USING TRI-LEXICAL DEPENDENCIES IN LFG PARSE DISAMBIGUATION

Aoife Cahill, Marion Weller, Christian Rohrer and Ulrich Heid
IMS, Universität Stuttgart, Germany

Proceedings of the LFG09 Conference

Miriam Butt and Tracy Holloway King (Editors)

2009

CSLI Publications

http://csli-publications.stanford.edu/

Abstract

In this paper we describe experiments where we automatically extract large lists of tri-lexical dependencies. We investigate how effective they are in PP attachment disambiguation by integrating them into a log-linear model for parse disambiguation. We show that we achieve a statistically significant improvement in parse accuracy with the new model that incorporates the tri-lexical dependencies.

1 Introduction

The use of lexical dependencies in parse disambiguation is not a new idea. For example, it is one of the key ideas behind the Collins (1999) parser. In that parser, bi-lexical dependencies are used, but in later work Bikel (2004) showed that the bi-lexical dependencies in fact had little or no impact on the parser decisions. One of the reasons these bi-lexical dependencies are thought to be ineffective is because of sparse data. In treebank-derived parsers, there are simply not enough instances of the dependencies for them to have a significant impact.

Intuitively, on the other hand, it seems that including lexical dependencies should help. In this paper, we consider tri-lexical dependencies between verbs, prepositions and nouns, automatically extracted from a very large corpus, independent from any training data the parser uses. Consider the Example (1) from German with the tri-lexical dependency between the verb *stehen* ('to stand'), the preposition *zu* ('to'), and the noun *Verfügung* ('disposition'), which correspond to the dependency 'to be available' in English:

(1) Dass das Geld zur Verfügung steht, wird angenommen.
That the money to the disposition stands, is assumed.
'The fact that the money is available is assumed.'

For a parser, there is ambiguity here about where to attach the PP *zur Verfügung*, either to the DP *das Geld* or to the verb *stehen* as shown in Figure 1. We observe that for the extracted dependencies with high log-likelihood values the PP almost never attaches to the DP.[1]

In this paper we attempt to incorporate this observation (and test its validity) in an LFG parse disambiguation scenario.

2 Automatically Extracting Tri-Lexical Dependencies

We used parsed text to extract multiword expressions with their morphosyntactic features, focusing on preposition-noun-verb triples (PNV) and verb-object combinations. These basic patterns can be expanded to include further components like

[1] One exception are PPs headed by *von* which occurs very frequently with a meaning corresponding to that of a genitive, and therefore cannot be included in this generalisation.

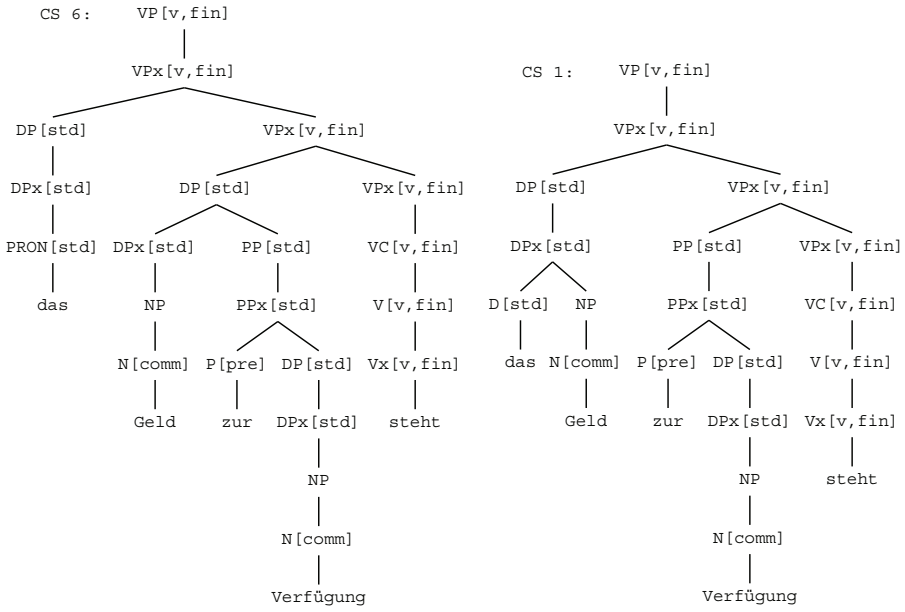

Figure 1: PP-Attachment ambiguity for German example (1) parsed with the LFG parser

adjectives or additional objects. The extracted PNV-triples were then ranked according to their log-likelihood values.

