Tools: Constraint Grammars PV110 Essay, Autumn 2012

Zdeňka Sitová, 324709

Parsing running text is an important task of NLP. The goal of this task is to assign a morphological and a syntactic feature to each word-form of the text. The biggest problem of assigning morphological or syntactic feature to a word-form is in the fact, that almost every word in the lexicon can have more than one feature. For example, a word *stop* can be either noun or verb; similar examples can be found for the morphology. The process of deciding, which feature should be chosen, is called **disambiguation**. Disambiguation is strongly dependent on the context, in which a word-form occurs. A **constraint grammar** is an ensemble of constraints, which define acceptable/unacceptable features (syntactic or morphological) for the word-form, according to its context and to the lexicon.

In this essay, I will start with a short review of Fred Karlsson's article about constraint grammars [Kar90]. In the second part, I will discuss usage of machine learning techniques for learning rules for constraint grammars.

1 Constraint grammar as a framework for parsing running text: a short review

This section is based on the article by Fred Karlsson [Kar90].

The process of parsing running text can be broken up into six subprocesses:

1. Preprocessing
2. Morphological analysis
3. Local morphological disambiguation
4. Morphosyntactic mapping
5. "Syntax proper"
   (a) Context-dependent morphological disambiguation
   (b) Determination of intrasentential caluse boundaries
   (c) Disambiguation of surface syntactic function

Constraint grammar corresponds to the fifth stage, "syntax proper". Optimally, all the stages should be executed sequentially, followed by paralell execution of three substages of the fifth stage.

A **cohort** is a set of readings for a word-form. Karlsson mentions some examples of local disambiguation strategies. There is a strategy, in which all reading with more than the smallest number of compound boundaries in current cohort are discarded. It means, that from possible readings of Swedish word *frukosten*, readings *fru-kost-en* and *fru-ko-sten* would be discarded; only reading

frukosten would be left in a cohort. Other examples of strategies are based on probabilities of the part-of-speech structure (e.g., NNN is much less probable than NVN).

Local disambiguation can be quite powerful in terms of reducing the number of readings. Karlsson mentions an example with a Swedish text of 840 000 word-form tokens: there were 3 830 word-forms with 6 reading before local disambiguation, 312 after it.

Karlsson implemented a constraint grammar parser in Lisp. In this implementation, a constraint is a quadruple `(domain, target, operator, context condition)`.

For example, a rule `(@w=0 "PREP" (-1 DET))` says "Discard a `PREP` reading from a cohort of any word-form, which is preceded by `DET`."

In this example,
Domain is `@w` (= any word-form)
Target reading is `PREP` (= preposition)
Operator is `=0` (= discard target reading)
Context condition is `-1 DET` (= there should be a determiner on the position before the target)

Unbounded dependencies are also allowed (e.g., `*1 VFIN` refers to `VFIN`, which can occur anywhere to the right of the target). There are three possible operators in Karlsson's constraint grammar:

- `=0` - discards target reading
- `=!` - iff all conditions are satisfied, this reading is correct
- `=!!` - iff all conditions are satisfied, this reading is correct, discard this reading otherwise

Constraints are not usually applied over sentence boundaries, but there is a possibility to do so. There are similar rules to decide, where is a clause boundary. For example, constraint `(@w=**CLB "<Conj>" (1 NOMHEAD)(2 VFIN))` would mean "There is a clause boundary before conjunction in a sentence such as *John eats and Bill drinks.*"

Syntactic constraints are similar to the constraints mentioned before. These constraints benefit from so-called **uniqueness principle**, which says that many syntactic features can occur only once in the clause. A typical example of such a feature is the subject.

Syntactic labels are assigned in three steps:

1. all the possible syntactic labels from the lexicon + morphological information are assigned (e.g. *he* can be a subject, *him* can be an object, ...),
2. morphosyntactic mapping, which takes into account context, as well as the morphological features, and

3. syntactic constrains, which are similar to context-dependent disambiguation, only they work with syntactic features and there is a `s` before the operator (e.g. `(@w=s0 "@+FMAINV" (*-1 VFIN))` says, that all the `@+FMAINV` readings should be discarded as a syntactic alternative, if there is a unique finite main verb to the left in the same clause).

Karlsson mentions, that he expects, that there will be only about 500 constraints needed for disambiguation in English.

2 Learning constraint-grammar style rules using Inductive Logic Programming

In Lindberg's article [Lin98], using Progol for learning constraint-grammar style rules is discussed. Progol is an implementation of Inductive Logic Programming.

Lindberg has tested Progol on a Swedish corpus of one million words. 7000 rules were induced, when tested on unseen data (42 925 words), 98 % of the words retained the correct tag. There were still some ambiguities left in the output, about 1.13 readings per a word.

In Lindberg's experiment, no grammatical background knowledge was given to the learner. Context has been limited to a window of maximally five words, when target word is in the middle of the window. Rules were induced for all Part-Of-Speech categories; within the scope of the window, the rule can look at all the word-forms and their Part-Of-Speech, morphological features, word-form itself and at the fact, whether there is an uppercase character in the word-form. For each of 24 Part-Of-Speech categories, a different set of training data was produced. A positive example is when a word is incorrectly tagged and the reading should be discarded; a negative example is correctly tagged word, for which the reading should be retained. For each Part-Of-Speech, there were about 4000 - 6000 randomly generated positive examples, with an equivalent number of negative examples.

An example of an induced rule is:
`remove(vb,A) :- constr(A,left,feats([dt]),`
which means that a word should not be read as a verb (`vb`) (i.e. this reading should be discarded), when it is immediately preceded by a word tagged as determiner (`dt`).

Lindberg compares his experiment with other taggers, trained on the same corpus: Brill tagger tagged 96.9 % words correctly, Oliver Mason's QTaggot tagged 96.3 % correctly. Neither of these taggers did leave any ambiguities though. Also, these taggers work with unknown words, which is not the case of Lindberg's tagger.

References

[Kar90] Fred Karlsson, *Constraint grammar as a framework for parsing running text*, Proceedings of the 13th conference on Computational linguistics

- Volume 3 (Stroudsburg, PA, USA), COLING '90, Association for Computational Linguistics, 1990, pp. 168–173.

[Lin98] Nikolaj Lindberg, *Learning constraint grammar-style disambiguation rules using inductive logic programming*, In Proc. COLING/ACL98, 1998, pp. 775–779.