Hsinchun Chen, and Zan Huang. A framework for authorship identification of online messages: Writing-style features and classification techniques. *Journal of the American Society for Information Science and Technology*, 57(3):378–393, 2006.
12. Rong Zheng, Yi Qin, Zan Huang, and Hsinchun Chen. Authorship analysis in cybercrime investigation. In *Proceedings of the 1st NSF/NIJ conference on Intelligence and security informatics*, ISI'03, pages 59–73, Berlin, Heidelberg, 2003. Springer-Verlag.

chared: Character Encoding Detection with a Known Language

Jan Pomikálek[1] and Vít Suchomel[2]

[1] Lexical Computing Ltd.
73 Freshfield Road, Brighton, UK
jan.pomikalek@sketchengine.co.uk

[2] Natural Language Processing Centre
Faculty of Informatics, Masaryk University
Botanická 68a, 602 00 Brno, Czech Republic
xsuchom2@fi.muni.cz

Abstract. chared is a system which can detect character encoding of a text document provided the language of the document is known. The system supports a wide range of languages and the most commonly used character encodings. We explain the details of the algorithm, describe the process of creating models for various languages and present results of an evaluation on a collection of Web pages.

Key words: character encoding, character encoding detection, charset, Unicode

1 Introduction

When creating monolingual text corpora from the Web, one of the problems which need to be addressed is character encoding detection. Web pages usually contain the information about the used character encoding in the meta tags. However, this information is not always available and not always correct, and in general cannot be relied upon. Nevertheless, the character encoding can usually be reliably guessed from the textual contents of a Web page.

The problem of character encoding detection can be tackled at two different levels. First, no assumption is made about the language of the input text. In this case both the character encoding and the language of the input may be guessed at the same time. Second, the language of the input is known and only the character encoding is detected. Existing systems which fall into the former category include for instance TextCat[1] and chardet[2]. A notable representative of the latter category is Enca[3].

The disadvantage of the existing systems is that they support only a limited number of languages and encodings. The system we designed, on the other

[1] http://www.let.rug.nl/vannoord/TextCat/
[2] http://pypi.python.org/pypi/chardet
[3] http://gitorious.org/enca

hand, covers a wide range of languages and frequently used encodings for each language.

2 chared

Our method assumes that the language of the input is known. For instance, when building monolingual corpora this is always guaranteed.[4] By narrowing the problem down to a single language, higher accuracy can be achieved.

2.1 Algorithm

The system works as follow. First, a model is created for each supported language. This requires two kinds of input:

1. The list of character encodings used for given language
2. A sample text in the language and in a known character encoding

We convert the sample text to all encodings from the list, creating *N* different files (where *N* is the number of encodings). Then, for each of the files we build a frequency list of all consequent triples of bytes. The *N* frequency lists (vectors) are then compared and the items (triples) with the same frequency in all the *N* vectors are discarded.

When detecting a character encoding of a document, its frequency vector is built and a scalar product is computed with each of the *N* model vectors. The character encoding associated with the model vector which produces the highest scalar product is returned as the encoding detected for the input document.

The advantage of building model vectors from the same data (only converted to *N* different encodings) is that the differences among the model vectors are only where the character encodings differ. If for instance all the *N* encodings are ascii compatible, then any triple of bytes containing only ascii characters will have the same frequency in all model vectors. Thus, this triple would add the same value to the scalar product with a tested document (vector) for all models. As such it cannot affect the order of the final scalar products (and the result of the classification) and thus it can be safely removed from all the model vectors. This both reduces the size of the vectors and speeds up the classification. Alternative approach which builds the models on *N* different texts in *N* different encodings prevents such pruning. Apart from that, similarity of the tested document to the texts used for training the models (regardless of their

[4] In fact, in a web corpus processing pipeline, character encoding detection has to be performed before language detection. Thus, the assumption that the input of the character encoding algorithm is in the language in question is not always valid. However, this is not a problem, since all texts in other languages will be refused by the language filter in the next step and it therefore does not matter whether their character encoding has been detected correctly or not.

encoding) may bias the results of the encoding detection. This bias would be the more severe the more ascii characters (or more generally, the more characters with the same representation in all the encodings in question) are used in the training texts (e.g. very severe for English).

