A FLEXIBLE FRAMEWORK FOR DEVELOPMENT OF ANNOTATED CORPORA

Svetla Koeva, Borislav Rizov, Svetlozara Leseva

Abstract: The paper describes the largely language-independent and task-independent framework for developing training corpora. The application utilises a corpus (arranged for the specific type of annotation) and various large coverage linguistic resources (linguistic networks, dictionaries, etc.) which provide certain linguistic information associated with the language units in the corpus. This involves association of certain units with certain set of linguistic features (tagset) in the linguistic resource database. The application is currently directed towards the creation of sense disambiguation training corpus derived from the Brown-designed Bulgarian Structured Corpus and annotated with word senses retrieved from the Bulgarian WordNet.

Keywords: WSD, corpora annotation, wordnet, flexible, multi-purpose language-independent framework

Introduction

The goal of this paper is to pres*ent a* Flexible Framework for development of annotated corpora called Chooser. The tool's functionality can be described to a great extent as language-independent and ask-independent provided it is oriented to the development of annotated training corpora irrelevant of their type and purpose.

One of the first problems encountered by any natural language processing system is ambiguity at different language levels: derivational, morpho-syntactic, syntactic, semantic, pragmatic. The resolution of a word's grammatical (morpho-syntactic) ambiguity has been largely solved for the purposes of natural language processing by part-of-speech (POS) taggers which perform high accuracy prediction of the syntactic category and particular grammatical features of words in running text [Chanod, 1994; Voutilainen, 1995]. Syntactic disambiguation (also known as structural) provides a means of choosing between alternative parses of the same string by calculating probabilities of the different phrase structures and gives preference to the most probable interpertation. The problem of resolving semantic ambiguity is generally known as word sense disambiguation (WSD) and has proved to be more difficult than part-of-speech and syntactic disambiguation [Resnik, 1997; Yarowsky, 1992; Yarowsky, 1995]. All levels of disambiguation are addressed within the framework of two main approaches – linguistic (rule-based) and probabilistic (data-driven) and their combinations in hybrid techniques. The rule-based approaches are usually characterized with high precision level and low recognition level; conversely the probabilistic approaches usually assign tags to all target language units but rate low in terms of accuracy. A flexible combination of these basic methods presupposes high values of both precision and recall evaluation criteria.

The statement that “a logical next step for the research community would be to direct efforts towards increasing the size of annotated training collections, while deemphasizing the focus on comparing different learning techniques trained only on small training corpora” [Banko, 2001] fully confirms our understanding of how POS and word sense disambiguation should be handled. Thus, the statistical approaches (completely or partially underlying any disambiguation process) need a well-defined tagged training data set.

A POS disambiguated training corpus has already been elaborated [Koeva, 2004b] and lately used in the development of the Brill-designed POS tagger [Doychinova, 2004]. At the moment our efforts are directed to the problems of WSD (illustrations below are from this domain). With a view to accomplishing this task we aimed at developing a framework which would be relatively independent of the annotation type, widely applicable for various NLP tasks and capable of incorporating various linguistic resources.

General description

Chooser was intended as a multi-purpose multi-functional platform covering various NLP tasks involving corpora annotation. It is a largely language-independent and task-independent visualisation and editing tool based on the Model-View-Controller (MVC) paradigm [Buschmann, 1996] affording automatic annotation and manual disambiguation of large volumes of text. The design is abstract enough to allow easy extension to all kinds of annotations.

The application utilises a corpus (arranged for the specific type of annotation) and various large coverage linguistic resources (linguistic networks, dictionaries, etc.) which provide certain linguistic information associated with the language units in the corpus. This involves association of certain units with certain set of linguistic features (tagset) in the linguistic resource database. The following terms will be used hereafter as follows:

Language unit (LU) is any natural language item which is target of annotation (especially disambiguation) tasks – i.e. word components (e.g. morphemes), simple words, compounds, expressions, language structures, etc.

