Exploiting Synergies Between Open Resources for German Dependency Parsing, POS-tagging, and Morphological Analysis

Rico Sennrich, Martin Volk and **Gerold Schneider**
Institute of Computational Linguistics
University of Zurich
Binzmühlestr. 14
CH-8050 Zürich
{sennrich,volk,gschneid}@cl.uzh.ch

Abstract

We report on the recent development of ParZu, a German dependency parser. We discuss the effect of POS tagging and morphological analysis on parsing performance, and present novel ways of improving performance of the components, including the use of morphological features for POS-tagging, the use of syntactic information to select good POS sequences from an n-best list, and using parsed text as training data for POS tagging and statistical parsing. We also describe our efforts towards reducing the dependency on restrictively licensed and closed-source NLP resources.

1 Introduction

German NLP tools such as part-of-speech taggers, morphology tools, and syntactic parsers often require licensing and suffer from usage restrictions, which makes the deployment of an NLP pipeline that combines several components cumbersome at best, impossible at worst (if no license can be obtained). Some restrictions are rooted in the copyright and/or licenses of the annotated corpora on which statistical taggers or parsers can be trained for German, such as TIGER (Brants et al., 2002) or Tüba-D/Z (Telljohann et al., 2004). There have been attempts to bypass these restrictions through corpus masking (Rehm et al., 2007), but for statistical models that require lexical information, this is not an option.

We discuss ParZu, a German dependency parser that relies on external tools for POS tagging and morphological analysis, and combines a hand-written grammar and a statistical disambiguation module that is trained on a treebank. We describe attempts to move towards components with freer licensing. We also discuss techniques to improve parsing performance by better exploiting the various resources, specifically by using morphological information in POS tagging, and through n-best POS tagging.

2 Parser Architecture

ParZu, first described in (Sennrich et al., 2009), is a hybrid dependency parser for German, which implements the grammar described by Foth (2005). It combines a hand-written grammar with a statistical disambiguation module, building on the same architecture as the English Pro3Gres parser (Schneider, 2008). The hand-written grammar is mostly unlexicalized and operates on the level of parts-of-speech.[1] To give the subject relation as example, the grammar constrains the possible parts-of-speech of the head (a finite verb) and the dependent (typically a noun or pronoun, but some other classes such as numbers are also allowed). The dependent must be in nominative case, and either agree with the verb in person and number, or be a coordinated structure. Since each word form may have multiple possible morphological analyses, the morphological constraints are unification-based and allow for underspecified representations. Also, at most one subject is allowed per finite verb, and some topological restrictions must be met, such as only one constituent being allowed in the Vorfeld.

The rules draw on part-of-speech information and morphological knowledge. For the former, Sennrich et al. (2009) use TreeTagger for POS tagging. For the latter, they use GERTWOL (Haapalainen and Majorin, 1995), a commercial morphology tool.

The statistical disambiguation module models lexical and positional preferences, and is trained on the TüBa-D/Z, a hand-annotated treebank of

[1] Among the lexicalized rules is a closed list of nouns which can head noun phrases with temporal function, such as *Er schläft **jeden Tag*** (English: 'he sleeps **every day**').

Proceedings of Recent Advances in Natural Language Processing, pages 601–609,
Hissar, Bulgaria, 7-13 September 2013.

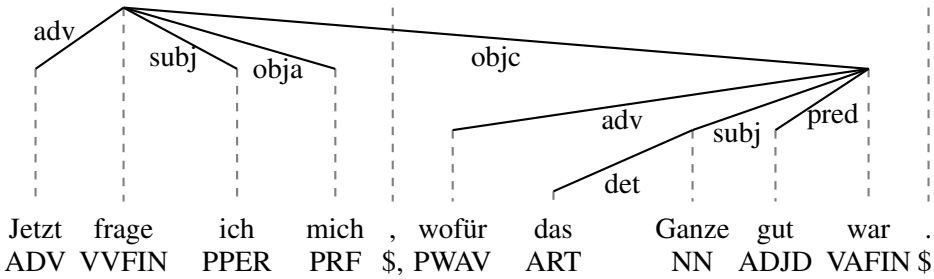

Figure 1: TüBa-D/Z parse tree in dependency format. (English: 'Now I ask myself what good it did.')

about 65 500 sentences from a German newspaper. Versley (2005) provides a conversion of the treebank into the dependency format that the parser implements. Figure 1 shows an example parse tree. Among others, the statistical disambiguation module performs a functional disambiguation of German noun phrases based on the verb's subcategorization frame, disambiguates the attachment of prepositional phrases and adverbs, and uses constant pseudo-probabilities to prefer some labels over others if both are permitted by the grammar.

