Comparing Learnability of Two Dependency Schemes: 'Semantic' (UD) and 'Syntactic' (SUD)

Ryszard Tuora
Institute of Computer Science, Polish Academy of Sciences
ryszardtuora@gmail.com

Adam Przepiórkowski
Institute of Computer Science, Polish Academy of Sciences
Faculty of Philosophy, University of Warsaw
Centre for Linguistics & Philology, University of Oxford
adamp@ipipan.waw.pl

Aleksander Leczkowski
Faculty of Psychology, University of Warsaw
a.leczkowski@student.uw.edu.pl

Abstract

This paper contributes to the thread of research on the learnability of different dependency annotation schemes: one ('semantic') favouring content words as heads of dependency relations and the other ('syntactic') favouring syntactic heads. Several studies have lent support to the idea that choosing syntactic criteria for assigning heads in dependency trees improves the performance of dependency parsers. This may be explained by postulating that syntactic approaches are generally more learnable. In this study, we test this hypothesis by comparing the performance of five parsing systems (both transition- and graph-based) on a selection of 21 treebanks, each in a 'semantic' variant, represented by standard UD (Universal Dependencies), and a 'syntactic' variant, represented by SUD (Surface-syntactic Universal Dependencies): unlike previously reported experiments, which considered learnability of 'semantic' and 'syntactic' annotations of particular constructions *in vitro*, the experiments reported here consider whole annotation schemes *in vivo*. Additionally, we compare these annotation schemes using a range of quantitative syntactic properties, which may also reflect their learnability. The results of the experiments show that SUD tends to be more learnable than UD, but the advantage of one or the other scheme depends on the parser and the corpus in question.

1 Introduction and Background

This paper compares the learnability of two approaches to dependency annotation. One, represented by Universal Dependencies (UD; http://universaldependencies.org/; Nivre et al., 2016), favours content words over function words as dependency heads, as this increases cross-linguistic uniformity of the resulting scheme; here we will call this approach 'semantic'.[1] Another, represented by Surface-Syntactic Universal Dependencies (SUD; https://surfacesyntacticud.github.io; Gerdes et al., 2018, 2019), uses purely syntactic criteria for determining headedness; hence the moniker 'syntactic'. The SUD scheme was designed as minimally different from – 'near-isomorphic to' – UD, and many UD treebanks have been converted to SUD, so differences in learnability between the two approaches should be relatively easy to assess and interpret. As is clear from Figure 1, which juxtaposes the UD basic tree (at the top) and the SUD tree (at the bottom), SUD generally adopts the principle that function words such as auxiliaries (e.g., *do*), subordinating conjunctions (*until*), copula (*'re*), and prepositions (*with*) are heads of relevant constructions. On the other hand, SUD representation of coordination is similar to that of UD, but where all non-initial conjuncts are attached to the head of the first one in UD, each conjunct is attached to the head of the previous conjunct in SUD; when there are just two conjuncts, annotations are the same.

Previous results suggest that the syntactic scheme should be more learnable. For example, Schwartz et al. (2012) compared six constructions, four of which are coded differently in UD

[1] It needs to be emphasised that the epithet 'semantic' is used here in this very technical sense, indicating preference for content words as heads, and does not in any way imply that UD is a semantic annotation scheme.

Findings of the Association for Computational Linguistics: EMNLP 2021, pages 2987–2996
November 7–11, 2021.