Linguistic data base / resource (LR) – any type of linguistic information base that serves as a source for introducing and resolving ambiguity.

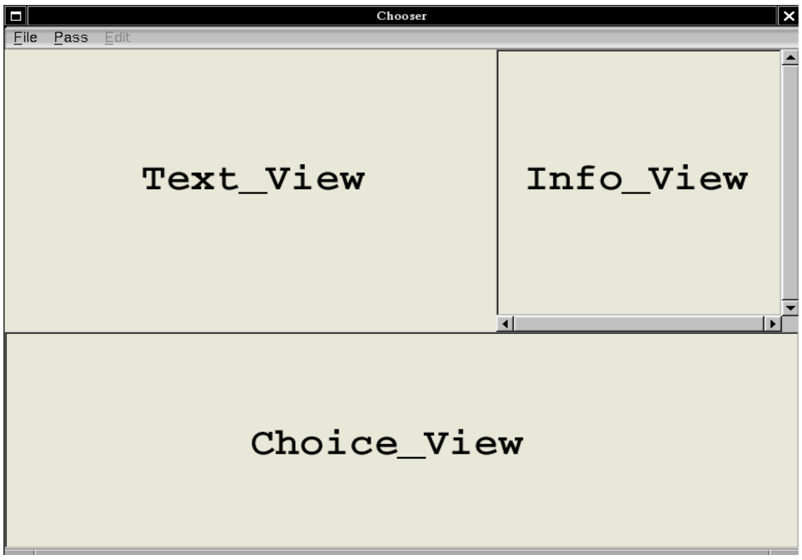
Language resources

The WSD implementation of Chooser makes use of two large linguistic resources – a training corpus of 75 000 words constituting a structured subset of the POS-tagged corpus of 150 000 words (itself being a subcorpus of the Brown-designed Corpus of Bulgarian) [Koeva, 2004b]. The subcorpora preserve the structure of the main corpus which consists of 500 text units of approximate length of 2000 words each, distributed proportionally to language use across 15 genres, thus forming an overall of about 1 000 000 words. The word forms in the training corpus are lemmatised and hand-checked where ambiguous forms have lead to the assignment of multiple lemmas.

The other resource – the Bulgarian WordNet (BulNet) is a lexical-semantic network of Bulgarian [Koeva, 2004a]. BulNet consists of over 23 500 synsets (synonym sets) and is continually being enlarged. Synsets are equivalence sets containing a number of obligatory elements: words (literals) with one and the same referential meaning, corresponding meaning definitions and a set of semantic relations such as hyperonymy, antonymy, meronymy, etc. as well as, morpho-semantic and some extralinguistic relations, pointing to other synsets. The lemmatised word forms in the training corpus are correlated with the corresponding sets of entries in the Bulgarian WordNet where the given lemmas feature as literals.

Visualisation and editing functionalities

The application's visualisation and editing functionalities provide text display management (Picture 1, left upper window, named Text view) and a number of other functions such as: text navigation according to various strategies, selection of particular options available for particular language units, group selection of adjacent or distant units (such as idioms, expressions whose constituents can be intervened by other words. Selection control is performed by the user who is enabled to navigate along a set of options associated with a particular language unit (displayed in the single bottom window of the application, Picture 1, named Choice_View) while simultaneously the information for the currently chosen (or default) option is shown in the right-hand window (Picture 1, named Info_View). The information displayed in the three windows is instantly synchronised on selecting a new item in the text, while the data in the right top window and the bottom window is also synchronised on clicking on a different list item in any of the two windows. The set of choices available for a given unit can be ordered according to different criteria, presently the adopted criterion is the frequency of selection of a given sense in the process of WSD.



Picture 1: Chooser's User Interface

The application design envisages a number of language units pass strategies such as: passing all language units in the corpus, stopping at language units which are associated with some information in the linguistic database, passing only ambiguous language units, language units that have been modified in the database since the last selection of the same item by the current user, or stopping at all occurrences of a particular item.

