Computational Linguistics: Summing up

Raffaella Bernardi

KRDB, Free University of Bozen-Bolzano

P.zza Domenicani, Room: 2.28, e-mail: bernardi@inf.unibz.it

Contents

1 Goals and Challenges 3
2 Morphology 4
3 Syntax 5
3.1 CFG for NL 6
3.2 Feature Structures 7
3.3 Parsing Techniques 8
4 Semantics 9
5 Discourse 10
6 LCT Colloquia 11

1. Goals and Challenges

Goals : **Ultimate goal**: To build computer systems that perform as well at using natural language as humans do. **Immediate goal** To build computer systems that can process text and speech more intelligently.

Challenges : deal with ambiguities at all levels (phonology, morphology, syntax, semantics, discourse, pragmatics.)

2. Morphology

We've seen that

- Morphology deals with the inner structure of words. Words consist of morphemes which can be composed together to form new words in two different ways, inflectional (same category) and derivational (new category) forms.
- one of the most common way to model morphotactics (the model of the morpheme ordering) is by means of **Finite State Automata** (FSA).
- FSA recognize/generate Regular Languages.
- Finite-state techniques cannot be used to model all aspects of NL; but for tasks where they do apply, they are extremely attractive. In fact, the flip side of their **expressive weakness** being that they usually behave very well computationally. If you can find a solution based on finite state methods, your implementation will probably be **efficient**.

3. Syntax

We have seen

- at the syntactic level, NL can be proved to be non Regular Language. (e.g. nested dependencies, **if ... then**.
- how to use Context Free Grammar to parse linguistic strings.
- how to use feature structures and unification to deal with agreement.
- different parsing techniques

We left open the two questions below

- **Is NL a Context Free Language or do we need a more expressive Formal Grammar?**
- **Can CFG deal with long-distance dependencies?**

3.1. CFG for NL

- Terminal: The terminal symbols are **words** (e.g. sara, dress ...).
- Non-terminal: The non-terminal symbols are **syntactic categories** (CAT) (e.g. *np*, *vp*, ...).
- Start symbol: The start symbol is the s and stands for sentence.

The production rules are divided into:

- Lexicon: They are of the form $np \rightarrow$ sara. They form the set LEX
- Grammatical Rules: They are of the type $s \rightarrow np\ vp$.

3.2. Feature Structures

We have used **attribute-value matrix (AVM)** to add agreement information to categories. E.g.

$$\begin{bmatrix} \texttt{CAT} & np \\ \texttt{AGR} & \begin{bmatrix} \texttt{NUM} & sg \\ \texttt{PERS} & 3 \end{bmatrix} \end{bmatrix}$$

as well as **sucategorization** information, e.g

$$\begin{bmatrix} \texttt{ORTH} & want \\ \texttt{CAT} & verb \\ \texttt{HEAD} & \begin{bmatrix} \texttt{SUBCAT} & \langle [\texttt{CAT}\ np], \begin{bmatrix} \texttt{CAT} & vp \\ \texttt{HEAD} & [\texttt{VFORM}\ INFINITIVE] \end{bmatrix} \rangle \end{bmatrix} \end{bmatrix}$$

We have used unification to check feature matching.

3.3. Parsing Techniques

Bottom up we **begin** with the concrete data provided by the input string — that is, the **words** we have to parse/recognize — and try to build bigger and bigger pieces of structure using this information. **We use our CFG rules right to left**.

Top down We **start at the most abstract level** (the level of sentences) and work down to the most concrete level (the level of words). **We use the CFG rule left to right.**

Depth first Search whenever there is more than one rule that could be applied at one point, we **explore one possibility and only look at the others when this one fails**.

Breadth first search we **carry out all possible choices at once**, instead of just picking one.

Left-corner parser we start with a **top-down prediction** fixing the category that is to be recognized, like for example s. Next, we take a **bottom-up step** and then alternates bottom-up and top-down steps until we have reached an s.

4. Semantics

We have seen that in Formal Semantics

1. The meaning of a sentence is considered to be its truth value
2. Its meaning is built compositionally starting from the lexicon, its represented by means of lambda terms and assembled by means of the lambda calculus.

We left open the following questions:

- How do we infer some piece of information out of another?
- **How do syntax and semantics relate?**

5. Discourse

We have looked at

- challenging problems for Discourse Model
- challenging problems for representation of Discourse Structures
- challenging problems for compositionally building DS.

We only mentioned Discourse Representation Theory.

6. LCT Colloquia

This afternoon:

Judith Knapp (EURAC)	**The application of CL tools for computer assisted language learning: Experiences with WordManager**
Time	16:00-17:00
Place	CS Seminar Room

Abstract available from the LCT Colloquia page: http://www.inf.unibz.it/mcs/lct/seminars.php