A PART-OF-SPEECH TAGGING BASED ON SUBTRACTION OF WORD-BASED REGULAR LANGUAGES

Aleksey Cheusov

Belorussian State University, Minsk, Belarus, e-mail: *vle@gmx.net*

Abstract. This paper presents an approach of a part-of-speech (POS) tagging based on subtraction of regular languages [2]. The basic idea is to implement a POS tagging by subtracting regular languages. The minuend is a language of finite state automaton representing an input sentence with assigned tags from a lexicon. The subtrahend is a language of a set of constraint rules written in word-based regular expression notation [1].

Description

Many statistical taggers tend to assign a single tag to each word from the sentence, that is, tagging may be defined as $T{:}S \to S^T$, where $S=(w_1,w_2,\ldots,w_N)$, $S^T=((w_1,t_1),(w_2,t_2),¼,(w_N,t_N))$, w_i is a word, t_i is a POS tag. A serious disadvantage of this approach is that often it is not possible to assign a single tag to an ambiguous word because the sentence itself is ambiguous, that is, the sentence has a number of correct tagging variants. Consider the sentence “Flying planes may be dangerous” as an example. It can be tagged in two ways. The word “flying” can be tagged either a gerund or an adjective. The simplest way to solve this problem is to keep one or several words from the sentence ambiguous, leaving final disambiguation for subsequent modules such as a parser. Many rule-based taggers work this way. But this approach is neither exempt of disadvantages. At worst, many words (potentially all) can be ambiguous, thus slowing down the parser. Another disadvantage is a significant complexity of creating a robust set of rules being created either manually or automatically.

Assuming that Σ is a POS tagset, the sentence $S=(w_1,w_2,\text{j}\ ,\omega_{\text{N}})$ where a set of tags $t_i \in 2^\Sigma (1 \le i \le N)$ (obtained from a lexicon or an unknown word guessing module) is assigned to each word, can be viewed as FSA $F^S=<Q,\Sigma,q,f,R>$, where $Q=\{0,1,\ldots,N\}$, initial state $q=0$, final state $f=N$ and a set of rules $R=\{\{s\text{-}1\}\times\{k_s\}\to\{s\}:1\le s\le N,k_s\in t_s\}$.

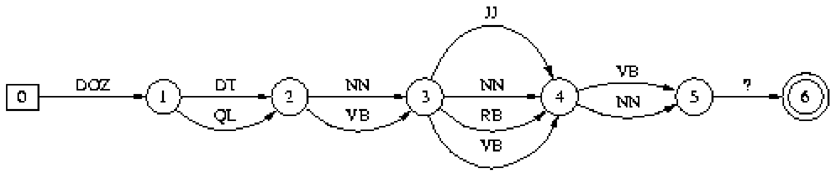

Figure 1: *Automaton F^S for the sentence “Does this program still work?”.*

See Figure 1 for example. A language of this automaton $L(F^S)$ is a set of all possible tagging variants of the given sentence , the number of which (actually the number of paths from initial state to the finite one in the automaton) is equal to $\prod_{i=1}^{N} |t_i|$, most of which are syntactically or semantically incorrect. To specify such incorrectly tagged variants a set of regular expressions $R=(R_1,R_2,\ldots,R_M)$ is created based on WRE [1] notation. Each rule matches the tagged sentences that contain incorrectness(es) of a certain type. Below are examples of such rules using LOB [9] POS tagset.

```
  # this rule matches the tagged sentences
  # beginning with words ``do'', ``does'' or
  # ``did'' and having no verb in its
  # infinitive form
  ^ DO|DOZ|DOD !HV!VB!BE + $

  # the following matches tagged sentences containing
  # quantifier followed by the verb,
  # m_any_verb_form is a macro expanding to BE|HV|VB|VBZ etc.
  QL m_any_verb_form

  # the following rule matches tagged sentences containing
  # verb excluding link verb followed by adjective which in turn
is
  # followed by the verb.
  m_any_verb_form/!@link_verb JJ m_any_verb_form

  # the following rule matches sentences containing two successive
verbs
  # on condition that the first one is not ``help'' and
  # that there is only one noun phrase which stands before them.
  # m_noun_phrase is a macro expanding to noun phrase,
  # pn_nn_adj is the macro expanding to any agent pronoun,
  # noun or adjective.
  ^ (. - m_pn_nn_adj) * m_noun_phrase (. - m_pn_nn_adj) *
VB/!"help" RB ? VB

  #
  QL NN * !JJ
```