The reason for using frequencies of tuples (triples in our case) of characters rather than single characters is that some bytes represent multiple different commonly used characters in different encodings. Such bytes may cause errors in the encoding detection. This is best illustrated on an example. The byte `a9` represents the character © (copyright symbol) in windows-1250 and the character *Š* (capital s with caron) in iso-8859-2. Both encodings are used for Czech language. The *Š* character is much more frequent in Czech texts than the copyright symbol. Thus, a windows-1250 text containing a copyright symbol may be easily misclassified as iso-8859-2 since the `a9` byte will have a higher frequency in the iso-8859-2 model. Since the two encodings are very similar, this kind of classification error is fairly likely, especially for short texts.

Building models on tuples of characters rather than on single characters helps to overcome this problem. The copyright symbol will be typically followed by a space whereas the *Š* character will be typically followed by another letter in a Czech text. Therefore, two bytes `a920` (© followed by a space in windows-1250) will certainly have a higher frequency in a windows-1250 model than in a iso-8859-2 model (for Czech). In our experiments, using triples of characters resulted in a higher accuracy of the encoding detection than using only pairs of characters.

2.2 Building models

In order to create models for our system, we collected a set of about 1000 Web pages for each language. This was done using Corpus Factory tools. [1] We identified the encoding of each page by using the information in the meta tags. Though we are well aware that this information is not always correct we believe the errors are so rare that they cause only an insignificant bias in the built models. We simply discarded all pages for which we have not been able to determine the character encoding from the meta tags.

First, for each language, we computed the relative frequencies of the encodings used in the Web pages and accepted those with a relative frequency above 0.5 % as the encodings commonly used for the language. We then created a model vector for each of these encodings by converting the Web pages to the encoding and building the frequency vector as described above.

3 Evaluation

To evaluate the system, we performed a 5-fold cross-validation on the training data. Table 1 contains results for selected languages.

The average accuracy is weighted by the frequency of occurrence of the encodings. The rationale of the weighting is that it is more important that the

	Czech		English		German		Greek		Italian		Norwegian	
	freq	accuracy	freq	accuracy	freq	accuracy	freq	accuracy	freq	accuracy	freq	accuracy
utf-8	60.2 %	100.0 %	56.9 %	95.8 %	54.6 %	100.0 %	68.5 %	100.0 %	54.2 %	100.0 %	63.0 %	100.0 %
windows-1250	32.2 %	100.0 %	0.3 %	n/a	0.1 %	n/a	0.2 %	n/a	0.0 %	n/a	0.1 %	n/a
windows-1252	0.4 %	n/a	9.4 %	97.5 %	6.5 %	97.3 %	3.1 %	75.8 %	7.1 %	95.7 %	7.0 %	97.4 %
windows-1253	0.0 %	n/a	0.0 %	n/a	0.0 %	n/a	14.3 %	99.3 %	0.0 %	n/a	0.0 %	n/a
iso-8859-1	1.0 %	89.5 %	32.8 %	90.9 %	37.1 %	85.8 %	1.7 %	71.2 %	37.9 %	85.1 %	29.3 %	88.2 %
iso-8859-2	6.0 %	99.6 %	0.0 %	n/a	0.1 %	n/a	0.0 %	n/a	0.1 %	n/a	0.1 %	n/a
iso-8859-7	0.0 %	n/a	0.0 %	n/a	0.0 %	n/a	12.0 %	97.2 %	0.0 %	n/a	0.0 %	n/a
iso-8859-15	0.0 %	n/a	0.0 %	n/a	1.2 %	85.6 %	0.0 %	n/a	0.0 %	n/a	0.4 %	n/a
training docs		801		668		773		879		771		740
w. avg accuracy		99.2 %		93.5 %		93.7 %		97.9 %		93.3 %		95.7 %

Table 1. Character encoding detection accuracy for selected languages.

algorithm correctly identifies the frequently used encodings than the rarely used ones.

Note that the results may be harmed by incorrect "annotation" of the data. We manually inspected some of the misclassified pages and often found out that the algorithm detected the encoding of the page correctly, but the character encoding specified in the meta tags was wrong. We therefore assume that the accuracy of our system is higher than indicated by the numbers in the table.

4 Conclusion

We have presented chared, a system for detecting character encoding for documents of which we know the language. Though our evaluation is only preliminary and lacks comparison with other systems, we are convinced that our approach works better than the general approach to the problem which makes no assumptions about the language of the input. The main advantage of our system compared to other available similar programs is a support for a wide range of languages (currently 51). chared is released under the BSD open source licence and is available for download from Google code[5].

