# Training Parsers on Incompatible Treebanks

**Richard Johansson**
Språkbanken, Department of Swedish, University of Gothenburg
Box 200, SE-40530 Gothenburg, Sweden
richard.johansson@gu.se

## Abstract

We consider the problem of training a statistical parser in the situation when there are multiple treebanks available, and these treebanks are annotated according to different linguistic conventions. To address this problem, we present two simple adaptation methods: the first method is based on the idea of using a shared feature representation when parsing multiple treebanks, and the second method on guided parsing where the output of one parser provides features for a second one.
To evaluate and analyze the adaptation methods, we train parsers on treebank pairs in four languages: German, Swedish, Italian, and English. We see significant improvements for all eight treebanks when training on the full training sets. However, the clearest benefits are seen when we consider smaller training sets. Our experiments were carried out with unlabeled dependency parsers, but the methods can easily be generalized to other feature-based parsers.

## 1 Introduction

When developing a data-driven syntactic parser, we need to fit the parameters of its statistical model on a collection of syntactically annotated sentences – a *treebank*. Generally speaking, a larger collection of examples in the training treebank will give a higher quality of the resulting parser, but the cost in time and effort of annotating training sentences is fairly high. Most existing treebanks are in the range of a few thousand sentences.

However, there is an abundance of theoretical models of syntax and there is no consensus on how treebanks should be annotated. For some languages, there exist multiple treebanks annotated according to different syntactic theories. Apart from German, Swedish, and Italian, which will be considered in this paper, there are important examples among the world's major languages, such as Arabic and Chinese.

To exemplify how syntactic annotation conventions may differ in even such a simple case as unlabeled dependency annotation, consider the Italian sentence fragment *la sospensione o l'interruzione* ('the suspension or the interruption') in Figure 1. As we will see in detail in §3.1.3, there are two Italian treebanks: the ISST and TUT. If annotating as in the ISST treebank (drawn above the sentence) determiners (*la, l'*) are annotated as dependents of the following nouns (*sospensione, interruzione*); in TUT (drawn below the sentence), we have the reverse situation. There are also differences in how coordinate structures are represented: in ISST, the two conjuncts are directly conjoined and the conjunction attached to the first of them, while in TUT the conjunction acts as a link between the conjuncts.

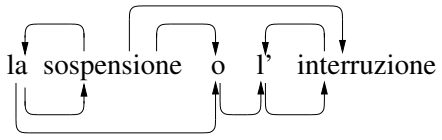



Figure 1: Differences in dependency annotation styles.

Given the high cost of treebank annotation and the importance of a proper amount of data for parser development, this situation is frustrating. How could we then make use of multiple treebanks when training a parser? A naïve way would be simply to concatenate them, but as we will see this results in a parser that performs badly on all the treebanks.

In this paper, we investigate two simple adaptation methods to bridge the gap between differing



*Proceedings of NAACL-HLT 2013*, pages 127–137,
Atlanta, Georgia, 9–14 June 2013. 

syntactic annotation styles, allowing us to use more data for parser training. The first approach treats the problem of parsing with multiple syntactic annotation styles as a multiview learning problem and addresses it by using feature representation that is partly shared between the views. In the second one we use a parser trained on one treebank to guide a new parser trained on another treebank. We evaluate these methods as well as their combination on four languages: German, Swedish, Italian, and English. In all four languages, we see a similar picture: the shared features approach is generally better when one of the treebanks is very small, while the guided parsing approach is better when the treebanks are more similar in size. However, for most training set sizes the combination of the the two methods achieves a higher performance than either of them individually.

## 2 Methods for Training Parsers on Multiple Treebanks

We now describe the two adaptation methods to leverage multiple treebanks for parser training. For clarity of presentation, we assume that there are two treebanks, although we can easily generalize to more. We use a common graph-based parsing technique (Carreras, 2007); the approaches described here could be used in transition-based parsing as well.

In a graph-based parser, for a given sentence $x$ the task of finding the top-scoring parse $\hat{y}$ is stated as an optimization problem of maximizing a linear objective function:

$$\hat{y} = \arg\max_{y} w \cdot f(x, y).$$

Here $w$ is a weight vector produced by some learning algorithm and $f(x, y)$ a *feature representation* that maps the sentence $x$ with a parse tree $y$ to a high-dimensional vector; the adaptation methods presented in this work is implemented as modifications of the feature representation function $f$. Since the search space is too large to be enumerated, the maximization must be handled carefully, and how this is done determines the expressivity of the feature representation $f$. In the parser by Carreras (2007) the maximization is carried out by a dynamic programming procedure relying on crucial independence assumptions to break down the search space into tractable parts. The factorization used in this approach allows $f$ to express features extracted not only from single edges, as McDonald et al. (2005), but also from sibling and grandchild edges.

