Language Independent Probabilistic Context-Free Parsing Bolstered by Machine Learning

Michael Schiehlen Kristina Spranger
Institute for Computational Linguistics
University of Stuttgart
D-70174 Stuttgart
Michael.Schiehlen@ims.uni-stuttgart.de
Kristina.Spranger@ims.uni-stuttgart.de

Abstract

Unlexicalized probabilistic context-free parsing is a general and flexible approach that sometimes reaches competitive results in multilingual dependency parsing even if a minimum of language-specific information is supplied. Furthermore, integrating parser results (good at long dependencies) and tagger results (good at short range dependencies, and more easily adaptable to treebank peculiarities) gives competitive results in all languages.

1 Introduction

Unlexicalized probabilistic context-free parsing is a simple and flexible approach that nevertheless has shown good performance (Klein and Manning, 2003). We applied this approach to the shared task (Buchholz et al., 2006) for Arabic (Hajič et al., 2004), Chinese (Chen et al., 2003), Czech (Böhmová et al., 2003), Danish (Kromann, 2003), Dutch (van der Beek et al., 2002), German (Brants et al., 2002), Japanese (Kawata and Bartels, 2000), Portuguese (Afonso et al., 2002), Slovene (Džeroski et al., 2006), Spanish (Civit Torruella and Martí Antonín, 2002), Swedish (Nilsson et al., 2005), Turkish (Oflazer et al., 2003; Atalay et al., 2003), but not Bulgarian (Simov et al., 2005). In our approach we put special emphasis on language independence: We did not use any extraneous knowledge; we did not do any transformations on the treebanks; we restricted language-specific parameters to a small, easily manageable set (a classification of dependency relations into complements, adjuncts, and conjuncts/coordinators, and a switch for Japanese to include coarse POS tag information, see section 3.4). In a series of post-submission experiments, we investigated how much the parse results can help a machine learner.

2 Experimental Setup

For development, we chose the initial n sentences of every treebank, where n is the number of the sentences in the test set. In this way, the sizes were realistic for the task. For parsing the test data, we added the development set to the training set.

All the evaluations on the test sets were performed with the evaluation script supplied by the conference organizers. For development, we used labelled F-score computed from all tokens except the ones employed for punctuation (cf. section 3.2).

3 Context Free Parsing

3.1 The Parser

Basically, we investigated the performance of a straightforward unlexicalized statistical parser, viz. BitPar (Schmid, 2004). BitPar is a CKY parser that uses bit vectors for efficient representation of the chart and its items. If frequencies for the grammatical and lexical rules in a training set are available, BitPar uses the Viterbi algorithm to extract the most probable parse tree (according to PCFG) from the chart.

Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL-X),
pages 231–235, New York City, June 2006.

3.2 Converting Dependency Structure to Constituency Structure

In order to determine the grammar rules required by the context-free parser, the dependency trees in the CONLL format have to be converted to constituency trees. Gaifman (1965) proved that projective dependency grammars can be mapped to context-free grammars. The main information that needs to be added in going from dependency to constituency structure is the category of non-terminals. The usage of special knowledge bases to determine projections of categories (Xia and Palmer, 2001) would have presupposed language-dependent knowledge, so we investigated two other options: Flat rules (Collins et al., 1999) and binary rules. In the flat rules approach, each lexical category projects to exactly one phrasal category, and every projection chain has a length of at most one. The binary rules approach makes use of the X-bar-scheme and thus introduces along with the phrasal category an intermediate category. The phrasal category must not occur more than once in a projection chain, and a projection chain must not end in an intermediate category. In both approaches, projection is only triggered if dependents are present; in case a category occurs as a dependent itself, no projection is required. In coordination structures, the parent category is copied from that of the last conjunct.

Non-projective relations can be treated as unbounded dependencies so that their surface position (antecedent position) is related to the position of their head (trace position) with an explicit coindexed trace (like in the Penn treebank). To find the position of trace and antecedent we assume three constraints: The antecedent should c-command its trace. The antecedent is maximally near to the trace in depth of embedding. The trace is maximally near to the antecedent in surface order.

Finally the placement of punctuation signs has a major impact on the performance of a parser (Collins et al., 1999). In most of the treebanks, not much effort is invested into the treatment of punctuation. Sometimes, punctuation signs play a role in predicate-argument structure (commas acting as coordinators), but more often they do not, in which case they are marked by special roles (e.g. “pnct”, “punct”, “PUNC”, or “PUNCT”). We used a general mechanism to re-insert such signs, for all languages but CH (no punctuation signs) and AR, CZ, SL (reliable annotation). Correct placement of punctuation presupposes knowledge of the punctuation rules valid in a language. In the interest of generality, we opted for a suboptimal solution: Punctuation signs are inserted in the highest possible position in a tree.

