Creating a Shallow-Parsed Hungarian Corpus with NooJ
Kata Gábor
Linguistics Institute, HAS

Introduction

In recent years unsupervised or semi-supervised learning of lexical syntactic or semantic information from corpora has gained more importance, especially for languages which lack extensive (hand-made) linguistic resources for NLP applications. However, the extraction of relevant lexical semantic information necessitates the use of big, (partially) parsed corpora. The only syntactically annotated Hungarian corpus, the Szeged Treebank, is limited in size, thus unsuitable for extracting lexical information of less frequent words. The creation of bigger, annotated corpora requires automatic parsing. The parser has to meet the following conditions:

- It has to robust enough to be applicable to big texts which cover several thematic domains.
- It has to give relevant output even if the complete analysis of the sentence is not available (i.e., it can be used as a chunker).

Basic Data

| Corpus size | 10.85M words |
| Text source | Hungarian National Corpus [Váradi, 2002] |

Morphological analysis: external analyser used in HNC (Humor), converted to NooJ dictionary and imported to NooJ.

Ambiguities: HNC is POS-tagged, but NooJ dictionaries reproduce the ambiguity. However, a list of unambiguous word forms or preferred analyses (+UNAMB) was added to the dictionary.

Structure: Chunking and dependency

Both input and output features are purely syntactical. Systematic ambiguities (e.g., past participles – finite verbs, postpositions – inflected nouns) are solved inside chunking grammars.

Chunking is performed by local grammars – implemented as FSTs – with a high precision.

Dependency annotation on the other hand benefits from NooJ’s enhancements, especially the lexical constraints. In order to be able to make use of this function, Hungarian NooJ dictionaries were completed by a set of lexical syntactic features. They define finer grained distributional categories than POS, making it possible to achieve higher precision than simple local grammars.

Lexical constraints in Dependency Parsing

Besides their usefulness in the precise formulation of syntactic rules, lexical constraints also provide very efficient ways to handle agreement and long-distance dependencies.

- PREP/POSTP + NP → morphosyntactic (defined in the context)
- V + Subj in number → morphosyntactic (defined in the context)
- V + Obj in definiteness → lexical
- V + Complements → long-distance

Evaluation

Gold standard: 53 sentences, manually annotated (chunks) from two different literary sources:

| Phrases in gold standard | 544 |
| Phrases in test set | 656 |
| Number of correct phrases | 420 |
| Precision | 64% |
| Recall | 79% |
| Partial matches (incl. head) | 516 |
| Precision | 78% |
| Recall | 94% |

The F-score of 57.78% reported by [Váradi, 2003] is now up to 70.3% in the NooJ-parsed corpus.

Argument Labeling

Both input and output features are purely syntactical. Systematic ambiguities (e.g., past participles – finite verbs, postpositions – inflected nouns) are solved inside chunking grammars.

Chunking is performed by local grammars – implemented as FSTs – with a high precision.

Dependency annotation on the other hand benefits from NooJ’s enhancements, especially the lexical constraints. In order to be able to make use of this function, Hungarian NooJ dictionaries were completed by a set of lexical syntactic features. They define finer grained distributional categories than POS, making it possible to achieve higher precision than simple local grammars.

References