To understand the machine learning problem of training parsers on incompatible treebanks, we compare it to the related problem of domain adaptation: training a system for a target domain, using a large collection of training data from a source domain combined with a small labeled or large unlabeled set from the target domain. Some algorithms for domain adaptation rely on the assumption that the differences between source and target distributions $P_s$ and $P_t$ can be explained in terms of a *covariate shift*: $P_s(y|x) = P_t(y|x)$ for all $x, y$, but $P_s(x) \neq P_t(x)$ for some $x$. In our case, we have the reverse situation: the input distribution is at least in theory unchanged between the two treebanks, while the input–output relation (i.e. the treebank annotation style) is different. However, domain adaptation and cross-treebank training can be seen as instances of the more general problem of *multitask learning* (Caruana, 1997). Indeed, one of the simplest and most well-known approaches to domain adaptation (Daumé III, 2007), which will also be considered in this paper, should more correctly be seen as a trick to handle multitask learning with any machine learning algorithm. On the other hand, there is no point in trying to use domain adaptation methods assuming a covariate shift, e.g. instance weighting, or any method in which the target data is unlabeled (Blitzer et al., 2007; Ben-David et al., 2010).

### 2.1 Sharing Feature Representations

Our first adaptation method relies on the intuition that some properties of two treebanks are shared, while others are unique to each of them. For instance, as we have seen in Figure 1 the two Italian treebanks annotate coordination differently; on the other hand, these treebanks also annotate several other linguistic phenomena in the same way. This observation can then be used to devise a model where we train two parsers at the same time and use a feature representation that is partly shared between the two models, allowing the machine learning algorithm to automatically determine which properties

of the two datasets are common and which are different. The idea of using features that are shared between the source and target training sets is a slight generalization of a well-known method for supervised domain adaptation (Daumé III, 2007).

In practice, this is implemented as follows. Assume that originally a sentence $x$ with a parse tree $y$ was represented as $f_1(x, y)$ if it came from the first treebank, and $f_2(x, y)$ if from the second treebank. We then add a *shared feature representation* $f_s$ to $f_1$ and $f_2$, and embed them into a single feature space. The resulting feature vectors then become

$$f_1(x, y) \oplus \mathbf{0}_2 \oplus f_s(x, y) \quad (1)$$

for a sentence from the first treebank, and

$$\mathbf{0}_1 \oplus f_2(x, y) \oplus f_s(x, y) \quad (2)$$

for the second treebank. Here, $\mathbf{0}_1$ means an all-zero vector with the dimensionality of the feature space of $f_1$, and $\oplus$ is vector concatenation. Using this new representation, the two datasets are combined and a single model trained. The hope is then that the learning algorithm will store the information about the respective particularities in the weights for $f_1$ and $f_2$, and about the commonalities in the weights for $f_s$. The result of this process is a symmetric parser that can handle both treebank formats: when we parse a sentence at test time, we just use the representation (1) if we want an output according to the first treebank and (2) for the second treebank.

In this work, $f_1$, $f_2$, and $f_s$ are identical: all of them correspond to the feature set described by Carreras (2007). However, it is certainly imaginable that $f_s$ could consist of specially tailored features that make generalization easier. In particular, using a generalized $f_s$ would allow us to use this approach in more complex cases than considered here, for instance if the dependencies would be labeled with two different sets of grammatical function labels, or if one of the treebanks would use constituents rather than dependencies.

### 2.2 Using One Parser to Guide Another

The second method is inspired by work in parser combination, an idea that has been applied successfully several times and relies on the fact that different parsing methods have different strengths and weaknesses (McDonald and Nivre, 2007), so that combining them may result in a better overall parsing accuracy. There are several ways to combine parsers; one of the simplest and most successful methods of parsing combination uses one parser as a *guide* for a second parser. This is normally implemented as a pipeline where the second parser extracts features based on the output of the first parser. Nivre and McDonald (2008) used this approach for combining a graph-based and a transition-based parser and achieved excellent results on test sets for several languages, and similar ideas were proposed by Martins et al. (2008).

We added guide features to the parser feature representation. However, the features by Nivre and McDonald (2008) are slightly too simple since they only describe whether two words are directly connected or not. That makes sense if the two parsers are trying to predict the same type of representation, but will not help us if there are systematic annotation differences between the two treebanks, for instance in whether to annotate a function word or a lexical word as the head. Instead, following work in semantic role labeling and similar areas, we use a generalized notion of syntactic relationship that we encode by determining a *path* between two nodes in a syntactic tree. We defined the function Path($x$, $y$) as a representation describing the steps required to traverse the parse tree from $x$ to $y$, first the steps up from $x$ to the common ancestor $a$ and then down from $a$ to $y$. Since we are working with unlabeled trees, the path can be represented as just two integers; to generalize to labeled dependency parsing, we could have used a full path representation as commonly used in dependency-based semantic role labeling (Johansson and Nugues, 2008).