3.3 Subcategorization and Coordination

The most important language-specific information that we made use of was a classification of dependency relations into complements, coordinators/conjuncts, and other relations (adjuncts).

Given knowledge about complement relations, it is fairly easy to construct subcategorization frames for word occurrences: A subcategorization frame is simply the set of the complement relations by which dependents are attached to the word. To give the parser access to these lists, we annotated the category of a subcategorizing word with its subcategorization frame. In this way, the parser can learn to associate the subcategorization requirements of a word with its local syntactic context (Schiehlen, 2004).

Coordination constructions are marked either in the conjuncts (CH, CZ, DA, DU, GE, PO, SW) or the coordinator (AR, SL). If conjuncts show coordination, a common representation of asyndetic coordination has one conjunct point to another conjunct. It is therefore important to distinguish coordinators from conjuncts. Coordinators are either singled out by special dependency relations (DA, PO, SW) or by their POS tags (CH, DU). In German, the first conjunct phrase is merged with the whole coordinated phrase (due to a conversion error?) so that determining the coordinator as a head is not possible.

We also experimented with attaching the POS tags of heads to the categories of their adjunct dependents. In this way, the parser could differentiate between e.g. verbal and nominal adjuncts. In our experiments, the performance gains achieved by this strategy were low, so we did not incorporate it into the system. Possibly, better results could be achieved by restricting annotation to special classes of adjuncts or by generalizing the heads’ POS tags.

3.4 Categories

As the treebanks provide a lot of information with every word token, it is a delicate question to de-

	Ch	Da	Du	Ge	Ja	Po	Sp	Tu
coarse POS	72.99	69.38	69.27	–	79.07		66.09	
fine POS	61.21	69.78	67.72	7.40	73.44	71.75		54.96
POS + feat	–	42.67	40.40	–				
dep-rel	76.61	72.77	70.70	70.31	78.12	72.93	66.93	65.03
coarse + dep-rel	77.61	67.56	69.43	–	81.36		64.03	
fine + dep-rel	51.21	57.72	68.55			46.28	36.59	54.97

Figure 1: Types of Categories (Development Results)

cide on the type and granularity of the information to use in the categories of the grammar. The treebanks specify for every word a (fine-grained) POS tag, a coarse-grained POS tag, a collection of morphosyntactic features, and a dependency relation (dep-rel). Only the dependency relation is really orthogonal; the other slots contain various generalizations of the same morphological information. We tested several options: coarse-grained POS tag (if available), fine-grained POS tag, fine-grained POS tag with morphosyntactic features (if available), name of dependency relation, and the combinations of coarse-grained or fine-grained POS tags with the dependency relation.

Figure 1 shows F-score results on the development set for several languages and different combinations. The best overall performer is dep-rel; this somewhat astonishing fact may be due to the superior quality of the annotations in this slot (dependency relations were annotated by hand, POS tags automatically). Furthermore, being checked in evaluation, dependency relations directly affect performance. Since we wanted a general language-independent strategy, we used always the dep-rel tags but for Japanese. The Japanese treebank features only 8 different dependency relations, so we added coarse-grained POS tag information. In the categories for Czech, we deleted the suffixes marking coordination, apposition and parenthesis (Co, Ap, Pa), reducing the number of categories roughly by a factor of four. In coordination, conjuncts inherit the dep-rel category from the parent.

Whereas the dep-rel information is submitted to the parser directly in terms of the categories, the information in the lemma, POS tag and morphosyntactic features slot was used only for back-off smoothing when associating lexical items with categories. A grammar with this configuration was used to produce the results submitted (cf. line labelled CF in Figures 2 and 3).

	Cz	Ge	Sp	Sw
dep-rel	52.66	70.31	66.93	72.91
new classific	58.92	74.32	66.09	61.59
new + dep-rel	56.94	78.40	64.03	66.32

Figure 4: Manual POS Tag Classes (Development)

Instead of using the category generalizations supplied with the treebanks directly, manual labour can be put into discovering classifications that behave better for the purposes of statistical parsing. So, Collins et al. (1999) proposed a tag classification for parsing the Czech treebank. We also investigated a classification for German[1], as well as one for Swedish and one for Spanish, which were modelled after the German classification. The results in Figure 4 show that new classifications may have a dramatic effect on performance if the treebank is sufficiently large. In the interest of generality, we did not make use of the language dependent tag classifications for the results submitted, but we will nevertheless report results that could have been achieved with these classifications.