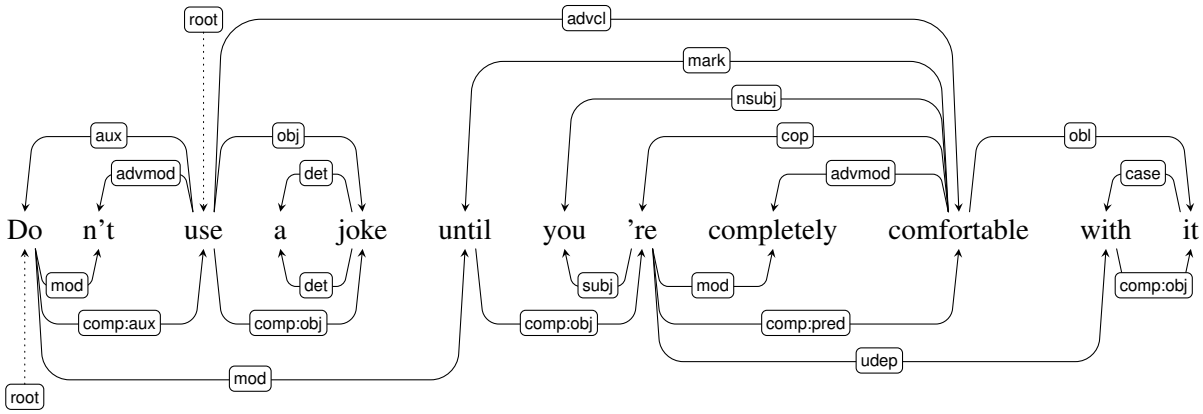

Figure 1: UD (top) and SUD (bottom) analyses of a sentence from the English GUM corpus (Zeldes, 2017)

and SUD:[2] preposition–noun (e.g., *of Rome*) – a class which also includes complementiser–clause constructions (e.g., *after you go*), *to*–infinitival (e.g., *to eat*), modal–verb (e.g., *can come*), and coordination. The experiments involved five different parsers (representing both transition-based and graph-based methodologies) and two different learnability measures (including one based on attachment scores). The results of these experiments favour SUD-like representations in all four cases. In the case of constructions involving a preposition or a complementiser, having them as heads – as in SUD, but unlike in UD – results in extremely strong ('unanimous') learnability improvements. The effect is weaker in the case of verb groups containing a modal and still weaker in the case of infinitivals introduced by *to*, but in both cases having the main lexical verb as the dependent – as in SUD, but unlike in UD – gives generally better results.

A similar range of constructions is inspected in Silveira and Manning (2015). For each kind of construction, 3 different variants of conversion from semantic to syntactic headedness are considered, depending on how many of the dependents of the semantic head are moved to the syntactic head. The best variant gives significant improvements in the learnability of the syntactic scheme in the case of preposition–noun (but not complementiser–clause), auxiliary–verb (rather than the more general modal–verb, considered in Schwartz et al., 2012) and – and this is were the improvement was most clear – in the case of copula–predicate constructions. Other papers that report better learnability of a more syntactic scheme converted automatically from a more semantic scheme include: Nilsson et al. (2006, 2007) (auxiliary–verb constructions in Arabic, Czech, Dutch and Slovene, small improvement observed in the case of the transition based MaltParser, but not with the graph-based MSTParser), Rosa (2015) (adposition–noun constructions in 30 languages), Kohita et al. (2017) (various constructions involving function and content words in 19 typologically varied languages), and Rehbein et al. (2017) (15 languages, although the extent of the improvements varied considerably, and in the exceptional case of Turkish regress was observed for all three parsers used in the experiments).[3]

On the other hand, de Lhoneux and Nivre (2016) report on an experiment involving 24 languages, in which the original UD representation of verb groups (modal–verb constructions) turns out to be more learnable by MaltParser than the converted representation with main verbs acting as dependents of modal verbs. In a similar vein, Wisniewski and Lacroix (2017) report that languages and particular constructions vary drastically in the extent to which the syntactic or the semantic approach to headedness is more or less learnable by their own transition-based parser. However, out of the seven constructions they consider (similar to those considered in Silveira and Manning, 2015), four differentiate UD and SUD, and out of these four, two (copula–predicate and case–noun, but not mark–verb) are more learnable in the syntactic encoding in the majority of languages – copula–predicate con-

[2]The other two constructions, which have the same representation in the two schemes, are: noun–noun (e.g., *John Doe*), determiner–noun (e.g., *the apple*).

[3]See also Ivanova et al. (2013) and Kirilin and Versley (2015), where a similar conclusion about better learnability of syntactic schemes is reached on the basis of comparison of different – rather than automatically converted – datasets.

structions by a wide margin (75% of languages). Unfortunately, the paper does not present the full results of the experiments, so it is not clear whether there is a correlation between, say, language family and learnability of particular representations of particular constructions.

The current paper is methodologically closest to Rehbein et al. (2017) and Kohita et al. (2017): it reports results of experiments performed on multiple corpora of typologically diverse languages, and it compares the learnability of different annotation schemes applied to the same underlying texts. However, the novelty of the current paper lies in comparing the learnability of two comprehensive linguistically-informed annotation schemes rather than a real scheme and an artificial scheme differing from it in the headedness of a single or a small number of constructions. That is, unlike previous experiments reported in the literature cited above, the experiments reported here were performed *in vivo* rather than *in vitro*. This matters, as any realistic annotation schema which employs a more 'syntactic' approach to headedness than UD will also differ from UD in the repertoire and distribution of dependency labels, and will also take into account the intrinsic linguistic interaction between various constructions. The co-existence of large and high-quality treebanks in their UD and SUD variants presents the unique opportunity to compare the learnability of 'semantic' and 'syntactic' annotation schemes in a realistic setup.

