Should Have, Would Have, Could Have

Investigating Verb Group Representations for Parsing with Universal Dependencies

Miryam de Lhoneux and **Joakim Nivre**
Uppsala University
Department of Linguistics and Philology
{miryam.de_lhoneux,joakim.nivre}@lingfil.uu.se

Abstract

Treebanks have recently been released for a number of languages with the harmonized annotation created by the Universal Dependencies project. The representation of certain constructions in UD are known to be suboptimal for parsing and may be worth transforming for the purpose of parsing. In this paper, we focus on the representation of verb groups. Several studies have shown that parsing works better when auxiliaries are the head of auxiliary dependency relations which is not the case in UD. We therefore transformed verb groups in UD treebanks, parsed the test set and transformed it back, and contrary to expectations, observed significant decreases in accuracy. We provide suggestive evidence that improvements in previous studies were obtained because the transformation helps disambiguating POS tags of main verbs and auxiliaries. The question of why parsing accuracy decreases with this approach in the case of UD is left open.

1 Introduction

Universal Dependencies[1] (henceforth UD) (Nivre, 2015) is a recent project that is attempting to harmonize syntactic annotation in dependency treebanks across languages. This is done through the development of annotation guidelines. Some guidelines have been hypothesized to be suboptimal for parsing. In the literature, certain representations of certain constructions have been shown to be better than their alternatives for parsing, for example in Schwartz et al. (2012). The UD guidelines however have been written with the intent to maximize cross-linguistic parallelism and this constraint has forced the guidelines developers to sometimes choose representations that are known to be worse for parsing (de Marneffe et al., 2014). For that reason, de Marneffe et al. (2014) suggest that those representations could be modified for the purpose of parsing, thus creating a *parsing representation*. Transforming tree representations for the purpose of parsing is not a new idea. It has been done for constituency parsing for example by Collins (1999) but also for dependency parsing for example by Nilsson et al. (2007). Nilsson et al. (2007) modified the representation of several constructions in several languages and obtained a consistent improvement in parsing accuracy. In this paper, we will investigate the case of the verb group construction and attempt to reproduce the study by Nilsson et al. (2007) on UD treebanks to find out whether or not the alternative representation is useful for parsing with UD.

2 Background

2.1 Tree Transformations for Parsing

Nilsson et al. (2006) have shown that modifying coordination constructions and verb groups from their representation in the Prague Dependency Treebank (henceforth PDT) to a representation described in Melčuk (1988) (Mel'čuk style, henceforth MS) improves dependency parsing for Czech. The procedure they follow is as follows:

1. Transform the training data.

[1] http://universaldependencies.github.io/docs/

Proceedings of the Workshop on Multilingual and Cross-lingual Methods in NLP, pages 10–19,
San Diego, California, June 17, 2016. ©2016 Association for Computational Linguistics

2. Train a model on that transformed data.

3. Parse the test data.

4. Transform the parsed data back to the original representation (for comparison with the original gold standard).

Nilsson et al. (2007) have shown that these same modifications as well as the modification of non-projective structures helps parsing in four languages. Schwartz et al. (2012) conducted a study over the alternative representations of 6 constructions across 5 parsing models for English and found that some of them are easier to parse than others. Their results were consistent across parsing models.

The motivations behind those two types of studies are different. Nilsson et al. (2006) have originally a representation that is more semantically oriented and potentially useful for NLP applications which they therefore wish their output to have, the PDT style, and change it to a representation that is more syntactically oriented, the MS style, because it is easier to parse. By contrast, Schwartz et al. (2012) have no a priori preference for any of the different alternatives of the constructions they study and instead study the effect of the different representations on parsing for the purpose of choosing one representation over the other. Their methodology is therefore different, they evaluate the different representations on their respective gold standard. They argue that accuracy within a representation is a good indicator of the learnability of that representation and they argue that learnability is a good criterion for selecting a syntactic representation among alternatives. In any case, these studies seem to show that such transformations can affect parsing for various languages and for various parsing models.