3.5 Markovization

Another strategy that is often used in statistical parsing is Markovization (Collins, 1999): Treebanks

[1] punctuation {$($" $, $.} adjectives {ADJA ADJD CARD} adverbs {ADV PROAV PTKA PTKNEG PTKVZ PWAV} prepositions {APPR APPO APZR APPRART KOKOM} nouns {NN NE NNE PDS PIS PPER PPOSS PRELS PRF PWS SYM} determiners {ART PDAT PIAT PRELAT PPOSAT PWAT} verb forms {VAFIN VMFIN VVFIN} {VAIMP VVIMP} {VAINF VMINF VVINF} {VAPP VMPP VVPP} {VVIZU PTKZU} clause-like items {ITJ PTKANT KOUS}

	Ar	Ch	Cz	Da	Du	Ge	Ja	Po	Sl	Sp	Sw	Tu	Bu
Best	66.91	89.96	80.18	84.79	79.19	87.34	91.65	87.60	73.44	82.25	84.58	65.68	87.57
Average	59.94	78.32	67.17	76.16	70.73	78.58	85.86	80.63	65.16	73.52	76.44	55.95	79.98
CF (submitted)	44.39	66.20	53.34	76.05	72.11	68.73	83.35	71.01	50.72	46.96	71.10	49.81	–
MaxEnt	59.16	61.65	63.28	73.25	64.47	73.94	82.79	80.30	66.27	69.73	72.99	47.16	–
combined	**61.82**	73.34	71.74	**79.64**	**75.51**	80.75	**88.15**	**82.43**	67.09	71.15	**76.88**	**53.65**	–
CF+Markov	45.37	70.76	55.14	74.49	72.55	68.87	84.57	71.89	55.16	47.95	71.18	51.64	–
CFM+newcl		73.84	62.10			77.76				49.61			–
combined		**76.84**	**72.76**			**82.59**			**69.38**	**72.57**			–
new rules (in %)	7.15	6.03	4.64	7.34	5.03	7.42	5.59	6.69	21.00	9.50	10.14	14.23	

Figure 2: Labelled Accuracy Results on the Test Sets

	Ar	Ch	Cz	Da	Du	Ge	Ja	Po	Sl	Sp	Sw	Tu
CF	41.91	76.61	52.66	72.77	70.69	70.31	81.36	72.76	49.00	66.93	72.91	65.03
CF+Markov	63.00	80.25	52.80	73.31	70.70	70.51	82.59	74.37	52.43	67.81	73.56	82.80
CFM+newcl		83.07	59.03			80.42				69.30		

Figure 3: F Score Results on the Development Sets

usually contain very many long rules of low frequency (presumably because inserting nodes costs annotators time). Such rules cannot have an impact in a statistical system (the line new-rules in Figure 2 shows the percentage of rules in the test set that are not in the training set); it is better to view them as products of a Markov process that chooses first the head, then the symbols left of the head and finally the symbols right of the hand. In a bigram model, the choice of left and right siblings is made dependent not only on the parent and head category, but also on the last sibling on the left or right, respectively. Formally the probability of a rule with left hand side C and right hand side $L_n \ldots L_1 H R_1 \ldots R_m$ is broken down to the product of the probability $P_h(H|C)$ of the head, the probabilities of the left siblings $P_l(L_i|L_{i-1}, H, C)$ and those of the right siblings $P_r(R_i|R_{i-1}, H, C)$. Generic symbols designate beginning (L_0, R_0) and end (L_{n+1}, R_{m+1}) of the sibling lists. The method can be transferred to plain unlexicalized PCFG (Klein and Manning, 2003) by transforming long rules into a series of binary rules:

$$
\begin{aligned}
&C \leftarrow L_n \ \langle C, H, L_n, L_{n-1}\rangle \\
&\langle C, H, L_{i+1}, L_i\rangle \leftarrow L_i \ \langle C, H, L_i, L_{i-1}\rangle \\
&\langle C, H, L_1, L_0\rangle \leftarrow [C, H, R_n, R_{n-1}] \ R_n \\
&[C, H, R_{i+1}, R_i] \leftarrow [C, H, R_i, R_{i-1}] \ R_i \\
&[C, H, R_1, R_0] \leftarrow H
\end{aligned}
$$

If the bigram symbols $[C, H, R_i, R_{i-1}]$ and $\langle C, H, L_i, L_{i-1}\rangle$ occur in less than a certain number of rules (50 in our case), we smooth to unigram symbols instead ($[C, H, R_i]$ and $\langle C, H, L_i\rangle$). We used a script of Schmid (2006) to Markovize infrequent rules in this manner (i.e. all rules with less than 50 occurrences that are not coordination rules).

