Creation of a Tagged Corpus for Less-Processed Languages with CLaRK System

**Kiril Simov, Petya Osenova, Alexander Simov,
Krasimira Ivanova, Ilko Grigorov, Hristo Ganev**

BulTreeBank Project
http://www.BulTreeBank.org
Linguistic Modelling Laboratory, Bulgarian Academy of Sciences
Acad. G. Bonchev St. 25A, 1113 Sofia, Bulgaria
kivs@bultreebank.org, petya@bultreebank.org, alex@bultreebank.org,
krassy_v@bultreebank.org, ilko@bultreebank.org,ico@bultreebank.org

Abstract

This paper addresses the problem of efficient resources compilation for less-processed languages. It presents a strategy for the creation of a morpho-syntactically tagged corpus with respect to such languages. Due to the fact that human languages are morphologically non-homogenous, we mainly focus on inflecting ones. With certain modifications, the model can be applied to the other types as well. The strategy is described within a certain implementational environment - the CLaRK System. First, the general architecture of the software is described. Then, the usual steps towards the creation of the language resource are outlined. After that, the concrete imlementational properties of the processing steps within CLaRK are discussed: text archive compilation, tokenization, frequency word list creation, morphological lexicon creation, morphological analyzer, semi-automatic disambiguation.

1. Introduction

A corpus annotated with morpho-syntactic information is one of the basic language resources for any language. This is especially true for languages with rich inflectional morphology. The existence of such a corpus is a prerequisite for the development of basic natural processing modules like morphological analyzers, taggers, chunk grammars. Thus, for less-processed languages the compilation of a morpho-syntactically annotated corpus is one of the primary tasks in the area of language resources creation. As a less-processed language we consider a language for which there are no electronic language resources at all or there are some partial ones.

We consider this task as a possibility for the creation of other resources of great importance for natural language processing like morphological lexicons, rule-based disambiguators, morphological guessers, baseline stochastic taggers. In this paper we present a strategy for the creation of a full form lexicon which to be used for morphological analysis of texts in a language. Additionally, the system offers mechanisms for rule- and manually-based disambiguation.

Needless to say, we are far from being pioneers in discussing such a problem. There is a vast amount of literature dealing with the creation of basic electronic resources for different languages: EAGLES initiative, BLaRK initiative. See also (Leech 1991), (Van Halteren 1999) among others. Our aim is not to summarize all the work devoted to this task, but, pointing to the troubleshoots, to contribute with a concrete strategy within a concrete implementational environment.

Languages are very diverse with respect to the complexity of their morphology. According to Bloomfield's classification (how do languages encode information in their morphology?) there are four types of languages: (1) *Isolating languages*; (2) *Agglutinative languages*; (3) *Polysynthetic languages*; and (4) *Inflecting languages*. As mentioned in (Allwood et. al. 2003, p. 4), it is difficult to capture all types of languages within one unified standardized scheme. For that reason, we present a model suitable for the group of inflecting languages. At the same time, this model is re-usable for other groups of languages, with the appropriate modifications. For example, it has been applied to Hungarian and Tibetan along with languages like Bulgarian, French, Croatian.

All the steps are realized in the CLaRK System. In order to facilitate the application of the strategy to a new language we provide a set of examples for English. These examples help the user to learn how to use the CLaRK System and, at the same time, they can be adapted to a new language.

The structure of the paper is as follows: in the next section the architecture of the CLaRK System is presented. In section 3 the general process of morpho-syntactic tagging is described. Section 4 concentrates on the steps of the morpho-syntactically annotated corpus within the CLaRK System. Section 5 outlines the conclusions.

2. CLaRK System

In this section we describe the basic technologies of the CLaRK System[1] — (Simov et. al. 2001). CLaRK is an XML-based software system for corpora development. It incorporates several technologies: *XML technology*; *Unicode*; *Regular Grammars*; and *Constraints over XML Documents*.

XML Technology

The XML technology is at the heart of the CLaRK System. It is implemented as a set of utilities for data structuring, manipulation and management. We have chosen the XML technology because of its popularity, its ease of understanding and its already wide use in description of linguistic information. In addition to the XML language

[1] For the latest version and the documentation of the system see `http://www.bultreebank.org/clark/index.html`.

(XML 2000) processor itself, we have implemented an XPath language (XPath 1999) engine for navigation in documents and an XSLT engine (XSLT 1999) for transformation of XML documents. We started with basic facilities for creation, editing, storing and querying XML documents and developed further this inventory towards a powerful system for processing not only single XML documents but an integrated set of documents and constraints over them. The main goal of this development is to allow the user to add the desirable semantics to the XML documents. The XPath language is used extensively to direct the processing of the document pointing where to apply a certain tool. It is also used to check whether some conditions are present in a set of documents.

