Multi-source Cross-lingual Delexicalized Parser Transfer: Prague or Stanford?

Rudolf Rosa
Charles University in Prague
Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics
Czech Republic
rosa@ufal.mff.cuni.cz

Abstract

We compare two annotation styles, Prague dependencies and Universal Stanford Dependencies, in their adequacy for parsing. We specifically focus on comparing the adposition attachment style, used in these two formalisms, applied in multi-source cross-lingual delexicalized dependency parser transfer performed by parse tree combination. We show that in our setting, converting the adposition annotation to Stanford style in the Prague style training treebanks leads to promising results. We find that best results can be obtained by parsing the target sentences with parsers trained on treebanks using both of the adposition annotation styles in parallel, and combining all the resulting parse trees together after having converted them to the Stanford adposition style (+0.39% UAS over Prague style baseline). The score improvements are considerably more significant when using a smaller set of diverse source treebanks (up to +2.24% UAS over the baseline).

1 Introduction

Dependency treebanks are annotated in various styles, with annotations based on Prague dependencies (Böhmová et al., 2003) and (Universal) Stanford Dependencies (De Marneffe and Manning, 2008; de Marneffe et al., 2014) being the most popular and widespread.[1] In last years, several treebank collections with unified annotation have been published. The largest of them, HamleDT, currently offers 30 treebanks, semi-automatically converted both to Prague dependencies and Universal Stanford Dependencies (Zeman et al., 2012; Rosa et al., 2014), and featuring morphological annotation using Interset (Zeman, 2008). Another collection, Google Universal Treebanks, contains 11 treebanks, generally annotated from scratch using a version of Stanford Dependencies (McDonald et al., 2013) and Universal POS (Petrov et al., 2012). Recently, these efforts have joined to produce Universal Dependencies (UD), which currently contain 18 treebanks annotated with a newly defined annotation scheme based on Universal Stanford Dependencies, Universal POS tags and Interset (Agić et al., 2015). UD are now becoming the de facto standard; however, we used the HamleDT collection for our experiments, as at the time of performing the experiments, HamleDT was much larger than UD, as well as more diverse in terms of language families represented.

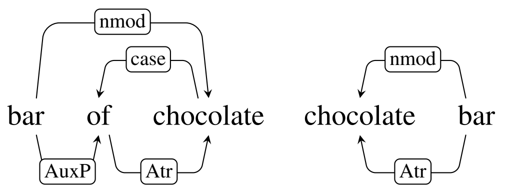

Figure 1: Stanford style (above) and Prague style (below) analysis of the phrases "bar of chocolate" and "chocolate bar". Note that in Stanford style, these phrases have a more similar structure, both featuring an *nmod* edge directly from "bar" to "chocolate". This shows the principle of constructions with a similar meaning also having a similar dependency structure.

1.1 Prague versus Stanford

One of the prominent features of Stanford style dependencies is their approach to function words. The general rule is that all function words, such

[1] We use the term *annotation style* to refer to the set of annotation conventions, as applied in annotating a given treebank, typically also defined by an annotation manual.

as adpositions[2] or conjunctions, are attached as leaf nodes. This is a result of a standpoint which favours direct dependency relations between lexical nodes, not mediated by function words. This also makes dependency structures more similar cross-lingually, as it is very common that the same function is expressed by an adposition in one language, but by other means, such as morphology or word order, in another language – or even within the same language, as shown in Figure 1. On the other hand, Prague style dependencies annotate adpositions as heads of adpositional groups.[3]

While Stanford style trees may be more useful for further processing in NLP applications, it has been argued that Prague style trees are easier to obtain by using statistical parsers. Among other differences, adpositions provide important cues to the parser for adpositional group attachment, which is one of the most notorious parsing problems. This information becomes harder to access when the adpositions are annotated as leafs. The issue of dependency representation learnability has been studied by several authors, generally reaching similar conclusions (Schwartz et al., 2012; Søgaard, 2013; Ivanova et al., 2013). The approach suggested by de Marneffe et al. (2014) is to use a different annotation style for parsing, with Prague style adposition annotation, among other, and to convert the dependency trees to full Stanford style only after parsing for subsequent applications.

Still, while the aforementioned observations seem to hold in the general case, in multilingual parsing scenarios, the higher cross-lingual similarity of Stanford style dependency trees may be of benefit. From all of the differences between Prague and Stanford, the adposition attachment seems to be the most interesting, as adpositions are usually very frequent and diverse in languages, as well as very important in parsing. Therefore, in this work, we evaluate the influence of adposition annotation style in cross-lingual multi-source delexicalized parser transfer.

[2]Adposition is a general term for prepositions, postpositions and circumpositions.