In summary, ParZu requires three components with licensing restrictions, for which we will discuss alternatives: a morphology tool (GERTWOL), a POS tagger (TreeTagger) and an annotated treebank (TüBa-D/Z). First, we present a baseline evaluation that compares parser performance with a statistical parser, and shows improvements to the grammar and statistical disambiguation module since the evaluation in (Sennrich et al., 2009).

2.1 Evaluation

This first evaluation serves three purposes: comparing parsing performance of ParZu with that of a state-of-the-art statistical parser, comparing the version of ParZu that we use to that of earlier publications, and evaluating the performance loss when moving from gold POS tags to automatically predicted ones. Note that our initial comments on limited deployability also apply to statistical parsers. Even if a statistical parser is released under a permissive license, it requires an annotated treebank for model training, and thus its deployment is hampered by the licensing restrictions of the treebank.

Of the 65 500 sentences in version 7 of TüBa-D/Z (1 230 000 tokens), we use the first 1000 for development purposes, the next 3000 for this evaluation, and the remaining 61 500 sentences for training. To represent state-of-the-art statistical parsing, we use MaltParser (Nivre, 2009), with settings optimized with MaltOptimizer (Ballesteros and Nivre, 2012). MaltParser is a tool for data-driven dependency parsing which implements various algorithms. For TüBa-D/Z, MaltOptimizer selects the stack projective algorithm (Nivre, 2009) with pseudo-projective pre- and postprocessing. The algorithm generates a parse tree through a sequence of transitions from an initial configuration (a NULL word on the stack, all words of the sentence in the buffer, and an empty set of labelled dependency arcs) to a terminal configuration (a NULL word on the stack, an empty buffer, and a set of labelled dependency arcs which forms the parse tree). For each configuration, three transitions are possible, either shifting the first word in the buffer to the stack, or labelling the last word in the stack a dependent of the second-to-last (removing the dependent from the stack), or vice versa. Each transition is predicted by a classifier which is trained on the training treebank.

For ParZu, we present results for version 0.11 – evaluated in (Sennrich et al., 2009) – and the last released version 0.21. The difference between these represents improvements in the core grammar and statistical disambiguation module.[2]

We measure labelled precision and recall, i.e. for how many tokens both the head and the dependency label are correctly predicted, compared to either the total number of predictions, or the number of relations in the treebank. Punctuation marks and ROOT are not considered in the evaluation – this means that if a system does not predict a head for a token, this harms its recall, but not the precision. We also report the f_1 score, the harmonic mean between precision and recall. For the evaluation, we use tokenization and sentence splitting of the treebank, but not the lemmas or morphological features. For MaltParser, we predict lem-

[2]The evaluation set was not used during development of these components.

system	precision	recall	f_1
TreeTagger			
MaltParser	84.7	85.1	84.9
ParZu v. 0.11	83.5	75.8	79.4
ParZu v. 0.21	85.4	83.2	84.3
gold tags			
MaltParser	88.0	88.4	88.2
ParZu v. 0.11	86.6	81.1	83.7
ParZu v. 0.21	89.7	89.1	89.4

Table 1: Parsing performance baseline results with automatically predicted tags (TreeTagger) and gold POS tags.

mas with TreeTagger, and use no morphological features, neither for training nor for parsing, since most morphological analyses are ambiguous, and we cannot easily provide MaltParser with disambiguated morphological analyses for parsing; for ParZu, we predict lemmas and extract morphological analyses with GERTWOL. We also compare using POS tagging with TreeTagger to using the gold tags from the treebank to show how parsing performance degrades because of tagging errors.

Results are shown in table 1. For Maltparser, the loss in performance (f_1) is 3.3 percentage points when moving from gold POS tags to automatically predicted ones.[3] We found that the automatic prediction of lemmas is less problematic than that of POS tags, with a difference of 0.3 percentage points in f_1 score between automatically predicted and gold lemmas.