Acknowledgements The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 248307 (PRESEMT project) and from the Czech Science Foundation under the project P401/10/0792.

References

1. Kilgarriff, A., Reddy, S., Pomikálek, J., PVS, A.: A corpus factory for many languages. Proc. LREC, Malta (2010)

[5] `http://code.google.com/p/chared/`

Words' Burstiness in Language Models

Pavel Rychlý

NLP Centre
Faculty of Informatics, Masaryk University
Botanická 68a, 602 00 Brno, Czech Republic
pary@fi.muni.cz

Abstract. Good estimation of the probability of a single word is a crucial part of language modelling. It is based on raw frequency of the word in a training corpus. Such computation is a good estimation for functional words and most very frequent words, but it is a poor estimation for most content words because of words' tendency to occur in clusters. This paper provides an analysis of words' burstiness and propose a new unigram language model which handles bursty words much better. The evaluation of the model on two data sets shows consistently lower perplexity and cross-entropy in the new model.

Key words: language models, words' probability, burstiness

1 Introduction

Language modelling is used in tagging, information retrieval, word sense disambiguation and many other natural language processing applications. A language model assigns a probability to a sequence of n words $P(w_1, w_2, \ldots, w_n)$. The simplest language model is the unigram one, it is based on probabilities of individual words without considering any context. Better language models are based on n-grams and/or other features of words (part of speech, lemma) but in all models the unigram probability of a single word $P(w)$ is very important.

Computation of words' probabilities are based on their estimation from a training corpus. There is an assumption that the probability is close to $P(w) = \frac{C_w}{N}$, where C_w is number occurrences of the word w in the training corpus of total size N. Advance techniques use smoothing of raw frequencies but they are also based on frequencies itself.

2 Burstiness of Words

The distribution of a word from a unigram model is roughly evenly distributed events (words), it could be modelled by a repeated Bernoulli trial with probability $P(w)$. This works well for most functional words, but most content words has very different distribution. Examples of distributions are displayed in Figure 1. The top one is a binomial distribution created by a random number gen-