[3]The lexical nodes are only directly connected in Prague tectogrammatical (deep-syntax) dependency trees, where function words are removed and their functions are captured via node attributes. It is worth noting that in general, there is little difference between representing information by means of node attributes or leaf nodes; thus, Stanford trees and Prague tectogrammatical trees are actually very similar in structure.

1.2 Delexicalized parser transfer

In the approach of single-source delexicalized dependency parser transfer (Zeman and Resnik, 2008), we train a parser on a treebank for a resource-rich *source language*, using non-lexical features, most notably part-of-speech (POS) tags, but not using word forms or lemmas. Then, we apply that parser to a POS-tagged corpus of an under-resourced *target language*, to obtain a dependency parse tree. Delexicalized transfer typically yields worse results than a fully supervised lexicalized parser, trained on a treebank for the target language. However, for a vast majority of languages, there are no manually devised treebanks, in which case it may be useful to obtain at least a lower-quality parse tree for tasks such as information retrieval or machine translation. Still, in this work, we do not apply delexicalized parser transfer to under-resourced languages, since there is no easy way of evaluating such experiments. Rather, we follow the usual way of using target languages for which there is a treebank available and thus the experiments can be easily evaluated, but we do not use the target treebank for training, thus simulating the under-resourcedness of the target language.

In *multi-source* delexicalized parser transfer, multiple source treebanks are used for training. McDonald et al. (2011) used simple treebank concatenation, thus obtaining one multilingual source treebank, and trained a multilingual delexicalized parser. In our work, we extend the method of Sagae and Lavie (2006), originally suggested for (monolingual) parser combination. In this approach, several independent parsers are applied to the same input sentence, and the parse trees they produce are combined into one resulting tree. The combination is performed using the idea of McDonald et al. (2005a), who formulated the problem of finding a parse tree as a problem of finding the maximum spanning tree (MST) of a weighted directed graph of potential parse tree edges. In the tree combination method, the weight of each edge is defined as the number of parsers which include that edge in their output (it can thus also be regarded as a parser voting approach). To find the MST, one can use e.g. the Chu-Liu-Edmonds algorithm (Chu and Liu, 1965; Edmonds, 1967), which was used by McDonald et al. (2005b) for non-projective parsing, and which we use in our work. The tree combination method can be easily ported from a monolingual to a multilingual setting, where the individual parsers are trained over different languages.

A possible research path which we do not follow in this work is the choice or weighting of the source languages according to their similarity to the target language, which has been successfully employed by several authors (Naseem et al., 2012; Søgaard and Wulff, 2012; Täckström et al., 2013; Rosa and Žabokrtský, 2015). This may have similar effect to our annotation style conversions, or it may be that these two approaches will behave rather orthogonally, as they might target different interlingual differences. Also, selection of a source language similar to the target may weaken the need for increasing annotation similarity, but this approach may still be useful for targets very dissimilar to the available sources. We believe these to be interesting questions that deserve further research.

1.3 This work

In this work, we use the HamleDT 2.0 collection and the MSTParser (McDonald et al., 2005b) to evaluate the potential benefit of employing Stanford style adposition attachment instead of the Prague style in parsing. We first show that in a monolingual setting, Prague style adposition annotation performs better than the Stanford style, both for lexicalized and delexicalized parsing. We also show that fully Stanfordized dependency trees perform even worse, but we further focus on adposition attachment only; the other annotation differences are of less interest for us, as they concern less frequent phenomena and/or do not seem so promising for cross-lingual experiments. We then perform extensive delexicalized parser transfer experiments, both using the full HamleDT collection as source treebanks (in a leave-one-out fashion), as well as using various smaller subsets consisting of languages with different adpositional characteristics. We also investigate a number of setups for parsing and combining the dependency trees with conversions between Prague style and Stanford style in between.

We conclude that the Stanford style of adposition attachment seems to be beneficial in multi-source cross-lingual delexicalized dependency parser transfer. Overall, best results are obtained by training parsers on source treebanks both in Prague and Stanford style, parsing the target text by all of the parsers, converting the Prague style parser outputs into Stanford style, and combining all of the parse trees. This approach achieves an average improvement of +0.39% UAS absolute over using Prague style only. When the set of source treebanks is small and the languages differ a lot in terms of adpositions, the improvements are even larger, up to +2.24% UAS absolute over the Prague style baseline.

2 Method

2.1 Dependency parser

Throughout this work, we use MSTperl (Rosa, 2015b), an implementation of the MSTParser of McDonald et al. (2005b), with first-order features and non-projective parsing. The parser is a single-best one, returning exactly one parse tree for each input sentence. It is trained using 3 iterations of MIRA (Crammer and Singer, 2003). The parser performs unlabelled parsing, returning only the dependency tree, with no dependency relation labels. We only evaluate unlabelled parsing in this work.