Figure 2: *WRE-based constraint rules*

So that, the tagging is now $T{:}S\rightarrow S^T$, where $S^T=L(F^S)-\cup_{i=1}^{M}L(R_i)$. This approach is obviously more flexible than conventional approaches because it allows to remove from the language $L(F^S)$ any particular tagging variant. As shown in [7], WRE grammar can be converted to an equivalent finite state automaton, therefore the subtraction of regular languages can be done by “subtracting” corresponding finite state automata. In view of the fact that the minuend F^S is acyclic and deterministic, an easy subtraction algorithm can be built. As a result of this subtraction an acyclic automaton F^T specifying all correct (matched by none of the constraint rules from *R*) tagging variants, is obtained. See Figure 3 for example.

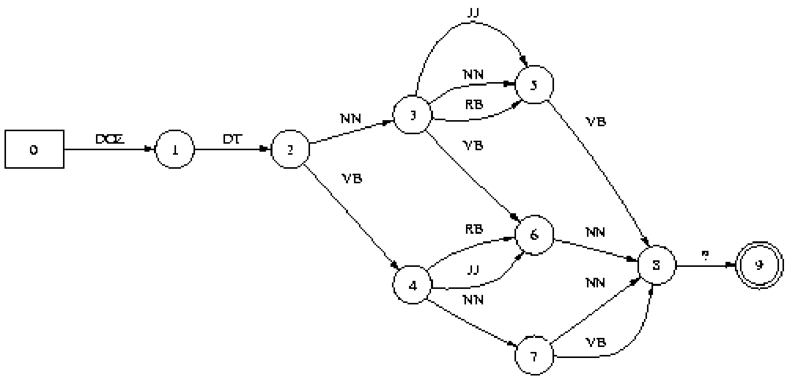

Figure 3: *Automaton having $L(F^S)-\cup_{i=1}^{M} L(R_i)$ language for the sentence "Does this program still work?"*

If a rule set *R* is complete enough and analyzed sentence is unambiguous, only one path from the initial state to the finite one exists. However, potentially several paths are possible. In this case the resulting automaton can be analyzed later on, for example, by a statistical tagger or by a parser able to process automaton, for instance, finite state transducer based parser as described in [8]. At worst, the complexity of this algorithm is $O(|\Sigma|^N)$, but in practice it is fast enough for use in productional systems aimed at processing huge amounts of texts. Another advantage of this approach is that manual creation of constraint rules is significantly easier (at least, for fixed word order languages) than using techniques used in several other systems such as [3, 4, 5, 6], because powerful and compact WRE notation allows to create less categorical and therefore less error-prone rules.

References

[1]. Aleksey Cheusov "The Word-based Regular Expressions in Computational Linguistics", *Proceedings of the seventh International conference "Pattern recognition and information processing"*, Minsk (2003).

[2]. W.Brauer, "Eine Enführung in die Theorie endlicher Automaten", *Stuttgart*, (1984).

[3]. Barbara B. Green, Gerald M. Rubin, "Automated grammatical tagging of English", *Department of Linguistics*, Brown University, (1971).

[4]. Z. Harris, "String analysis of sentence structure", *Language*, 22(3), 161-183, (1962).

[5]. Fred Karlsson, Atro Voutilainen et al., "Constraint Grammar", *Mouton de Gruytor*, Berlin, (1995).

[6]. Emmanuel Roche, Yves Schabes, "Deterministic part-of-speech tagging with finite state transducers", Mitsubishi Electric Research Laboratories, (1995).

[7]. Denis Postanogov, "Effective implementation of word-based regular expressions notation in natural language processing", *Proceedings of the seventh International conference "Pattern recognition and information processing"*, Minsk, (2003).

[8]. Emmanuel Roche, "Parsing with Finite State Transducers", Technical Report, Mitsubishi Electric Research Laboratories, (1996).

[9]. http://www.scs.leeds.ac.uk/amalgam/tagsets/lob.html