ISSN 1392 – 124X INFORMATION TECHNOLOGY AND CONTROL, 2005, Vol. 34, No. 3

RULE-BASED ANNOTATION OF LITHUANIAN TEXT CORPORA

Jurgita Kapočiūtė, Gailius Raškinis

Vytautas Magnus University
Vileikos 8, LT-3035 Kaunas, Lithuania

Abstract. In this paper we present an algorithm that automatically recognizes and annotates person and place names, contractions, acronyms, foreign language phrases, dates and sentence boundaries in Lithuanian texts. The algorithm is based on a set of manually developed template matching rules and a few specialized lexicons. The algorithm performs annotation by making several passes over the text. It can operate in automatic and semi-automatic annotation modes. In the semi-automatic annotation mode, the user is allowed to intervene in cases where automatic decision is uncertain. Users' feedback is memorized and stored in the lexicons. Rules and lexicons were developed after a careful examination of the text corpus of 600 thousand words. The algorithm was evaluated on a separate corpus of 400 thousand words and achieved ~93% annotation accuracy.

Keywords: text corpora, automatic annotation, tagging.

1. Indroduction

Text corpora are large text collections that store many millions of running words [7]. They are used as a basis for verification of hypotheses about language. Complex hypotheses related to language modeling, speech synthesis, morphological disambiguation and machine translation require annotated corpora, i.e. corpora to which additional linguistic, morphological, and/or syntactic information is added. Although text corpora can be annotated manually, such annotation requires many person-months of human labor.

In this paper we describe a rule-based approach to automatic recognition and annotation of text entities, such as: proper nouns, dates, sentence boundaries, contractions, acronyms, and foreign words. The recognition of these entities is complex due to the ambiguity inherent to all natural languages. Specific ambiguities also arise due to the highly inflected nature and free word order of Lithuanian.

The rest of the paper is organized as follows. Section 2 describes existing approaches to automatic annotation. In section 3 annotation problems are exemplified and the structure of the proposed algorithm is outlined. Section 4 provides experimental results. Section 5 contains some concluding remarks. Appendix provides more formalized description of annotation rules.

2. Related work

The methods of automatic annotation of various text entities can be divided into those based on empirically stated rules and those based on probabilistic models estimated on annotated text corpora.

Simple template matching rules are described by Wang and Huang [12]. Authors suggest that the pattern consisting of a lower-case string, followed by any of the symbols ".", "?", "!" and further by a capital letter accounts for the most part of sentence boundaries in the text. Grover at al. [4] and Ignat at al. [5] present simple templates for date detection. Their templates are enriched by the references to the lexicon of month names and check for prepositions and words frequently occurring in date phrases. Gawronska [3] recognizes nouns of inflected Polish with the help of the list of possible noun endings for all genders, numbers and cases. Pouliquen at al. [9] and Dimitrova and Dicheva [1] detect foreign language phrases that are inserted in the text, referring to the foreign language lexicons.

Kiss and Strunk [6] use rather sophisticated template matching rules for sentence boundary vs. contraction disambiguation. The authors use parameterized rules, where parameters are extracted from annotated text corpora. Rules are based on many features such as: length of a word, number of internal periods it has, number of times each word goes at the end of a sentence in the training corpus, number of times each word begins with a capital or lower-case letter; list of words that most frequently go at the beginning of a sentence. Candidate pattern is classified either as a contraction or as a sentence boundary depending on whether the rule applied to the candidate pattern results in a value exceeding specified threshold.

Wang and Huang [12] are addressing sentence boundary detection problem within a probabilistic framework. The authors compare Hidden Markov Models (HMM) trained on an annotated text corpus and the maximum entropy approach. In the latter case, word collocations and their frequencies at the beginning and at the end of a sentence are used as features and integrated into the formula of maximum entropy that is used for identifying sentence boundaries in a text. Tajima et al. [10] use similar probabilistic methods for identifying sentence boundaries: phrases are analyzed, examining how often certain words, phrases or morphological tags can go at the end of a sentence. Pham and Tran [8] present N-gram based language recognition method capable of assigning whole text documents to some particular language.