We added the following path-based feature templates, assuming we have a potential head $h$ with dependent $d$, a sibling dependent $s$ and grandchild (dependent-of-dependent) $g$:

- POS($h$)+POS($d$)+Path($h$, $d$)
- POS($h$)+POS($s$)+Path($h$, $s$)
- POS($h$)+POS($d$)+POS($s$)+Path($h$, $s$)
- POS($h$)+POS($g$)+Path($h$, $g$)
- POS($h$)+POS($d$)+POS($g$)+Path($h$, $g$)

To exemplify, consider again the example *la sospensione o l'interruzione* shown in Figure 1. As-

sume that we are parsing according to the ISST representation (drawn above the sentence) and we consider adding an edge with *sospensione* as head and *la* as dependent, and another parser following the TUT representation (below the sentence) has created an edge in the opposite direction. The first feature template above would then result in a feature NOUN+DET+(1,0), where (1,0) represents the path relationship between the two words in the TUT tree (one step up, no step down). Similarly, when the ISST parser adds the coordination edge between *sospensione* and *interruzione*, it can make use of the information that these two nouns are *indirectly* connected in the output by the TUT parser; this is represented as a path (1,3). This is an example of a situation where we have a systematic correspondence where a single edge in one representation corresponds to several edges in the other.

Like the multiview approach described above, this method is trivially adaptable to more complex situations such as labeled dependency parsers with differing label sets, or dependency/constituent parsing.

### 2.3 Combining Methods

The two adaptation methods are orthogonal and can easily be combined. When trying to improve the performance of a parser trained on the primary treebank $T_1$ by leveraging a supporting treebank $T_2$, we then use $T_2$ in two different ways: first by training a guide parser, and secondly by concatenating it to $T_1$ using a shared feature representation.

## 3 Experiments

We carried out experiments to evaluate the cross-framework adaptation methods. The evaluations were carried out using the official CoNLL-X evaluation script using the default parameters. Since our parsers do not predict edge labels, we report unlabeled attachment scores in all tables and plots.

### 3.1 Treebanks Used in the Experiments

In our experiments, we used four languages: German, Swedish, Italian, and English. For each language, we had two treebanks. Our approaches currently require that the treebanks use the same tokenization conventions, so for Italian and Swedish we automatically retokenized the treebanks. We also made sure that the two treebanks for one language used the same part-of-speech tag sets, by applying an automatic tagger when necessary.

#### 3.1.1 German: Tiger and TüBa-D/Z

For German, there are two treebanks available: Tiger (Brants et al., 2002) and TüBa-D/Z (Telljohann et al., 2004). These treebanks are constituent treebanks, but dependency versions are available: TüBa-D/Z (version 7.0) includes the dependency version in the distribution, while for Tiger we used the version from CoNLL-X (Buchholz and Marsi, 2006). The constituent annotation styles in the two treebanks are radically different: Tiger uses a very flat structure with a minimal amount of intermediate nodes, while TüBa-D/Z uses a more elaborate structure including topological field information. However, the dependency versions are actually quite similar, at least with respect to attachment. The most common systematic difference we observed is in the annotation of coordination.

Both treebanks are large: for Tiger, the training set was 31,243 sentences and the test set 7,973 sentences, and for TüBa-D/Z 40,000 and 11,428 sentences respectively. We did not use the Tiger test set from the CoNLL-X shared task since it is very small. We applied the TreeTagger POS tagger (Schmid, 1994) to both treebanks, using the pre-trained German model.

#### 3.1.2 Swedish: Talbanken05 and Syntag

As previously noted by Nivre (2002) *inter alia*, Swedish has a venerable tradition in treebanking: there are not only one but two treebanks which must be counted among the earliest efforts of that kind. The oldest one is the Talbanken or MAMBA treebank (Einarsson, 1976), which has later been reprocessed for modern use (Nilsson et al., 2005). The original annotation is a function-tagged constituent syntax without phrase labels, but the reprocessed release includes a version converted to dependency syntax. The dependency treebank was used in the CoNLL-X Shared Task (Buchholz and Marsi, 2006), and we used that version version in this work.

The second treebank is called Syntag (Järborg, 1986). Similar to Talbanken, its representation uses function-tagged constituents but no phrase labels. We developed a conversion to dependency trees, which was straightforward since many constituents

have explicitly defined heads (Johansson, 2013).

The two treebank annotation styles have significant differences. Most prominently, the Syntag annotation is fairly semantically oriented in its treatment of function words such as prepositions and subordinating conjunctions: in Talbanken, a preposition is the head of a prepositional phrase, while in Syntag the head is the prepositional complement. There are also some domain differences: Talbanken consists of student essays and public information, while Syntag consists of news text.