Tokenization

The CLaRK System supports a user-defined hierarchy of tokenizers. At the very basic level the user can define a tokenizer in terms of a set of token types. In this basic tokenizer each token type is defined by a set of UNICODE symbols. Above this basic level tokenizers the user can define other tokenizers for which the token types are defined as regular expressions over the tokens of some other tokenizer, the so called parent tokenizer. For each tokenizer an alphabetical order over the token types is defined. This order is used for operations like the comparison between two tokens, sorting and similar.

Regular Grammars

The regular grammars in CLaRK System (Simov, Kouylekov and Simov 2002) work over token and element values generated from the content of an XML document and they incorporate their results back in the document as XML mark-up. The tokens are determined by the corresponding tokenizer. The element values are defined with the help of XPath expressions, which determine the important information for each element. In the grammars, the token and element values are described by token and element descriptions. These descriptions could contain wildcard symbols and variables. The variables are shared among the token descriptions within a regular expression and can be used for the treatment of phenomena like agreement. The grammars are applied in cascaded manner. The evaluation of the regular expressions, which define the rules, can be guided by the user. We allow the following strategies for evaluation: 'longest match', 'shortest match' and several backtracking strategies.

Constraints over XML Documents

The constraints that we have implemented in the CLaRK System (see (Simov, Simov and Kouylekov 2003)) are generally based on the XPath language. We use XPath expressions to determine some data within one or several XML documents and thus we evaluate some predicates over the data. Generally, there are two modes of using a constraint. In the first mode the constraint is used for validity check, similar to the validity check, which is based on a DTD or an XML schema. In the second mode, the constraint is used to support the change of the document to satisfy the constraint. The constraints in the CLaRK System are defined in the following way: `(Selector, Condition, Event, Action)`, where the selector defines to which node(s) in the document the constraint is applicable; the condition defines the state of the document when the constraint is applied. The condition is stated as an XPath expression, which is evaluated with respect to each node, selected by the selector. If the result from the evaluation is improved, then the constraint is applied; the event defines when this constraint is checked for application. Such events can be: selection of a menu item, pressing of key shortcut, an editing command; the action defines the way of the actual constraint application.

Cascaded Processing

The central idea behind the CLaRK System is that every XML document can be seen as a "blackboard" on which different tools write some information, reorder it or delete it. The user can arrange the applications of the different tools to achieve the required processing. This possibility is called **cascaded processing**.

3. Morpho-Syntactic Tagging

Morpho-syntactic tagging means assigning both: a part-of-speech and the bundle of all relevant grammatical features to the tokens in a corpus. Hence, the existence of an appropriate language-specific tagset is needed as well as an initial corpus in the language in question. When considering a less-processed language, several things have to be taken into account with respect to time-frame and financial constraints: (1) re-usability (i.e. the possibility to reuse an already existing resource), (2) representative tagset construction (i.e., when in a group of related languages there exists a tagset for one language, it can be used as a base for the other ones), and (3) using linguistic knowledge background, to minimize the size of the tagset for easier management.

Apart from these prerequisites, several steps have to be performed for the successful morpho-syntactic analysis. They are as follows:

1. tokenization
2. morpho-syntactic tagging
3. morpho-syntactic disambiguation
4. named-entity recognition

These steps require the construction of a set of tools for processing language corpora. The tokenization step requires a hierarchy of tokenizers for handling various cases. The possible combinations should meet some requirements, such as: (1) flexibility with respect to different token types (words, multiwords, punctuation, special symbols); (2) normalization (suppressing the difference between capital and small letters when necessary); and (3) modularity (tokenizing texts of mixed languages).

The morpho-syntactic step can be performed in various ways depending on the language and the existent language resources. The tools can be: taggers, regular grammars and guessers, used separately or in various combinations. A tagger can rely on linguistic knowledge, that is - consulting morphological dictionary. In this case a rule-based guesser is additionally needed to handle the unrecognized words. It usually relies on word-formation principles

and graphical prompts (capitalization, punctuation). On the other hand, a tagger can rely on statistical approaches. In this case it depends on frequency metrics and certain linguistic regularities between words. In state-of-the-art tools, both approaches (knowledge-based and stochastic) are often successfully combined. The regular grammars can be used at least for the following subtasks: (1) tagging multiword expressions (when one morphological word consists of more than one orthographical words), and (2) encoding rich knowledge resources, such as dictionaries.