Many of the abovementioned approaches claim to be language-independent. Kiss and Strunk [6] applied their method to 8 Indo-European languages as well as to Estonian and Turkish and reported 98.93% – 99.72% and 90.52% – 99.92% annotation accuracy for sentence boundaries and contractions respectively. Ignat at al. [5] applied methods to English, French, German, Spain, Italian, Portuguese, and Romanian and reported 64-100% recall and 86-100% precision. Accuracy of annotating English sentences boundaries reported by Wang and Huang [12] varies from 91.43% to 99.56%. Tajima et al. [10] report 77.24% accuracy for annotating Japanese texts.

Automatic detection of Lithuanian text entities has never been attempted. In this paper, we present the first attempt to build such an algorithm using the rule-based approach and a few specialized lexicons. The algorithm aims at recognizing sentence boundaries, contractions, acronyms, proper nouns, foreign language insertions, and dates. The algorithm is also designed to distinguish between 2 subcategories of proper nouns (person names and place names), 3 types of foreign language phrases (English, Russian, Other), 4 types of dates (simple dates, date sets, time intervals, ages of an individual).

3. Automatic annotation

3.1. Annotation problems

Automatic recognition and annotation of text entities faces the following major problems:

Ambiguity problems

a. Person name vs. place name ambiguity (word "Roma" can stand for both person and place name).
b. Proper noun vs. generic noun ambiguity (word "Eglė" at the beginning of a sentence can stand for person name and generic noun).
c. Dash separated place name ambiguity (words "Kaunas" and "Vilnius" represent two place names within "Autostrada Kaunas – Vilnius", while "Adis-Abeba" represent one composite place name).
d. Period "." related ambiguity. Period may indicate sentence boundary, contraction, or both.
e. Language ambiguity ("bet" – may be Lithuanian or English word). Language ambiguity also results from international words, especially Slavonic words, spelled in Latin ("echo" may be Lithuanian, English, or Russian word).
f. Ambiguities related to categorization of short dates if no context is available ("50 metų" may be a simple date – "50 metų [įvykiai]", a time interval –"[truko] 50 metų" and the age of an individual – "[sukako] 50 metų").

Problems of recognizing single/compound entities

a. Some entities can occur only in isolation ("Lietuva"; "psl.").
b. Some entities always constitute a part of a compound entity ("Saudo" is always the part of "Saudo Arabija"; "habil." is the part of "habil. dr.").
c. Some entities can occur both in compound phrases and in isolation, depending on the context ("Britanija" may occur in isolation or as a part of the compound proper noun "Didžioji Britanija").

Problems of notational variety:

a. Contractions may start both with capital and lower-case letter ("Ha", "ha"); they can end with or without an external period ("psl.", "psl"). Contractions aren't necessarily short ("tūkstm.", "apskr.").
b. Person names, especially person names of foreign origin, are spelled in many different ways. They can contain short lower-case particles, apostrophes and hyphens ("Liudvigas van Bethovenas", "O'Neal", "Bush'as", "Vitkutė-Adžgauskienė"). They can be preceded or succeeded by a variable number of name contractions ("Dž. Bušas", "M.K. Čiurlionis", "Petras I", "Kenedis Dž."). Composite person names should be annotated as one entity in all abovementioned examples.
c. Dates belonging to all four subcategories may be written in many different formats: simple dates ("2005 m. sausio 1 diena", "1980 10 08"); date sets ("1991, 1992 ir 1993 metai", "7-as ir 8-as šimtmečiai"); time intervals ("Nuo 2001 metų vasario iki 2003 vasaros pradžios", "Nuo VII amžiaus pr. Kr. iki XI amžiaus po Kr. IX septinto dešimtmečio pirmos pusės"); ages of an individual ("25-erių metų" asmuo, "1,5 mėnesio" kūdikis, "4-5 metukų" vaikas).

3.2. Annotation algorithm

Our algorithm is based on the assumption that the abovementioned annotation problems can be solved by the carefully designed set of annotation rules. Rules must specify the template against which running

texts are matched as well as additional conditions, which may refer to an external linguistic knowledge stored in lexicons. For instance, "is the text token a word form of standard Lithuanian?", "does the text token is known to be a person name?" are typical examples of knowledge the templates may require. A typical annotation rule is illustrated by Figure 1.