ParZu version 0.21 performs markedly better than version 0.11, which an improvement of about 3 percentage points in terms of precision, and 8 in terms of recall. This is mostly due to continued development on the core components, i.e. the grammar and the disambiguation module. With gold tags, ParZu outperforms MaltParser by 1.2 percentage points in f_1 score (88.2% $\rightarrow$ 89.4%). Note that, despite the similar total performance, the parsers have different strengths and weaknesses. ParZu is consistently better than MaltParser in the functional disambiguation of noun phrases, i.e. relations such as subject, object, and genitive modifier, while MaltParser finds more coordinations, albeit with lower precision. Some selected f_1 values (ParZu and MaltParser, respectively): SUBJ 94.5 vs. 90.3; OBJA 87.9 vs. 80.7; OBJD 77.8 vs. 49.6; GMOD 93.8 vs. 88.9.

[3]We follow the suggestion of Seeker et al. (2010) to tag the training data with the same tool used for decoding. With TreeTagger used only for parsing, but gold tags for training, performance is lower with 82.6% precision and 83.0% recall.

```
> Bewegungen
bewegen<V>ung<SUFF><+NN><Fem><Acc><Pl>
bewegen<V>ung<SUFF><+NN><Fem><Dat><Pl>
bewegen<V>ung<SUFF><+NN><Fem><Gen><Pl>
bewegen<V>ung<SUFF><+NN><Fem><Nom><Pl>
```

Figure 2: SMOR analysis of *Bewegungen*.

When moving from gold POS tags to automatically predicted ones, recall of ParZu drops by 5.9 percentage points, which is a bigger loss than that of MaltParser. Note that the drop is bigger in terms of recall than precision, which indicates that ParZu tends to make fewer labelling decisions, and generate more partial parses, when confronted with mistagged sentences. This is because the correct structure may be considered ungrammatical by the grammar on the basis of POS tags. While this can be perceived as a disadvantage compared to the data-driven MaltParser, which can learn the idiosyncrasies of the tagger when trained on automatically tagged data, we will try to exploit this behaviour to correct tagging errors in an n-best tagging workflow.

3 Morphology

For parsing, morphology tools provide two useful types of information. Lemma information allows for less sparse representation of statistical data, and inflectional analyses can be used to enforce agreement constraints, and for the functional disambiguation of German noun phrases.

As alternatives to GERTWOL, we investigate two morphology tools, both based on the SMOR grammar (Schmid et al., 2004), which is open source and licensed under GPL v2. The first is the SMOR grammar with the lexicon of the University of Stuttgart (consequently referred to as SMOR). The lexicon is closed-source, and can be licensed for research purposes. Secondly, we investigate Morphisto (Zielinski and Simon, 2009), which combines the SMOR grammar with an open-source lexicon, provided under the Creative Commons 3.0 BY-SA Non-Commercial license.

One problem with the SMOR grammar is that the morphology does not produce conventional lemmas, but derivational analyses as shown in figure 2. Specifically, the word form *Bewegungen* (English: 'movements') is shown to be composed

morphology tool	precision	recall	f_1
none	86.0	85.7	85.9
GERTWOL	89.7	89.1	89.4
SMOR	89.8	89.3	89.5
Morphisto	89.8	89.3	89.5

Table 2: ParZu parsing performance with different morphology tools (gold POS tags).

of the verb stem *bewegen* and the suffix *-ung*. In order to obtain a more traditional lemma, namely a form that corresponds to the nominative singular form (for nouns), we produce a pseudo-lemma by selecting the last morpheme in the analysis string, and concatenating it with the unnormalized stem. We separate the stem which we want to retain, and the ending which we substitute with the normalized form, through a longest common subsequence match between the original word form and the last morpheme in the SMOR analysis. In the example above, the last morpheme in the analysis is *ung*, which means that our pseudo-lemma is the concatenation of *Beweg* and *ung*, thus obtaining *Bewegung* as lemma.

Table 2 shows results for the three morphology systems. We can see that, for the purposes of parsing, the three tools perform similarly well, with SMOR/Morphisto performing 0.1 percentage points better than GERTWOL. The difference to not using any morphological information is about 3.5 percentage points. Note that ParZu relies heavily on these external analyses, and that some of the loss could be mitigated by using more lexicalized statistics instead. The performance difference is greatest for noun phrase relations, such as dative or accusative object.

