Notes on the Evaluation of Dependency Parsers Obtained Through Cross-Lingual Projection

Kathrin Spreyer
Department of Linguistics
University of Potsdam
spreyer@uni-potsdam.de

Abstract

In this paper we address methodological issues in the evaluation of a projection-based framework for dependency parsing in which annotations for a source language are transfered to a target language using word alignments in a parallel corpus. The projected trees then constitute the training data for a data-driven parser in the target language. We discuss two problems that arise in the evaluation of such cross-lingual approaches. First, the annotation scheme underlying the source language annotations – and hence the projected target annotations and predictions of the parser derived from them – is likely to differ from previously existing gold standard test sets devised specifically for the target language. Second, the standard procedure of cross-validation cannot be performed in the absence of parallel gold standard annotations, so an alternative method has to be used to assess the generalization capabilities of the projected parsers.

1 Introduction

The manual annotation of treebanks for natural language parsing is time-consuming and expensive, but the availability of such resources is crucial for data-driven parsers, which require large amounts of training examples. A technique known as *annotation projection* (Yarowsky and Ngai, 2001) provides a means to relax this resource bottleneck to some extent: In a word-aligned parallel corpus, the text of one language (*source language*, SL), say English, is annotated with an existing parser, and the word alignments are then used to transfer (or *project*) the resulting annotations to the other language (*target language*, TL). The projected trees, albeit noisy, can then constitute the training data for data-driven TL parsers (Hwa et al., 2005; Spreyer and Kuhn, 2009). Finally, in order to assess the quality of the projected parser, its output needs to be compared to held-out TL test data.

Two problems arise in the evaluation of such approaches. First, the annotations projected from the SL usually differ stylistically from those found in the TL test data, rendering any immediate comparison between the predictions of the projected parser and the gold standard meaningless. We discuss the use of tree transformations for evaluation purposes, namely to consolidate discrepancies between the annotation schemes. We then present experiments that investigate the influence of the annotation scheme used in training on the generalization capabilities of the resulting parser. We also briefly address the interaction between annotation style and parsing algorithm (transition-based vs. graph-based).

The second problem addressed here is the assessment of variance in the training data, and hence in parser quality. The standard procedure for this purpose would be *cross-validation*. However, the popular data sets used for benchmarking parsers, such as those that emerged

Coling 2008: Poster Volume, pages 1176–1184,
Beijing, August 2010

from the CoNLL-X shared task on dependency parsing (Buchholz and Marsi, 2006), are typically based on monolingual text. This means that cross-validation is unavailable for projection-based frameworks, because no projection can be performed for the training splits in the absence of a translation in the SL. We therefore propose a validation scheme which accounts for training data variance by training a parser multiple times, on random samples drawn from the projected training data. Each of the obtained parsers can subsequently be evaluated against a fixed, held-out test set independent of the projection step, and the array of accuracy measurements thus obtained can be further subjected to significance testing to verify that observed performance differences are not merely random effects.

The paper is structured as follows. Section 2 describes the projection framework we are assuming. Section 3 summarizes and contrasts the characteristics of four different annotation schemes underlying our SL parsers (English, German) and TL test data (Dutch, Italian). Experiments with different annotation schemes and parsing algorithms are presented in Section 4. In Section 5 we discuss variance assessment in more detail. Section 6 concludes.

2 The Projection Framework

This section briefly describes how we obtain dependency parsers for new languages via annotation projection in a parallel corpus. A detailed discussion can be found in Spreyer and Kuhn (2009).

We use the Europarl corpus (Koehn, 2005) as our parallel corpus. It comprises parallel data from 11 languages; in this paper, we present experiments with English and German as SLs, and Dutch and Italian as TLs.

First, the bitexts for the language pairs under consideration (English-Dutch, English-Italian, German-Dutch, and German-Italian) are word-aligned using Giza++ (Och and Ney, 2003), and all texts are part-of-speech tagged with the TreeTagger (Schmid, 1994) according to pre-trained models.[1]

[1]Available from `http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/DecisionTreeTagger.html`.

