Automatic Structuring of Written Texts

Marek Veber, Aleš Horák, Rostislav Julinek, Pavel Smrž

Faculty of Informatics
Masaryk University
Botanická 68a, 60200 Brno, Czech Republic**

Abstract. This paper deals with automatic structuring and sentence boundary labelling in natural language texts. We describe the implemented structure tagging algorithm and heuristic rules that are used for automatic or semiautomatic labelling. Inside the detected sentence the algorithm performs a decomposition to clauses and then marks the parts of text which do not form a sentence, i.e. headings, signatures, tables and other structured data. We also pay attention to the processing of matched symbols in the text, especially to the analysis of direct speech notation.

1 Introduction

In order to reduce the time and memory demands of syntactic analysis, POS tagging, aligning parallel corpora and other NLP tasks, one first needs to divide the analyzed text into parts which are then analysed separately. The first suitable division points are paragraph boundaries. After appropriate pre-analysis it is possible go even deeper and segment the text to sentences and then to particular clauses. The analysis is facilitated by demarcation of those word groups that cannot or should not be divided any further like data, personal names, URL addresses etc.

In sentence boundary labelling we meet the problem of meaning ambiguity of the full-stop mark (a dot). Either it can denote a sentence end or it can be a part of an abbreviation or it can even bear both of these meanings (according to statistical results in English [3]: 90% — sentence end, 9.5% abbreviation and 0.5% both; in the Czech corpus DESAM we have 92% — sentence end, 5.5% abbreviation and 2.5% both meanings). Common approaches to solving the problem of labelling these hierarchical structures use regular expressions or finite automata with look-ahead that bear on several simple clues in text (like capitalisation) with a list of abbreviation and exceptions (see e.g. [1]). Other approaches are based on regressive trees [2] and artificial neuron networks [3] which make use of contextual information about POS tags in the surroundings of the potential structure boundary. However, those approaches cannot be easily applied in the analysis of Czech language because of the extent of the Czech tagset [4, 5].

** The research is sponsored by the Czech Ministry of Education under the grant VS 97028.

2 Structuring Algorithm

Our approach not only uses all the common methods, it also takes advantage of hierarchical processing in several adjoint phases. The algorithm first verticalises the input plain text to elementary lexical symbols (words, numbers and punctuation marks). Then it joins basic groups of elementary symbols into complexes that form further indivisible parts. Besides information contained in the text the analysis exploits the morphological information of particular symbols that is either determined during disambiguation or, when the information is missing, it is acquired by means of the morphological analyser. Above all we are interested in symbols, which are potential abbreviations, coordinate and subordinate conjunctions or which can form a verb group. The selection of the candidates for division runs hierarchically with the use of backtracking.

The heuristic rules generate possible text divisions together with appropriate probabilities, which then enables us to ask for consultation with human expert (linguist) only in cases where the division probability overpasses a given threshold value. The expert can then approve the result or reject it. This possibility can be advantageously used in semiautomatic annotation of training corpora.

Tagging relies on the fact that sentence boundaries do not exceed the paragraph limits and the clause boundaries stay between the sentence tags. The text is processed one paragraph at a time, where we seek for sentence boundaries. Every found sentence is then processed by the clause separation algorithm. Thus at the beginning and the end of paragraph we obtain positions that do certainly form the beginning or the end of a sentence and the same for clause boundaries at the sentence bounds.

Within the scope of the paragraph we mark all the possible candidates for a structural mark. Then we apply a set of partial parsing rules that increase or decrease the values of the positions of selected candidates in the block. Eventually we divide the whole paragraph according to the strongest candidates so that the required conditions would be satisfied. If such division is not possible, we try a different division according to some weaker candidates to the sentence boundaries.

3 Partial Parsing Rules

Sentence boundary The potential candidates for a sentence boundary are the positions consisting of a full-stop ('.'), a question mark ('?'), an exclamation mark ('!'), three dots ('...') followed by a word beginning with an upper case letter or a non-alphanumeric character (an opening quotation mark or a left parenthesis) and a closing quotation mark which is preceded by a full-stop, a question mark or an exclamation mark. The candidates are not sought among positions inside of any matched characters like parentheses, quotation marks or structural marks denoting data or e-mail addresses.

During this run we also seek for the candidates for direct speech, which are found as pairs of quotation marks that contain at least one punctuation character

and more than two positions in between. The direct speech candidate is marked by <sx> tag and after this step the block of direct speech text is divided to sentences and clauses.