We conclude that despite the unorthodox notion of lemmas, SMOR and Morphisto can be usefully deployed in a parser and are a suitable replacement for the commercial GERTWOL tool. This positive result is somewhat surprising given that, in a manual evaluation, a large performance gap between Morphisto and GERTWOL was found (Mahlow and Piotrowski, 2009). We plan to extract a fully free morphological lexicon from Wiktionary in future work, in order to have even more permissive licensing.

4 Tagging

The baseline experiments in table 1 show that tagging errors account for about a third of total parsing errors. As a consequence, we investigate ways to improve tagging, and mitigate the effect of tagging errors on parsing performance through n-best-tagging.

4.1 Conditional Random Field Tagging with Morphological Features

A major problem in statistical POS tagging for German is the complex morphology of German, which results in many inflected or compounded forms which have never been observed during training. We aim to improve performance by using a conditional random field (CRF) tagger that uses morphological features, similar to the first applications of CRFs described by Lafferty, McCallum and Pereira (Lafferty et al., 2001), and the model described in (Seeker et al., 2010). Conditional random fields are undirected graphical models that operate in a maximum entropy framework, and have the advantage over classical hidden Markov models (HMM) that they relax independence assumptions and allow for the inclusion of arbitrary features.

We use the following features:

- seven features representing the word form, the surrounding word forms (up to two words to the left and right), and the bigram of word plus left/right neighbour
- a bigram feature (on the level of labels)
- the lowercased word form
- is the word capitalized? (binary)
- is the word form alphanumeric (including dashes)? (binary)
- all possible POS tags of the word form as produced by a morphology tool

The set of possible POS tags is extracted from a morphology tool by mapping all analyses of the word form into the STTS tag set. Internally, all features are binarized.

4.2 Evaluation

We evaluate the CRF model without morphology, and with morphological analyses extracted from SMOR or Morphisto.[4] We use the CRF toolkit

[4]The feature extraction scripts, and configuration files necessary to reproduce our results are available on `https://github.com/rsennrich/clevertagger`.

tagger	morphology	TüBa	Sofies Welt
TreeTagger	-	94.9	95.0
TnT	-	97.0	94.7
CRF	-	96.2	94.7
CRF	Morphisto	97.6	96.6
CRF	SMOR	97.8	96.7

Table 3: POS tagging accuracy (in percent). N=53 935 (TüBa-D/Z) / 7416 (Sofies Welt).

tagger	morphology	TüBa	Sofies Welt	
				NE=NN
TnT	-	89.5	58.0	84.0
CRF	-	80.8	60.6	85.2
CRF	Morphisto	90.9	89.1	91.3
CRF	SMOR	92.6	89.6	91.3

Table 4: POS tagging accuracy for out-of-vocabulary words (in percent). N=3936 (TüBa-D/Z) / 393 (Sofies Welt).

Wapiti for training and decoding (Lavergne et al., 2010). We compare tagging performance to TreeTagger, a decision tree tagger, and TnT, a trigram HMM tagger.

We train TnT and the CRF models on the same 61 500 sentences from TüBa-D/Z that we used for the parsing evaluation; for TreeTagger, we use the published model for German. We evaluate performance on a 3000-sentence evaluation set from TüBa-D/Z, and a corpus of 529 sentences from "Sofies Welt", which is part of the Smultron parallel treebank (Volk et al., 2010).[5]

As the results in table 3 show, TnT performs better than TreeTagger on TüBa-D/Z (97% versus 94.9%), but slightly worse on Sofies Welt (94.7% versus 95.0%). This indicates that the TnT model is slightly domain-specific, and performance on Sofies Welt may better reflect out-of-domain performance. The CRF tagger without morphological features performs slightly worse than TnT, while the CRF models with morphological features perform best overall, with an accuracy of 97.6-8% on TüBa-D/Z, and 96.6-7% on Sofies Welt. This is an improvement of 1.6–2 percentage points compared to TreeTagger, TnT, and a CRF tagger without morphological features. The difference between using Morphisto and the original SMOR system to obtain morphological features is small.