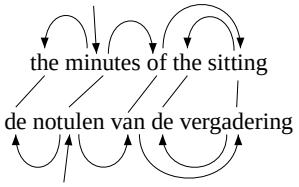


Figure 1: Dependency tree projection from English to Dutch.

Second, we annotate the SL portions, i.e., the German and English texts, with MaltParser dependency parsers (Nivre et al., 2006) trained on standard data sets for the two languages; specifically, we are using the baseline parsers of Øvrelid et al. (2010). The English training data consists of the Wall Street Journal sections 2–24 of the Penn Treebank (Marcus et al., 1993), converted to dependencies (Johansson and Nugues, 2007). The treebank data used to train the German parser is the Tiger Treebank (Brants et al., 2002), in the version released with the CoNLL-X shared task (Buchholz and Marsi, 2006).

Given the SL dependency trees, we project the dependencies to the corresponding (i.e., aligned) TL elements as shown in Figure 1. The links between the English and Dutch words indicate the word alignment. We postulate edges between TL words (e.g., *de* and *notulen*) if there is an edge between their respective SL counterparts (*the* and *minutes*).

The projected dependencies are then used as training data for TL (Dutch and Italian) dependency parsers. In order to account for the fact that many of the projected dependency structures are incomplete due to missing alignments or non-parallelism of the translation, we employ fMalt (Spreyer and Kuhn, 2009), a modified version of the MaltParser which handles fragmented training data. We restrict the admissible fragmentation to three fragments per sentence, for sentences with four or more words, based on early experiments with automatically labeled Dutch data. Sentences that receive more fragmented analyses are discarded.

Finally, we evaluate the projected TL parsers against gold standard test sets by parsing the TL test data and comparing the parser output to

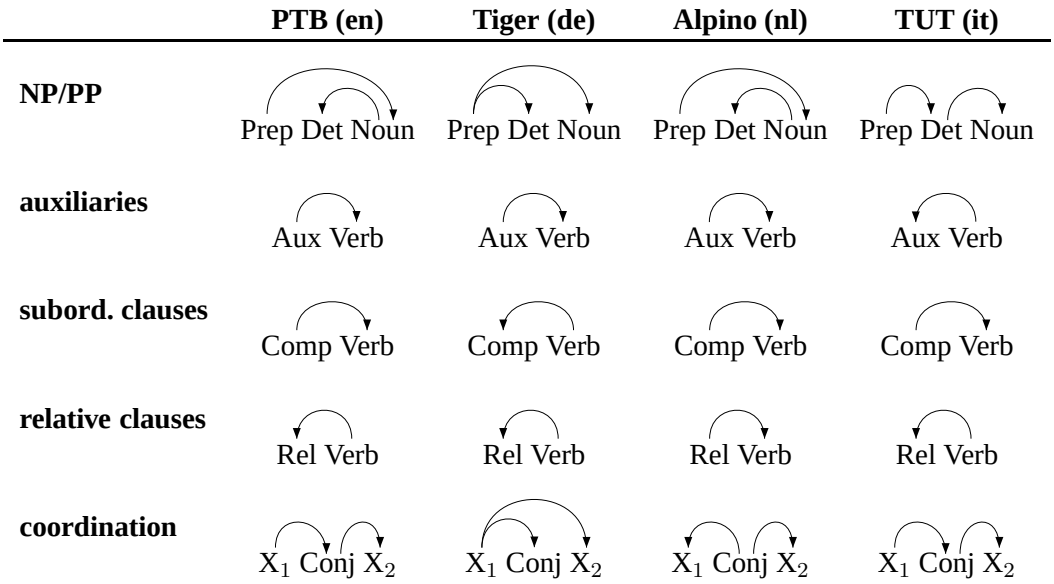


Table 1: Different annotation schemes in dependency-converted treebanks.

the reference annotations. However, we discuss below how differences in annotation style prohibit a direct comparison, and how the annotation schemes affect the learnability of the grammar and therefore the accuracy of the derived parsers.