Silveira and Manning (2015) were the first to obtain negative results from such transformations. They attempted to modify certain constructions in a UD treebank to improve parsing for English but failed to show any improvement. Some transformations even decreased parsing accuracy. They observe that when they transform their parsed data back to the original representation, they can amplify parser errors. As a matter of fact, a transformation can be prompted by the presence of only one dependency relation but involve transformations of many surrounding dependency relations. The verb group transformation is such an example and will be described in section 3. If, then, a wrong dependency relation prompts a transformation in the parsed data, its surrounding items which might have been correct become wrong. A wrong parse can then become worse. They take this as partial explanation for the results that are inconsistent with the literature. However, the same problem can have arisen in Nilsson et al. (2006) and may have downplayed the effects that those studies have observed. It therefore seems that this explanation is not enough to account for those results.

This raises the question of whether this phenomenon actually happened in the study by Nilsson et al. (2007). It would be interesting to know if the effects they observed were affected by this kind of error amplification. It seems that there is still a lot to do to study the impact of different representations on parsing with UD as well as on dependency parsing more generally. We propose to take one step in that direction in this paper.

2.2 Error Analysis for Dependency Parsing

McDonald and Nivre (2007) conducted an extensive error analysis on two parsers in order to compare them. They compare the effect of sentence length on the two models, the effect of the structure of the graph (i.e. how close to the root individual arcs are) on the two models as well as the accuracy of the models on different POS tags and on different dependency relations. These comparisons allow them to provide insights into the strengths and weaknesses of each model. Conducting such an error analysis that compares baseline models with their transformed version could provide some further insights into the effects obtained with tree transformations. Attempting such a detailed error analysis is beyond the scope of this project but some steps will be taken in that direction and are described in Section 4.

2.3 Verb Groups

In the PDT, main verbs are the head of auxiliary dependencies, as in Figure 1. Nilsson et al. (2007) show that making the auxiliary the head of the dependency as in Figure 2 is useful for parsing Czech and Slovenian.

Schwartz et al. (2012) also report that, in English,

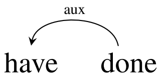

Figure 1: PDT representation of verb groups

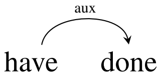

Figure 2: MS representation of verb groups

verb groups are easier to parse when the auxiliary is the head (as in PDT) than when the verb is the head (as in MS). Since UD adopts the PDT style representation of verb groups, it would be interesting to find out whether or not transforming them to MS could also improve parsing. This is what will be attempted in this study.

Nilsson et al. (2006) describe algorithms for such a transformation as well as its back transformation. However, their back transformation algorithm assumes that the auxiliary appears to the left of the verb which is not always the case. In addition, it is unclear what they do with the cases in which there are two auxiliaries in a verb group. For these reasons, we will use a slightly modified version of this algorithm that we describe in Section 3.

3 Methodology

3.1 General Approach

We will follow the methodology from Nilsson et al. (2007), that is, to transform, parse and then detransform the data so as to compare the original and the transformed model on the original gold standard. The method from Schwartz et al. (2012) which consists in comparing the baseline and the transformed data on their respective gold standard is less relevant here because UD is believed to be a useful representation and that the aim will be to improve parsing within that representation. However, as was argued in that study, their method can give an indication of the learnability of a construction and can potentially be used to understand the results obtained by the parse-transform-detransform method. For this reason, this method will also be attempted. In addition, the original parsed data will also be transformed into the MS gold standard for comparison with the MS parsed data on the MS gold standard. Comparing the two can potentially help find out if the error amplifications described in the background section are strongly influencing the results. As a matter of fact, if the transformed model is penalized by error amplifications on the original gold standard, it is expected that the original model will be penalized in the same way on the transformed gold standard.

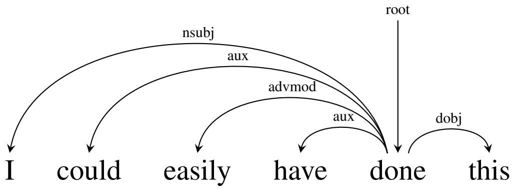

Figure 3: Example (1) annotated in UD style

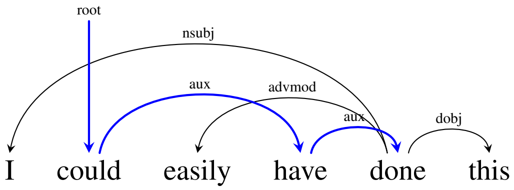

Figure 4: Intermediate representation between UD and MS of Example (1). Thick blue dependencies are those that changed compared to Figure 3.