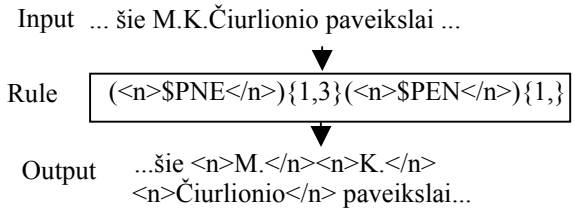

Figure 1. Illustration of a template matching rule

The rule sounds: "Search text for the pattern consisting of two parts, such that the first part consists of an abbreviated first name ($PNE) repeated no more than 3 times ({1,3}) and the second part consists of at least one ({1,}) token that matches some entry in the lexicon of person names ($PEN). For every such pattern found in the text put labels at the beginning (<n>) and at the end (</n>) of each text token".

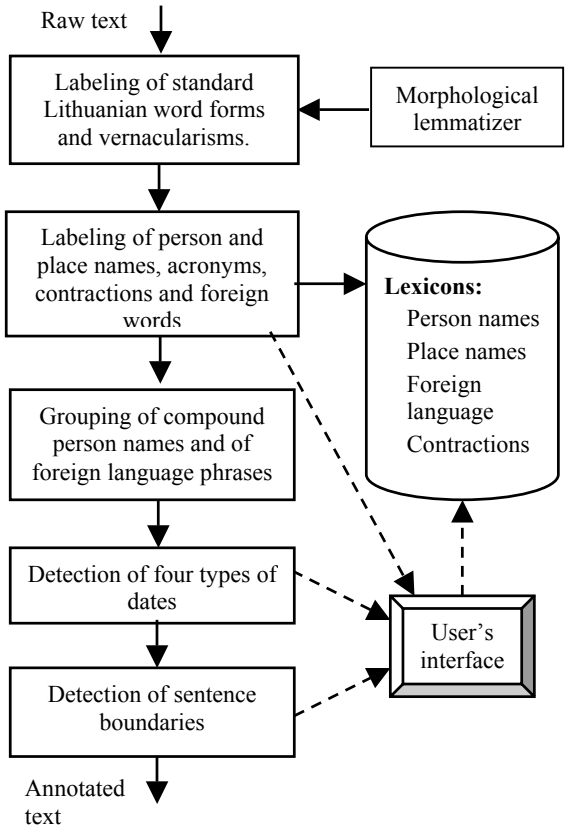

Figure 2. The architecture of the text annotating algorithm. Dotted lines represent human-computer communications in a semi-automatic annotation mode

The proposed algorithm can operate in automatic and semi-automatic processing modes. When operating in semi-automatic processing mode, the algorithm may ask humans to provide the correct decision. This happens if lexicon lookup fails and its own processing templates cannot achieve required certainty. Human's answers are always stored in lexicons, thus extending linguistic knowledge and reducing the number of appeals for human help in the future (Figure 2).

The algorithm of automatic text annotation consists of a few consecutive text processing steps:

Labeling of standard Lithuanian word forms. During the first pass over the text, all text tokens are tested for being possible word forms of standard Lithuanian[1]. Tokens failing this test are tested for being vernacularisms, which are recognized by a simple replace-match rule. Typical endings of vernacularisms ("on", "oj", "im", etc.) are replaced by the appropriate standard Lithuanian endings (e.g., "on"→"a" "rankon"→"ranka") and are tested for being possible word forms of standard Lithuanian again.

Lexicon lookup stage. All text tokens found in the specialized lexicons of person names, place names, acronyms, contractions and foreign words are correspondingly labeled. Lexicon lookup procedure takes into account the fact that proper nouns in the text may have another inflection than the ones stored in the lexicon. Foreign words are assigned to English, Russian and Unknown categories. As lexicons of place names and contractions directly store compound entities (e.g. "Dramblio Kaulo Krantas", "t.y.") no additional processing is required for "assembling" them from individual tokens. Compound place names and sequences of foreign language tokens still need to be "assembled" into a single entity before their annotation can take place.

At this point, the majority of text tokens are labeled, i.e. their role in the text is known. When operating in the semi-automatic annotation mode, a user is queried about the role of remaining text tokens.