To make the two treebanks compatible on the token level, we retokenized Syntag – which handles punctuation in an idiosyncratic way – and applied a POS tagger trained on the Stockholm–Umeå Corpus (Gustafson-Capková and Hartmann, 2006) to both treebanks. For Talbanken, we used 7,362 sentences for training and set aside a new test set of 3,680 sentences since the CoNLL-X test set is too small for serious experimental purposes – only 389 sentences. For Syntag, we split the treebank into 3,524 sentences for training and 1,763 sentences for testing.

#### 3.1.3 Italian: ISST and TUT

There are two Italian treebanks. The first is the Italian Syntactic–Semantic Treebank or ISST (Montemagni et al., 2003). Here, we used the version that was prepared (Montemagni and Simi, 2007) for the CoNLL-2007 Shared Task (Nivre et al., 2007).

The TUT treebank[1] is a more recent effort. This treebank is available in multiple constituent and dependency formats, and we have used the CoNLL-formatted dependency version in this work. The representation used in TUT is inspired by the Word Grammar theory (Hudson, 1984) and tends to be more surface-oriented than that of ISST. For instance, as pointed out above in the discussion of Figure 1, TUT differs from ISST in its treatment of determiner–noun constructions and coordination. It has been noted (Bosco and Lavelli, 2010; Bosco et al., 2010) that the TUT representation is easier to parse than the ISST representation.

We simplified the tokenization of both treebanks. In ISST, we split multiwords into separate tokens and reattached clitics to nonfinite verb forms. For instance, a single token *a_causa_di* was converted into three tokens *a, causa, di*, and the three tokens *trovar-se-lo* into a single token *trovarselo*. In TUT, we applied the same conversions and also recomposed preposition–article and multiple-clitic contractions that had been split by the annotators, e.g. *della, glielo* etc.[2] After changing the tokenization, we applied the TreeTagger POS tagger (Schmid, 1994) to both treebanks, using the pre-trained Italian model with the Baroni tagset[3].

After preprocessing the data, we created training and test sets. For ISST, the training set was 2,239 and the test set 1,120 sentences, while for TUT the training set was 1,906 and the test set 954 sentences.

#### 3.1.4 English: Two Different Conversions of the Penn Treebank

For English, there is no significant dependency treebank so we followed most previous work in using dependency trees automatically derived from constituent trees in the large Penn Treebank WSJ corpus (Marcus et al., 1993). Due to the fact that there is a highly parametrizable constituent-to-dependency conversion tool available (Johansson and Nugues, 2007), we could create two dependency treebanks with very different annotation styles.

The first training set was created from sections 02–12 of the WSJ corpus. By default, the conversion tool outputs a treebank using the annotation style of the CoNLL-2008 Shared Task (Surdeanu et al., 2008); however we wanted to create a more surface-oriented style for this treebank, so we turned on options to make *wh*-words heads of relative clauses, and possessive markers heads of noun phrases. This corpus had 20,706 sentences, and will be referred to as WSJ Part 1 in the experimental section.

The second training treebank was built from sections 13–22. For this treebank, we inverted the value of most options in order to get a more semantically oriented treebank where content words are connected directly. In this treebank, we also used "Prague-style" annotation of coordination: the conjuncts are annotated as dependents of the conjunction. This set contained 20,826 sentences, and will

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

[2] It should be noted that these conversions also make sense from a practical NLP point of view, since a number of contractions are homonymic with other words.

[3] http://sslmit.unibo.it/~baroni/collocazioni/itwac.tagset.txt

be called WSJ Part 2.

We finally applied both conversion methods to sections 24 and 23 to create development and test sets. The development set contained 1,346 and the test set 2,416 sentences. We did not change the tokenization or part-of-speech tags of the WSJ corpora. Here, we should note that we have a slightly more synthetic and controlled experimental setting than for Swedish and German: the parsers are evaluated on the same test set, so we know that there is no difference in test set difficulty. We also know *a priori* that performance differences are not due to any significant differences in genre, since all texts come from the same source (the Wall Street Journal) and tend to focus on business-oriented news.

### 3.2 Baseline Parsing Performance

As a starting point, we trained parsers on all treebanks. In addition, we created a parser using a naïve adaptation method by combining the training sets for each language, and training parsers on those three sets. We then applied all three parsers for every language on both test sets for that language. The results for German, Swedish, Italian, and English are presented in Table 1.

Every parser performed well on the test set annotated in the same annotation style as its training set. As has been observed previously, surface-oriented styles are easier to parse than semantically oriented styles: The Talbanken and WSJ Part 1 parsers all achieve much higher performance on their respective test sets than the Syntag and WSJ Part 2 parsers. The better performance of the Talbanken parser is also partly explainable by the fact that its training set is more than twice as large as the Syntag training set. Similarly for German, we see slightly higher performance for TüBa-D/Z than for Tiger.