3.2 Transformation Algorithm

The transformation algorithm is illustrated by Figures 3, 4 and 5 which represent the transformation of a sentence with a verb group given in Example (1). Figure 3 is the original UD representation of this example, Figure 4, an intermediate representation and Figure 5 is the final MS representation.

(1) I could easily have done this

The transformation first looks for verb groups in a dependency graph. Those verb groups are collected in the set V. A verb group V_i has a main verb $V_{i_{mv}}$ (*done* in the example) and a set of auxiliaries $V_{i_{aux}}$ with at least one element (*could* and *have* in the example). Verb groups are collected by traversing the sentence from left to right, looking at auxiliary dependency relations. An auxiliary dependency relation $w_{aux} \xleftarrow{aux} w_{mv}$ is a relation where the main verb is the head and the auxiliary is the dependent. Only auxiliary dependency relations between two verbal forms are considered. When such a dependency relation is found, if there is a V_i in V that has the head of the dependency relation (w_{mv}) as main verb $V_{i_{mv}}$,

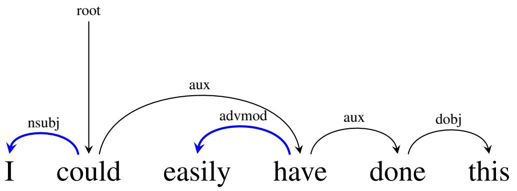

Figure 5: Example (1) annotated in MS style. Thick blue dependencies are those that changed compared to Figure 4.

w_{aux} is added to that V_i's set of auxiliaries $V_{i_{aux}}$. Otherwise, a new V_i is created and added to V.

After that, for each V_i in V, if there is only one auxiliary in $V_{i_{aux}}$, the direction of the dependency relation between that auxiliary and the main verb $V_{i_{mv}}$ is inverted and the head of $V_{i_{mv}}$ becomes the head of the auxiliary. When there are several auxiliaries (like in example (1)), the algorithm attaches the closest one to $V_{i_{mv}}$ and the head of $V_{i_{mv}}$ becomes the head of the outermost one. Any auxiliary in-between is attached in a chain from the outermost to the one that is closest to the verb. In the example, the main verb *done* gets attached to the closest auxiliary *have* and the head of the main verb *done* which was the root becomes the head of the outermost auxiliary, *could*.

Next, dependents of the main verb are dealt with to make sure projectivity is maintained. As a matter of fact, as can be seen from Figure 4, the previous changes can introduce non-projectivity in an otherwise projective tree, which is undesirable. Dependents to the left of the leftmost verb of the whole verb group (i.e. including the auxiliaries and the main verb) get attached to the leftmost verb. In the example, *I* gets attached to *could*. Dependents to the right of the rightmost verb of the verb group get attached to the leftmost verb. In the example, *this* remains attached to the main verb *done*. Any remaining dependent gets attached to the auxiliary that is closest to the verb. In the example, *easily* gets attached to *have*.

3.3 Back Transformation Algorithm

The back transformation algorithm works similarly to the transformation algorithm. A set of verb groups V is first collected by traversing the sentence from left to right, looking at auxiliary dependency relations. An auxiliary dependency relation $w_d \xleftarrow{aux} w_h$ between a dependent w_d and a head w_h in MS can be between an auxiliary and the main verb or between two auxiliaries. When one such relation is found, if its head w_h is not already in a $V_{i_{aux}}$ in V, a new verb group V_i is created and w_h is added to $V_{i_{aux}}$. What the algorithm does next depends on the direction of that dependency relation. If it is right-headed, the dependent w_d of that dependency relation is the main verb and the algorithm recurses the chain of auxiliary dependency relations through heads: it looks at the head w_h of dependency relations and adds them to $V_{i_{aux}}$ until it finds a head that is not itself the dependent of an auxiliary dependency relation. If it is left-headed, the algorithm recurses the chain of auxiliary dependency relations through the dependents. It looks at dependents of dependency relations until it finds the main verb $V_{i_{mv}}$, i.e. a w_{i_d} that is not the head of an auxiliary dependency relation, each time adding the head of the relation w_{i_h} to $V_{i_{aux}}$. After that, for each V_i in V, the head of the auxiliary that is furthest from the main verb becomes the head of the main verb. The main verb becomes the head of all auxiliaries and their dependents.

In the previous example, Figure 5 can be transformed back to Figure 3 in this way: *done* is identified as the main verb of the verb group and *could* as its furthest auxiliary. The head of *could* therefore becomes the head of *done* and the two auxiliaries of the sentence as well as their dependents get attached to the main verb.