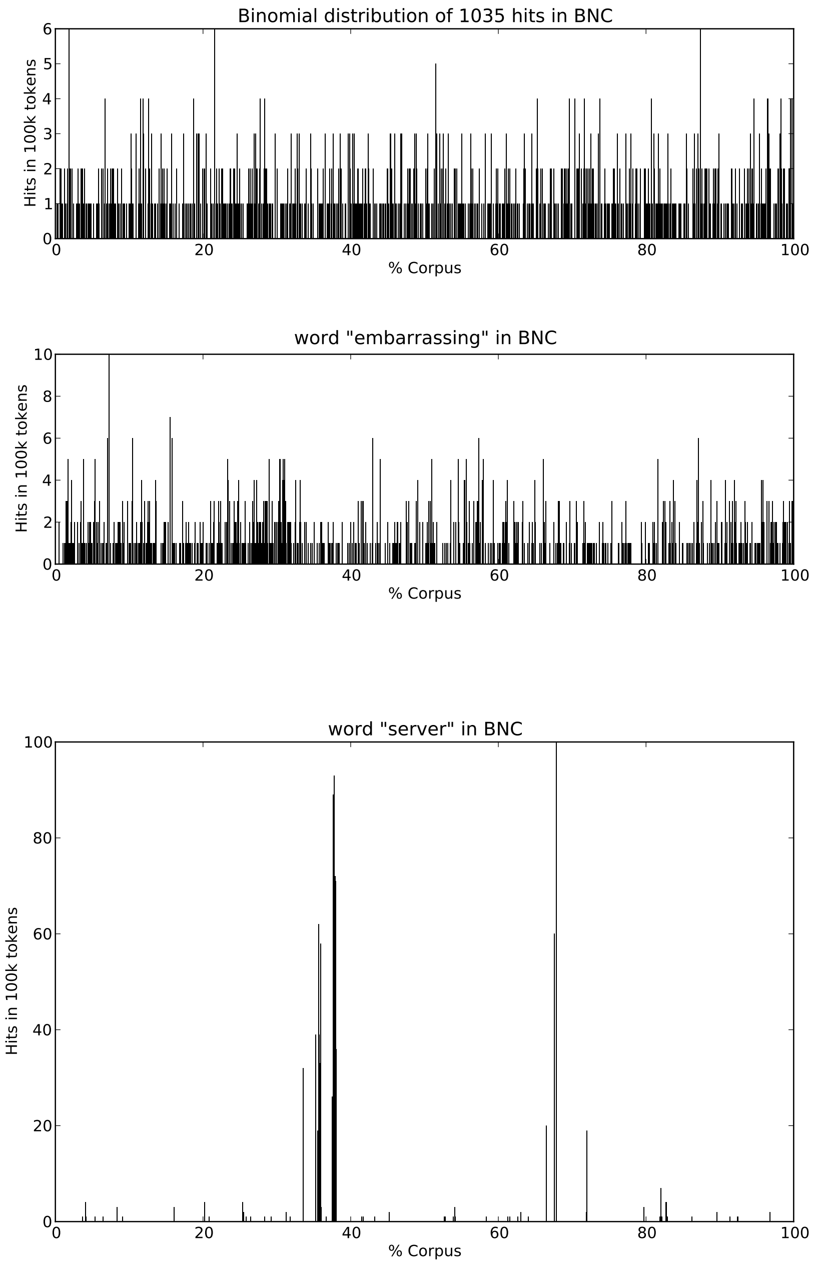


Fig. 1. Number of hits in a continuous part o 100k tokens. All plots contains 1035 hits in the British National Corpus. From top: Binomial, word *embarrassing*, word *server*.

erator, the following ones displays hits of words *embarrassing* and *server* in BNC [1]. All three has the same frequency: 1035 hits in the whole corpus. In the last plot, we can see bursts of occurrences, for each occurrence there is a cluster of several or many other occurrences of the same word.

The burstiness of a word does not depend on its raw frequency. The Figure 2 displays distributions of several words with the same total frequency (1035). In general, names and terms has many hits in one cluster. On the other hand distribution of general words is more even. Another view provides Figure3. It plots two frequency distributions of 1000 words in BNC. The upper one (with '+' marks) is formed from words with the highest C_w/ARF_w ratio (< 1.726), these words has the biggest burstiness. The lower one (with '.' marks) is formed from words with the lowest C_w/ARF_w ratio (> 29.5), these words are evenly distributed in the corpus. The plot is in log-log scale. We can see that the average raw frequency of bursty words is lower but both lines fits Zipf's law very well – both sets contains words from the whole frequency range, like all words in the corpus.

The initial assumption that from a bigger training corpus we have better estimation of words' probabilities is not valid for most words [2]. On the other hand, the relative frequency of the number of clusters is quite stable and relative frequency of the given word in its cluster is also stable.

In many applications a document frequency (number of documents containing the given word at least ones) instead of raw frequency is used to estimate $P(w)$. Especially in cases where all documents has the same size, it works well [3]. The disadvantage of such approach is the impossibility to distinguish frequencies of functional words because they all have document frequency equal to the total number of documents. Hence, it could be used only for applications in the field of information retrieval, where we want to model occurrences of clusters and not individual words.

3 Bursting Language Model

This paper propose a bursting language model in which the probability of clusters and probability of words within a cluster are separated. The estimation of the word's cluster probability is based on Average Reduced Frequency (ARF) [4]. It is defined by the following formula:

$$ARF_w = \frac{1}{v}\sum_{i=1}^{C_w} \min\{d_i, v\}$$

where $v = N/C_w$, N is the size of the corpus, and d_i is the distance between two consecutive occurrences of the given word in the corpus, hence,