Grouping of compound person names and of foreign language phrases. The application of specific rules for assembling compound person names and foreign language phrases is followed by the application of one general rule of merging sequences of tokens of the same type into a single entity. Specific rules for compound person names incorporate the first name contractions ("Dž. Bush"), typical lower case particles ("Emanuelis de la Costa"), extra additions ("O'Neal'as") and multiple standalone person names ("Valdas Adamkus") into a single compound person name. Specific rules for foreign language incorporate digits, acronyms, person names making up the same foreign language phrase. Rules take into account language-ambiguous words. For instance, tokens "to"

[1] Text token is identified as a word form of standard Lithuanian if it is recognized by the morphological lemmatizer of Lithuanian [13].

and "be" can be both English and Lithuanian words, but the phrase "to be or not to be" is correctly recognized as an English phrase.

Date identification. Initially, date detection rules check for the presence of digits accompanied by some date-related text strings, such as: names of months, years, centuries, days, weeks, etc. Thereafter, date components are assembled into one entity. Finally, dates are assigned to subcategories, the decision being based on the context, i.e. on the presence of certain prepositions ("nuo" … "iki"), conjunctions ("ir", "arba", "ar"), punctuation signs (",") endings ("25-erių", "14-metė"). The user may be queried about category assignment decisions.

Detection and annotation of sentence boundaries is performed by the set of specific rules as the last step of automatic annotation.

Appendix provides more formalized description of annotation rules.

4. Results

The algorithm described above was investigated on the manually annotated test corpus containing 400 thousand words extracted from the VMU Lithuanian text corpus. While operating in automatic annotation mode the algorithm was allowed to skip "risky" annotation decisions. Human-annotated texts were compared with the machine-annotated texts on a tag-by-tag basis. The performance was estimated by using the recall and precision metrics. The recall R was estimated as the number of decisions that were actually taken (A) divided by the number of decisions that were required to be taken (Q). The precision P was estimated as the ratio of the number of correct decisions to the total number of decisions:

$$R = \frac{A}{Q} \times 100\%, \quad P = \frac{A - I - D}{A} \times 100\%,$$

where *I*, and *D* denote false (inserted), and missed (deleted) annotations respectively.

While operating in semi-automatic annotation mode the algorithm made 15-52 and 0.68-12 queries per 1000 tokens on test and development corpora respectively (depending on the functional style of the text).

Detailed analysis of annotation errors revealed they are due to the following main reasons:

Lack of semantic analysis. Annotation rules are based on surface word forms and partially on morphological and syntactic templates. This information alone is not sufficient to disambiguate all annotation decisions. For instance, the token "Saulė" (case-sensitive) occurring in the middle of a sentence is annotated as a person name by default. Nevertheless, this token could also be a proper name of the star ("The Sun") in some cases.

While operating in semi-automatic annotation mode the algorithm made 15-52 and 0.68-12 queries per 1000 tokens on test and development corpora respectively (depending on the functional style of the text).

Table 1. Recall and precision of automatic annotation measured per entity type. Rows marked by T and D give the results on test and development corpora respectively.

Entity Type		Semi-automatic	Automatic	
		P	P	R
Proper nouns	T	97.61	89.70	99.14
	D	99.57	90.78	99.31
Acronyms	T	99.07	96.82	100.00
	D	99.98	98.67	100.00
Contractions	T	95.52	94.80	100.00
	D	99.45	96.28	100.00
Sentence boundaries	T	99.07	91.50	100.00
	D	99.89	93.11	100.00
Dates	T	97.34	89.25	87.21
	D	99.89	93.73	78.17
Foreign language	T	90.42	86.30	100.00
	D	94.30	87.56	100.00

Detailed analysis of annotation errors revealed they are due to the following main reasons:

Lack of semantic analysis. Annotation rules are based on surface word forms and partially on morphological and syntactic templates. This information alone is not sufficient to disambiguate all annotation decisions. For instance, the token "Saulė" (case-sensitive) occurring in the middle of a sentence is annotated as a person name by default. Nevertheless, this token could also be a proper name of the star ("The Sun") in some cases.

Adaptation to the development corpus. Annotation rules were empirically designed to deal with annotation problems found in the development corpus. As it was finite unseen patterns and contexts occurring in test corpus fail to be annotated correctly.

Spelling mistakes in the text. The algorithm assumes there are no spelling mistakes in texts. Misspelled contractions, for instance, fail to be annotated.