3.4 Data

We ran all experiments on UD 1.2 (Nivre et al., 2015). Treebanks that had 0.1% or less of auxiliary dependency relations were discarded. Japanese was also discarded because the Japanese treebank is not open source. Dutch was discarded because the back transformation accuracy was low (90%). This is due to inconsistencies in the annotation: verb groups are annotated as a chain of dependency relations. This leaves us with a total of 25 out of the 37 treebanks. For comparability with the study in Nilsson et al. (2007), and because we used a slightly modified version of their algorithm, we also tested the approach on the versions of the Czech and Slovenian treebanks that they worked on, respectively version 1.0 of the PDT (Hajic et al., 2001) and the 2006 version of SDT (Deroski et al., 2006). Table 1 gives an

Treebank	#S	#W	%A
SDT	1,936	35K	9.45
PDT	80,407	1,382K	1.38
Basque	7,194	97K	8.51
Bulgarian	10,022	141K	1.03
Croatian	3,757	84K	3.87
Czech	77,765	1,333K	0.92
Danish	5,190	95K	2.29
English	14,545	230K	2.85
Estonian	1,184	9K	0.73
Finnish	12,933	172K	1.49
Finnish-FTB	16,913	143K	2.89
French	16,148	394K	1.45
German	14,917	282K	1.05
Greek	2,170	53K	0.36
Hebrew	5,725	147K	0.15
Hindi	14,963	316K	3.27
Italian	12,188	260K	1.87
Norwegian	18,106	281K	2.60
Old_Church_Slavonic	5,782	52K	0.35
Persian	5,397	137K	1.40
Polish	7,500	76K	0.97
Portuguese	9,071	207K	0.20
Romanian	557	11K	2.88
Slovenian	7,206	126K	4.57
Spanish	15,739	424K	0.89
Swedish	4,807	76K	2.37
Tamil	480	8K	5.30

Table 1: Data sets: train + development; S= sentence, W=word; A=auxiliary dependency relation.

overview of the data used for the experiments.

3.5 Software

For comparability with previous studies, we used MaltParser (Nivre et al., 2006) with default settings, training on the training set and parsing on the development set for all the languages that we investigated. For enhanced comparability of the results, we used the UD POS tags instead of the language specific POS tags. MaltEval (Nilsson and Nivre, 2008) was used for evaluation. The transformation code has been released as part of the python package oDETTE version 1.0[2] (DEpendency Treebank Transformation and Evaluation). The package can be used to run the whole pipeline, from transformation to evaluation. It can work on several treebanks in parallel which enables quick experiments. (We trained and parsed the data for the 25 treebanks in 9 minutes on an 8-core machine).

4 Results

4.1 Effect of VG Transformation on Parsing

As mentioned before, we converted training data in all treebanks involved, trained a parser with that transformed training set, parsed the test data and transformed the parsed data back to the original representation. Parsing accuracy of that transformed parsed data can then be compared with the parsed data obtained from the baseline, the unmodified model. Results are given in Table 2. All results report Labeled Attachment Scores (henceforth LAS) using MaltParser. As Unlabeled Attachment Scores (UAS) showed similar tendencies to LAS, they are not included for clarity. All experiments report significance levels for the McNemar test as obtained by MaltEval.[3] A 100% accuracy was obtained for the back transformation of all data sets except for UD Spanish, Portuguese, Romanian and Hindi (99.9, 99.7, 99.4 and 99% respectively). As can be seen from the table and contrary to expectations, by and large the results decrease significantly with the transformed version of the treebank with a few exceptions but no result increases significantly.

As mentioned in Section 2.1, results on the original representation are the ones that we care about because it is the UD representation that we are interested in and because those results are directly comparable with each other. However, as was also said, results on the transformed gold standard can give an indication on the learnability of a construction. For this reason, they are reported in Table 3. Table 3 also reports results of the parsing model trained on UD representations where the parsed data have been transformed to the MS representation. As was said in Section 3.1, this is to find out if error amplifications have a strong influence on the results: if error amplifications were the main source of added errors from the baseline on UD to the back transformed UD, it would be expected that the original parsed test set transformed into MS would perform worse on the MS gold standard than the test set parsed by the model trained on MS. As can be seen from Table 3 however, this is not the case: the original model generally beats the transformed model even on the transformed gold standard. As can also be seen from the

[2] `https://github.com/mdelhoneux/oDETTE/archive/v1.0.tar.gz`

[3] In the tables, $* = p < .05$ and $** = p < .01$.