[5]Both corpora use the STTS tag set, and we conflate non-standard tags: for pronominal adverbs, TüBa-D/Z uses *PROP*, Smultron *PROAV*, and TreeTagger *PAV*.

tagger	morphology	precision	recall	f_1
TreeTagger	-	85.6	83.7	84.6
TnT	-	87.1	85.2	86.2
CRF	-	86.3	84.8	85.5
CRF	Morphisto	87.9	86.7	87.3
CRF	SMOR	88.1	86.9	87.5

Table 5: Parsing performance with different POS taggers. ParZu with SMOR.

A large part of the performance difference can be attributed to the handling of unknown words. Table 4 shows tagging accuracy for words that do not occur in the TüBa-D/Z training set. TnT uses suffix analysis to estimate the class of unknown words, and on TüBa-D/Z, strongly outperforms a CRF model that has neither smoothing for unknown words nor morphological features. On Sofies Welt, TnT performs poorly due to frequent names (like *Sofie*) being tagged as *NN* instead of *NE*. We also present results with *NN* and *NE* conflated into a single POS tag.

The morphological features yield a big performance boost for the CRF tagger. With morphological features from SMOR, performance on TüBa-D/Z for out-of-vocabulary words is 12 percentage points better than without morphological features, and 3 percentage points better than that of TnT. On Sofies Welt, the difference is even more marked, with a gain of about 30 percentage points through morphological features, compared to either TnT or a CRF model without morphological features. Even if we conflate *NN* and *NE* into a single category, we observe a gain of 6-7 percentage points for the models with morphological features.

Improvements to POS tagging have a direct effect on parsing performance, as table 5 shows. we observe a difference of 3.2 percentage points in recall, and 2.5 percentage points in precision, between the worst tagger (TreeTagger) and the best one (CRF with SMOR).

4.3 N-best-tagging

While morphological analyses help the tagging of unknown words, the tag of some word forms cannot be predicted based on local features alone. Examples are the distinction between finite and infinitive verbs (e.g. *erhalten*), between relative pronouns and articles (e.g. *der*), and between prepositions and separated verb particles (e.g. *um*).

Consider examples 1–3 to see the single word form *erhalten* (English: 'receive') as three different parts-of-speech.

1. Sie feiern, wenn sie [...] erhalten/VVFIN.
   They celebrate if they receive [...]

2. Sie wollen [...] erhalten/VVINF.
   They want to receive [...]

3. Sie haben [...] erhalten/VVPP.
   They have received [...]

The gap [...] can be filled with a direct object and multiple adjuncts and thus be arbitrarily long, for instance *dieses Jahr viele Geschenke* (English: 'many gifts this year'). In such a case, a trigram Hidden Markov Model, which only considers a history of two words, would be unable to distinguish between the examples and assign the same label to *erhalten* in all of them.

The parser evaluation in table 1 shows that tagging errors affect the recall of ParZu more strongly than its precision. This indicates that ParZu tends to give no label at all, rather than the wrong label, if the POS sequence is ungrammatical. We propose to use this characteristic to choose the best tag sequence from an n-best list by preferring complete analyses over partial ones.

For each input sentence, we generate the n-best tag sequences with the CRF model, parse each, and then perform parse selection based on a number of features:

- The probability of the POS sequence
- The rank of the POS sequence
- The number of unattached nodes
- The number of "bad" labels (apposition or coordination, see below)

The features are combined into a single score in a log-linear framework, with weights set to optimize parsing performance on a development set of 1000 sentences. The probability feature obtains a positive weight (higher is better); all other features a negative one (higher is worse). Appositions and coordinations are considered bad labels because they are frequent in mistagged sentences. If a verb is mistagged as noun, noun phrases cannot be analyzed as subject, object etc., but will instead be labelled appositions of each others.

n-best	parsing performance			tagging accuracy
	precision	recall	f_1	
1	(no parsing)			97.8
1	88.1	86.9	87.5	98.1
50	88.2	87.9	88.0	98.3

Table 6: Parsing performance and tagging accuracy with n-best tagging. CRF with SMOR for tagging, ParZu with SMOR for parsing.