3 Annotation Schemes

In a projection setting like the one described above, we deal with two sets of annotations: those projected from the SL, and those marked up in the TL gold standard. The four annotation schemes we compare here are those used in the Penn Treebank (PTB; WSJ sections) (Marcus et al., 1993) for English, the Tiger Treebank (Brants et al., 2002) for German, the Alpino Treebank (van der Beek et al., 2002) for Dutch, and the Turin University Treebank[2] (TUT) for Italian.

Table 1 illustrates the most obvious differences among the annotation schemes. Note that we compare annotations in the dependency-converted format. This restricts the comparison to attachment decisions and eliminates the bracket bias inherent to constituent-based comparisons (Carroll et al., 1998; Rehbein and van Genabith, 2007). Again, we use the dependency-converted data sets of the CoNLL-X shared task.

As shown in the table, both the English and the Dutch treebank annotate prepositional phrases hierarchically, with an embedded NP. The flat annotation scheme of the German treebank, on the other hand, makes every word in the PP a dependent of the preposition (with some exceptions). The Italian annotation scheme assumes a hierarchical structure like English and Dutch, but declares the determiner rather than the noun as the head of nominal phrases. Another idiosyncrasy of the Italian annotation scheme is the treatment of fused prepositions such as *della* which incorporate the determiner of the embedded NP: In the dependency-converted TUT, these fused prepositions are represented as two separate tokens, one tagged as a preposition, the other as a determiner.

Next, auxiliaries take the lexical verb as their dependent in all treebanks except the Italian TUT, which inverts the dependency, resulting in a flat structure with the lexical verb as its head. The structure of subordinate clauses is hierarchical according to the English, Dutch and Italian annotation schemes, but flat in Tiger, with the complementizer as a dependent of the embedded verb. Relative clauses, on the other hand, are assigned a flat structure in all but the Dutch scheme, where the relativizer is the head of the embedded verb. Finally, coordination is annotated in three different ways: While the treebanks for English and Italian implement a strictly right-branching strat-

[2] http://www.di.unito.it/~tutreeb

egy, the German annotation scheme attaches both the conjunction and the second conjunct to the first conjunct. The Dutch treebank annotates coordinations as flat structures, with all conjuncts depending on the conjunction.

In order to evaluate projected parsers, any differences in the source and target annotations need to be consolidated. A straightforward way of doing so is by means of tree transformations. Naturally, this begs the question of where such transformations should take place: One could transform the projected annotations to conform to the reference annotations encountered in the test set; alternatively, one can manipulate the test set to reflect the annotation decisions adopted in the source annotations. A variant of the former approach has been implemented by Hwa et al. (2005). They apply post-projection transformations to Chinese training data projected from English in order to infuse TL-specific information which has no counterpart in the source language.

We argue in favor of the alternative, since in a practical application scenario, where rapid, inexpensive development plays a prominent role, it is conceivable that the SL annotation scheme would be adopted unaltered for the TL parser. Consider, for instance, an architecture for multilingual syntax-based information retrieval which is based on parsers for various TLs, all to be derived from a single SL. Devising a tailored annotation scheme for each of the TLs would require linguistically trained personnel with extensive knowledge of the languages at hand. By contrast, adhering to the SL annotation scheme results in homogeneous parser output across the TLs and thus facilitates streamlined higher-level processing.

In Section 4 we present experiments that involve the language pairs English–Dutch, German–Dutch, English–Italian, and German–Italian. For each of the TLs Dutch and Italian, we therefore derive transformed test sets for each SL: one version according to the English PTB annotation style to evaluate the parsers projected from English, and another version according to the German Tiger-style annotations to evaluate parsers projected from German. As an example, Table 2 illustrates the transformations performed on the Italian test set for the parser projected from English.