However, as can be expected every parser performed very poorly when applied to the test set using the annotation style it was not trained on. For Swedish and English, the accuracy figures are in the range of 50-60, while the figure are a bit less poor for German since the two treebanks are more similar. We also see, again unsurprisingly, that the naïve combination baseline performs poorly in all situations: we just get a "worst-of-both-worlds" parser that performs badly on both test sets.

| GERMAN | Acc. on Tiger | Acc. on TBDZ |
|---|---|---|
| Tiger | 87.8 | 72.0 |
| TüBa-D/Z | 71.8 | 89.4 |
| Tiger+TBDZ | 77.7 | 87.7 |

| SWEDISH | Acc. on ST | Acc. on TB |
|---|---|---|
| Syntag | 81.4 | 52.6 |
| Talbanken | 50.3 | 88.2 |
| Syntag+Talbanken | 61.8 | 82.7 |

| ITALIAN | Acc. on ISST | Acc. on TUT |
|---|---|---|
| ISST | 81.1 | 57.4 |
| TUT | 55.9 | 84.0 |
| ISST+TUT | 73.9 | 71.6 |

| ENGLISH | Acc. on WSJ 1 | Acc. on WSJ 2 |
|---|---|---|
| WSJ part 1 | 92.6 | 57.4 |
| WSJ part 2 | 57.4 | 89.5 |
| WSJ parts 1+2 | 75.3 | 72.1 |

Table 1: Baseline performance figures.

### 3.3 Evaluation on the Full Training Sets

We trained new parsers using the shared features and guided parsing adaptation methods described in §2. Additionally, we trained parsers using both methods at the same time; we refer to these parsers as *combined*. Including the baseline parsers, this gave us 24 parsers to evaluate on their respective test sets.

The results for German are given in Table 2. Here, we see that all three adaptation methods give statistically significant[4] improvements over the baseline when parsing the Tiger treebank. In particular, the combined method gives a strong 0.7-point improvement, a 6% error reduction. For TüBa-D/Z, the improvements are smaller, although still significant except for the guided parsing method.

| Method | Acc. on Tiger | Acc. on TüBa-D/Z |
|---|---|---|
| Baseline | 87.8 | 89.4 |
| Shared | **88.1** | **89.6** |
| Guided | **88.4** | 89.5 |
| Combined | **88.5** | **89.6** |

Table 2: Performance figures for the German adapted parsers. Results that are significantly different from the baseline performances are written in boldface.

[4]At the 95% level. The significance levels of differences were computed using permutation tests.

| Method | Acc. on ST | Acc. on TB |
|---|---|---|
| Baseline | 81.4 | 88.2 |
| Shared | 81.3 | 88.3 |
| Guided | **82.5** | **88.4** |
| Combined | **82.5** | **88.5** |

Table 3: Performance of the Swedish adapted parsers.

For Swedish, we have a similar story: we see stronger improvements in the weak parser. Since the Talbanken treebank is twice as large as the Syntag treebank and has a surface-oriented representation that is easier to parse, this parser is useful as a guide for the Syntag parser: the improvements of the guided and combined Syntag parsers are statistically significant. However, it is harder to improve the Talbanken parser, for which the baseline is much stronger. 3 shows the results for the Swedish parsers.

| Method | Acc. on ISST | Acc. on TUT |
|---|---|---|
| Baseline | 81.1 | 84.0 |
| Shared | **81.5** | **84.4** |
| Guided | **81.7** | **84.3** |
| Combined | **81.8** | **84.7** |

Table 4: Performance of the Italian adapted parsers.

When we turn to the English corpora, the adaptation methods again gave us a number of very large improvements. The results are shown in Table 5. The shared features and combined methods gave statistically significant improvements for the WSJ Part 1 parser, and the guided parsing method an improvement that is nearly significant. However the most dramatic change is the 1.2-point improvement of the WSJ Part 2 parser, given by the guided parsing and combined methods. It is possible that this result partly can be explained by the fact that this experiment is a bit cleaner: in particular, as outlined in §3.1.4, there are no domain differences.

| Method | Acc. on WSJ 1 | Acc. on WSJ 2 |
|---|---|---|
| Baseline | 92.6 | 89.5 |
| Shared | **92.8** | 89.5 |
| Guided | 92.8 | **90.7** |
| Combined | **92.9** | **90.7** |

Table 5: Performance of the English adapted parsers.