UD language	**Orig**	**Transf**
Basque	**64.4**	63.8**
Bulgarian	**83.4**	83.2*
Croatian	**75.9**	74.6**
Czech	**80.0**	76.5**
Danish	**75.9**	75.2**
English	**81.7**	80.4**
Estonian	77.1	**77.8**
Finnish	**66.9**	66.4*
Finnish-FTB	**71.3**	70.4**
French	**82.1**	81.6**
German	**76.6**	76.0**
Greek	75.2	**75.3**
Hebrew	**78.4**	77.9**
Hindi	**85.4**	84.2**
Italian	**83.8**	83.6
Norwegian	**84.5**	82.0**
Old_Church_Slavonic	**68.8**	68.7
Persian	**81.1**	79.8**
Polish	**79.4**	79.1
Portuguese	81.3	**81.5**
Romanian	**64.2**	62.5*
Slovenian	**80.8**	79.7**
Spanish	**81.5**	81.2**
Swedish	**76.8**	75.7**
Tamil	**67.2**	67.1

Table 2: LAS with the original and transformed treebanks.

UD language/ Gold	**Orig UD**	**Orig MS**	**Transf MS**
Basque	**64.4**	**64.4**	64.0
Bulgarian	**83.4**	82.9	*82.5*
Croatian	**75.9**	**75.9**	73.7
Czech	**80.0**	*79.9*	76.4
Danish	**75.9**	*75.8*	74.8
English	**81.7**	*81.5*	80.2
Estonian	*77.1*	77.0	**77.6**
Finnish	**66.9**	*66.4*	65.9
Finnish-FTB	71.3	**72.5**	*72.1*
French	**82.1**	*81.8*	81.3
German	**76.6**	*76.1*	75.4
Greek	**75.2**	**75.2**	75.1
Hebrew	*78.4*	**78.5**	77.9
Hindi	**85.4**	*85.2*	84.9
Italian	**83.8**	*83.6*	83.3
Norwegian	**84.5**	**84.5**	81.7
Old_Church_Slavonic	*68.8*	**68.9**	68.7
Persian	**81.1**	**81.1**	79.8
Polish	**79.4**	*79.3*	79.0
Portuguese	**81.3**	**81.3**	81.6
Romanian	*64.2*	**64.6**	64.0
Slovenian	**80.8**	**80.8**	79.8
Spanish	**81.5**	*81.4*	81.2
Swedish	**76.8**	*76.7*	75.6
Tamil	67.2	**67.5**	*67.4*

Table 3: LAS on original and transformed treebanks on original gold standard (UD) and transformed (MS). Highest score in bold and second highest in Italics.

table, the scores are overall higher for the UD parsing model on the UD gold standard than the transformed parsing model on the transformed gold standard. This potentially indicates that the verb group transformation makes the UD representation harder to learn and might help give a partial explanation of why it decreases parsing accuracy on the original gold standard. This is not entirely surprising as, as can be seen from the Figures illustrating the transformation above, the original representation is flatter than the transformed representation. Further work is needed to explore that more in-depth. In any case, the original model beats the transformed model on several metrics and it seems safe to conclude that the verb group transformation hurts UD parsing at least with MaltParser.

4.2 Comparing Dependency Relations

Turning to the error analysis, one thing that is striking when looking at the performance of different dependency relations is that punctuation performs consistently worse in the transformed version of the parsed data compared to the baseline as can be seen in Figure 6.[4]

Because punctuation is most often attached to the main verb, it can be hypothesized that identifying the main verb of the sentence is crucial for avoiding this kind of errors and that the transformation hurts the identification of the main verb in the case of UD. A close examination of about a third of errors containing an auxiliary dependency relation in English further reinforced that hypothesis.

4.3 Comparison with SDT and PDT

What is noticeable in the results we have seen so far is that the accuracy decreased for languages for which accuracy has been shown to increase in the past: Czech, Slovenian and English. This indicates that the UD style is making a difference. For that reason, we are now attempting a comparison be-

[4] Punctuation is often excluded from evaluation for several reasons so it is important to say that although punctuation is affected by the score, the overall trend in the evaluation does not change if it is excluded which indicates that its decrease in accuracy is a symptom of what is going on.