The data from which we automatically extracted the PNV-triples consists of 230 million tokens of parsed newspaper text.[2] We use the `fspar` parser (Schiehlen, 2003) to create dependency structures for each sentence, where ambiguities remain unresolved. This is especially the case with PP-attachment and case marking on nouns, as can be seen in the example in Table 1.

2.1 Extraction

The extraction of PNV-triples begins with first identifying the full verb of a sentence and then systematically 'collecting' every relevant item annotated as a dependent of this verb. Extraction steps are illustrated in Table 1.

The verb *legen* ('to put') in line 8 and the auxiliaries in lines 2 and 9 form a verbal complex (*war... gelegt worden*) which is referred to by the prepositions *nach* – 'after' (line 3) and *auf* – 'on' (line 6). The ambiguous attachment of *nach* to the noun in line 5 is ignored. Triples are extracted by combining the verb, the preposition and head noun of the object of the preposition. The first PNV-triple we extract is built from *liegen*, *auf* and *Eis*; the second is built from *liegen*, *nach* and

[2]Note that while pronouns were extracted during this process, PNV-triples containing pronouns were ignored.

line	token	gloss	POS	lemma	features	dependencies
0	Dieses	this	PDAT	dies		1
1	Abkommen	agreement	NN	Abkommen	Nom:N:Sg	8/9/2
2	war	was	VAFIN	seinP	1|3:Sg:Past:Ind	-1
3	nach	after	APPR	nach	Dat	8/9/2
4	dem	the	ART	d		5
5	U-Boot-Zwischenfall	submarine incident	NN	U-Boot-Zwischenfall	Dat:M:Sg	3
6	auf	on	APPR	auf	Dat|Akk	8/9/2||5
7	Eis	ice	NN	Eis	Dat:N:Sg|Akk:N:Sg	6
8	gelegt	put	VVPP	legen	PPart	8/9/2
9	worden	passive	VAPP	werdenP	PPart	8/9/2
10	.	.	$.	.		-1

Table 1: Slightly simplified example of a sentence parsed with `fspar`

U-Boot-Zwischenfall. Of the two identified triples, only *auf Eis legen* (lit. 'to put on ice': to put something on hold) is a valid, idiomatic multiword expression, while *nach U-Boot-Zwischenfall legen* is a random combination of a preposition, noun and verb. For each one of the extracted triples, we will compute a log-likelihood association score to identify associated word combinations. The higher the log-likelihood association, the more likely the triple is to be an idiomatic multiword expression.

A list of 760 manually checked triples that was created as part of the B3 project of the Stuttgart SFB732 research project is used to handle the ambiguous attachment of a preposition to two or more verbs: if one of the possible combinations is known to be valid, the remaining ones can be discarded; this leads to a lower number of trivial triples.

2.2 Log-Likelihood

In order to distinguish (highly) associated PNV-triples from random cooccurrences, triples were ranked according to their log-likelihood-scores using the UCS-toolkit[3] (Evert, 2004) to compute the scores. Log-likelihood is based on the cooccurrences and individual occurrences of word pairs. For this reason, the extracted triples needed to be reduced to pairs: We tried two different settings by adding the preposition to the noun or to the verb resulting in N-PV and NP-V pairs (cf. Heid et al. (2008)). Table 2 gives a sample of the dependencies extracted, together with their log-likelihood values.

3 Parsing System

In our experiments we use the handcrafted German LFG of Rohrer and Forst (2006) coupled with a log-linear disambiguation component (Riezler et al., 2002; Forst, 2007). This is a robust large-scale grammar that has been implemented within the XLE system and achieves complete spanning parses for around 80% of newspaper text.