$$\sum_{i=1}^{C_w} d_i = N.$$

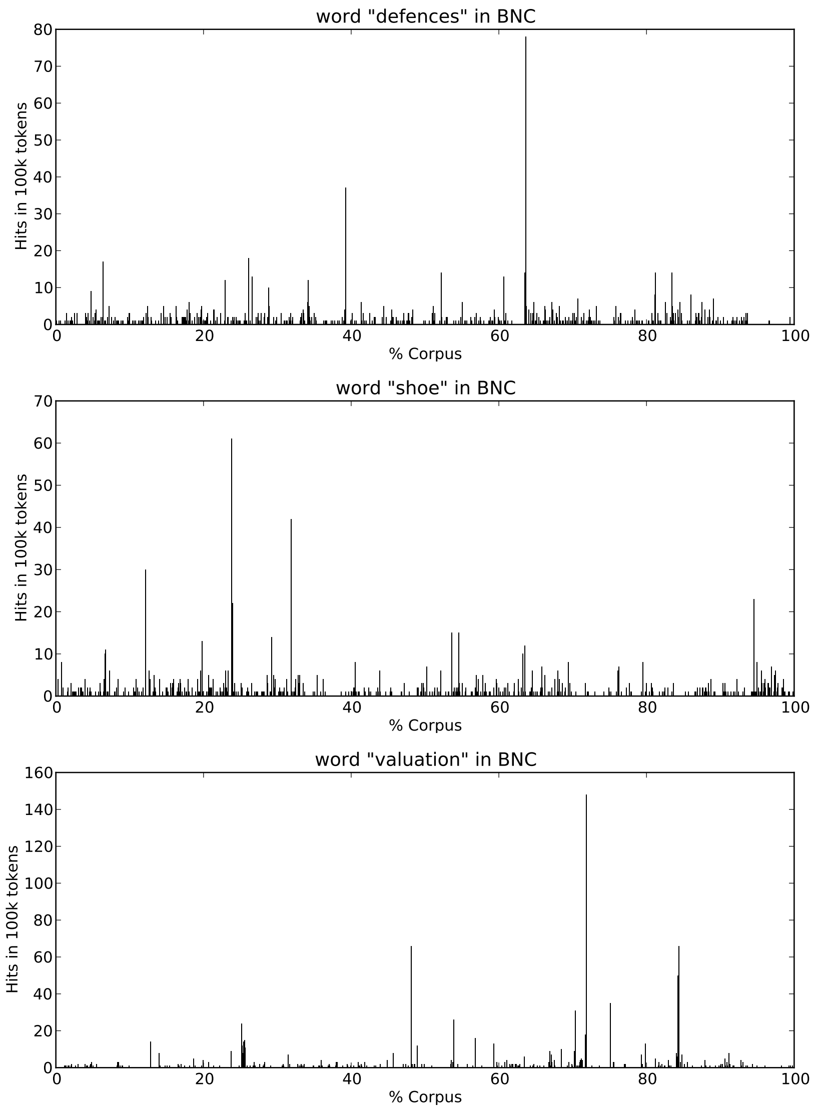


Fig. 2. Number of hits in a continuous part o 100k tokens. All plots contains 1035 hits in the BNC. Words from top: *shoe*, *defences*, *valuation*.

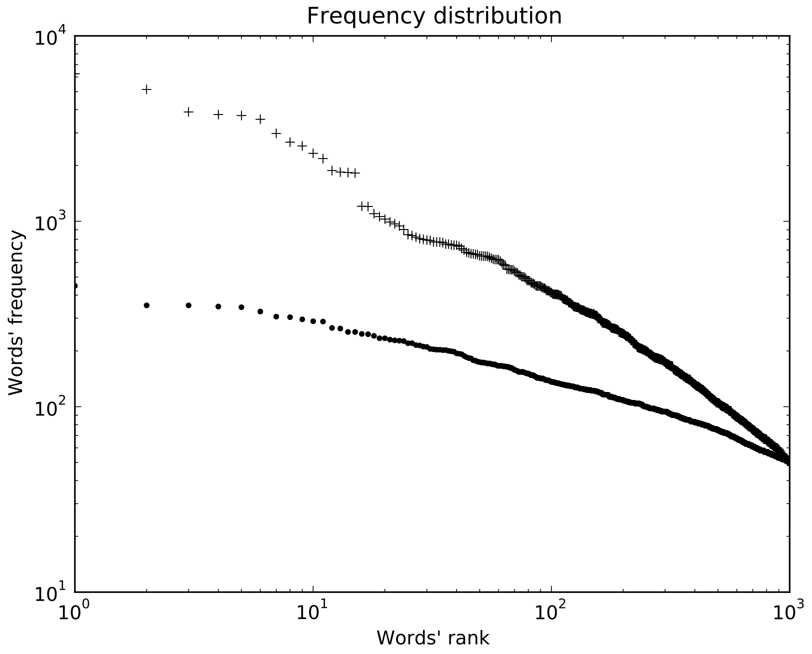


Fig. 3. Frequency distribution of 1000 words with biggest (upper +) and lowest (lower .) $freq/ARF$ ratio in BNC. Both lines fits the Zipf's law very well.

We can see that the definition of ARF does not depend on documents, it could handle documents of different sizes.