Our delexicalized feature set is based on (McDonald et al., 2005a), with lexical features removed, and consists of various conjunctions of the following features:

POS tags We use the coarse 12-value Universal POS Tagset (UPT) of Petrov et al. (2012).[4] For an edge, we use information about the POS tag of the head, dependent, their neighbours, and all of the nodes between them.

Token distance We use signed distance of head and dependent ($order_{head} - order_{dependent}$), bucketed into the following buckets:
$+1;\ +2;\ +3;\ +4;\ \geq{+5};\ \geq{+11};$
$-1;\ -2;\ -3;\ -4;\ \leq{-5};\ \leq{-11}.$

We use exactly the same settings of the parser in all experiments. For lexicalized parsing, we also include the word form and word lemma of the head and dependent node, in various conjunctions with the POS tags and token distance as well as with each other. The configuration files that contain the feature sets and other settings, as well as the scripts we used to conduct our experiments, are available in (Rosa, 2015a).

[4]These 12 values are: NOUN, VERB, PUNCT, ADJ, ADP, PRON, CONJ, ADV, PRT, NUM, DET, X.

	Language	Size (kTokens) Train	Test
ar	Arabic	250	28
bg	Bulgarian	191	6
bn	Bengali	7	1
ca	Catalan	391	54
cs	Czech	1,331	174
da	Danish	95	6
de	German	649	33
el	Greek	66	5
en	English	447	6
es	Spanish	428	51
et	Estonian	9	1
eu	Basque	138	15
fa	Persian	183	7
fi	Finnish	54	6
grc	Ancient Greek	304	6
hi	Hindi	269	27
hu	Hungarian	132	8
it	Italian	72	6
ja	Japanese	152	6
la	Latin	49	5
nl	Dutch	196	6
pt	Portuguese	207	6
ro	Romanian	34	3
ru	Russian	495	4
sk	Slovak	816	86
sl	Slovenian	29	7
sv	Swedish	192	6
ta	Tamil	8	2
te	Telugu	6	1
tr	Turkish	66	5

Table 1: List of HamleDT 2.0 treebanks.

Please note that our conclusions are only valid for the MSTperl parser, and may not hold e.g. for higher order graph based parsers or transition based parsers. In this work, we decided to focus on breadth of evaluated parsing and combination setups; we intend to evaluate a wider range of parsers in future.

2.2 Dataset and its conversions

We use the HamleDT 2.0 collection of 30 dependency treebanks, which had been semi-automatically harmonized to Prague dependencies and then Stanfordized into Universal Stanford Dependencies. We list the treebanks and their sizes in Table 1. More information about the treebanks contained in the dataset, as well as the dataset itself, can be obtained online.[5]

In most experiments, we use the Prague style version of HamleDT, as the Stanford version performs much worse for parsing (see Section 3.1). Instead of using the full Stanford version, we only focus on one of its prominent features – adposition attachment. Thus, we alternate between Prague adposition attachment as head (denoted "P"), and Stanford adposition attachment as leaf node (denoted "S"), using simple conversion scripts.

[5]`https://ufal.mff.cuni.cz/hamledt`

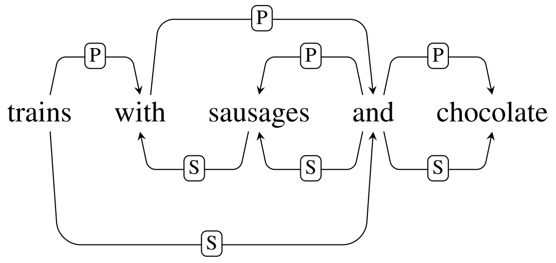

Figure 2: Original Prague style adposition analysis (above), and Stanford style adposition analysis as produced by the conversion (below). Note that the coordination stays in the Prague style. Edge labels are not shown as we do not use them in this work.

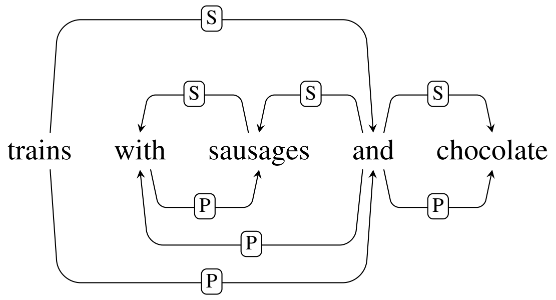

Figure 3: Stanford style adposition analysis (above), and Prague style adposition analysis as produced by the conversion (below). Together with Figure 2, this shows a case where our conversion is imperfect, as we are unable to obtain the original structure after the conversion roundtrip.