2 Experimental Setup[4]

2.1 Data

Treebanks. Experiments were performed on a subset of UD 2.6 treebanks and the corresponding SUD 2.6 treebanks created by the SUD team. 21 treebanks (in each annotation scheme) representing 18 languages were selected on the basis of three criteria. First of all, emphasis was put on the quality of treebanks, so only those – mainly Indo-European – that have the quality score higher than 70 percent were used (as evaluated by the official UD script: https://github.com/UniversalDependencies/tools/blob/master/evaluate_treebank.pl by Dan Zeman). Second, in order to obtain robust results, only relatively large corpora, over 70k tokens, were selected. Third, due to the limited computational power, upper bounds on the treebank size had to be set – 1000k tokens. Three languages – Italian, Polish and Swedish – are represented by two treebanks each, which may give some insight into how stable certain trends are within one language.

Preprocessing. The original UD and SUD treebanks had been preprocessed before the experiments were carried out. In particular, the representation of multitoken words was normalised to the format where, say, the French form *du* is represented in the conllu scheme by two lines (one corresponding to *de* with information 'SpaceAfter=No' and the other to *le*) rather than three (one for *de*, another for *le* and another for their contraction *du*). This was done to remove some inconsistencies between training and testing subsets of some corpora. Additionally, all tokens with PUNCT as their UPOS tag were removed, unless they had dependents.

Pretrained embeddings. Where possible (i.e., in the case of UDPipe, UUParser, and COMBO), pretrained *fasttext* word embeddings were utilised (https://fasttext.cc/docs/en/crawl-vectors.html; Grave et al., 2018) as opposed to learning embeddings during the training process. The *fasttext* architecture is based on embeddings of character n-grams, but only the resulting word-level vectors were used in the training procedure, as all of the selected systems which offer an option of including external embeddings can work with word embeddings only. Each embedding model was pruned to 300,000 most frequent forms, to ease the computational load.

2.2 Parsers

Two transition-based and three graph-based parsers were used in the experiments. Some of these tools offer robust pipelines for NLP, including tokenisation, lemmatisation and tagging, but in the current experiments only the parser component of the tool was trained; in particular, POS tags were extracted from the gold standard and used as features. Below, training procedures of each parser are described separately, including only information about parsers' hyperparameters that differ from the default setting.[5]

UDPipe. Version 1.2.0 (http://ufal.mff.

[4] The code necessary to perform all of the actions described in this section can be found on our Github page: https://github.com/ryszardtuora/ud_vs_sud

[5] In total 1764 models were trained (882 with UDPipe, 294 with Mate, 210 with each graph-based and transition-based UUParser, and 168 with COMBO).

cuni.cz/udpipe; Straka and Straková, 2017) of this transition-based parser was used *without* the default values of various hyperparameters, as these were fitted on UD, and thus could skew the results against SUD. Instead, 21 models were trained on each treebank (for either annotation scheme). That is, for each transition system available – projective, swap, link2 – seven models were trained using random hyperparameter search – a feature provided by UDPipe that randomises some of the training hyperparameters.

Mate. Version 3.62 (Bohnet, 2010) of the graph-based parser was utilised; it was adapted from the version 3.61 (available here: http://code.google.com/p/mate-tools/) to our study.[6] Seven models were trained for each treebank (and each annotation scheme), and in every training run a different non-projective approximation threshold was selected from the following list: 0.75, 0.5, 0.4, 0.3, 0.2, 0.15, 0.1.

UUParser. Version 2.4 (https://github.com/UppsalaNLP/uuparser; de Lhoneux et al., 2017) of both graph- and transition-based methodologies were applied in the experiment. UUParser is an adaptation of the BIST parser (Kiperwasser and Goldberg, 2016). In UUParser, swap transition and Eisner algorithms were implemented, among others, in place of their projective counterparts – used by BIST parser – in transition- and graph-based versions respectively. Universal POS tags dimension was set to 20 and external word embedding dimension was adapted to the size of the embeddings used. Five models with different random seeds were trained, and the one which performed best as measured by LAS on the dev set, was then selected for testing.