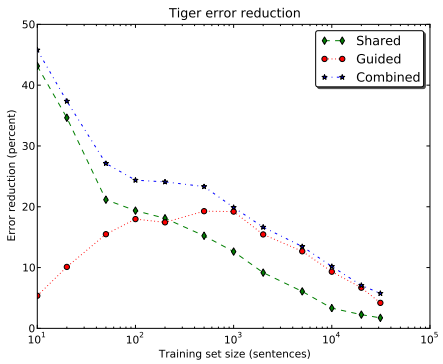
For WSJ Part 2, we analyzed the differences between the baseline and the best adapted parser. While there were improvements for all POS tags, the most notable one was in the attachment of conjunctions, where we got an increase from 69% to 75% in attachment accuracy, an 18% relative error reduction. Here we saw a very clear benefit of guided parsing: since this treebank uses "Prague-style" coordination annotation (i.e. the conjunction governs the conjuncts), it is hard for the parser to handle valencies and selectional preferences when there is a conjunction involved. It has been noted (Nilsson et al., 2007) that this style of annotating coordination is hard to parse. Since the WSJ Part 1 parser uses a coordination style that is easier to parse, the WSJ Part 2 parser can rely on its judgment.

Although conclusions must be very tentative since we are testing on just four languages, we can make a few general observations.

- The largest improvements (absolute and relative) all happen in treebanks that are harder to parse. In particular, Syntag and WSJ Part 2 are harder to parse due to their representation, and to some extent this may be true for Tiger as well – its learning curve rises more slowly than for TüBa-D/Z. Of course, in some cases (in particular Syntag, but also Tiger) this may partly be explained by the training set being smaller, but not for WSJ Part 2. In these cases, the guided parsing method seems to be more effective.
- The languages where the shared features method gives significant improvement for both treebanks are German and Italian, where we do not have the situation that one treebank is much larger or much easier to parse.
- The combination of the two methods gave significant improvements in all eight cases, and had the highest performance in six cases.

### 3.4 The Effect of the Training Set Size

In order to better understand the differences between the adaptation methods, we analyzed the impact of training set size on the improvement given by the respective methods. Let us refer to the training treebank annotated according to the same style as the test set as the *primary treebank*, and the other one as the *supporting treebank*. We carried out the experiments in this section by varying the number of



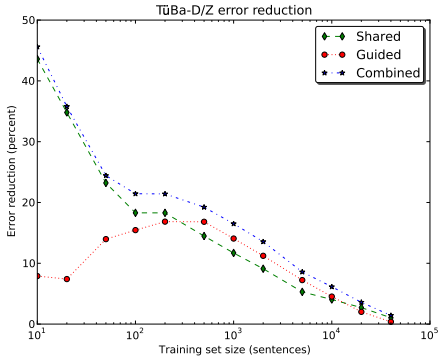
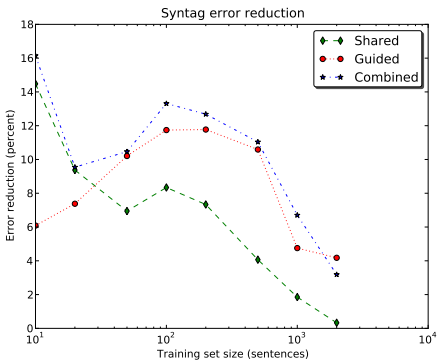




Figure 2: Error reduction by training set size, German.



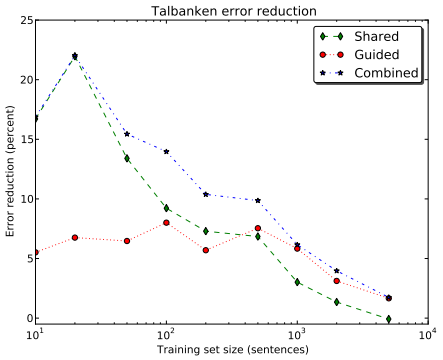



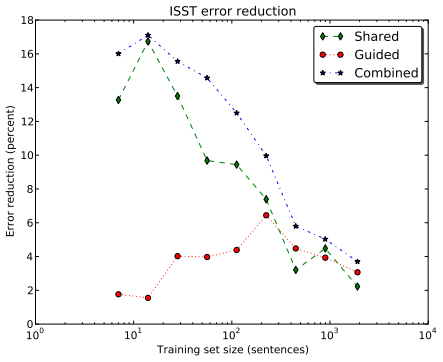

Figure 3: Error reduction by training set size, Swedish.

training sentences in the primary treebank and keeping the size of the supporting treebank constant.

In order to highlight the differences between the three adaptation methods, we show error reduction plots in Figures 2, 3, 4, and 5 for German, Swedish, Italian, and English respectively. For each training set size on the $x$ axis, the plot shows the reduction in relative error with respect to the baseline.

We note that *every single one* of the 24 adapted parsers learns faster than the corresponding baseline parser. While we saw a number of significant improvements in §3.3 when using the full training sets, the relative improvements are much stronger when the training sets are small- and medium-sized.

These plots illustrate the different properties of the two methods. Using a shared feature representation tends to be very effective when the primary treebank is small: the error reductions are over 40 percent for German and over 25 percent for English. Guided parsing works best for mid-sized sets, and the relative effectiveness of both methods decreases as the size of the primary treebank increases. Again, we see that guided parsing is less effective if the guide uses an annotation style that is hard to parse.