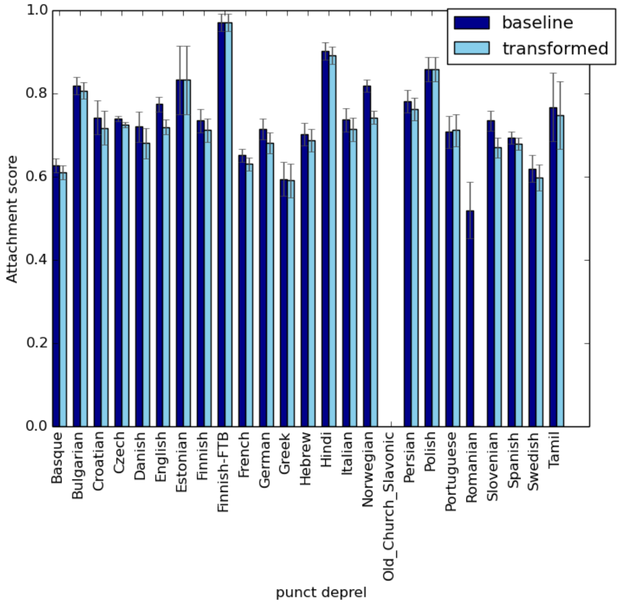

Figure 6: F1 score and error margin in parsed test set

	Orig	**Transf**
UD_Czech	**80.0**	76.5**
PDT	68.5	**68.8****
UD_Slovenian	**80.5**	79.1**
SDT	65.7	**66.2**

Table 4: LAS with the original and transformed treebanks.

tween the effect of the approach on SDT and on UD_Slovenian as well as between its effect on PDT and UD_Czech. As shown in Table 4, similar improvements to the original study were obtained on SDT and PDT.

As was just mentioned, it can be hypothesized that identifying the main verb is crucial for avoiding the kind of errors that were observed in the UD transformed version. It can then be hypothesized that the transformation helps to identify the main verb in PDT and SDT whereas it makes it harder in UD. When observing some examples in SDT, the transformation seems to help disambiguating POS tags. As a matter of fact, more than 90% of auxiliaries in SDT have the tag *Verb-copula* but also more than 20% of the main verbs involved in auxiliary dependency relations have that same POS tag. POS tags therefore do not give enough information to distinguish between the main verb and an auxiliary.

The experiment we are now turning to suggested that this is a reasonable hypothesis. We tested the approach on three different versions of PDT and SDT (i.e. we changed both training and test data, trained the model on the transformed training set, parsed on transformed test set and compared with the transformed gold standard). In the original version τ_o, we did not change anything. We created a disambiguated version τ_d, in which main verbs are tagged as *Verb-main* for SDT, and *Vp* for PDT and where auxiliaries are tagged *aux*. We created an ambiguous version τ_a, where we made verbs fully ambiguous, i.e. all verbal tags are transformed to the same *verb* POS tag.

	Orig	**Transf**	Δ
SDT τ_d	**67.8**	67.4	-0.4
SDT τ_o	65.7	**66.2**	0.5
SDT τ_a	64.2	**65.4***	1.2
PDT τ_d	69.2	69.2	0.0
PDT τ_o	68.5	**68.8****	0.3
PDT τ_a	68.2	**68.4***	0.2

Table 5: LAS on the original and transformed treebanks with different levels of POS tag ambiguity. Δ = Transf - Orig

As appears from Table 5, the results in SDT support the hypothesis: when verbs are made fully ambiguous, the transformation improves the results more than when they are partially ambiguous. When they are disambiguated, the approach does not work, the accuracy even decreases. The picture is slightly less clear with PDT where disambiguating the POS tags makes the approach ineffective but making them ambiguous does not make the approach more useful. Ambiguating the tags seems to affect PDT less than it affects SDT however which might indicate that PDT suffers from ambiguity even more than SDT in the original treebank. This might be due to the fact that the POS tags used in the PDT experiments are automatically predicted whereas the tags used for SDT are gold tags. This idea is further explored in Section 4.4.