COMBO. The graph-based dependency parsing component from Version 1.0.1 https://gitlab.clarin-pl.eu/syntactic-tools/combo of the COMBO (Rybak and Wróblewska, 2018) pipeline was utilized, with word embeddings, characters, and gold UPOS tags as features. For each treebank four models with different combinations of learning rate (0.001 or 0.002) and dropout probability (0.4 or 0.25) have been trained, for 100 epochs each.

[6]The implemented change forces the parser to produce only one root in each sentence. We thank Bernd Bohnet for adjusting the parser to our needs and for allowing to share the new Mate 3.62 version on our Github page.

2.3 Evaluation

In each case, models produced by the parsers on the basis of training sets were used to parse the test parts of the respective treebanks. Hyperparameter selection (based on LAS) and early-stopping was performed on the development set.

The official `conll18_ud_eval.py` script (http://universaldependencies.org/conll18/evaluation.html) was used to calulate UAS and LAS scores both during hyperparameter selection and during final testing. Due to the differences in the annotation of labels in UD and SUD, modifications had to be implemented in the script. In UD, syntactic relations are divided with a colon into two parts. The first part refers to the universal dependency taxonomy. The second part, after the colon, is a relation subtype which is specific to one language or a group of related languages. For example, advmod is a general UD relation that refers to adverbial dependents, while advmod:arg is specific for Polish and referes to obligatory adverbial arguments, advmod:df is specific for Chinese and Cantonese and refers to durative and frequentative noun phrases, etc. In SUD, on the other hand, some general relation names contain the colon; e.g., comp:pred is used for copulae and comp:aux for auxiliary verbs.

The `conll18_ud_eval.py` script ignores the part after the colon during evaluation (if a parser predicts advmod:df instead of just advmod, or vice versa, that counts as a match). In the case of SUD, leaving out the part after the colon would result in incomplete labels. Hence, the evaluation script was modified so that labels were processed differently in the case of SUD: if the part of the relation after the colon is either aux, pred, obj, or obl, it will not be split off, and a full match of the predicted relation will be necessary.[7] These label manipulations are applied only at the stage of evaluation; during the training phase, parsers are learning the full spectrum of dependency labels.

3 Results

3.1 UAS and LAS scores

The results of the experiments are presented in Table 1 (on the next page). Out of 210 comparisons, 58 gave statistically significant results using the

[7]We would like to thank Bruno Guillaume and the whole SUD team for a discussion on an unbiased evaluation method.

strict version of McNemar's test, with $\alpha = 0.001$.[8] Out of these, 46 favour SUD, and 12 – UD; this confirms the generally – but not unanimously – higher learnability of the 'syntactic' scheme. (Taking into account all 210 comparisons, the result is 146:64 in favour of SUD.) There is a clear difference between the transition- and graph-based parsers in this respect. The former – UDPipe and transition-based UUParser – have no clear preferences: their SUD:UD scores in statistically significant differences are 4:4 and 6:4, respectively (and in all differences: 23:19 and 23:19). The latter – Mate, COMBO, and graph-based UUParser – strongly prefer SUD, with respective significant scores 14:1, 8:2, and 14:1 (and all scores: 34:8, 32:10, and 34:8). Particular parsers show similar preferences for SUD or UD in terms of UAS and LAS, apart from Mate, whose preference for SUD is 5:1 in terms of significant UAS differences and 9:0 in terms of LAS.

Moreover, the mean difference of UAS and LAS results is similar for all parsers. In the case of UAS, the mean differences between UD and SUD are -0.03, -0.61, -0.36, -0.04, and -0.48 for UDPipe, Mate, COMBO, transition-based and graph-based UUParser respectively (i.e., SUD is preferred on the average), and in the case of LAS, the differences are 0.01, -0.86, -0.42, -0.14, -0.60 (that is, apart from UDPipe, parsers tend to prefer SUD). The highest difference between these two metrics concerns Mate's results ($\delta = 0.20$); however, the difference is minimal in the case of, for instance, UDPipe's results ($\delta = 0.04$).

As to particular corpora, when one scheme is more learnable according to one parser, it tends to be more learnable also according to other parsers. Only in the case of the Polish PDB treebank do different parsers have significantly different preferences: the two transition-based parsers and COMBO significantly prefer UD (both with respect to the LAS score), while the graph-based UUParse prefers SUD (with respect to UAS).