- The conversion from P to S takes each adposition and attaches it as a dependent of its left-most non-adpositional child, together with all of its other non-adpositional children. Thus, the adposition becomes a leaf node, unless it has adpositional dependent nodes (typically this signifies a compound adposition). Coordinating conjunctions are passed through (recursively) – if the left-most non-adpositional child is a coordinating conjunction, then its dependent leftmost non-adpositional conjunct is used instead as the new head of the adposition (see Figure 2).

- In the conversion from S to P, each adposition with a non-adpositional head is attached as a dependent of its head's head, and its original head is attached as its dependent (see Figure 3).

The roundtrip of the conversion (UAS after converting from P to S and back) is around 98% in total, and around 94% for adposition nodes alone.

2.3 Parse tree combination

The inference component of the MSTParser is also applied to perform the combination of parse trees obtained for a sentence from different parser instances. In that setting, each potential dependency edge is assigned a score equal to the number of input parse trees in which it is contained. The MST algorithm then finds and returns a dependency tree in which the edges are confirmed by the highest possible number of input trees.

The general experiment setup is as follows. One of the 30 treebanks is taken as the target treebank, and the remaining 29 treebanks become source treebanks. Then, delexicalized parsers are trained on the source treebanks, resulting in 29 trained parser models. Next, each of the parsers is applied to each sentence in the test section of the target treebank. And finally, the obtained parse trees for each sentence are combined together as has been described above, and the resulting dependency tree is evaluated using the target treebank.

Note that there are three places where a conversion from one annotation style to another may take place – conversion of the source treebank before training a parser, conversion of the parser output before the parse tree combination, and conversion of the parse tree combination output. We will denote the setups using the pattern“X/Y/Z”, where “X” denotes the annotation style used for parser training and parsing, “Y” is the style into which the parser outputs are converted before being combined, and “Z” is the style into which the result of the combination is converted. Furthermore, “P,S/Y/Z” will refer to parsing both with “P” style parsers and “S” style parsers, thus resulting in 58 trees for each sentence to combine, rather than 29. In many setups, there is no conversion after the combination, or there is even no combination performed (in monolingual setups); therefore, we will often omit the last part of the pattern, using only “X/Y”.

3 Experiments and Evaluation

We use the training sections of the treebanks for parser training and their testing sections for evaluation. We report the results using UAS (unlabelled attachment score).

Setup	Lex	Delex	Transfer
Prague	80.54	74.12	56.68
Stanford full	76.47	69.53	48.91
Prague non-punct	80.23	74.00	56.08
Stanford full non-punct	76.84	70.66	50.15

Table 2: Prague versus full Stanford annotation style, UAS averaged over 30 target languages.
The *Lex*icalized and *Delex*icalized parsers are monolingual. The *Transfer* parser is a combination of 29 sources parsers applied to the remaining target language.

3.1 Full Universal Stanford Dependencies

As a preliminary experiment, we compared the Prague version with its fully Stanfordized version. The results are shown in Table 2. It can be seen that the Stanford version performs much worse than the Prague one – its results are lower by around 5% UAS absolute.

Closer inspection showed that many of the errors are actually due to sentence-final punctuation attachment. In Stanford style, sentence-final punctuation is to be attached as a dependent node of the root node of the sentence (typically the main predicate). However, this is difficult for the first-order parser, as it has no knowledge of the root node when scoring the potential edges, and thus the punctuation gets often attached to some other verb. In Prague style, the sentence-final punctuation is attached to the technical root node, which is marked by special values of the node features, and thus the assignment is very easy to make. While this is an important point to keep in mind when parsing into full Stanford style, it is of little relevance to the goal of this paper – punctuation attachment is rarely important in NLP applications, and is not very likely to significantly contribute to cross-lingual dependency structure similarity either. For this reason, we also include UAS measured only on non-punctuation nodes. Still, adposition attachment, which we are mostly interested in, accounts for only a part of the score difference.

3.2 Prague versus Stanford adpositions

Further on, we only use the Prague style annotation of the treebanks, with adpositions annotated either in Prague style (P) or Stanford style (S).

3.2.1 Supervised parsers

We first evaluate supervised monolingual lexicalized and delexicalized parsers, alternating between the P and S annotation styles of adpositions. The results in Table 3 show that in the lexicalized setting, the UAS of the P style parser is +0.77%

Setup	Lexicalized	Delexicalized
P/P	**80.54**	**74.12**
S/P	78.44	72.65
S/S	79.77	73.91
P/S	80.23	73.94

Table 3: Average UAS of supervised monolingual parsers, both lexicalized and delexicalized.