These strategies are easily extendable and redefinable with respect to the particular task, thus enabling the users to adopt different techniques towards disambiguation of the LU, as well as towards enhancing, enlarging, validating, etc. the information in the linguistic database.

Chooser is a multiple-user platform that performs dynamic interaction between the local users. Users communication is implemented by means of a server that takes care of a number of activities in two principal directions:

- Interaction between the local users and the linguistic database:

This type of interaction includes various requests by the local clients sent to and automatically performed by the server such as updates.

- Interaction between the local users:

This type of interaction involves the automatic notification of the server about local clients' data status, processing of clients' data and feedback to local users. For example this functionality is employed in weight assignment to options used in option lists array according to frequency.

The Architecture of the Chooser

The application's architecture integrates specific features and functionalities required for the implementation of NLP tasks. Linguistic annotation involves aligning language units in corpora with certain tags available in the linguistic resources. This process involves associating specific items in different sources and, generally, subsequent performance of decision-making procedures. In this respect, Chooser's architecture is meant in the first place to ensure fast and reliable visualisation and editing environment for language data as well as to implement the set of particular requirements imposed by the concrete tasks.

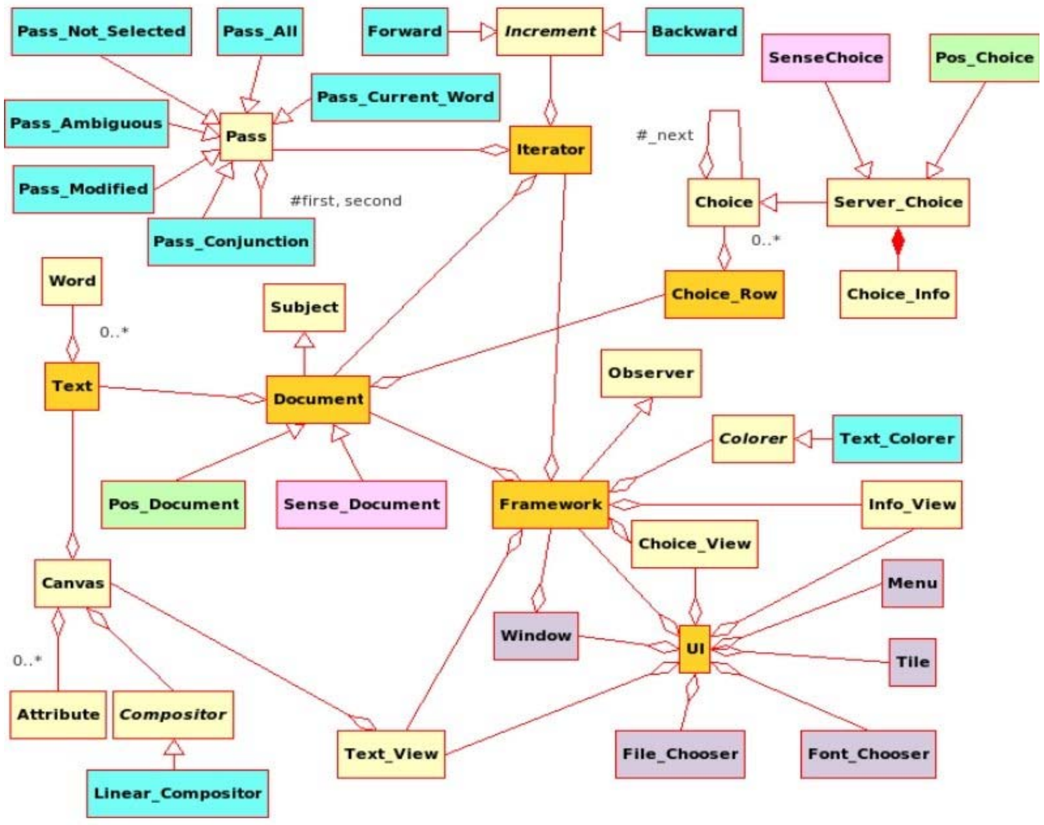
The Model-View-Controller paradigm and some well-known design patterns such as Strategy, Chain of responsibility, Observer, Iterator afford sufficient and flexible design solution for the requirements posed by this NLP task [Gamma, 1994].