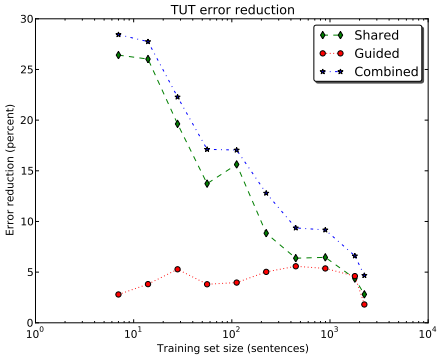




Figure 4: Error reduction by training set size, Italian.

In particular, for Swedish the Syntag parser never gives a very large improvement when guiding the Talbanken parser, and this is also true of both Italian parsers. To a smaller extent, this also holds for English and German: the WSJ Part 2 and Tiger parsers are less useful as guides than their counterparts.

The combination method generally performs very well: in all eight experiments, it outperforms the other two for almost every training set size. Its performance is very close to that of the guided parsing method for larger training sets, when the effect of the shared features method is less pronounced.

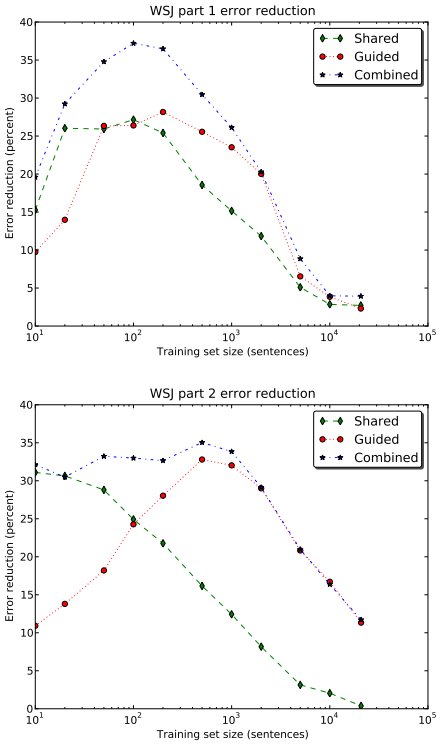



Figure 5: Error reduction by training set size, English.

## 4 Conclusion

We have considered the problem of training a dependency parser on incompatible treebanks, and we studied two very simple methods for addressing this problem, the shared features and guided parsing methods. These methods allow us to use more than one treebank when training dependency parsers. We evaluated the methods on eight treebanks in four languages, and had statistically significant improvements in all eight cases. In particular, for English we saw a strong 1.2-point absolute improvement (an 11% relative error reduction) in the performance of a semantically oriented parser when trained on the full training set. For German, we also had very strong results for the Tiger treebank: a 6% error reduction. For Swedish, the parser trained on the small Syntag treebank got a boost from a guide parser trained on the larger Talbanken. In general, it seems to be easier to improve parsers that use representations that are harder to parse.

For all eight treebanks, both methods achieved large improvements for small training set sizes, while the effect gradually diminished as the training set size increased. The shared features method was the most effective for very small training sets, while guided parsing surpassed it when training sets got larger. The combination of the two methods was also effective, in most cases outperforming both methods on their own. In particular, when using the full training sets, this was the only method that had statistically significant improvements for all treebanks.

While this work used an unlabeled graph-based dependency parser, our methods generalize naturally to other parsing approaches, including transition-based dependency parsing. Labeled parsing with incompatible label sets is easy to implement in the shared features framework by removing the label information from the shared feature representation $f_s$, and similar modifications of $f_s$ could be carried out to handle more complex situations such as combined constituent and dependency parsing. Furthermore, the paths used by the feature extractor in the guided parser can be extended without much effort as well. The models presented here are very simple, and in future work we would like to explore more complex approaches such as quasi-synchronous grammars (Smith and Eisner, 2009; Li et al., 2012) or automatic treebank transformation (Niu et al., 2009).

## Acknowledgements

I am grateful to the anonymous reviewers, whose feedback has helped to clarify the description of the methods. This research was supported by University of Gothenburg through its support of the Centre for Language Technology and Språkbanken. It has been partly funded by the Swedish Research Council under grant number 2012-5738.

## References


Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A theory of learning from different domains. *Machine Learning*, 2010(79):151–175.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, Bollywood, Boom-boxes and Blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 440–447, Prague, Czech Republic.

Cristina Bosco and Alberto Lavelli. 2010. Annotation schema oriented validation for dependency parsing evaluation. In *Proceedings of the Ninth Workshop on Treebanks and Linguistic Theories (TLT9)*, Tartu, Estonia.