The morpho-syntactic disambiguation step can be viewed either as a part of the morpho-syntactic tagging, or as a separate module. In the latter case it is performed by a disambiguator, which, similarly to the tagger, can be statistically-based or rule-based. For the creation of a stochastic device, a manually analyzed training corpus is needed. For the construction of a rule-based tool, a preliminary observation over the linguistic phenomena of the language in question is necessary.

Named-entity recognition step can be performed as part of the tokenization level, as part of morpho-syntactic tagging, or as a separate module. At the tokenization level the 'general token classification' can be applied (Osenova and Simov 2002), which distinguishes between common words, names, abbreviations, punctuation, special symbols, errors. Being a part of morpho-syntactic tagging means incorporating gazetteers of names and abbreviations into the tagger. As a separate module named-entity recognition can be organized into grammars and applied over raw or morphologically tagged text.

The order and combinations of the steps, listed above, depend on the language specificities, the aimed granularity of the analysis and on the existent initial resources for the language in question. One possible solution that we propose is generalized and described in the next section.

4. Implementing of Morpho-Syntactic Annotation

In this section we present a strategy for the creation of a morpho-syntactically tagged corpus for a language with little or no language resources. Most of the steps allow for more than one solution, because they depend heavily on the type of the language. Thus, we present a very simple solution which can be a basis for the development of a more sophisticated solution in each concrete case. We give a prompt how the corresponding step can be implemented in the CLaRK System.

4.1. Text Archive Compilation

We consider collecting electronic texts in the language in question as a prerequisite for the creation of a corpus. In order the text to be processed by CLaRK System, it has to be represented in XML format in an encoding appropriate for the language. CLaRK System recognizes Unicode encodings (UTF-8 and UTF-16) and several 8 bits standards for alphabet encodings. It also supports entity converters for several alphabets (ISO 8879).

In the text archive each document has to be marked-up at least to the structural level: chapters, articles, paragraphs. Some additional meta-information would be useful. For this level one can consult: TEI or CES guidelines.

Usually the texts for a given language are available in a plain text, HTML or RTF format. CLaRK System can read plain text or RTF documents directly and converts them into XML documents. The conversion of HTML to XML has to be done outside the CLaRK System, because CLaRK System can read only well-formed XML documents.

4.2. Tokenization

As it was mentioned above, the tokenization is the process of segmentation of the text into sentences and words: (Grefenstette and Tapanainen 1994). In general, the task is quite complex and in CLaRK System we divided it between two tools: tokenizers and regular grammars. The tokenizers work in a cascaded manner. First, a primitive tokenizer is applied which assigns a category to each Unicode symbol, then a sequence of complex tokenizers is applied. Each tokenizer in the sequence works over the output of the previous tokenizer. The tokens for each complex tokenizer are defined via regular expressions. The result of the tokenization is a list of tokens and their categories. This list is an input to the regular grammar tool which actually annotates the text if necessary. At the tokenization level our goal is to segment the text into a list of potential words, punctuation, numerical expressions. We assume that abbreviations, sentences, dates and similar entities are processed at the next level although one can try to do this directly at the tokenization level. Some of the trickier cases like several words forming one token or multi-token words can be processed later.

4.3. Frequency Word List Creation

Having a text archive and a reasonable tokenizer we can select some tokens as a basis for the creation of a morphological lexicon for automatic morphological annotation. This can be done in several ways. We consider the creation of a frequency list of tokens from the electronic texts as a good initial start. Such a word list can be constructed in CLaRK System with the help of Statistical tool. The tool counts the number of occurrences for each token in the selected textual elements. The result is an XML document which contains a table. Each row represents the token itself, its category, the number of occurrences. Additionally, the tokens can be normalized.

4.4. Morphological Lexicon Creation and Morphological Analyzer

Each regular grammar in the CLaRK System has a representation as an XML document. Using XSL transformation it is easy to construct a regular grammar over the word list produced in the previous processing step. Each rule in this grammar searches for a token in the text and substitutes it with an XML fragment. The XML fragment represents the morpho-syntactic annotation. For instance, the rule for the English word 'cost' has the following form:

```
"cost" -> <w aa="NN;VB;VPP;VPT">\w</w>
```

Here the token is on the left side and the XML fragment on the right. The fragment substitutes the token in the text when found. `\w` is a variable for

the input found in the XML document, thus when the rule succeeds the token 'cost' will be substituted with `<w aa="NN;VB;VPP;VPT">cost</w>`. The value of the attribute `aa` encodes all possible morpho-syntactic tags for the given token.