The User Interface (UI) ensures text visualisation and navigation by means of the following objects: ChoiceView, InfoView, TextView. The latter defines a vertically scrollable area (displayed in the left-hand top window) and an object called Canvas responsible for displaying the text loaded for annotation and for storing information about the colour and the position of the language units. The LU themselves are stored in the Text object as a collection of Words and are displayed by the TextView.

Because of the employment of various texts ordering strategies the strategy management is delegated to a special object called Compositor. Compositor receives a collection of the sizes of the objects for arrangement and the width of the display frame and returns a collection of positions for the objects. ChoiceView is a control object of the kind "listview" which displays the set of options available for the language units (in the bottom window) while InfoView visualises additional (linguistic) information for the selected option in the right-hand top window.

The sets of options for WSD are in fact sets of BulNet synsets in which the lemma assigned to a wordform in the training corpus features as a literal. The additional linguistic information includes the meaning definition of the corresponding synset, the other (where any) members of the synset, the semantic relations pointing to other synsets.

Framework serves as the application controller which centralises retrieval and invocation of request-processing components. It performs action and view management between the User Interface and Document by responding to the user's actions and to updates in the document data, including such made by external applications, thus notifying the user of modifications in the Bulgarian WordNet. Document is an abstract class that on the one hand defines an interface for loading from and saving to a stream, and on the other provides access to LUs and the options available for them by means of two objects – Text and Choice_Row which are indexable collections of LUs and choices, respectively. In this way a given LU in the corpus text is associated with a corresponding (co-indexed) Choice object. On the basis of the current instance of Choice an object Colorer called by the Framework supplies information about the current LU. Its minimum functionality is to show the current LU, in the WSD implementation – words and compounds, and basically serves to distinguish annotated, ambiguous, modified words, etc. Another function of the controller is to provide interface for users searching and editing the LU in the text (spelling, typing errors, etc.) and the citation forms (in cases of wrong lemmatisation, etc.).



Picture 2. Class diagram representing Chooser’s architecture

Navigation along the text is managed also by Framework by means of an object called Iterator which encapsulates different pass strategies delegated to the objects Pass and Increment. Skipping of particular LU is determined by Pass object using the LU's corresponding Choice object while Increment determines the direction (right, left) of the movement along the text. The Iterator defines an interface for beginning the iteration, moving to the next LU (as requested by the user), moving to an arbitrary LU and checking if the iteration is finished.

Choice objects are generic and complex components central to the application’s design that enable different strategies for concrete user actions, request retrieval, storage and management. Some basic responsibilities of Choice include call and view of the available options and their weights for a LU and additional information for every option by making requests to a centralised linguistic database, storage and view of the selected option. The mechanism for selecting compound (adjacent or non-adjacent) LU (treated as one unit for the concrete task) is realised by cyclic lists of Choice objects so that every Choice object stores reference to the next. In such a way the users are enabled to group words (e.g. compounds, idiom constituents, etc.) and select them as one unit regardless of whether they are adjacent or are separated by other words.

Basically, disambiguation task requires that the linguistic database be centralised. This problem is solved by a successor of Choice Server_Choice. It interacts with certain other classes and encapsulates centralized passing of the information for a LU from a server program which stores and maintains the information available for LU and options to an instance of the application. The server extracts linguistic information from the linguistic database (for WSD this is the wordnet) and sends it to every client connected to it. Changes in the database are reported to the server which passes the differences to clients so that the users are enabled to immediately see the changes. This is achieved by creating a single instance of the class Choice_Info which is responsible for storing all the information for LUs and the corresponding options while at the same time maintaining connection with the server and “listening” for updates.