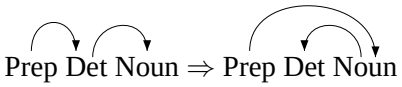
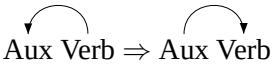
TUT (it) ⇒ PTB (en)	
NP/PP	
auxiliaries	
fused prepositions	

Table 2: Transformations performed on the Italian test set for the parser projected from English.

a.

lang	orig	PTB	Tiger
nl	–	**69.21**	67.38
it	–	**66.44**	53.09

b.

lang	orig	PTB	Tiger
nl	79.23	**80.79**	79.19
it	**88.52**	86.88	84.02

Table 3: Unlabeled attachment scores obtained by training MaltParsers on (a) projected and (b) gold standard dependencies according to different annotation schemes.

English.

4 Annotation Scheme Experiments

4.1 Learnability

If the annotation style is carried over from the source language as we suggest above, we may ask: Is one annotation scheme more appropriate than the other? When more than one source language (annotation scheme) is available, will one produce more "learnable" TL annotations than the other? We explore these questions experimentally. Table 3a shows the performance of Dutch ('nl') and Italian ('it') MaltParsers trained on annotations projected from English ('PTB') and German ('Tiger'), as evaluated against the respective transformed Dutch and Italian gold standards.

Looking at the results for Dutch, we find that there is indeed a significant difference between the parser projected from English and the one projected from German. The former, generating PTB-style dependencies, achieves 69.21% unla-

lang.	words/sent	words/frag	frags/sent
en→nl	27.83	1.95	14.25
de→nl	27.55	1.98	13.92
en→it	28.86	2.26	12.79
de→it	28.79	1.66	17.33

Table 4: Average fragmentation in the projected dependencies.

parser	orig	PTB	Tiger
MST	81.41	**83.01**	**83.87**
		Tiger ≈ *PTB* > *orig*	
Malt	79.23	**80.79**	79.19
		PTB > *orig* > *Tiger*	

Table 5: UAS of the Dutch MST parsers trained on gold standard dependencies. (MaltParser results repeated from Table 3b.)

beled attachment score (UAS). According to a t-test (cf. Section 5), this is significantly ($p<0.01$) better than the parser projected from German Tiger-style annotations, which achieves 67.38%.

Turning to Italian, the parser projected from the English PTB-style annotations again performs better. However, the huge difference of 13.35% UAS suggests a more fundamental underlying problem with the word alignment between the German and Italian sentences. And indeed, inspection of the degree of fragmentation in the Italian projected dependencies (Table 4) confirms that considerably more edges are missing in the dependencies projected from German than from English. Missing edges are an indication of missing word alignment links.

In order to control such factors and focus only on the learnability of the different annotation schemes, we report in Table 3b the results of training on gold standard monolingual treebank data (distinct from the test data), transformed – like the test sets – to conform with the English and German annotation scheme, respectively.[3] In addition, the column labeled 'orig' shows the performance obtained when the original (dependency-converted) Alpino/TUT annotation scheme is used. For Italian, the results corroborate those obtained with the projected parsers: training on the PTB-transformed treebank is significantly[4] ($p<0.01$) more effective than training on the Tiger-transformed treebank. The original TUT scheme is even more effective ($p<0.01$), which comes as no surprise given that the TUT guidelines were tailored to the traits of the Italian language.

The Dutch parser, too, responds better to the hierarchical PTB-based annotation scheme than to the flat Tiger scheme ($p<0.01$). In fact, it also outperforms the parser trained with the original Alpino annotations ($p<0.01$). This demands for further investigation, reported in the following section.

4.2 Interaction with Parsing Algorithms

The results in Table 3 affirm that the performance of a parser hinges on the annotation scheme that it is trained on. However, the learnability of a given scheme depends not only on the annotation decisions, but also on the parsing algorithm implemented by the parser. For instance, it has been noted (Joakim Nivre, p.c. 2008) that flat coordination structures like those in the Alpino Treebank generally pose a challenge to incremental, deterministic parsers like MaltParser.