Setup	P/P			S/S	
	UAS	>S/S	?	>P/P	UAS
Lexicalized	80.54	13	16	1	79.77
Delexicalized	74.12	11	16	3	73.91

Table 4: Pairs of supervised parser setups.
"UAS" = Average UAS as in Table 3
">S/S", ">P/P" = Number of languages for which the setup performed significantly better
"?" = Number of languages for which neither setup performed significantly better

above the S style parser, and Table 4 confirms that the P parser is significantly better than the S parser for nearly half of the languages.[6] Actually, to obtain S style parse trees, it is better to parse the text using a parser trained on a P style treebank, and then convert the output parse trees (this yields a +0.46% higher UAS than parsing directly using an S style parser). Here, the adpositions clearly provide important information to the parser, and their annotation as heads benefits the results.

In the delexicalized setting, the P style parser scores higher than the S style one only by a small margin (+0.21% UAS), although still being significantly better for a third of the languages. Moreover, parsing directly using the S style is now comparable to parsing using P style and then converting to S style. This suggests that the most important piece of information for correctly attaching an adposition is its lemma, and delexicalizing a parser thus reduces the advantage of P style annotation for correct adposition attachment.

3.2.2 29-to-1 delexicalized parser transfer

We now move on to the main focus of our work, evaluating the effect of adposition annotation style in multilingual transfer of 29 delexicalized source parsers to a target language using parse tree combination.

Table 5 shows that using either P or S for everything leads to comparable results, with the S style now achieving a slightly better score (+0.20% UAS absolute on average). The results tend to get worse when additional conversions are performed; we thus omit such setups from further evaluation. Interestingly, slight improvements can be obtained by applying both P parsers and S parsers and combining them after conversion of the resulting trees to the S style, achieving a total average increase of +0.39 UAS absolute over the P style baseline.

[6]We used McNemar's test with significance level 5%.

P style output		S style output	
Setup	UAS	Setup	UAS
P/P/P	56.68	S/S/S	56.88
P/S/P	55.43	S/P/S	56.31
S/S/P	55.51	P/P/S	56.48
S/P/P	55.84	P/S/S	56.80
P,S/P/P	**56.81**	P,S/S/S	**57.07**
P,S/S/P	55.71	P,S/P/S	56.67

Table 5: Average UAS of various setups of delexicalized parser transfer, always using 1 language as target and the remaining 29 languages as source.

Setup A					Setup B	
	UAS	>B	?	>A	UAS	
S/S	56.88	8	13	9	56.68	P/P
P,S/P	56.81	5	22	3	56.68	P/P
P,S/S	57.07	9	19	2	56.88	S/S
P,S/S	57.07	8	17	5	56.81	P,S/P
P,S/S	57.07	7	20	3	56.68	P/P

Table 7: Pairs of delexicalized transfer setups.
"UAS" = Average UAS as in Table 5
">B", ">A" = Number of target languages for which the setup performed significantly better
"?" = Number of target languages for which neither setup performed significantly better

Table 6 shows detailed results of the better-performing transfer setups for all target languages, together with the results of the supervised monolingual methods. Table 7 compares several pairs of the transfer setups by reporting the number of target languages (out of the total 30) for which one setup was significantly better than the other setup.

We can now see that the improvements obtained by employing S style parsers are not only low, but also usually statistically insignificant – the highest scoring P,S/S setup is significantly better than the baseline P/P setup only for 7 target languages, while also being significantly worse for other 3 target languages. Still, we believe that the sole fact that in this setting, employing the S style annotation leads to comparable or slightly better results (which is not true for the supervised monolingual parsers) indicates a potential benefit of the S style annotation in a cross-lingual setting, presumably due to the increased similarity of the dependency structures across languages.