The abstract classes afford concrete implementation that is either universal or purpose-specific and in any case largely applicable for all kinds of disambiguation tasks. For the current application of Chooser WSD, several concrete successors have been designed. Linear_Compositor is a concrete and fast (linear) text arrangement strategy. Text_Colorer colours the current LUs (words) so as to distinguish ambiguous items for which selection has been made from items with only one option or no option at all, as well as compound items. In this way the interface is more user-friendly as different colouring facilitates users' decision-making process.

With a view to enabling users to employ different strategies for going over LUs the following concrete Pass successors have been designed:

- Pass_All asks Iterator to stop at every LU in the text;
- Pass_Ambiguous enables passing over ambiguous LUs (for which there are more than one options);
- Pass_Not_Selected enables selection only of ambiguous words for which no selection has been made so far (already disambiguated LUs are skipped);
- Pass_Modified is an object that requests Iterator to stop at LUs which have had their corresponding items of information (including available options and additional options' information in the linguistic database) modified at a later moment than the last selection of an option for this LU by the user. Thus, the user receives update of the changes in the database that concern the choices made be him/her;
- Pass_Current_Word asks Iterator to process all occurrences of the current LU (selected by the user);
- Pass_Conjunction is a conjunction of two other strategies, specifically applied to combine Pass_Current_Word and some of the other four;
- Increment interface is currently implemented by two classes Forward and Backward that determine the direction of pass.

Finally, Chooser uses a number of task-specific objects closely-bound to the particular file format underlying the text - Sense_Document, POS_Document and Sense_Choice, POS_Choice. POS_Document and Sense_Document are task-specific variants of the document layout for POSD and WSD respectively. POS_Choice and Sense_Choice are more complex both in terms of design and implementation. Sense_Choice and Pos_Choice inherit Server_Choice. The interaction with the server is concretely implemented in the following manner: on the creation of Choice_Info a second thread responsible for the establishing and maintaining of the connection and retrieval of updates is initiated. All Server_Choice objects perform options and option details (information) requests by Choice_Info in the main thread. In this way the centralised linguistic database and the local Chooser instances interacts bidirectionally.

Implementation

The framework is a platform-independent implementation written in C++. The User Interface uses FLTK (Fast Light Toolkit – a cross-platform open source library for GUI, see www.fltk.org). Other system-dependent features like the use of threads are implemented with Boost (free open source and cross-platform library, see www.boost.org). The utilisation of these libraries along with the server's implementation in Perl makes possible for Chooser to be recompiled for and run on a number of operating systems (UNIX/LINUX (X11), MacOS, Windows, OS/2, etc.).

Functionality of the application

The already discussed features of the framework can be summarized as:

Multipurpose – it can be easily extended to different linguistic (and non-linguistic) projects involving data annotation, although created for specific NLP tasks.

Multi-user – the application has a centralized design able to maintain and interact between multiple users-;

User-friendly – the design affords easy and compatible incorporation of new features, visualization and editing strategies, etc.