In order to see to what extent our results are influenced by characteristics of the MaltParser, we repeated the experiments with the MST parser (McDonald et al., 2005), focusing on Dutch parsers from gold standard training data.[5]

The MST parser is a graph-based dependency parser which considers all possible edges to find the globally optimal tree. The results of the MST experiments are given in Table 5, together with the corresponding Malt results repeated from Table 3b. We observe that the relative learnability ranking among the three annotation schemes is indeed different with MST. While in the transition-based paradigm the original Alpino annotations still appeared more adequate for training than the

[3] We did not attempt parameter optimization, so the figures reported here do *not* represent the state-of-the-art in dependency parsing for either language.

[4] According to Dan Bikel's Randomized Parsing Evaluation Comparator: `http://www.cis.upenn.edu/~dbikel/software.html#comparator`

[5] With projected training data for Dutch, and in all experiments with Italian, MST produced the same pattern of relative performance as Malt.

trans	Malt	MST
none	79.23	81.41
$\text{coordination}_{\text{en}}$	**80.91**	**83.01**
$\text{relative}_{\text{en}}$	79.21	81.81
all_{en}	**80.79**	**83.01**
$\text{coordination}_{\text{de}}$	79.39	82.19
$\text{relative}_{\text{de}}$	79.21	81.81
$\text{subord}_{\text{de}}$	79.47	**82.67**
np/pp_{de}	**80.73**	**83.83**
all_{de}	79.19	**83.87**

Table 6: Impact of individual transformations on Dutch treebank parsers. Significant improvements ($p<0.01$) over original Alpino annotation ('none') are in bold face.

Tiger trees, it is now outperformed by both the PTB and the Tiger trees under the graph-based approach. There is no significant difference between the Tiger-based and the PTB-based parser.

To shed some light on the unexpected ranking of the Alpino annotation scheme, we look at the impact of the individual transformations separately in Table 6. The upper part of the table shows how the transformations of the Alpino data towards PTB-style annotations affects learnability. We find that both the MaltParser and the MST parser benefit from the right-branching coordination markup of the PTB scheme. The attachment of relativizers in relative clauses seems to play only a minor role and makes no significant difference.

Turning to the Tiger-style transformations, first note that the semi-flat coordination adopted in the German treebank does not seem to be superior to the flat annotations in Alpino: no significant improvement is achieved for either parser by using the former ('$\text{coordination}_{\text{de}}$'). Surprisingly, both parsers benefit from the flat annotation of prepositional phrases ('np/pp_{de}'). The MST parser, but not the MaltParser, further takes advantage of the flat subordination structure annotated in Tiger. As mentioned earlier, this is in line with the fundamentally different parsing paradigms represented by Malt and MST.

We tentatively conclude that the MST parser is in fact better at exploiting the flat aspects of the Tiger annotations, while both parsers largely benefit from the highly hierarchical coordination structure of the PTB annotation scheme. A more detailed exploration of these issues is clearly in order, and subject to future research.

4.3 Discussion

Kübler et al. (2008) present an extensive comparison of two German treebanks: the Tiger treebank with its rather flat annotation scheme, and the TüBa/DZ treebank with more hierarchical structures. They find that the flat Tiger annotation scheme is more easily learned by constituent-based (PCFG) parsers when evaluated on a dependency level. Our results suggest the opposite, but this may well be due to the differences in the experimental setup: Our training data represent dependency trees directly, and we learn incremental, deterministic dependency parsers rather than PCFGs.

5 Variance Assessment

The second question we address in this paper is the assessment of variance in the training data, and hence in parser quality. The standard procedure for this purpose would be *cross-validation*. To perform k-fold cross-validation, the data is partitioned into k splits of equal size, and one of the splits is used as test data, while the remaining k-1 splits serve as training data. The train–test cycle is repeated until each of the k subsamples has been used as test data exactly once.