In order to construct such rules for the tokens in the word list the user needs a tagset for the language. The XSL transformation converts the word list into a set of empty rules like:

```
"cost" -> <w aa="">\w</w>
```

The user has to fill the appropriate morpho-syntactic information in them. The help which the system can provide is different sorting over the XML representation of the rules. The sorting can ensure better observation over the tokens. Here especially the reverse sorting (comparing the tokens from right to left) can be useful for grouping the tokens on the basis of their endings. Additionally, one can write conditional insertion operations in CLaRK which to fill the appropriate tags. Such a rule can be *if the token ends in 'lly' then it is an adverb*. Depending on the goal each wordform can be assigned also a lemma. In this way a morphological dictionary for the language is created.

The set of the ready rules is a regular grammar in CLaRK System. It is compiled into a minimized deterministic finite state automaton and can be used for morphological analysis of the texts.

4.5. Semi-automatic Disambiguation

Disambiguation is done with the help of constraints. In the example we use 'some attribute' value constraints. The constraints of this kind determine the value of some attribute on the basis of the context in which the attribute appears. In our case the target attribute is `ana` attribute which represents the morpho-syntactic analysis of the word. The value of the attribute depends on two things: (1) the value of the attribute `aa` for the word in our case which determines all the possible tags, and (2) the analyses of the other words in the text. Thus the first very general constraint states that the value of `ana` attribute is a member of the tokenized value of the `aa` attribute for the same word. This constraint can be used, as it was mentioned above, in two modes: validation mode and insertion mode. When used in insertion mode it will support the manual disambiguation of the annotated text by stopping at each ambiguous word, tokenizing the value of `aa` attribute and offering the user possibility for choosing the right tag. In validation mode the constraint checks whether the values of `ana` attribute is among the tags encoded in the value of `aa` attribute.

Additionally the user can write rules for automatic disambiguation imposing in the constraint more restrictions on the context in which the word appears. For instance, if the word before 'cost' is the determiner 'the', then the value of `ana` attribute for 'cost' is `NN`. Such rules can significantly reduce the amount of human intervention in the process of compiling a morpho-syntactically annotated corpus.

5. Conclusion

In the paper we presented a strategy for the creation of a morpho-syntactically tagged corpus for less-processed languages. The strategy is described within a certain implementational environment — the CLaRK System. The implementation is done as a sequence of steps. All these steps are done in CLaRK System for English. They are described as demos and are part of the distribution of the system. Although the described strategy requires a lot of manual work we think it is a good starting point for the development of more sophisticated approach in CLaRK System. The advantage is that the users have in one place all the necessary machinery for the implementation of each step. It is worth mentioning that the XPath engine of the system also provides an extensive library of mathematical functions which allows the implementation of statistical taggers in the system as well.

6. References

Jens Allwood, Leif Grünqvist and A.P. Hendrikse. 2003. *Developing a tag set and tagger for the African languages of South Africa with special reference to Xhosa.* To be published in the South African Journal of Linguistics and Applied Language.

Gregory Grefenstette and Pasi Tapanainen. 1994. *What is a word, What is a sentence? Problems of Tokenization.* In: Proc. of The 3rd International Conference on Computational Lexicography (COMPLEX'94). Budapest, Hungary. pp 79–87.

Geoffrey Leech. 1991. *The state of the art in corpus linguistics.* In: Aijmer & Altenberg (eds.), English Corpus Linguistics: Studies in honour of Jan Svartvik. pp 8–29.

Petya Osenova and Kiril Simov. 2002. *Learning a token classification from a large corpus. (A case study in abbreviations).* In: *Proc. of the ESSLLI Workshop on Machine Learning Approaches in Computational Linguistics*, Trento, Italy.

Kiril Simov, Zdravko Peev, Milen Kouylekov, Alexander Simov, Marin Dimitrov, Atanas Kiryakov. 2001. *CLaRK - an XML-based System for Corpora Development.* In: Proc. of the Corpus Linguistics 2001 Conference. pp 558–560.

Kiril Simov, Milen Kouylekov, Alexander Simov. *Cascaded Regular Grammars over XML Documents.* In: Proc. of the 2nd Workshop on NLP and XML (NLPXML-2002), Taipei, Taiwan.

Kiril Simov, Alexander Simov, Milen Kouylekov. *Constraints for Corpora Development and Validation.* In: Proc. of the Corpus Linguistics 2003 Conference, pages: 698-705.

Hans van Halteren (ed.). 1999. *Syntactic Wordclass-Tagging.* Kluwer Academic Publishers.

XML. 2000. *Extensible Markup Language (XML) 1.0 (Second Edition).* W3C Recommendation. http://www.w3.org/TR/REC-xml

XPath. 1999. *XML Path Language (XPath) version 1.0.* W3C Recommendation. http://www.w3.org/TR/xpath

XSLT. 1999. *XSL Transformations (XSLT). version 1.0.* W3C Recommendation. http://www.w3.org/TR/xslt