Cristina Bosco, Simonetta Montemagni, Alessandro Mazzei, Vincenzo Lombardo, Felice Dell'Orletta, Alessandro Lenci, Leonardo Lesmo, Giuseppe Attardi, Maria Simi, Alberto Lavelli, Johan Hall, Jens Nilsson, and Joakim Nivre. 2010. Comparing the influence of different treebank annotations on dependency parsing. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, pages 1794–1801, Valletta, Malta.

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

Sabine Buchholz and Erwin Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, pages 149–164, New York City, United States.

Xavier Carreras. 2007. Experiments with a higher-order projective dependency parser. In *Proceedings the CoNLL Shared Task*, pages 957–961, Prague, Czech Republic.

Rich Caruana. 1997. Multitask learning. *Machine Learning*, 28(1):41–75.

Hal Daumé III. 2007. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 256–263, Prague, Czech Republic.

Jan Einarsson. 1976. Talbankens skriftspråkskonkordans. Department of Scandinavian Languages, Lund University.

Sofia Gustafson-Capková and Britt Hartmann. 2006. Manual of the Stockholm Umeå Corpus version 2.0. Stockholm University.

Richard Hudson. 1984. *Word Grammar*. Blackwell.

Jerker Järborg. 1986. Manual för syntaggning. Department of Linguistic Computation, University of Gothenburg.

Richard Johansson and Pierre Nugues. 2007. Extended constituent-to-dependency conversion for English. In *NODALIDA 2007 Conference Proceedings*, pages 105–112, Tartu, Estonia.

Richard Johansson and Pierre Nugues. 2008. The effect of syntactic representation on semantic role labeling. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 393–400, Manchester, United Kingdom.

Richard Johansson. 2013. Bridging the gap between two Swedish treebanks. *Northern European Journal of Language Technology*. Submitted.

Zhenghua Li, Ting Liu, and Wanxiang Che. 2012. Exploiting multiple treebanks for parsing with quasi-synchronous grammars. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 675–684, Jeju Island, Korea.

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

André F. T. Martins, Dipanjan Das, Noah A. Smith, and Eric P. Xing. 2008. Stacking dependency parsers. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 157–166, Honolulu, United States.

Ryan McDonald and Joakim Nivre. 2007. Characterizing the errors of data-driven dependency parsing models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 122–131, Prague, Czech Republic.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 91–98, Ann Arbor, United States.

Simonetta Montemagni and Maria Simi. 2007. The Italian dependency annotated corpus developed for the CoNLL-2007 shared task. Technical report, ILC-CNR.

Simonetta Montemagni, Francesco Barsotti, Marco Battista, Nicoletta Calzolari, Ornella Corazzari, Alessandro Lenci, Antonio Zampolli, Francesca Fanciulli, Maria Massetani, Remo Raffaelli, Roberto Basili, Maria Teresa Pazienza, Dario Saracino, Fabio Zanzotto, Nadia Mana, Fabio Pianesi, and Rodolfo Delmonte. 2003. Building the Italian Syntactic–Semantic

Treebank. In Anne Abeillé, editor, *Building and Using Syntactically Annotated Corpora*. Kluwer, Dordrecht.

Jens Nilsson, Johan Hall, and Joakim Nivre. 2005. MAMBA meets TIGER: Reconstructing a Swedish treebank from antiquity. In *Proceedings of NODALIDA Special Session on Treebanks*.

Jens Nilsson, Joakim Nivre, and Johan Hall. 2007. Generalizing tree transformations for inductive dependency parsing. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 968–975, Prague, Czech Republic.

Zheng-Yu Niu, Haifeng Wang, and Hua Wu. 2009. Exploiting heterogeneous treebanks for parsing. 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 46–54, Suntec, Singapore.

Joakim Nivre and Ryan McDonald. 2008. Integrating graph-based and transition-based dependency parsers. In *Proceedings of ACL-08: HLT*, pages 950–958, Columbus, United States.

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932, Prague, Czech Republic.

Joakim Nivre. 2002. What kinds of trees grow in Swedish soil? A comparison of four annotation schemes for Swedish. In *Proceedings of the First Workshop on Treebanks and Linguistic Theories (TLT2002)*, Sozopol, Bulgaria.

Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of International Conference on New Methods in Language Processing*, Manchester, United Kingdom.

David A. Smith and Jason Eisner. 2009. Parser adaptation and projection with quasi-synchronous grammar features. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 822–831, Suntec, Singapore.

Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. 2008. The CoNLL-2008 shared task on joint parsing of syntactic and semantic dependencies. In *CoNLL 2008: Proceedings of the Twelfth Conference on Natural Language Learning*, pages 159–177, Manchester, United Kingdom.

Heike Telljohann, Erhard Hinrichs, and Sandra Kbler. 2004. The Tüba-D/Z treebank: Annotating German with a context-free backbone. In *In Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC 2004)*, pages 2229–2235.