For both training and evaluation of the log-linear disambiguation models described in this paper, we use data constructed with the help of the TIGER Treebank (Brants et al., 2002). Our training data consists of 11,504 pairs of labelled and unlabelled packed representations of c- and f-structures which have been produced by our grammar. The labelled representations were constructed by matching the f-structure part of the unlabelled representations produced by the grammar against packed f-structure representations that were derived from the original TIGER Treebank graphs (Forst, 2003). Only sentences for which a proper subset of the readings is compatible with the treebank annotations (and can be determined as such in a reasonable amount of time) were included in the training data, since only these

[3] http://www.collocations.de

N-PV	Log-likelihood	NP-V	Log-likelihood
Verfügung – zu stehen 'to be available'	7.1814156e+04	Verfügung zu – stehen 'to be available'	3.9224998e+04
Verfügung – zu stellen 'to make available'	5.1476639e+04	Verfügung zu – stellen 'to make available'	3.1667310e+04
Leben – um kommen 'to die'	4.5470218e+04	Leben um – kommen 'to die'	2.9221682e+04
Mittelpunkt – in stehen 'to be central'	3.7863808e+04	Mittelpunkt in – stehen 'to be central	2.7858338e+04
Anspruch – in nehmen 'to make use of'	2.8404239e+04	Anspruch in – nehmen 'to make use of'	2.5917412e+04
Vordergrund – in stehen 'to be to the fore'	2.4039711e+04	Grenze in – halten (refl) 'to keep within reasonable limits'	1.8137896e+04

Table 2: Example PNV-triples and their log-likelihoods

Feature Type	Sample feature	Log-likelihood
A	VERB_PP_ATTACH_DP	-1.1977411824394557094
	VERB_PP_ATTACH_NODP	-0.365429809372071146
B	NOUN_PP_ATTACH_DP	-3.6529562031401150435
	NOUN_PP_ATTACH_NODP	4.4348141926322766082
C	ACC_PP_ATTACH_DP	0.26424665120489210235
D	VERB_PP_ATTACH_NODP_zu	7.4110203067279920575
	NOUN_PP_ATTACH_NODP_ab	17.000320083423069661

Table 3: Example of each feature type with log-likelihoods after training

are useful for discriminative training. For evaluation, we use the TiGer Dependency Bank (TiGer DB) (Forst et al., 2004), a dependency-based gold standard for German parsers.

4 Experiments

We carry out a number of experiments to test the effectiveness of the tri-lexical dependencies in parse disambiguation. There are a number of ways to integrate the tri-lexical dependencies into the log-linear model. We design four features to achieve this and experiment with various combinations of the features. Given an ambiguous PP attachment decision, where the PP headed by preposition `prep` and with object head noun `noun` can either attach to the VP headed by `verb` or the DP,[4] we design the following four feature types:

A PP attached to VP or DP and log-likelihood of N-PV

B PP attached to VP or DP and log-likelihood of NP-V

C If the NP in the PP is in accusative case, does it attach to DP or VP?

D PP headed by `prep` attached to VP or DP and log-likelihood of N-PV or NP-V

Table 3 gives an example of each feature type along with the log-likelihood it is assigned after training.

We automatically extract these feature types using the 40,000 most-likely PNV dependencies, in addition to the standard parse disambiguation features described in Forst (2007) and train a standard log-linear model. We tune the parameters of the log-linear model on a development set of 362 sentences and carry out the final testing on 1451 sentences. Table 4 gives the results for various combinations of feature types.[5] The results show that combining all four types of features results in

[4] We do not take the head noun of this DP into account at the moment.

[5] The missing results for feature combinations C and CD are due to problems with training.

Features	F-Score	Features	F-Score
None	79.54	BC	79.54
A	79.53	BD	79.47
B	79.54	ABC	79.54
D	79.47	ABD	79.47
AB	79.54	ACD	79.47
AC	79.54	BCD	79.46
AD	79.47	ABCD	**79.99**

Table 4: Initial Results of incorporating tri-lexical dependencies into parse disambiguation

the biggest improvement over the baseline system. Although the difference is small (0.45), the Approximate Randomization significance test (Noreen, 1989) shows that it is statistically significantly better. Some feature combinations in particular cause the results to degrade. In particular, feature type D that attempts to learn attachment preferences for each preposition does badly. However, in combination with all other feature types, it leads to an improvement.

An example sentence where the new model performs better than the baseline is given in (2). The most-probable solution according to the baseline model attached the PP 'on Tuesday' to the NP 'the Federal Constitutional Court', whereas the most probable solution according to the new model (ABCD) correctly attaches the PP to the verb 'to decide'.