We tested the same approach on the UD treebanks for Czech and Slovenian to see if they can also be affected by ambiguity in some way. In the case of UD, τ_d is the same as τ_o since the tags are already disambiguated. As can be seen from the top part of Table 6, the opposite effect is found: the transformation hurts accuracy more when the tags are ambiguous than when they are not. However, because of the similarity between copulas and auxiliaries in UD, representing them differently might make it confus-

	Orig	Transf	Δ
UD_Slovenian τ_o	80.8	79.7**	-1.1
UD_Slovenian τ_a	79.7	77.1**	-2.6
UD_Czech τ_o	77.1	76.6**	-0.5
UD_Czech τ_a	76.7	76.1**	-0.6
Without copula dependency relations			
UD_Slovenian τ_o	85.4	83.6**	-1.8
UD_Slovenian τ_a	83.6	82.2**	-1.4
UD_Czech τ_o	79.1	78.7**	-0.4
UD_Czech τ_a	78.3	78.1**	-0.2

Table 6: LAS on the original and transformed treebanks with different levels of POS tag ambiguity. Δ = Transf - Orig

ing for the parser. It would be interesting to try the approach and change the representation of copulas as well as auxiliaries. We tested something simpler: we tested the same experiment on the treebanks without copulas, i.e. we removed all sentences that have a copula dependency relations both in the training and the test sets. As can be seen from the bottom of Table 6, doing so gives the expected results: the transformation affects accuracy less when the tags are ambiguous than when they are not. The transformation still does not help parsing accuracy however.

4.4 Predicted vs gold POS tags

An issue that has been ignored so far is that in the PDT, the parser used predicted POS tags for parsing the test sets whereas in UD (and in SDT), we have been using gold POS tags. It was said in the previous section that the experiment about ambiguity on the PDT seems to indicate that tags are of poorer quality in the original experiment. It is possible that this is due to the fact that they are predicted rather than gold tags. It would be interesting to find out if the transformation approach works on UD parsing using predicted tags. This is slightly difficult to test as there does not exist taggers for all UD treebanks yet. There does exist one for Swedish however, which is why we tested this hypothesis on UD_Swedish. As can be seen from Table 7, using predicted POS tags does have an impact on the effect of the transformation as the transformation hurts parsing accuracy less than it does on data with gold POS tags. The transformation still does not help parsing accuracy however.

Overall then, the results suggest that there is something about the UD representation that makes this transformation infelicitous. It seems then that in the case of UD, it is better to keep the main verbs as heads of auxiliary dependency relations. There are other factors that may play a role in the results. For example, as appears from Table 1, the original SDT has a much higher percentage of auxiliary dependencies. This could be caused by the domain of the treebank.

POS tag	Orig	Transf	Δ
gold	76.8	75.7**	-1.1
predicted	76.4	75.6**	-0.8

Table 7: LAS on the original and transformed UD_Swedish treebank with predicted and gold POS tags. Δ = Transf - Orig

5 Conclusion and Future Work

In this paper, we have attempted to reproduce a study by Nilsson et al. (2007) that has shown that making auxiliaries heads in verb groups improves parsing but failed to show that those results port to parsing with Universal Dependencies. Contrary to expectations, the study has given evidence that main verbs should stay heads of auxiliary dependency relations for parsing with UD. The benefits of error analyses for such a study have been highlighted because they allow us to shed more light on the different ways in which the transformations affect the parsing output. Experiments suggest that gains obtained from verb group transformations in previous studies have been obtained mainly because those transformations help disambiguating between main verbs and auxiliaries. It is however still an open question why the VG transformation hurts parsing accuracy in the case of UD. It seems that the transformation makes the construction harder to learn which might be because it makes it less flat. Future work could carry out an error analysis that is more detailed than was the case in this study. Repeating those experiments with other tree transformations that have been shown to be successful in the past, such as making prepositions the head of prepositional phrases, as well as looking at other parsing models would provide more insight into the relationship between tree transformations and parsing.

Acknowledgements

We thank the three anonymous reviewers for useful feedback. We would also like to thank Jimmy Callin and Huener Kasikara for useful comments on a first version of this paper. Discussions with the participants of the course Language Technology: Research and Development in Uppsala also provided valuable help for this project. Thanks to Aaron Smith for providing the predicted POS tags data set for UD_Swedish.

References

Michael Collins. 1999. *Head-driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.