Interestingly, this relative stability in preferences for SUD or UD concerns particular corpora (and, to some extent, languages; especially the two Swedish corpora behave similarly on most parsers) but not

[8]The strict version of McNemar's test was employed here in order to minimise the false discovery rate; as Table 1 reports the results of 210 comparisons, the weaker test, with $\alpha = 0.05$, would likely produce some false significance claims, while with the stricter version the probability that all statistically significant claims are correct is over 0.94.

	UDPipe						**Mate**						**COMBO**						**UUParser (transition-based)**						**UUParser (graph-based)**					
	UAS			LAS			UAS			LAS			UAS			LAS			UAS			LAS			UAS			LAS		
	UD	SUD	Δ	UD	SUD	Δ	UD	SUD	Δ	UD	SUD	Δ	UD	SUD	Δ	UD	SUD	Δ	UD	SUD	Δ	UD	SUD	Δ	UD	SUD	Δ	UD	SUD	Δ
bg	90.95	91.13	−0.18	86.76	86.94	−0.18	91.25	92.15	−0.90	86.47	87.90	**−1.43**	93.23	93.74	−0.51	89.40	90.30	−0.91	91.51	92.25	−0.74	87.44	88.34	−0.90	92.22	92.70	−0.48	88.30	89.13	−0.83
cs	89.14	88.47	0.67	86.22	83.86	**2.36**	88.89	89.15	−0.25	83.64	82.87	0.76	92.26	92.15	0.11	88.77	87.74	**1.03**	90.71	90.07	0.65	86.80	84.93	**1.87**	91.30	91.50	−0.20	87.52	86.66	0.86
de	85.10	83.14	**1.96**	80.07	79.06	1.00	85.37	84.08	**1.29**	79.47	79.16	0.32	87.87	86.97	0.91	82.74	82.81	−0.07	84.90	84.36	0.54	79.61	79.83	−0.23	86.17	84.92	**1.25**	81.04	80.78	0.25
en	83.65	84.70	−1.05	81.08	82.42	**−1.34**	83.72	85.63	**−1.90**	81.35	83.49	**−2.15**	85.94	86.90	−0.95	82.97	84.03	**−1.06**	83.58	84.81	**−1.23**	80.86	82.58	**−1.72**	85.26	86.27	−1.01	82.82	83.98	**−1.16**
es	90.42	90.22	0.20	87.32	87.41	−0.09	90.45	90.93	−0.48	87.23	87.99	**−0.76**	92.81	93.22	−0.41	90.66	90.95	−0.29	91.29	91.44	−0.14	88.68	88.76	−0.08	91.71	91.99	−0.29	89.10	89.53	−0.43
et	83.60	83.72	−0.12	80.53	80.48	0.05	79.24	81.24	**−2.00**	71.98	74.99	**−3.00**	86.13	87.16	**−1.03**	82.78	83.85	**−1.07**	83.21	83.91	**−0.70**	79.68	80.49	**−0.81**	83.95	85.12	**−1.17**	80.34	81.71	**−1.37**
hr	86.26	86.69	−0.43	81.52	82.31	−0.79	85.96	86.61	−0.65	79.67	81.03	**−1.36**	88.67	89.39	−0.72	83.17	84.54	**−1.37**	86.59	86.18	0.41	80.84	80.93	−0.09	87.78	88.67	**−0.90**	82.24	83.74	**−1.50**
it	92.12	92.09	0.03	89.97	90.35	−0.38	92.65	92.54	0.11	90.32	90.71	−0.39	93.98	93.64	0.35	92.39	92.09	0.30	92.99	92.05	0.94	91.10	90.34	0.76	93.37	93.19	0.17	91.54	91.47	0.06
it	88.59	87.62	**0.97**	85.08	84.55	0.53	88.46	88.53	−0.07	85.00	85.68	−0.68	90.41	90.43	−0.02	86.98	87.61	−0.63	89.37	88.72	0.65	85.74	85.65	0.09	89.99	89.48	0.52	86.48	86.57	−0.09
la	93.90	93.69	0.21	92.81	92.67	0.14	94.71	94.53	0.18	93.33	93.37	−0.04	95.75	96.01	−0.26	94.75	94.97	−0.23	94.75	93.76	**0.99**	93.54	92.60	**0.94**	95.57	95.80	−0.23	94.46	94.85	−0.40
lt	71.18	72.25	−1.07	66.87	66.94	−0.07	66.96	67.74	−0.77	60.16	60.00	0.16	73.65	74.83	−1.17	69.08	68.11	0.97	70.99	72.14	−1.15	65.85	66.36	−0.51	74.41	75.92	**−1.51**	69.78	69.90	−0.11