However, the popular data sets used for benchmarking parsers, such as the CoNLL-X shared task data used here, are typically based on monolingual text. This means that cross-validation is unavailable for projection-based frameworks, because no projection can be performed for the training splits in the absence of a translation in the SL.

Moreover, the expected noise level in the projected dependencies requires that there be a considerable amount of training data for an evaluation to be meaningful. So even if parallel test data is available, the data partitioning performed in cross-validation may compromise the results.

We therefore propose a validation scheme which (i) does not reduce the amount of test data by partitioning (this may be a problem when only a small number of gold standard annotations is

	nl_{ptb}	nl_{tig}	it_{ptb}	it_{tig}
	68.51	67.25	66.56	54.01
	70.07	66.79	66.45	54.21
	69.21	68.13	66.07	53.37
	69.45	68.29	66.47	52.77
	68.47	67.31	66.74	52.55
	69.07	66.97	66.20	53.66
	69.99	67.87	66.56	52.70
	69.71	66.43	66.37	52.70
	68.77	67.11	66.05	52.08
	68.83	67.67	66.96	52.82
mean	69.21	67.38	66.44	53.09
sd	0.58	0.60	0.29	0.69

Table 7: Intra-system variance assessment.

available), (ii) does not require parallel test data and is independent of the projection step, and (iii) takes advantage of the fact that training data is cheap and therefore abundant in projection-based settings. Specifically, given that we have plenty of training data, we can train a particular parser multiple (say, k) times, each time sampling a fixed number of training examples from the pool of training data. The k parsers can then each parse the unseen test set, and subsequent comparison against the gold standard annotations yields k values of the performance metric at hand (here, UAS). As in conventional cross-validation, these k values are then averaged to provide an aggregated score, and they can be used to derive standard deviations etc. The arrays of measurements for different systems can further be subjected to significance tests such as the two-sample t-test to verify that observed performance differences are not merely random effects.

5.1 Experiments

We use the validation procedure just described (with k=10) to investigate the variance in the projected parsers discussed in the previous section (Table 3a). Table 7 lists the scores obtained by the individual parsers, each trained on a different random sample of 100,000 words, drawn from the pool of all projected annotations. We also show the standard deviation and repeat the mean UAS. We observe that, for a given language, standard deviation seems to correlate negatively with mean UAS; in other words, the better parsers also seem to be more robust towards variance in the training data.

5.2 Discussion

Classical cross-validation and the validation method described here do measure slightly different things. First, in cross-validation it is not only the training data that is varied, but the test data as well. Second, when two systems are compared under the cross-validation regime, the k rounds can usually be considered *paired* samples because both systems are trained and evaluated on identical partitionings of the data. In contrast, projection-based settings typically involve some form of filtering on the basis of the projected annotations; in our case, the filter restricts the degree of fragmentation in the projected dependency tree. This filtering makes it all but impossible to pair the training samples without seriously diminishing the pool from which the samples are drawn. For instance, when comparing the Italian parser projected from English (it_{ptb}) and the one projected from German (it_{tig}), a training sentence may receive a complete analysis from the English translation, and hence be included in the training pool for it_{ptb}; but the same (Italian) sentence may receive a highly fragmented analysis under projection from German (e.g., due to missing alignment links) and be discarded from the training pool for it_{tig}.

With samples that cannot be paired, it is also not obvious how evaluation strategies like the randomized comparison mentioned above (fn. 4) could be employed in a sound way (by non-statisticians).

6 Conclusions

We have discussed two issues that arise in the evaluation of frameworks that involve cross-lingual projection of annotations. We focused on the projection of dependency trees from German and English to Dutch and Italian, and presented experiments that compare parsers trained on the projected dependencies. The parsers differ in the annotation scheme they follow: When they are projected from German, they employ the flat Tiger annotation scheme of the source language; pro-

jected from English, they learn the more hierarchical PTB structures. In order to evaluate the projected parsers against target language (Dutch, Italian) gold standard annotations, we convert the test sets to the annotation scheme employed in the respective source language.