Erroneous human input. The algorithm assumes that all human answers to its queries are error-free (semi-automatic annotation) and stores results of such queries in appropriate lexicons. For instance, if human erroneously declared some string to be a person name instead of the place name, all future occurrences of this string will be automatically and erroneously annotated as person names.

5. Conclusions

This paper presents the first algorithm for automatic annotation of various entities in Lithuanian texts. The precision of the proposed algorithm is

evaluated by comparing machine annotated texts with texts annotated by a human expert. Working in an automatic mode the algorithm achieved over 92% of precision.

The algorithm is adaptive in the sense that in semi-automatic operating mode human's answers are always stored in one of the external lexicon. Thus linguistic knowledge becomes extended and the number of future appeals reduced.

The processing architecture used for annotating proper nouns, contractions, acronyms, foreign language insertions, dates, and sentence boundaries can be extended for detection and annotation other linguistically distinct text elements: spelling mistakes, words, grammatical forms, etc.

References

[1] **V. Dimitrova, D. Dicheva.** Learning Terminology in a Foreign Language. *Proceedings of the International Conference Resent Advances in Natural Language Processing. Tzigov Chark, Bulgaria*, 12-15, 1997. http://www-it.fmi.uni-sofia.bg/larflast/papers/ranlp97.pdf.

[2] Electronic text center. *Last updated* 17 *March*, 2005. http://etext.lib.virginia.edu/.

[3] **B. Gawronska.** Extracting Semantic Classes and Morphosyntactic Features for English-Polish Machine Translation. *Proceedings of the* 9*th International Conference on theoretical and Methodological Issues in Machine Translation. Keihanna, Japan*, 13-17 *March*, 2002. http://www.eamt.org/archive/tmi2002/conference/07_gawronska.pdf.

[4] **C. Grover, C. Matheson, A. Mikheev, M. Moens.** LT TTT – A Flexible Tokenization Tool. *Proceedings of Second International Conference on Language Resources and Evaluation,* 2000. http://www.ltg.ed.ac.uk/ papers/00tttlrec.pdf.

[5] **C. Ignat, B. Pouliquen, A. Ribeiro, R. Seinberger.** Extending an Information Extraction Tool Set to Central and Eastern European Languages. *Proceedings of the International Workshop Information Extraction for Slavonic and other Central and Eastern European Languages* (IESL'2003). *Borovets, Bulgaria*, 8 - 9 *September*, 2003, 33-39. http://hosting.jrc.cec.eu.int/langtech/Documents/EuroLan-03_Pouliquen-Steinberger-et-al.pdf.

[6] **T. Kiss, J. Strunk**. Multilingual Least-Effort Sentence Boundary Disambiguation. *Under review*, 2003. http://www.linguistics.ruhr-uni-bochum.de/~strunk/ks2003FINAL.pdf.

[7] **R. Marcinkevičienė.** Corpus linguistics in theory and practice. *Darbai ir Dienos*, *VDU* 24, 2000, 6-63. http://donelaitis.vdu.lt/publikacijos/marcinkeviciene.pdf.

[8] **T. Pham, D. Tran.** VQ-Based Written Language Identification. *Proceedings of the Seventh International Symposium on Signal Processing ant its Applications. Paris, France*, 2003, 513-516. http://www.ise.canberra.edu.au/DatT/Publications/1032.pdf.

[9] **B. Pouliquen, R. Steinberger, C. Ignat.** Automatic Annotation of Multilingual Text Collections with a Conceptual Thesaurus. *Proceedings of the Workshop Ontologies and Information Extraction at the Summer School The Semantic Web and Language Technology - Its Potential and Practicalities (EUROLAN'2003). Bucharest, Romania*, 28 *July* - 8 *August*, 2003. http://www.jrc.cec.eu.int/langtech/Documents/EuroLan-03_Pouliquen-Steinberger-et-al.pdf.

[10] **S. Tajima, H. Nanba, M. Okumura.** Detecting Sentence Boundaries in Japanese Speech Transcriptions Using a Morphological Analyzer. *Proceedings of the First International Joint Conference on Natural Language Processing*, 2004, 207-212. http://www.nlp.its.hiroshima-cu.ac.jp/~nanba/pdf/ijcnlp_tajima.pdf.