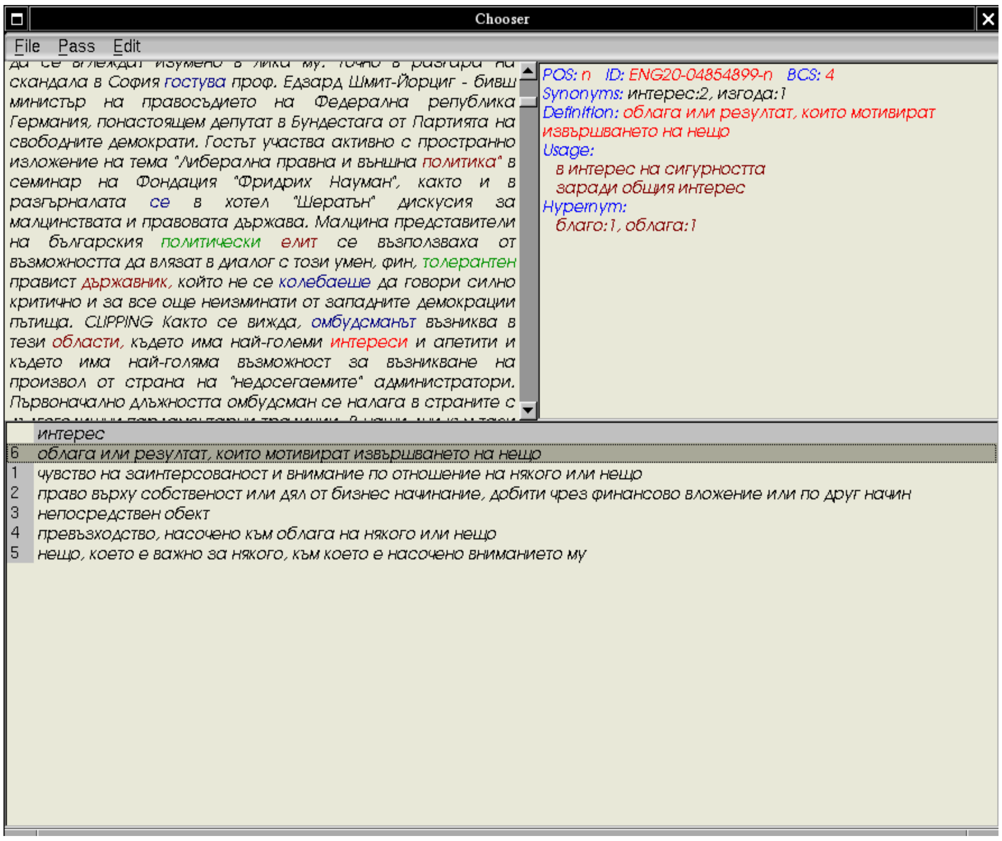
Language-independent – different languages can be integrated (the language dependent parameters and their pertaining values can be easily changed – reformulated, added or deleted, if necessary).

The functionalities of the tool with respect to the WSD provide for options such as labeling compounds referring to a single concept (i.e. *New York, sulphuric acid,* etc.), assigning the correct wordnet sense of a particular single or compound word from the list of senses available in the Bulgarian wordnet and adding / deleting / correcting senses in the wordnet, if necessary.

After the processing of the Bulgarian semantic corpus with the tool all semantically ambiguous LU available in the corpus are merged with the correct senses in the Bulgarian wordnet and the following outputs are produced:

- For function words - Word, Lemma, POS tag;

- For nouns, verbs, adjectives and adverbs - Word, Lemma, POS tag, Sense tag;
- For compounds - Compound, Lemma, POS tag (equivalent to the head word POS), Sense tag.



да се вглеждат изумено в лика му. Точно в разгара на скандала в София гостува проф. Едзард Шмит-Йорциг - бивш министър на правосъдието на Федерална република Германия, понастоящем депутат в Бундестага от Партията на свободните демократи. Гостът участва активно с пространно изложение на тема "Либерална правна и външна политика" в семинар на Фондация "Фридрих Науман", както и в разгърналата се в хотел "Шератън" дискусия за малцинствата и правовата държава. Малцина представители на българския политически елит се възползваха от възможността да влязат в диалог с този умен, фин, толерантен правист държавник, който не се колебаеше да говори силно критично и за все още неизминати от западните демокрации пътища. CLIPPING Както се вижда, омбудсманът възниква в тези области, където има най-големи интереси и апетити и където има най-голяма възможност за възникване на произвол от страна на "недосегаемите" администратори. Първоначално длъжността омбудсман се налага в страните с