While our experiments with gold standard treebank data affirm that the annotation scheme that is being learned has some influence on the performance of the parser, one should bear in mind that in a projection scenario, the quality of the word alignment plays at least an equally important role when it comes to chosing a suitable source language and annotation scheme.

We have further proposed a validation scheme which unlike cross-validation does not require parallel test data. Instead, it exploits the fact that training data is usually available in abundance in projection scenarios, so parsers can be trained on multiple random samples and evaluated against a single, independent test set which need not be further partitioned.

Acknowledgments

The work reported in this paper was supported by the Deutsche Forschungsgemeinschaft (DFG; German Research Foundation) in the SFB 632 on Information Structure, project D4 (Methods for interactive linguistic corpus analysis).

References

Brants, Sabine, Stefanie Dipper, Silvia Hansen, Wolfgang Lezius, and George Smith. 2002. The TIGER treebank. In *Proceedings of the Workshop on Treebanks and Linguistic Theories*, pages 24–41.

Buchholz, Sabine and Erwin Marsi. 2006. CoNLL-X Shared Task on Multilingual Dependency Parsing. In *Proceedings of CoNLL-X*, pages 149–164, New York City, June.

Carroll, John, Ted Briscoe, and Antonio Sanfilippo. 1998. Parser evaluation: a survey and a new proposal. In *Proceedings of LREC 1998*, pages 447–454, Granada, Spain.

Hwa, Rebecca, Philip Resnik, Amy Weinberg, Clara Cabezas, and Okan Kolak. 2005. Bootstrapping parsers via syntactic projection across parallel texts. *Natural Language Engineering*, 11(3):311–325.

Johansson, Richard and Pierre Nugues. 2007. Extended constituent-to-dependency conversion for English. In Nivre, J., H.-J. Kaalep, and M. Koit, editors, *Proceedings of NODALIDA 2007*, pages 105–112.

Koehn, Philipp. 2005. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Proceedings of the MT Summit 2005*.

Kübler, Sandra, Wolfgang Maier, Ines Rehbein, and Yannick Versley. 2008. How to Compare Treebanks. In *Proceedings of LREC 2008*, pages 2322–2329.

Marcus, Mitchell P., Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.

McDonald, Ryan, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of HLT-EMNLP 2005*.

Nivre, Joakim, Johan Hall, Jens Nilsson, Gülşen Eryiğit, and Svetoslav Marinov. 2006. Labeled pseudo-projective dependency parsing with support vector machines. In *Proceedings of CoNLL-X*, pages 221–225, New York City, June.

Och, Franz Josef and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Øvrelid, Lilja, Jonas Kuhn, and Kathrin Spreyer. 2010. Cross-framework parser stacking for data-driven dependency parsing. To appear in TAL 2010 special issue on Machine Learning for NLP 50(3), eds. Isabelle Tellier and Mark Steedman.

Rehbein, Ines and Josef van Genabith. 2007. Treebank annotation schemes and parser evaluation for German. In *Proceedings of EMNLP-CoNLL 2007*, pages 630–639, Prague, Czech Republic, June. Association for Computational Linguistics.

Schmid, Helmut. 1994. Probabilistic part-of-speech tagging using decision trees. In *International Conference on New Methods in Language Processing*, pages 44–49, Manchester, England.

Spreyer, Kathrin and Jonas Kuhn. 2009. Data-driven dependency parsing of new languages using incomplete and noisy training data. In *Proceedings of CoNLL 2009*, pages 12–20, Boulder, CO, June.

van der Beek, Leonoor, Gosse Bouma, Robert Malouf, and Gertjan van Noord. 2002. The Alpino dependency treebank. In *Computational Linguistics in the Netherlands (CLIN)*.

Yarowsky, David and Grace Ngai. 2001. Inducing Multilingual POS Taggers and NP Bracketers via Robust Projection across Aligned Corpora. In *Proceedings of NAACL 2001*, pages 200–207.