[11] TEI-Text Encoding Initiative. *Last updated* 7 *March*, 2002. http://helmer.aksis.uib.no/tonemerete/forelesninger/Datalingvistikk/om_tei_2002_03_07.html

[12] **H. Wang, Y. Huang.** Bondec – a sentence boundary detector. CS224N/*Ling237 Final Projects*, 2003. http://nlp.stanford.edu/courses/cs224n/2003/fp/huangy/final_project.doc.

[13] **V. Zinkevičius.** Lemuoklis – tool for morphological analysis. *Darbai ir Dienos*, *VDU* 24, 245-274, 2000. http://donelaitis.vdu.lt/publikacijos/zinkevicius_.pdf.

Appendix

Formalized description of the annotation algorithm

Let:

\A	denote the beginning of a new line[2];
\Z	denote the end of a new line;
\B	denote the beginning of a new sentence;
\M	denote position other than beginning of a new sentence;
DD	be the set of digits (written in digital form) {0,1,…9};
DL	be the set of digits (written in literal form) ("vienas", "du"…)[3];
RD	be the set of Roman numerals {I, II, III,…}.
	# Definitions for detecting proper nouns,
	# abbreviations and foreign language words:
ACR	be the lexicon of acronyms;
AMW	be an ambiguous word (GR any text token);
CAT	be the set of known contractions occurring anywhere in the text (with external period);
CON	be the lexicon of contractions (other than the first name and paragraph numbering contractions);
FLW	be the set of foreign language words (English, Russian or others);
PEN	be the lexicon of standalone person names;

[2] The algorithm is presented in a slightly simplified form using Perl inspired notation. Lexicon names are prefixed by '@', variable names are prefixed by '$', i.e. $LEX is any element from lexicon @LEX.

[3] Lexicon lookup procedure takes into account the fact that inflected words in the text may have another inflection than the ones stored in the lexicon.

PLN be the lexicon of standalone and compound place names;

PNE denote the first name contractions (ending with an external period);

PNW denote the first name contractions (ending without an external period);

SWN denote the set of lower case particles proper to compound person names (“van”, “fon”, “de”, etc.);

TT denote the set of text tokens of standard Lithuanian.

 # Definitions for detecting dates:

ABD be the set of words that indicate abstract date (“šis”, “praeitas”, “ateinantis”, “žalvario”, “geležies”, “nepriklausomybės”, etc.);

AWY be the set of words associated with the word “metai” (“aštuoniasdešimtmetis”, “pusmetis”, “šešiolikametis”, “vienuolikmetis”, “tūkstantmetis”, etc.);

CCE be the set of words and contractions that indicating era (“prieš mūsų erą”, “po Kristaus”, etc.);

END be the set of endings, which explicitly indicate the inflection of a numeral (“ieji”, “ąją”, e.g.: “1980-ieji”, “15-ąją”, etc.);

HD be the set of huge numerals in literal form (“tūkstantis”, “šimtas”, “milijonas”, “milijardas”);

PAR be the set of particles (“netgi”, “net”, “pat”, etc.);

PRE be the set of prepositions (“nuo”, “prieš”, “per”, “iki”, “ligi”, etc.);

SW be the set of words that indicate short period of time (“diena”, “para”, “savaitė”, etc.);

WCE is the list that indicates word (“amžius”);

WMO is the list that indicates word (“mėnuo”);

WW is the list of words (“pusė”, “pradžia”, “vidurys”, “ketvirtis”, “pabaiga”);

WY is the list that indicates word (“metai”) as isolate word;

YM be the set of months (“sausis”, “vasaris”, etc.);

YS be the set of seasons (“pavasaris”, “vasara”, “ruduo”, “žiema”).

 # Definitions for detecting sentence boundaries:

CNE be the set of contractions never occurring at the end of a sentence (“dr.”, “gerb.”, etc.);

PAG be the set of paragraphs numeration symbols (ending with external period);

SEP be the set of token separators [,.?!:-/()[]{}<>|%`'"¨–*];

SLC be the set of lower-case letters;

SUC be the set of upper-case letters;

USS be the set of common sentence separators [.!?].