In the following experiments, we perform n-best tagging with $n = 50$, then pruning all tag sequences which are less probable than the best sequence by a factor of 20 or more. This makes the size of the n-best list elastic in practice. If the tag sequence is unambiguous, all but the 1-best tag sequence are immediately discarded; for sentences with many ambiguities, we allow n of up to 50, which happens 13 times in the 3000-sentence evaluation set. On average, n (after pruning) is around 4, which also means that the number of sentences being parsed, and thus the runtime of the parser, is increased by a factor of 4. The baseline is the system with SMOR as morphology tool, both for the parser and the CRF tagging model.

We can see in table 6 that n-best tagging not only improves parsing recall by about 1 percentage point, but also improves tagging accuracy by 0.5 percentage points (97.8% $\rightarrow$ 98.3%). Some improvement in tagging accuracy is already visible with 1-best tagging, due to heuristic rules in the parser itself, i.e. forcing the last verb in a subordinated clause to be finite (VVFIN), even if the tagger predicts it to be infinite (VVINF) or a participle (VVPP), if the morphology system allows the analysis as VVFIN and the conjunction does not govern an infinitive.

With n-best tagging, parsing performance (f_1) is 88.0%, which is 1.4 percentage points below that with gold POS tags, and 3.7 percentage points better than in our baseline experiments in table 1 (84.3 $\rightarrow$ 88.0% $\rightarrow$ 89.4%). This means that n-best tagging, and the use of a CRF tagger rather than TreeTagger, has markedly reduced the number of parsing errors that are caused by tagging errors, compared to the baseline.

We will look more closely at the tagging results with SMOR and n-best tagging. The jump from 97.8% to 98.3% in tagging accuracy represents a relative reduction in tagging errors by 20%. Table 7 shows the change in tagging errors grouped

error type	tagging	+ parsing		change
		1-best	50-best	
verbs	299	146	112	-62.5%
nouns/names	372	372	381	+2.4%
pronouns	114	114	94	-17.5%
other	391	385	350	-10.5%
total	1176	1017	937	-20.3%

Table 7: Tagging errors grouped by gold POS. N=53935. CRF SMOR for tagging, ParZu with SMOR for parsing.

by different POS types. For verbs in German, tagging decisions are especially difficult to make locally because the part-of-speech tags encode some inflectional information, and the correct inflection may depend on non-local context. Both the heuristics in 1-best-tagging and tag sequence selection from n-best tagging help to reduce the number of verb tagging errors markedly, in total by 62.5%.

There are smaller improvements for pronouns and other parts-of-speech, including a better disambiguation between articles and pronouns. As an example of an ambiguity that is resolved through n-best-tagging, consider German *der*, which can mean 'the' (article), 'who' (relative pronoun), or 'this one' (demonstrative pronoun). 30–40% of tagging errors are due to confusions of NN (normal noun), NE (proper noun), and FM (foreign word). Parsing does not improve tagging accuracy for these parts-of-speech, mainly because ParZu makes little distinction between them.

In summary, we have demonstrated that we can perform n-best tagging, and use syntactic features extracted from ParZu for the selection of the best tag sequence. This allows us to disambiguate tagging ambiguities based on syntactic information, which improves both tagging accuracy and parsing performance.

5 Parsed Corpora as Training Treebanks

A third hurdle to the deployment of ParZu, and any data-driven parsers, is the limited availability of treebanks. We found that one complicating factor in the distribution of treebanks is that the creators of the treebank, i.e. the syntactic annotation layer, typically do not own the copyright to the original text. We thus investigate if it would be a viable alternative to use automatically annotated corpora as a training resource. Such an automatically annotated corpus would serve the same purpose as corpus masking (Rehm et al., 2007), i.e. allowing for the distribution of the annotation layer without infringing on the copyright of the original corpus, but while corpus masking loses lexical information, we can learn fully lexicalized statistics from automatically annotated corpora, at the cost of noise in the form of tagging/parsing errors.

In parsing, training on automatically parsed text is known as self-training. Self-training typically yields worse results than training on manually annotated data, with performance depending on the underlying parsing model (Steedman et al., 2003). There are, however, cases where self-training may be beneficial performance-wise, namely as a way to adapt systems to new domains (Steedman et al., 2003; Bacchiani et al., 2006), when using a re-ranker (Charniak and Johnson, 2005) or when considering the confidence score of the parser (Schneider, 2012).