Tgt	Lexicalized supervised			Delexicalized supervised			Delexicalized transfer			
lang	P/P	S/S	P/S	P/P	S/S	P/S	P/P	P,S/P	S/S	P,S/S
ar	**77.47**	76.32	77.17	**69.61**	69.29	69.50	44.61	**44.99**	43.16	44.13
bg	**87.95**	87.50	87.61	**83.87**	82.76	83.32	**73.17**	72.72	72.24	72.65
bn	82.27	**82.39**	82.27	77.59	**78.82**	77.59	59.98	60.34	**60.47**	60.22
ca	**86.11**	84.37	85.49	**79.71**	79.03	79.33	**66.45**	66.38	65.61	66.07
cs	**80.87**	80.31	80.63	**70.99**	70.69	70.69	64.06	**64.14**	63.62	63.93
da	**85.66**	84.42	85.12	**81.13**	80.31	80.67	**63.74**	63.53	62.82	63.09
de	**84.65**	83.57	84.53	**77.52**	76.92	77.47	52.58	55.17	**55.95**	55.32
el	**80.68**	80.20	80.18	**75.40**	75.15	74.73	67.05	67.69	67.63	**67.78**
en	**84.71**	84.37	84.05	**76.57**	76.19	76.03	46.13	**48.23**	47.65	47.09
es	**85.46**	83.55	84.74	**79.75**	78.52	79.25	**69.73**	69.61	68.85	69.17
et	85.15	**86.30**	85.46	80.96	**82.85**	80.75	71.34	72.07	74.06	**74.48**
eu	**75.28**	75.07	**75.28**	68.34	**68.41**	68.34	46.12	45.92	**46.15**	46.07
fa	**82.27**	80.21	81.70	70.44	**71.72**	70.78	54.69	54.77	56.41	**56.69**
fi	71.17	70.80	**71.21**	63.10	62.51	**63.13**	**51.48**	51.17	50.60	51.08
grc	**56.98**	56.61	56.56	48.92	**49.10**	48.80	46.24	46.38	46.48	**46.50**
hi	90.40	86.43	**90.42**	**80.55**	80.52	80.52	30.12	29.64	33.23	**33.64**
hu	**77.60**	77.07	77.40	**72.54**	71.79	72.34	59.68	59.89	60.50	**60.81**
it	**81.46**	80.57	81.22	**77.49**	76.57	76.92	64.52	**65.13**	64.44	64.50
ja	**91.17**	89.65	90.79	81.72	84.03	**84.35**	44.23	42.64	44.02	**44.88**
la	47.55	**48.72**	47.36	44.08	**44.12**	43.81	41.14	41.28	41.34	**41.47**
nl	**80.90**	80.05	80.11	**74.02**	73.70	73.57	62.47	62.04	**63.81**	63.80
pt	**83.50**	82.21	82.97	**80.14**	78.68	79.77	71.35	**71.60**	71.14	71.26
ro	**89.62**	88.79	**89.62**	85.19	**85.34**	84.85	59.66	**59.85**	58.52	58.67
ru	**83.98**	83.49	83.75	**73.08**	72.70	72.90	**63.82**	63.65	62.43	63.13
sk	**79.02**	78.70	78.63	**71.38**	70.88	70.93	63.66	**63.73**	63.36	63.62
sl	**81.19**	80.94	80.95	72.91	**72.93**	72.69	**54.40**	53.68	53.80	53.68
sv	**83.20**	81.93	82.48	**78.84**	77.97	78.18	62.08	62.18	**62.22**	61.60
ta	**72.70**	72.60	72.30	**68.17**	67.92	67.62	38.76	**39.01**	37.66	38.91
te	**87.60**	86.93	**87.60**	**85.59**	84.09	85.59	66.83	66.16	**67.00**	66.50
tr	**79.48**	79.02	79.26	**73.99**	73.72	73.72	40.39	40.82	**41.28**	41.26
Avg	**80.54**	79.77	80.23	**74.12**	73.91	73.94	56.68	56.81	56.88	**57.07**

Table 6: UAS of supervised lexicalized monolingual parsers, supervised delexicalized monolingual parsers, and delexicalized transfer parsers.

Subset	ADP freq.	Language
High	15%	Spanish
	19%	Hindi
	19%	Japanese
Med	9%	Czech
	8%	English
	9%	Swedish
Low	0%	Basque
	4%	Ancient Greek
	1%	Hungarian
Mix	15%	Spanish
	9%	Swedish
	1%	Hungarian

Table 8: Subsets of source treebanks, selected according to their frequency of adposition tokens.

Setup	High	Med	Low	Mix	All9
P/P	40.53	52.00	44.53	41.03	54.98
P,S/P	41.29	52.57	45.00	41.75	55.37
S/S	41.36	51.64	43.69	41.95	54.85
P,S/S	**42.77**	**52.67**	**46.41**	**42.66**	**55.42**

Table 9: UAS of delexicalized parser transfer, averaged over 21 target languages, with the specified subset treebanks as sources.

3.2.3 Smaller source treebank subsets

For a deeper insight and further confirmation of our findings, we also performed a set of experiments with smaller 3-member subsets of the treebank collection. We selected several treebank groups, based on the ratio of adposition tokens to all tokens. We also only chose large enough treebanks (more than 100,000 tokens). The subsets are listed in Table 8; we also used a larger "All9" set of all the 9 selected treebanks. Only these were then used for training; the remaining 21 languages were used for testing as target languages.