(2) Das entschied das Bundesverfassungsgericht (BVG) am Dienstag.
That decided the Federal Constitutional Court (BVG) on Tuesday.
'The Federal Constitutional Court decided that on Tuesday'

4.1 How Much Data?

An arbitrary decision was made to use the first 40,000 tri-lexical dependencies in the experiments above. However, an interesting question is how many tri-lexical dependencies do you need? The log-likelihood values are an indication of the reliability of the dependencies, and so the more that are used, the more noise that is introduced. The log-linear model does take the log-likelihood into account, however at some point, one would expect the noise to drown out the reliable and useful tri-lexical dependencies. We carry out an ablation experiment to test the effect of the number of tri-lexical dependencies used on the overall f-score. The results are presented in Figure 2.

The graph shows that almost the same f-score can be achieved with only 10,000 tri-lexical dependencies: it is even slightly higher. Table 5 gives the p-values from applying the approximate randomization test to each pair of feature sets. It shows that using 10,000 dependencies is not statistically significantly better than using

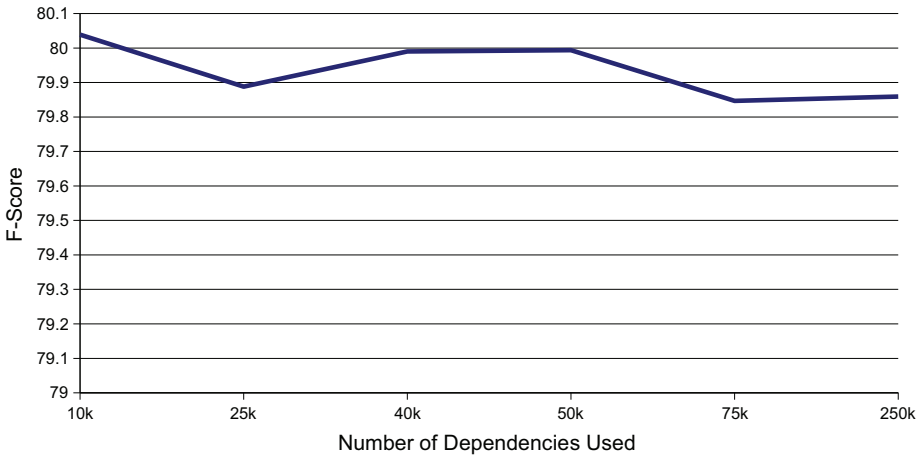

Figure 2: Increasing the number of tri-lexical dependencies

	10k	25k	40k	50k	75k	250k
10k		0.019	0.131	0.724	0.001	< 0.0001
25k			0.131	0.243	0.252	< 0.0001
40k				0.975	0.019	< 0.0001
50k					0.057	< 0.0001
75k						< 0.0001

Table 5: P-Values for testing significance between pairs of feature lists

40,000 or 50,000; however is it statistically significantly better than 25,000, 75,000 and 250,000. We see very few significant differences in the table. What is noteworthy, however, is that all feature lists perform significantly better than the list with 250,000 features. We conclude from this that by adding in so many dependencies, we have introduced too much noise into the model and this is causing the model to degrade. It seems sensible therefore to choose the model with 10,000 features, since this performs best (albeit only slightly) and is the most effective in terms of performance. The dip in performance between 10,000 and 25,000 is due to 9 sentences with PP-attachment ambiguity that receive an improved f-score but 11 sentences with worse f-scores.

In total, there are only 27 sentences in the larger test set where there is a tri-lexical dependency involved in an ambiguous PP attachment decision. These sentences alone make up a very small test set, and further evaluation on a larger, more targeted test suite would be required for complete evaluation. In the meantime, however, we look at the performance of each of the models on only these 27 sen-

Model	# Sentences		PP Attachment changes	
	incr. f-score	decr. f-score	correct	incorrect
A	0	0	0	0
B	0	0	0	0
D	2	0	1	0
AB	0	0	0	0
AC	0	0	0	0
AD	2	0	1	0
BC	0	0	0	0
BD	2	0	1	0
ABC	0	0	0	0
ABD	2	0	1	0
ACD	2	0	1	0
BCD	2	0	1	0
ABCD	11	5	9	3

Table 6: The numbers of sentences with tri-lexical dependencies involved in ambiguous attachment decisions

tences, and show how many of the sentences have a higher f-score than the baseline system and how many have a lower f-score. We also show of the sentences with a changed PP attachment, how many of these are now correct and how many are incorrect. Table 6 shows the results.

