Some applications of a statistical tagger for Swedish

Johan Carlberger* Viggo Kann*

Keywords: part-of-speech tagging, statistical tagging, grammar checking, word prediction.

We will briefly describe a part-of-speech (POS) tagger for Swedish and discuss some applications: rule-based and probabilistic grammar checking, word prediction and keyword extraction.

In POS tagging of a text, each word and punctuation mark in the text is assigned a morphosyntactic tag. We have designed and implemented a tagger based on a second order Hidden Markov Model [1]. Given a sequence of words $w_{1..n}$, the model finds the most probable sequence of tags $t_{1..n}$ according to the equation:

$$\mathcal{T}(w_{1..n}) = \arg\max_{t_{1..n}} \prod_{i=1}^{n} P(t_i|t_{i-2}, t_{i-1})P(w_i|t_i). \tag{1}$$

Estimations of the two probabilities in this equation are based on the interpolation of relative counts of sequences of 1, 2 and 3 tags and word-tag pairs extracted from a large tagged corpus.

For unknown words, we use a statistical morphological analysis adequate for Swedish and other moderately inflecting languages. This analysis is based on relative counts of observed tags for word types ending with the same 1 to 5 letters. This captures both inflections (tense *-ade* in *hämtade* (fetched)) and derivations (nounification *-ning* in *hämtning* (pick-up)).

We also perform an analysis that finds the last word form of compounds, which are common in Swedish. The possible tags of the last word form indicates possible tags (and probability estimation) for an unknown compound word. These two analyses are heuristically combined to get estimations of $P(w_i|t_i)$, which enables unknown words to work in the model. This method combines morphological information for unknown words with contextual information of surrounding words, and resulted in a tagger that tags 98 % of known and 93 % of unknown words correctly [2].

Grammar checking

The tagger was developed to be a part of a program for grammar checking of Swedish text. The text is first tagged, and then checked for grammatical errors by so called error rules. An agreement error detecting rule could for example state that if a determiner is followed by any number of adjectives and then a noun where the noun disagrees in gender, then the determiner should be corrected, and the correct inflection should be suggested. The grammar checker is efficient and has good recall and precision [3]. At `http://www.nada.kth.se/theory/projects/granska/` there is a web version of the tagger and grammar checker.

By letting the grammar checker use a syntacticly disambiguated instead of an undisambiguated text, error rule writing becomes easier and tests showed an increase in both recall and precision. The drawback is when incorrect tagging causes false alarms, but such can be avoided to a great extent by writing "tagging correction rules" and by other means.

It is inefficient to try each error rule at each position in the text. We therefore perform a statistical optimization, where each rule is analyzed in advance. For each position in the rule the possible matching words and taggings (tag pairs) are computed. Then, using statistics on word and tag pair relative frequencies, the position of the rule that is least probable to match a Swedish text is determined. This means that this rule is checked by the matcher *only* at the positions in the text where the words or tag pairs of this least probable position in the rule occur.

Probabilistic grammar and spell checking

Using error rules in grammar checking makes it hard to cover all possible combinations of ungrammatical constructions. This fact led us to investigate a method for finding errors with a probabilistic approach.

A suspicious sentence is identified using some measure related to the probability given by Equation 1. Changes are made to the sentence in order to make it less suspicious, and if a good enough alternative sentence is found, it is suggested as an alternative to the original sentence. This

*Address: Nada, Numerical Analysis and Computing Science, Royal Institute of Technology, SE–100 44 Stockholm, SWEDEN. Fax +46-8-7900930. E-mail `jfc@nada.kth.se`, `viggo@nada.kth.se`

method may capture not only errors that spell checkers findm, but also errors where the misspelled word happens to be a word in them word list, or when adjacent are words interchanged, or when a is omitted, or when a an extra word has slipped into the sentence by mistake.

The simple algorithm we devised successfully identified and corrected errors of these types in very short sentences, but whether this method can successfully treat more complicated sentences remains to be investigated. Anyhow, the approach seems promising, at least for identifying suspicious sentences.

Keyword extraction

A problem with indexing of documents and keyword extraction is that words appear in different forms and as parts of compound words. In a typical web search engine, if one uses the noun *lagar* (laws) as a keyword, one will not find documents containing the word *lag* (law) or *grundlagar* (constitutional laws). A solution to this problem could be to index documents with the base forms of occurring words and use the base forms of keywords as actual keywords.

This approach will improve coverage, but precision can be further improved. Not knowing the syntactic function of an occurrence of *lagar* in a text, the possible base forms are *lag* (law) as well as the verb *laga* (mend). To improve precision, a better solution is to tag the texts and only use base forms implied by the tagging.

With the compound analysis algorithm included in our tagger, compound words can also be mapped to their correct base forms as well as to the base forms of their constituents. For example, *grundlagar* is mapped to *grund*, *lag* and *grundlag*.

Word prediction

Word prediction is the problem of guessing which word will appear after a given sequence of words. For example, the word following *The boy will not* is more likely *be* than *been*. A word prediction algorithm ranks the words in a lexicon according to an estimated probability of each word to appear after a given sequence of words.

As the tagging algorithm estimates probabilities of sequences of words, it can be used as for word prediction. Lexicon words are ranked by the probabilities given by tagging the given sequence of words with each lexicon word inserted at the end. However, this an inefficient solution, since there are as many different sequences to examine as there are lexicon words.

In most applications, only the most probable words are of interest, and there can be restrictions on what words are selectable. A more efficient approach would therefore be to only investigate words that are among the most probable ones. Our solution involves two simple steps: First, categorize words according to restriction criterion and possible tag. Second, select the most common in each category and group these words by restriction criterion, and use only selected words in the algorithm. This solution effectively reduces the number of words to consider. Since $P(w|t)$ is the only term of Equation 1 that depends on w, this method gives (almost) the same result as the naive approach.

The algorithm was improved by including $P(w_i|w_{i-1})$ in the tagging equation, by promoting previously occurring words in the text and by adjusting the probability estimations for unknown word for more suitable rank among known words. An implementation used for speeding up typing showed to save almost 50 % of keystrokes.

References

[1] Charniak, E., Hendrickson, C., Jacobson, N., Perkowitz, M. (1993). Equations for part-of-speech tagging. In *11th National Conf. Artificial Intelligence*, pages 784–789.

[2] Carlberger, J., Kann, V. (1999). Implementing an Efficient Part-Of-Speech Tagger. In *Softw. Pract. Exper.* 29(9), pages 815–832.

[3] Domeij, R., Knutsson, O., Carlberger, J., Kann, V. (2000). Granska – an efficient hybrid system for Swedish grammar checking. To appear in Proc. *Nodalida-99*.