Let the functions:

Length(x) return the length of an argument x in characters;

Count(x) return the number of occurrences of x (including all its inflected forms) in the current text (Count(x)=0 meaning x is absent from the current text);

LCase(x) return the lower-case equivalent of x;

ID(x) return 0, if x can be both generic and proper noun and 1, if x can be just a proper noun;

X{nmin, nmax} denotes that the string/variable X is allowed to be adjacently repeated from nmin to nmax times. If X{nmin,} then nmax=∞. If X{nmin} then nmax=nmin.

Rules for automatic annotation of person names (<n>), place names (<pl>), foreign phrases (<f>):

```
1. (<n>$PNE</n>){1,3} (<n>$PEN</n>){1,}
2. (<n>$PNW</n>){1,3} (<n>$PEN</n>){1,}
3. (<n>$PEN</n>){1,} (<n>$PNE</n>){1,3}
4. (<n>$PEN</n>){1,} (<n>$PNW</n>){1,3}
5. (<n>$PEN</n>){1,} (<n>$RD</n>){1}
6. <n>($PEN){1}</n> (<n>($SWN)</n>){1,2}
   <n>($PEN){1}</n>
7. <n>($PNE){1,3}</n>
   (<n>($SWN)</n>){1,2} <n>($PEN)</n>
8. \B(<n>($PEN){1}</n>), if ID($PEN)=1
   and Count(LCase($PEN))=0 and
   Count(\M($PEN))>0
9. \M((<n>($PEN)</n>){1,}), if
   ID($PEN)=0
10. \B(<?>[4]($AMW){1}<?>), if $AMW in @PEN
    and $AMW in @PLN
11. \M(<?>($AMW){1}<?>), if $AMW in @PEN
    and $AMW in @PLN
12. \B(<pl>($PLN){2,}</pl>)
13. \M(<pl>($PLN){1,}</pl>), if
    ID($PLN)=1
14. \B(<?>($PLN){1}<?>), if ID($PLN)=1
15. \B(<?>($PEN){1}<?>), if ID($PEN)=1
    and Count(LCase($PEN))>0 and
    Count(\M($PEN))>0
16. \B(<?>($PEN){1}<?>), if ID($PEN)=1
    and Count(LCase($PEN))=0 and
    Count(\M($PEN))=0
17. <acronym>$ACR</acronym>
18. <contraction>$CON</contraction>
19. (<f>$FLW</f>){1,}
20. (<f>$FLW</f>){1,}(<f>$AMW</f>){1,3}
    (<f>$FLW</f>){1,}, if $AMW{1,3} in
    @FLW and $AMW{1,3} in @TT
21. (<f>$FLW</f>){1,}(<?>$AMW<?>){1,3}
    (<f>$FLW</f>){1,}, if $AMW{1,3} not
    in @FLW and $AMW{1,3} in @TT
```

[4] <?> denotes the annotation case which should be resolved by a human (in semi-automatic annotation mode).

22. (<f>$FLW</f>){1,}(<f>$ACR</f>){1} (<f>$FLW</f>){1,}
23. (<f>$FLW</f>){1,}(<f>$DD{1,}</f>) (<f>$FLW</f>){1,}
24. (<f>$FLW</ f>){1,}(<f>$PEN</f>){1,3} (<f>$FLW</f>){1,}
25. (<f>$FLW</f>){1,}(<f>$PLN</f>) (<f>$FLW</f>){1,}
26. ([()][5] or [‟] or [‘]) (<f>$AMW</f>){1,2} (<f>$FLW</f>){1,}, if $AMW{1,2} in @FLW and $AMW{1,2} in @TT
27. (<f>$FLW</f>){1,} (<f>$AMW</f>){1,2} ([)] or [”] or [’]), if $AMW{1,2} in @FLW and $AMW{1,2} in @TT
28. ([() or [‟] or [‘]) (<f>DD</f>){1,} (<f>$FLW</f>){1,}
29. (<f>$FLW</f>){1,} (<f>$DD{1,}</f>){1,}([)] or [”] or [’])
30. ([() or [‟] or [‘]) (<f>$PEN</f>){1,2} (<f>$FLW</f>){1,}
31. (<f>$FLW</f>){1,} (<f>$PEN</f>){1,2}([)] or [”] or [’])
32. ([() or [‟] or [‘]) (<f>$PLN</f>){1} (<f>$FLW</f>){1,}
33. (<f>$FLW</f>){1,} (<f>$PLN</f>){1} ([)] or [”] or [’])
34. \A(<f>$TT</f>){1,2} (<f>$FLW</f>){1,}, if $TT in $FLW
35. (<f>$FLW</f>){1,} (<f>$TT</f>){1,2}\Z, if $TT in $FLW
36. (<?>$TT</?>){1,2} (<f>$FLW</f>){1,}, if Length($TT) < 4
37. (<f>$FLW</f>){1,} (<?>$TT</?>){1,2}, if Length($TT) < 4