These results are very interesting. They show a different pattern to the results presented in Table 4, where most feature combinations involving the *D*-type features performed worse. Here we see now that these feature combinations are actually performing better than the baseline on the sentences with ambiguous PP attachment, while other feature combinations perform at the same overall level. On closer inspection, we see that this improvement is due to improved PP attachment in only one of those sentences. It is still clear, that the combination of all feature types is what gives the most improvement, also in terms of PP attachment. Of the 11 sentences with improved f-score, nine of these are due to now correct PP attachments. In the five sentences with lower f-scores than the baseline, three of these are due to the new model now incorrectly attaching the PP.

5 Generalising the dependencies

Although we achieved a statistically significant improvement in overall accuracy of the most probable f-structure, we wondered if we could increase performance even more. One of the reasons often cited for the relatively unsuccessful performance of lexical dependencies in parse disambiguation is sparse data. Many of the lexical dependencies extracted are simply too infrequent to be useful in most cases. Our

number of possible categories	2	3	4	5	6
number of words with *n* possible categories	2243	278	42	18	3

Table 7: Ambiguous assignment to GermaNet-classes

attempt to combat this issue was to backoff from the lexical level to a more general level. We used the German version of WordNet, GermaNet to do the backoff.

5.1 GermaNet

GermaNet defines 23 categories, which we used to generalise the head nouns in our dependency lists. We also introduced an additional category to account for nouns not included in GermaNet. As the heads of nouns are specified in the parse output, words that could not be found in the GermaNet-lists could be searched for with their head-nouns. Unknown words ending with *-ist* (as in *Linguist*) were tagged as *Mensch* ('human').

Another obvious problem is word sense disambiguation. It would be impossible to choose the correct category in isolation given several alternatives (e.g. *ice* → *nutrition*, *substance*); we randomly chose an assignment to one of the possible categories (e.g. *ice* → *nutrition*). Table 7 shows the amount of ambiguity for the GermaNet categories.

Table 8 contains the GermaNet-classes, the number of entries in each class and its most frequent word. While most of the example entries seem reasonable for their respective classes, some are not very intuitive: The word *Mark* can mean 'bone-marrow' (and thus qualify for the category *body*) or the currency *Mark* which would be the intended meaning in most cases. The example entry in the REST class is an ambiguous lemma that can mean either cabinet or a sort of vine.

5.2 GermaNet Experiments

We carry out the same experiments as in Section 4, choosing the feature set that performs best, ABCD. We evaluate the most probable parse against the same test set and achieve an f-score of 79.83. This is 0.16 f-score points lower than the previous results, although it is not statistically significant. This result is disappointing. We had hoped that by backing off to a more general level, our results would have improved. We also combine the two models and achieve an f-score of 79.78, even lower than the GermaNet model alone.

6 Conclusions

In this paper we presented a method for automatically extracting tri-lexical dependencies and ranking them using log-likelihood. In order to evaluate how effective these dependencies were for the PP attachment disambiguation, we carried out a

frequency	Measure Class		most frequent word		
1755565	Artefakt	*artefact*	43364	Tag	*day*
1730719	Geschehen	*event*	32502	Fall	*case/fall*
826886	Zeit	*time*	311257	Jahr	*year*
784217	Gruppe	*group*	23170	Welt	*world*
642875	Kommunikation	*communication*	29837	Frage	*question*
543302	Ort	*location*	23074	Bereich	*area*
533880	Mensch	*human*	13053	Mensch	*human*
528944	Besitz	*property*	51407	Land	*land*
517856	Attribut	*attribute*	18453	Weise	*way*
511247	Kognition	*cognition*	13768	Ansicht	*view*
466077	REST	*REST*	2012	Kabinett	*Cabinet*
				|Kabinettwein	*vine*
419586	Menge	*quantity*	154686	Prozent	*percent*
251135	Koerper	*body*	102769	Mark	*Mark*
155851	Gefuehl	*feeling*	10722	Sinn	*sense*
136961	Form	*form*	9610	Kreis	*circle*
98682	Relation	*relation*	9073	Gegensatz	*contrast*
71626	natPhaenomen	*natural phenomenon*	8269	Tod	*death*
49975	Nahrung	*nutrition*	4650	Wasser	*water*
48201	Substanz	*substance*	5034	Luft	*air*
46696	Motiv	*motive*	20584	Leben	*life*
17711	natGegenstand	*natural object*	3688	Erde	*earth*
15088	Tier	*animal*	1910	Tier	*animal*
10291	Pflanze	*plant*	838	Wurzel	*root*
5811	Tops	*top level*	2739	Ding	*thing*