We parsed the German portion of the Europarl corpus (Koehn, 2005) with ParZu, and extracted new statistics from this automatically parsed corpus. We chose Europarl because it has been used extensively in NLP research, especially in Statistical Machine Translation, and comes with no known usage restrictions. We compare three training sets: the original TüBa-D/Z, a training set of equal size (in terms of numbers of tokens: 1 million) from the parsed Europarl corpus, and the full Europarl corpus (1.8 million sentences; 47 million tokens).

We trained POS taggers and parsers on these corpora. For POS tagging, results are shown in table 8. While the taggers perform worse when trained on a segment of Europarl that is the same size as the TüBa-D/Z training corpus, this can be compensated by using the full Europarl corpus. For Sofies Welt, tagging accuracy almost reaches the level of the manually annotated training set, with a performance difference of 0.2-0.3 percentage points. On the TüBa-D/Z test set, the difference remains greater. However, this difference may be partially due to a second effect, namely that the TüBa-D/Z training corpus is in-domain in respect to the TüBa-D/Z test set, but Europarl is not.

Table 9 shows the performance for parsers trained on different training sets. We can see that the performance of MaltParser drops markedly when trained on parsed text, with a drop in f_1

tagger	treebank	TüBa	Sofies Welt
TnT	TüBa-D/Z	97.0	94.7
TnT	Europarl (1)	94.0	93.0
TnT	Europarl (47)	96.0	94.4
CRF	TüBa-D/Z	97.6	96.6
CRF	Europarl (1)	95.4	95.8
CRF	Europarl (47)	96.9	96.4

Table 8: POS tagging accuracy (in percent) with models trained on automatically annotated corpora. CRF with Morphisto.

system	treebank	parsing performance		
		precision	recall	f_1
ParZu	TüBa-D/Z	89.8	89.3	89.5
ParZu	Europarl (1)	89.0	88.5	88.7
ParZu	Europarl (47)	89.2	88.6	88.9
MaltParser	TüBa-D/Z	88.0	88.4	88.2
MaltParser	Europarl (1)	81.0	78.7	79.8
MaltParser	Europarl (47)	[training failed]		

Table 9: parsing performance (in percent) with models trained on automatically parsed text (gold POS tags; Morphisto).

by 8 percentage points. Performance of ParZu is more stable, and decreases by 0.6 percentage points when trained on the parsed Europarl corpus. The reason for this stability is that the role of statistical data in ParZu is limited to the disambiguation of some structures, with the grammar and morphology system constituting two other central knowledge sources for parsing, while MaltParser depends entirely on the data, and is thus more susceptible to noise. We also suspect that the fact that Europarl is from a different domain than the evaluation set accounts for some of the decrease in performance.

We conclude that the performance on self-trained data strongly depends on the statistical models, and also on the domains of the respective training and test sets. The CRF models with morphological features have shown to be more robust than a HMM tagger, and ParZu more robust than MaltParser in a self-training setting.

6 Conclusion

This paper discusses various interactions of three types of NLP tools: dependency parsers, POS taggers, and morphology tools. We demonstrate that the knowledge of morphology tools can be integrated into POS taggers through conditional random field (CRF) models, yielding very accurate models, which are also better at handling unknown words than conventional taggers. While the quality of POS tagging is important for parsing, POS tagging can also be improved with the help of a parser. We show that using n-best tagging, and parse selection based on syntactic features, can improve tagger accuracy. In our experiments, we measured an improvement of 0.5 percentage points in tagging accuracy, starting from a very competitive baseline of 97.8%. Our best system obtains a tagging accuracy of 98.3%, and labelled parsing f_1 of 88.0% on a TüBa-D/Z test set, compared to a baseline tagging accuracy of 94.9%, and labelled parsing f_1 of 84.9%, when using TreeTagger for POS tagging and MaltParser for parsing.

We also discuss and evaluate open alternatives to closed NLP resources. We perform an application-oriented evaluation of morphology tools, which shows that SMOR, both with the official Stuttgart lexicon and Morphisto, are competitive with GERTWOL for the purpose of extracting grammatical constraints, despite some technical challenges such as the idiosyncratic conception of lemmas in the SMOR grammar. Finally, we automatically annotate free corpora in order to use them for model training. These corpora can be distributed without infringing on the copyright of the corpora on which treebanks are based. Training models on these corpora leads to decreased performance compared to the manually annotated treebank, but performance is more robust with the models that integrate other knowledge sources, namely the CRF taggers with morphological features, and ParZu, which contains a hand-written grammar.