Marie-Catherine de Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D. Manning. 2014. Universal stanford dependencies: A cross-linguistic typology. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014), Reykjavik, Iceland, May 26-31, 2014.*. pages 4585–4592.

Sao Deroski, Toma Erjavec, Nina Ledinek, Petr Pajas, Zdenk abokrtsk, and Andreja ele. 2006. Towards a slovene dependency treebank. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006)*. ELRA, Genova, Italy, pages 1388–1391.

Jan Hajic, Eva Hajicova, Petr Pajas, Jarmila Panevova, and Petr Sgall. 2001. Prague dependency treebank 1.0. LDC, 2001T10.

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.

I.A. Melčuk. 1988. *Dependency Syntax: Theory and Practice*. SUNY series in Linguistics. State University of New York Press.

Jens Nilsson and Joakim Nivre. 2008. MaltEval: An evaluation and visualization tool for dependency parsing. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC)*. pages 161–166.

Jens Nilsson, Joakim Nivre, and Johan Hall. 2006. Graph transformations in data-driven dependency parsing. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA, ACL-44, pages 257–264.

Jens Nilsson, Joakim Nivre, and Johan Hall. 2007. Generalizing tree transformations for inductive dependency parsing. In *ACL 2007, Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, June 23-30, 2007, Prague, Czech Republic*.

Joakim Nivre. 2015. Towards a universal grammar for natural language processing. In *Computational Linguistics and Intelligent Text Processing - 16th International Conference, CICLing 2015, Cairo, Egypt, April 14-20, 2015, Proceedings, Part I*. pages 3–16.

Joakim Nivre, Željko Agić, Maria Jesus Aranzabe, Masayuki Asahara, Aitziber Atutxa, Miguel Ballesteros, John Bauer, Kepa Bengoetxea, Riyaz Ahmad Bhat, Cristina Bosco, Sam Bowman, Giuseppe G. A. Celano, Miriam Connor, Marie-Catherine de Marneffe, Arantza Diaz de Ilarraza, Kaja Dobrovoljc, Timothy Dozat, Tomaž Erjavec, Richárd Farkas, Jennifer Foster, Daniel Galbraith, Filip Ginter, Iakes Goenaga, Koldo Gojenola, Yoav Goldberg, Berta Gonzales, Bruno Guillaume, Jan Hajič, Dag Haug, Radu Ion, Elena Irimia, Anders Johannsen, Hiroshi Kanayama, Jenna Kanerva, Simon Krek, Veronika Laippala, Alessandro Lenci, Nikola Ljubešić, Teresa Lynn, Christopher Manning, Ctlina Mrnduc, David Mareček, Héctor Martínez Alonso, Jan Mašek, Yuji Matsumoto, Ryan McDonald, Anna Missilä, Verginica Mititelu, Yusuke Miyao, Simonetta Montemagni, Shunsuke Mori, Hanna Nurmi, Petya Osenova, Lilja Øvrelid, Elena Pascual, Marco Passarotti, Cenel-Augusto Perez, Slav Petrov, Jussi Piitulainen, Barbara Plank, Martin Popel, Prokopis Prokopidis, Sampo Pyysalo, Loganathan Ramasamy, Rudolf Rosa, Shadi Saleh, Sebastian Schuster, Wolfgang Seeker, Mojgan Seraji, Natalia Silveira, Maria Simi, Radu Simionescu, Katalin Simkó, Kiril Simov, Aaron Smith, Jan Štěpánek, Alane Suhr, Zsolt Szántó, Takaaki Tanaka, Reut Tsarfaty, Sumire Uematsu, Larraitz Uria, Viktor Varga, Veronika Vincze, Zdeněk Žabokrtský, Daniel Zeman, and Hanzhi Zhu. 2015. Universal dependencies 1.2. LINDAT/CLARIN digital library at Institute of Formal and Applied Linguistics, Charles University in Prague.

Joakim Nivre, Johan Hall, and Jens Nilsson. 2006. MaltParser: A data-driven parser-generator for dependency parsing. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC)*. pages 2216–2219.

Roy Schwartz, Omri Abend, and Ari Rappoport. 2012. Learnability-based syntactic annotation design. In *COLING*. volume 24, pages 2405–2422.

Natalia Silveira and Christopher Manning. 2015. Does Universal Dependencies need a parsing representation? An investigation of English. In *Proceedings of the Third International Conference on Dependency Linguistics (Depling 2015)*. pages 310–319.