Table 8: List of GermaNet-categories and the number of entries as well as the most frequent word in each category.

number of parse-disambiguation experiments. We integrated these dependencies into the log-linear disambiguation model by means of four different feature types and achieved a statistically significant improvement over a baseline when all four feature types were combined. We experimented with backing off the individual word dependencies to try and tackle the sparse data problem. We used classification from GermaNet as a backoff: however we found that this did not improve results. We also combined both models, which also did not lead to an improvement. We found that in our experiments, the 10,000 most likely dependencies contributed most to the improved f-score, and that 250,000 introduced too much noise.

Our initial results were encouraging: we achieved a small improvement in f-

score without using any backoff. However, the results with the GermaNet backoff were disappointing. They are consistent, however, with other experiments using WordNet/GermaNet resources as a backoff for lexical dependencies (Bikel, 2004).

References

Bikel, Daniel M. 2004. A Distributional Analysis of a Lexicalized Statistical Parsing Model. In D. Lin and D. Wu (eds.), *Proceedings of EMNLP 2004*, pages 182–189, Barcelona, Spain: Association for Computational Linguistics.

Brants, Sabine, Dipper, Stefanie, Hansen, Silvia, Lezius, Wolfgang and Smith, George. 2002. The TIGER Treebank. In E. Hinrichs and K. Simov (eds.), *Proceedings of the first Workshop on Treebanks and Linguistic Theories (TLT'02)*, pages 24–41, Sozopol, Bulgaria.

Collins, Michael. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph. D. thesis, MIT.

Evert, Stefan. 2004. *The Statistics of Word Cooccurrences: Word Pairs and Collocations*. Ph. D. thesis, University of Stuttgart.

Forst, Martin. 2003. Treebank Conversion – Creating a German f-structure bank from the TIGER Corpus. In *Proceedings of the LFG03 Conference*, pages 205–216, Saratoga Springs, NY: CSLI Publications.

Forst, Martin. 2007. *Disambiguation for a Linguistically Precise German Parser*. Ph. D. thesis, University of Stuttgart.

Forst, Martin, Bertomeu, Núria, Crysmann, Berthold, Fouvry, Frederik, Hansen-Schirra, Silvia and Kordoni, Valia. 2004. Towards a dependency-based gold standard for German parsers – The TiGer Dependency Bank. In *Proceedings of the COLING Workshop on Linguistically Interpreted Corpora (LINC '04)*, Geneva, Switzerland.

Heid, Ulrich, Fritzinger, Fabienne, Hauptmann, Susanne, Weidenkaff, Julia and Weller, Marion. 2008. Providing Corpus Data for a Dictionary for German Juridical Phraseology. In A. Storrer, A. Geyken, A. Siebert and K. Wrzner (eds.), *Proceedings of the 9th Conference on Natural Language Processing, KONVENS 2008*, Berlin, Germany: Mouton de Gruyter.

Noreen, Eric W. 1989. *Computer Intensive Methods for Testing Hypotheses: An Introduction*. New York: Wiley.

Riezler, Stefan, King, Tracy H., Kaplan, Ronald M., Crouch, Richard, Maxwell, John T. III and Johnson, Mark. 2002. Parsing the Wall Street Journal using a

Lexical-Functional Grammar and Discriminative Estimation Techniques. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 271–278, Philadelphia, Pennsylvania, USA: Association for Computational Linguistics.

Rohrer, Christian and Forst, Martin. 2006. Improving Coverage and Parsing Quality of a Large-scale LFG for German. In *Proceedings of the Language Resources and Evaluation Conference (LREC-2006)*, Genoa, Italy.

Schiehlen, Michael. 2003. A Cascaded Finite-State Parser for German. In *Proceedings of the Research Note Sessions of the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL'03)*, Budapest.