POS: n ID: ENG20-04854899-n BCS: 4
Synonyms: интерес:2, изгода:1
Definition: облага или резултат, които мотивират извършването на нещо
Usage:
в интерес на сигурността
заради общия интерес
Hypernym:
благо:1, облага:1

	интерес
6	облага или резултат, които мотивират извършването на нещо
1	чувство на заинтерсованост и внимание по отношение на някого или нещо
2	право върху собственост или дял от бизнес начинание, добити чрез финансово вложение или по друг начин
3	непосредствен обект
4	превъзходство, насочено към облага на някого или нещо
5	нещо, което е важно за някого, към което е насочено вниманието му

Picture 3. WSD with Chooser

Conclusions

The work along WSD of Bulgarian has been under way within the framework of developing a training set for a system for machine translation. Along these lines have elaborated a fully sense-disambiguated corpus carried out entirely with the described linguistic resources and assisted by the here presented application. Improvements of Chooser's design have already been made since the time it was first presented at the COMTOOCI workshop and proved the initial expectations of being an easily upgradable application.

Bibliography

[Banko, 2001] Michele Banko, Eric Brill: Scaling to Very Very Large Corpora for Natural Language Disambiguation. ACL, 2001: 26-33.

[Buschmann, 1996] Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Machael Stal. Pattern-Oriented Software Architecture. John Wiley and Sons,1996.

[Chanod,1994] Jean-Pierre Chanod, Pasi Tapananen. Statistical and costraind-based taggers for French. In Technical report MLTT-016. Rank Xerox Research Centre, Grenoble, 1994.

[Gamma, 1994] E. Gamma; R. Helm, R. Johnson, J. Vlissides.. Design Patterns, Addison-Wesley: Reading. MA, 1994.

[Doychinova, 2004] Veselka Doychinova, Stoyan Mihov: High Performance Part-of-Speech Tagging of Bulgarian. In: Proceedings of Eleventh International Conference on Artificial Intelligence: Methodology, Systems, Applications (AIMSA-2004), LNAI #3192. 2004:246-255.

[Koeva, 2004a] Svetla Koeva, Tinko Tinchev, Stoyan Mihov. Bulgarian Wordnet-Structure and Validation. In: Romanian Journal of Information Science and Technology, Volume 7, No. 1-2, 2004:61-78.

[Koeva 2004b] Svetla Koeva. Modern Language Technologies. In: Lows of/for language. Heyzal, Sofia, 2004:111- 157.

[Resnik,1997] Philip Resnik and David Yarowsky. A perspective on word sense disambiguation methods and their evaluation. In Marc Light, editor, Tagging Text with Lexical Semantics: Why, What and How? Washington, April. SIGLEX (Lexicon Special Interest Group) of the ACL, 1997:79.86.

[Voutilainen, 1995]: Atro Voutilainen, A syntax-based part-of-speech analyser. In Proceedings of the Seventh Conference of the European Chapter of the Association for Computational linguistics. Dublin, 1995.

[Yarowsky, 1992] David.Yarowsky. Word-sense disambiguation using statistical models of Roget's categories trained on large corpora. In: COLING 92. Nantes, 1992.

[Yarowsky, 1995]: David.Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In: ACL 95. MIT, 1995:189-196.

Author information

Svetla Koeva, Department of Computational Linguistics, IBL, BAS, Sofia 1113, 52 Shipchenski prohod blvd., e-mail: svetla@ibl.bas.bg

Borislav Rizov, Department of Computational Linguistics, IBL, BAS, Sofia 1113, 52 Shipchenski prohod blvd., e-mail: bobby@ibl.bas.bg

Svetlozara Leseva, Department of Computational Linguistics, IBL, BAS, Sofia 1113, 52 Shipchenski prohod blvd., e-mail: zara@ibl.bas.bg