Signature First, we determine whether the input block of text is formed by a signature. The condition for the block to be a signature is that the whole text must be formed by proper names or abbreviations that denote an academic or a military title. If the whole block satisfies this condition, we mark it with the <sign> tag.

Heading The fact that the block represents a heading is recognised by the condition that the block does not end with a full-stop and contains only one candidate for a sentence end. Such paragraph is marked as <head>.

4 Data Set and Results

Our results are based on the facts from the DESAM corpus (see [4, 5]) which is a tagged corpus consisting of more than 1 200 000 positions collected from Czech newspaper articles.

The following two kinds of tags are used in the DESAM corpus: *structural* and *grammatical* tags. The structural tags mark boundaries of documents, paragraphs, sentences, headers and signatures. Each position in the corpus is tagged with the following two grammatical tags: a *lemma* — the basic form of a particular word and a *tag* — representing its grammatical categories. Both the grammatical and structural tags have been manually disambiguated in the DESAM corpus.

The automatic structure tagger uses five tags: <s> for regular sentences (<sx> for direct speech and <c> for clauses — these tags were not included in the manual structure tagging), <head> for headings, <sign> for signatures and <table> otherwise. We have compared an output of the automatic tagging on the DESAM corpus with a manual tagging of the same texts. The results are summarised in the following table:

	number of blocks		number of positions		percentage of whole corpus		average length of one block	
tag	auto	man.	auto	man.	auto	man.	auto	man.
<s>	54030	51092	1054142	996430	84.6 %	80 %	19.5	19.5
<head>	5936	10802	28134	49821	2.2 %	4 %	4.7	4.6
<sign>	1885	1003	5289	2491	0.4 %	0.2 %	2.8	2.7
<table>	6741	8520	155958	196790	12.5 %	15.8 %	23.1	23.0

The next table displays the percentage of errors made either by automatic tagger or by a human labeller:

percentage of corpus	kind of error	manual tagging	automatic tagging
0.3 %	error in data, full-stop is missing	OK	ERR
0.1 %	error in data, extra (useless) full-stop	OK	ERR
0.8 %	error in data, sentence starts with lowercase	OK	ERR
2.2 %	bad tag	OK	ERR
2.7 %	bad tag or consistency err. (missing matched tag)	ERR	OK

Those differences display the necessity to make corrections not only to the tagging algorithm but also to the training data set. We also need to improve the error detection algorithm, which now only finds errors of matched characters.

The most problematic task (70% of the differences) for the automatic tagger is the decision whether the selected text should be marked as `<table>` or `<head>`. We suppose that this task can be solved if the tagger takes the surrounding context of the block into account.

5 Conclusions

The automatic sentence boundary labelling can be demonstrated on a morphologically disambiguated data as well as on a plain corpus text without tags, in which case a morphological analyser Lemma (see [6]) is used. The results of automatic labelling have been compared to a manually labelled corpus.

In the presented approach we have concentrated on accuracy, efficiency and robustness so as the structuring does not slow down the text processing. Based on comparisons that we have made so far, we can say that the automatic algorithm achieves even better (more consistent) results than the human labellers.

References

1. Cutting, D., Kupiec, J., Pedersen, J., Sibun, P.: A practical part-of-speech tagger. In *the 3rd Conference on Applied Natural Language Processing*, Trento, Italy 1991.
2. Riley, M., D.: Some applications of tree-based modeling to speech and language indexing. In *Proceedings of the DARPA Speech and Natural Language Workshop*, pages 339-352, Morgan Kaufmann 1989.
3. Palmer, D., D., Hearst, M., A.: Adaptive Sentence Boundary Disambiguation. In *The Proceedings of the ANLP '1994*, Stuttgart, Germany, October 1994.
4. Pala, K., Rychlý, P., Smrž, P.: DESAM— Approaches to Disambiguation. Technical Report FIMU-RS-97-09, Faculty of Informatics, Masaryk University, Brno, 1997.
5. Pala, K., Rychlý, P., Smrž, P. : DESAM— Annotated Corpus for Czech. In *Proceedings of SOFSEM'97*.
6. Ševeček, P.: *LEMMA* morphological analyzer and lemmatizer for Czech, program in "C", Brno, 1996. (manuscript).
7. Julinek, R.: Automatic Detection of Sentence Boundaries, Master thesis, Masaryk University, Brno, April 1999.