Acknowledgements

This research was funded by the Swiss National Science Foundation under grant 105215_126999.

References

Michiel Bacchiani, Michael Riley, Brian Roark, and Richard Sproat. 2006. Map adaptation of stochastic grammars. *Comput. Speech Lang.*, 20(1):41–68.

Miguel Ballesteros and Joakim Nivre. 2012. MaltOptimizer: A system for MaltParser optimization.

In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey.

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

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 173–180, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Killian A. Foth. 2005. *Eine umfassende Constraint-Dependenz-Grammatik des Deutschen*. University of Hamburg, Hamburg.

Mariikka Haapalainen and Ari Majorin. 1995. GERTWOL und Morphologische Disambiguierung für das Deutsche. In *Proceedings of the 10th Nordic Conference of Computational Linguistics*, Helsinki.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of the 10th Machine Translation Summit*, pages 79–86, Phuket, Thailand.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Thomas Lavergne, Olivier Cappé, and François Yvon. 2010. Practical very large scale CRFs. In *Proceedings the 48th Annual Meeting of the Association for Computational Linguistics*, pages 504–513, Uppsala, Sweden. Association for Computational Linguistics.

Cerstin Mahlow and Michael Piotrowski. 2009. A target-driven evaluation of morphological components for German. In *Searching Answers – Festschrift in Honour of Michael Hess on the Occasion of his 60th birthday*, pages 85–99. MV-Wissenschaft, Münster.

Joakim Nivre. 2009. Non-Projective Dependency Parsing in Expected Linear Time. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 351–359, Suntec, Singapore. Association for Computational Linguistics.

G. Rehm, A. Witt, H. Zinsmeister, and J. Dellert. 2007. Corpus masking: Legally bypassing licensing restrictions for the free distribution of text collections. *Digital Humanities*, pages 166–170.

Helmut Schmid, Arne Fitschen, and Ulrich Heid. 2004. A German computational morphology covering derivation, composition, and inflection. In *Proceedings of the IVth International Conference on Language Resources and Evaluation (LREC 2004)*, pages 1263–1266.

Gerold Schneider. 2008. *Hybrid Long-Distance Functional Dependency Parsing*. Ph.D. thesis, Institute of Computational Linguistics, University of Zurich.

Gerold Schneider. 2012. Using semantic resources to improve a syntactic dependency parser. In *SEM-II workshop at LREC 2012*, pages 67–76, Istanbul, Turkey.

Wolfgang Seeker, Bernd Bohnet, Lilja Øvrelid, and Jonas Kuhn. 2010. Informed ways of improving data-driven dependency parsing for German. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 1122–1130, Beijing, China. Association for Computational Linguistics.

Rico Sennrich, Gerold Schneider, Martin Volk, and Martin Warin. 2009. A New Hybrid Dependency Parser for German. In *Proceedings of the German Society for Computational Linguistics and Language Technology 2009*, pages 115–124, Potsdam, Germany.

Mark Steedman, Steven Baker, Jeremiah Crim, Stephen Clark, Julia Hockenmaier, Rebecca Hwa, Miles Osborne, Paul Ruhlen, and Anoop Sarkar. 2003. Semi-supervised training for statistical parsing. Technical Report CLSP WS-02 Final Report, Johns Hopkins University.

Heike Telljohann, Erhard W. Hinrichs, and Sandra Kübler. 2004. The TüBa-D/Z treebank: Annotating German with a context-free backbone. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation*, Lisbon, Portugal.

Yannick Versley. 2005. Parser evaluation across text types. In *Fourth Workshop on Treebanks and Linguistic Theories (TLT)*, Barcelona, Spain.

Martin Volk, Anne Göhring, Torsten Marek, and Yvonne Samuelsson. 2010. SMULTRON (version 3.0) – The Stockholm MULtilingual parallel TReebank. http://www.cl.uzh.ch/research/paralleltreebanks.html.

Andrea Zielinski and Christian Simon. 2009. Morphisto – an open source morphological analyzer for German. In *Proceedings of the 2009 conference on Finite-State Methods and Natural Language Processing: Post-proceedings of the 7th International Workshop FSMNLP 2008*, pages 224–231, Amsterdam. IOS Press.