The summary results are to be found in Table 9; the statistical significance of the setups is evaluated in Table 10. For these datasets, the advantage of employing the S style in combination with P style becomes clearly visible, frequently leading to significantly better results than when using only the P style (however, using only S style parsers performs rather poorly). Moreover, converting the parse trees to S style before combining them is also often significantly better than converting them to P style. The improvements are large especially for the High, Low and Mix datasets. This suggests that the role of Stanford style is stronger with small and highly diverse datasets, where its benefit of making the dependency trees more similar becomes rather important.[7] For the High dataset,

Source	Setup A			Setup B	
subset	UAS	>B	?	>A	UAS
	S/S			P/P	
High	41.36	12	8	1	40.53
Med	51.64	1	16	4	52.00
Low	43.69	2	8	11	44.53
Mix	41.95	9	10	2	41.03
All9	54.85	3	12	6	54.98
	P,S/P			P/P	
High	41.29	9	12	0	40.53
Med	52.57	8	13	0	52.00
Low	45.00	6	14	1	44.53
Mix	41.75	8	13	0	41.03
All9	55.37	6	14	1	54.98
	P,S/S			S/S	
High	42.77	15	6	0	41.36
Med	52.67	15	6	0	51.64
Low	46.41	15	5	1	43.69
Mix	42.66	11	9	1	41.95
All9	55.42	9	12	0	54.85
	P,S/S			P,S/P	
High	42.77	15	6	0	41.29
Med	52.67	4	11	6	52.57
Low	46.41	15	6	0	45.00
Mix	42.66	9	10	2	41.75
All9	55.42	3	12	6	55.37
	P,S/S			P/P	
High	42.77	19	2	0	40.53
Med	52.67	5	16	0	52.00
Low	46.41	15	6	0	44.53
Mix	42.66	13	8	0	41.03
All9	55.42	7	11	3	54.98

Table 10: Pairs of delexicalized transfer setups using specific source treeebank subsets.
"UAS" = Average UAS as in Table 9
">B", ">A" = Number of target languages for which the setup performed significantly better
"?" = Number of target languages for which neither setup performed significantly better

the best result surpasses the Prague-only baseline by as much as +2.24% UAS absolute on average, yielding a significantly better result for 19 of the 21 target languages.

4 Conclusion

In this work, we investigated the usefulness of Stanford adposition attachment style as an alternative to the Prague style in dependency parsing, using a large set of 30 treebanks for evaluation. We especially focused on multi-source cross-lingual delexicalized parser transfer, as one of the targets behind the design of Universal Stanford Dependencies is to be more cross-lingually consistent

[7]Of course, this is only a speculation, as there are many other properties of the source treebank subsets which we were unable to factor out that may influence the results – for example, the High and Low subsets contain genealogically highly varied languages, but we were unable to find such a varied subset among the languages with a medium frequency of adpositions.

than other annotation styles.

We managed to confirm that for supervised parsing, Prague annotation style is favourable over Stanford style, as has been already stated in literature. However, in the parser transfer setting, Stanford style adposition attachment tends to perform better than the Prague style, presumably thanks to its abstraction from the high interlingual variance in adposition usage. Best results are achieved by at once combining outputs of parsers trained on treebanks of both Prague and Stanford adposition attachment style, reaching an average improvement of +0.39% UAS absolute over the Prague style baseline. Our results are further confirmed by experiments using smaller and more diverse subsets of training treebanks, where the advantage of combining Prague and Stanford adposition annotation style becomes even more pronounced, reaching average improvements of up to +2.24% UAS absolute over the Prague style baseline.

We used the first-order non-projective MST-Parser in all experiments; therefore, our conclusions are valid only for that parser. The next logical research step is thus to apply other parsers in a similar setting to determine whether our findings can be further generalized, or whether using a different parser leads to different effects when comparing and combining Prague and Stanford adposition annotation styles.

Acknowledgments

This research was supported by the grants GAUK 1572314, SVV 260 224, and FP7-ICT-2013-10-610516 (QTLeap). This work has been using language resources developed, stored and distributed by the LINDAT/CLARIN project of the Ministry of Education, Youth and Sports of the Czech Republic (project LM2010013).

References

Željko Agić, Maria Jesus Aranzabe, Aitziber Atutxa, Cristina Bosco, Jinho Choi, Marie-Catherine de Marneffe, Timothy Dozat, Richárd Farkas, Jennifer Foster, Filip Ginter, Iakes Goenaga, Koldo Gojenola, Yoav Goldberg, Jan Hajič, Anders Trærup Johannsen, Jenna Kanerva, Juha Kuokkala, Veronika Laippala, Alessandro Lenci, Krister Lindén, Nikola Ljubešić, Teresa Lynn, Christopher Manning, Héctor Alonso Martínez, Ryan McDonald, Anna Missilä, Simonetta Montemagni, Joakim Nivre, Hanna Nurmi, Petya Osenova, Slav Petrov, Jussi Piitulainen, Barbara Plank, Prokopis Prokopidis, Sampo Pyysalo, Wolfgang Seeker, Mojgan Seraji, Natalia Silveira, Maria Simi, Kiril Simov, Aaron Smith, Reut Tsarfaty, Veronika Vincze, and Daniel Zeman. 2015. Universal dependencies 1.1. `http://hdl.handle.net/11234/LRT-1478`.