Rules for automatic annotation of dates:

1. [()<d>$DD{4} [-] $DD{4}</d>[)]
2. <d>($DD{4})$_1$ [-] ($DD{4})$_2$</d>, if ($DD{4})$_2$- ($DD{4})$_1$< 100
3. <?>($DD{4})$_1$ [-] ($DD{4})$_2$<?>, if ($DD{4})$_2$ - ($DD{4})$_1$≥ 100
4. [()<d>$DD{4}</d>[)]
5. <d>($PRE{1} $PAR{0,1}){0,1} (($DD{1,} or $DL{1} or $DD{1,} $END{1}){1,} ($PRE{1} $PAR{0,1} or [,]){1}){1,} ($AWY{1} or $SW{1} or $WY{1} or $WCE{1}){1}</d>
6. <d>($PRE{1} $PAR{0,1}){0,1} $DD{4} ([.] or [ ]) ($DD{1,2} or $RD{1,4}){1} (([.] or [ ]) $DD{1,2} $SW{0,1}){0,1}</d>
7. <d>($PRE{1} $PAR{0,1}){0,1} $YM{1} $WMO{0,1} ($DD{1,2} ([-] $END{1}) {0,1} or $DD{1,2}){1} $SW{0,1}</d>
8. <d>($PRE{1} $PAR{0,1}){0,1} ($DD{1,} or ($DD{1,}[-] $END{1}){1} or $RD{1,} or $DL{1} or $ABD{1}){1,} ($WCE{1} $CCE{0,1}){1} (($DD{1,} or $RD{1,} or $DL{1} or $ABD{1}){0,1} $AWY{1}){0,1} ($DD{1,} or (($DD{1,}[-] $END{1}){1} or $RD{1,} or $DL{1} or $ABD{1}){1,} $WW{1}){0,1}</d>
9. <d>($PRE{1} $PAR{0,1}){0,1} ($DD{1,} or ($DD{1,}[-] $END{1}){1} or $RD{1,} or $DL{1} or $ABD{1}){0,1} $CCE{0,1} ($AWY{1} or $WY{1} or $HD{1}$WY{1}){1} $CCE{0,1} ($YS{1} or $YM{1} $WMO{0,1}){0,1} (($DD{1,} or ($DD{1,}[-] $END{1}){1} or $RD{1,} or $DL{1}){1,} $SW{0,1}){0,1} $WW{0,1}</d>

Rules for automatic annotation of sentence boundaries:

1. \A<s>$SEP{0,}$SUC{1,}
2. \A<s>$SEP{0,}$SLC{1,}
3. \A<s>$SEP{0,}$DD{1,}
4. $SUC{1,}$SEP{0,} </s>\Z
5. $SLC{1,}$SEP{0,} </s>\Z
6. $DD{1,}$SEP{0,} </s>\Z
7. $SLC{1,}$SEP{0,}$USS{1,}</s><s>$SEP{0,} $SUC{1,}
8. $CAT{1}</s><s>$TT{1}
9. $CAT{1}<?>$PEN{1} or $CAT{1}<?>$PLN{1}
10. $DD{1,}$SEP{0,}$USS{1,}<?>$SEP{0,} $SUC{1,}
11. $SLC{1,} SEP{0,}$USS{1,}<?>$SEP{0,}$DD{1,}
12. $SUC{1,} $SEP{0,}$USS{1,}<?>$SEP{0,}$DD{1,}
13. $DD{1,} $SEP{0,}$USS{1,}<?>$SEP{0,}$DD{1,}

[5] Separators are enclosed within brackets to distinguish them from separators used to specify annotation rules.

DOI: 10.5755/j01.itc.34.3.12012