A word's cluster starts at the first occurrence of the given word. Up to that point the probability of the word is estimated by the probability of the word's cluster. From that point on, the probability of the word is estimated by the probability of the word within the cluster. Cluster probability is higher (for bursty words much higher) then within-cluster probability. Such elevated probability of the given word means slightly lower probability of all other words, at the start of each cluster probabilities of all words have to be adjusted to match the basic feature that sum of probabilities of all words is 1. This probability elevation longs only for a word's cluster size, after given number of tokens, the probability of the word drops down back to the cluster probability and all probabilities have to be adjusted again.

For each word we define the following parameters:

$$P(cluster_w) = \frac{ARF_w}{\hat{N}}$$

$$P(wincluster) = \frac{C_w^2}{ARF_w \hat{N}}$$

$$clustersize_w = \frac{N}{10C_w}$$

where $\hat{N}$ is the sum of all current counts, that is ARF_w if w is in its elevated cluster and C_w^2/ARF_w otherwise.

4 Evaluation

An evaluation of language models could be done by computing cross-entropy or perplexity on an existing text. During an experiment, language model is trained (probabilities are estimated) on a training part of the evaluation data a cross-entropy is computed on an evaluation part (which could be much smaller).

The evaluation of the proposed model was done on two data sets:

1. training on BNC, cross-entropy on the corpus Susanne, [5].
2. 10-fold cross-validation on Word Street Journal corpus (WSJ) [6]. The corpus was divided into 10 parts, for each part two corpora was created, one containing only the given part and second containing the rest of the corpus. The bigger corpus was used for training, the smaller one for evaluation.

In both cases, all computation was done on word forms, Manatee system [7] was used for computing raw frequencies and ARF for all words.

The proposed language model was compared with the simple unigram model. The results on WSJ are listed in Table 1. The overall results are summarised in Table 2.

Table 1. Ten-fold cross-validation on WSJ

unigram model	10.15	10.13	10.17	10.12	10.17	10.19	10.18	10.16	10.17	10.22
bursting model	9.96	9.94	9.98	9.94	9.98	10.00	9.99	9.97	9.98	10.02
difference	0.19	0.19	0.19	0.19	0.19	0.19	0.19	0.19	0.19	0.20

Table 2. Evaluation results on both data sets

	BNC	WSJ
Cross-entropy of unigram model	10.71	10.17
Cross-entropy of bursting model	10.39	9.97
Perplexity of unigram model	1676	1149
Perplexity of bursting model	1337	1006

We can see that using bursting model instead of unigram one has stable results with significantly lower cross-entropy and perplexity.

5 Conclusion

The proposed bursting language model provides better estimation of words' probabilities. The perplexity on evaluation data is about 20% lower with the proposed model compared to a standard unigram model.

Acknowledgements This work has been partly supported by the Czech Science Foundation under the projects P401/10/0792 and 407/07/0679.

References

1. Aston, G., Burnard, L.: The BNC handbook: exploring the British National Corpus with SARA. Edinburgh University Press (1998)
2. Curran, J., Osborne, M.: A very very large corpus doesn't always yield reliable estimates. In: proceedings of the 6th conference on Natural language learning-Volume 20, Association for Computational Linguistics (2002) 1–6
3. Church, K., Gale, W.: Poisson mixtures. Natural Language Engineering **1**(2) (1995) 163–190
4. Savický, P., Hlaváčová, J.: Measures of word commonness. Journal of Quantitative Linguistics **9**(3) (2002) 215–231
5. Sampson, G.: English for the Computer: The SUSANNE Corpus and Analytic Scheme. Clarendon Press (1995)
6. Eugene Charniak, e.a.: Bllip 1987-89 wsj corpus release 1. (2000)
7. Rychlý, P.: Manatee/Bonito–A Modular Corpus Manager. In: 1st Workshop on Recent Advances in Slavonic Natural Language Processing, Masaryk University (2007) 65–70

Author Index

RASLAN 2011

Fifth Workshop on Recent Advances in Slavonic Natural Language Processing

Editors: Aleš Horák, Pavel Rychlý
Typesetting: Adam Rambousek
Cover design: Petr Sojka

Printed and published by Tribun EU s. r. o.
Cejl 32, 602 00 Brno, Czech Republic

First edition at Tribun EU
Brno 2011

ISBN 978-80-263-0077-9

www.librix.eu