Alena Böhmová, Jan Hajič, Eva Hajičová, and Barbora Hladká. 2003. The Prague dependency treebank. In *Treebanks*, pages 103–127. Springer.

Yoeng-Jin Chu and Tseng-Hong Liu. 1965. On shortest arborescence of a directed graph. *Scientia Sinica*, 14(10):1396.

Koby Crammer and Yoram Singer. 2003. Ultraconservative online algorithms for multiclass problems. *The Journal of Machine Learning Research*, 3:951–991.

Marie-Catherine De Marneffe and Christopher D Manning. 2008. The Stanford typed dependencies representation. In *Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation*, pages 1–8. Association for Computational Linguistics.

Marie-Catherine de Marneffe, Natalia Silveira, Timothy Dozat, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D. Manning. 2014. Universal Stanford dependencies: A cross-linguistic typology. In *Proc. of LREC'14*, Reykjavk, Iceland. European Language Resources Association (ELRA).

Jack Edmonds. 1967. Optimum branchings. *Journal of Research of the National Bureau of Standards B*, 71(4):233–240.

Angelina Ivanova, Stephan Oepen, and Lilja Øvrelid. 2013. Survey on parsing three dependency representations for English. In *ACL (Student Research Workshop)*, pages 31–37.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005a. Online large-margin training of dependency parsers. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 91–98. Association for Computational Linguistics.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005b. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 523–530. Association for Computational Linguistics.

Ryan McDonald, Slav Petrov, and Keith Hall. 2011. Multi-source transfer of delexicalized dependency parsers. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 62–72, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. 2013. Universal dependency annotation for multilingual parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 92–97.

Tahira Naseem, Regina Barzilay, and Amir Globerson. 2012. Selective sharing for multilingual dependency parsing. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 629–637, Stroudsburg, PA, USA. Association for Computational Linguistics.

Slav Petrov, Dipanjan Das, and Ryan McDonald. 2012. A universal part-of-speech tagset. In *Proc. of LREC-2012*, pages 2089–2096, Istanbul, Turkey. European Language Resources Association (ELRA).

Rudolf Rosa and Zdeněk Žabokrtský. 2015. KL_{cpos^3} – a language similarity measure for delexicalized parser transfer. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Short Papers*, Stroudsburg, PA, USA. Association for Computational Linguistics, Association for Computational Linguistics.

Rudolf Rosa, Jan Mašek, David Mareček, Martin Popel, Daniel Zeman, and Zdeněk Žabokrtský. 2014. HamleDT 2.0: Thirty dependency treebanks Stanfordized. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Hrafn Loftsson, Bente Maegaard, and Joseph Mariani, editors, *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC 2014)*, pages 2334–2341, Reykjavík, Iceland. European Language Resources Association.

Rudolf Rosa. 2015a. MSTperl delexicalized parser transfer scripts and configuration files. `http://hdl.handle.net/11234/1-1485`.

Rudolf Rosa. 2015b. MSTperl parser (2015-05-19). `http://hdl.handle.net/11234/1-1480`.

Kenji Sagae and Alon Lavie. 2006. Parser combination by reparsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 129–132. Association for Computational Linguistics.

Roy Schwartz, Omri Abend, and Ari Rappoport. 2012. Learnability-based syntactic annotation design. In *Proceedings of COLING 2012: Technical Papers*.

Anders Søgaard and Julie Wulff. 2012. An empirical etudy of non-lexical extensions to delexicalized transfer. In *COLING (Posters)*, pages 1181–1190.

Anders Søgaard. 2013. An empirical study of differences between conversion schemes and annotation guidelines. In *Proceedings of the Second International Conference on Dependency Linguistics (DepLing 2013), Prague, Czech Republic: Charles University in Prague, Matfyzpress*, pages 298–307.

Oscar Täckström, Ryan McDonald, and Joakim Nivre. 2013. Target language adaptation of discriminative transfer parsers. *NAACL HLT 2013*, pages 1061–1071.

Daniel Zeman and Philip Resnik. 2008. Cross-language parser adaptation between related languages. In *IJCNLP 2008 Workshop on NLP for Less Privileged Languages*, pages 35–42, Hyderabad, India. Asian Federation of Natural Language Processing, International Institute of Information Technology.

Daniel Zeman, David Mareček, Martin Popel, Loganathan Ramasamy, Jan Štěpánek, Zdeněk Žabokrtský, and Jan Hajič. 2012. HamleDT: To parse or not to parse? In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Mehmet Uur Doan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, May. European Language Resources Association (ELRA).

Daniel Zeman. 2008. Reusable tagset conversion using tagset drivers. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC 2008)*, pages 213–218, Marrakech, Morocco. European Language Resources Association.