For time reasons, Markovization was not taken into account in the submitted results. We refer to Figures 2 and 3 (line labelled CF+Markov) for a listing of the results attainable by Markovization on the individual treebanks. Performance gains are even more dramatic if in addition dependency relations + manual POS tag classes are used as categories (line labelled CFM+newcl in Figures 2 and 3).

3.6 From Constituency Structure Back to Dependency Structure

In a last step, we converted the constituent trees back to dependency trees, using the algorithm of Gaifman (1965). Special provisos were necessary for the root node, for which no head is given in certain treebanks (Džeroski et al., 2006). To interpret the context-free rules, we associated their children with dependency relations. This information was kept in a separate file that was invisible to the parser. In cases there were several possible interpretations for a context

free rule, we always chose the most frequent one in the training data (Schiehlen, 2004).

4 Machine Learning

While the results coming from the statistical parser are not really competitive, we believe that they nevertheless present valuable information for a machine learner. To give some substance to this claim, we undertook experiments with the Zhang Le's MaxEnt Toolkit[2]. For this work, we recast the dependency parsing problem as a classification problem: Given some feature information on the word token, in which dependency relations does it stand to which head? While the representation of dependency relations is straightforward, the representation of heads is more difficult. Building on past experiments (Schiehlen, 2003), we chose the "nth-tag" representation which consists of three pieces of information: the POS tag of the head, the direction in which the head lies (left or right), and the number of words with the same POS tag between head and dependent. We used the following features to describe a word token: the fine-grained POS tag, the lemma (or full form) if it occurs at least 10 times, the morphosyntactic features, and the POS tags of the four preceding and the four following word tokens. The learner was trained in standard configuration (30 iterations). The results for this method on the test data are shown in Figure 2 (line MaxEnt).

In a second experiment we added parsing results (obtained by 10-fold cross validation on the training set) in two features: proposed dependency relation and proposed head. Results of the extended learning approach are shown in Figure 2 (line combined).

5 Conclusion

We have presented a general approach to parsing arbitrary languages based on dependency treebanks that uses a minimum overhead of language-specific information and nevertheless supplies competitive results in some languages (Da, Du). Even better results can be reached if POS tag classifications are used in the categories that are optimized for specific languages (Ge). Markovization usually brings an improvement of up to 2%, a higher gain is reached in Slovene (where many new rules occur in the testset) and Chinese (which has the highest number of dependency relations). Comparable results in the literature are Schiehlen's (2004) 81.03% dependency f-score reached on the German NEGRA treebank and Collins et al.'s (1999) 80.0% labelled accuracy on the Czech PDT treebank. Collins (1999) used a lexicalized approach, Schiehlen (2004) used the manually annotated phrasal categories of the treebank.

Our second result is that context-free parsing can also boost the performance of a simple tagger-like machine learning system. While a maximum-entropy learner on its own achieves competitive results for only three languages (Ar, Po, Sl), competitive results in basically all languages are produced with access to the results of the probabilistic parser.

Thanks go to Helmut Schmid for providing support with his parser and the Markovization script.

[2] http://homepages.inf.ed.ac.uk/s0450736/maxent_toolkit.html

References

S. Buchholz, E. Marsi, A. Dubey, and Y. Krymolowski. 2006. CoNLL-X shared task on multilingual dependency parsing. In *CoNLL-X*. SIGNLL.

Michael J. Collins, Jan Hajiĉ, Lance Ramshaw, and Christoph Tillmann. 1999. A Statistical Parser for Czech. In *ACL'99*, College Park, MA.

Michael J. Collins. 1999. *Head-Driven Statistical Methods for Natural Language Parsing*. Ph.D. thesis, Univ. of Pennsylvania.

Haim Gaifman. 1965. Dependency Systems and Phrase-Structure Systems. *Information and Control*, 8(3):304–337.

Dan Klein and Christopher Manning. 2003. Accurate Unlexicalized Parsing. In *ACL'03*, pages 423–430.

Michael Schiehlen. 2003. Combining Deep and Shallow Approaches in Parsing German. In *ACL'03*, pages 112–119, Sapporo, Japan.

Michael Schiehlen. 2004. Annotation Strategies for Probabilistic Parsing in German. In *COLING '04*, pages 390–396, Geneva, Switzerland, August.

Helmut Schmid. 2004. Efficient Parsing of Highly Ambiguous Context-Free Grammars with Bit Vectors. In *COLING '04*, Geneva, Switzerland.

Helmut Schmid. 2006. Trace Prediction and Recovery with Unlexicalized PCFGs and Gap Threading. Submitted to *COLING '06*.

Fei Xia and Martha Palmer. 2001. Converting dependency structures to phrase structures. In *HLT 2001*.