lv	83.57	83.63	−0.06	79.81	80.15	−0.34	79.16	79.90	−0.74	72.64	73.91	**−1.28**	84.49	85.37	**−0.88**	80.54	81.68	**−1.14**	82.73	83.01	−0.28	78.57	78.96	−0.39	83.93	84.90	**−0.97**	79.98	81.14	**−1.16**
pl	95.38	94.66	0.72	93.41	92.67	0.74	94.30	94.33	−0.03	88.83	89.56	−0.73	96.04	95.99	0.05	93.43	93.96	−0.53	95.04	94.74	0.29	91.91	92.16	−0.25	95.74	95.56	0.18	92.83	93.00	−0.17
pl	89.60	89.41	0.19	87.24	85.51	**1.73**	88.01	88.64	−0.63	82.72	82.54	0.18	92.18	92.66	−0.48	89.33	88.65	**0.68**	89.60	89.86	−0.26	86.27	85.26	**1.00**	90.28	91.34	**−1.07**	86.96	86.80	0.16
ro	86.77	85.74	1.03	80.94	80.87	0.07	87.53	87.17	0.36	81.25	81.99	−0.74	91.00	90.88	0.11	85.97	86.69	−0.72	89.08	88.98	0.10	83.90	84.67	−0.77	89.91	90.04	−0.13	84.73	85.74	**−1.01**
ru	84.44	85.96	**−1.52**	80.26	82.31	**−2.04**	83.79	84.27	−0.47	77.39	78.16	−0.76	86.69	86.71	−0.02	80.75	81.75	−1.00	84.74	85.68	−0.94	78.66	80.31	**−1.65**	86.60	87.86	**−1.26**	80.61	82.59	**−1.98**
sk	86.78	87.99	**−1.21**	84.41	83.99	0.42	85.72	88.21	**−2.50**	80.73	81.65	−0.92	88.61	90.14	**−1.52**	85.03	84.90	0.13	86.29	87.58	**−1.29**	82.10	82.00	0.10	87.36	89.16	**−1.80**	83.41	83.66	−0.25
sl	89.80	90.07	−0.26	87.69	88.09	−0.40	90.46	90.58	−0.12	86.93	87.59	−0.66	92.89	93.32	−0.43	90.26	91.17	−0.91	90.57	90.23	0.34	87.48	87.70	−0.22	91.87	92.21	−0.34	89.07	89.66	−0.59
sr	87.73	87.53	0.20	83.87	84.12	−0.25	88.50	89.01	−0.51	83.48	84.96	**−1.48**	90.43	90.92	−0.49	86.31	87.63	**−1.32**	89.05	87.85	1.20	84.94	84.65	0.29	90.10	90.64	−0.54	85.99	87.47	**−1.48**
sv	86.68	86.99	−0.32	82.26	82.67	−0.40	87.05	88.13	**−1.09**	82.52	83.94	**−1.42**	88.63	88.63	0.00	84.25	84.56	−0.31	87.78	87.44	0.34	83.61	83.36	0.25	88.27	88.37	−0.10	83.92	84.51	−0.59
sv	87.89	88.40	−0.51	84.49	85.05	−0.56	88.00	89.64	**−1.64**	84.56	86.16	**−1.60**	90.03	90.16	−0.13	86.79	87.10	−0.31	88.65	89.15	−0.50	85.39	86.05	−0.66	89.74	90.00	−0.26	86.58	87.35	−0.77

Table 1: UAS and LAS results of the five parsing systems on the UD and SUD versions of the corpora. Differences are in green if UD annotation is more learnable and in red if SUD annotation is more learnable; statistically significant ($p < 0.001$) differences are additionally in bold. See Table 2 for full names of corpora

language families. This is especially clear in the case of Germanic languages. While English and, to a smaller extent the two Swedish treebanks, show strong preference for the 'syntactic' scheme across all five parsing systems, the German treebank favours 'semantic' scheme in nearly all cases. The proportion of statistically significant differences (to all comparisons) is high in the group of Germanic languages (14 out of 40).

Slavic languages, out of which nine treebanks were included in the study, appear to be following a similar pattern, altough to a smaller extent. Czech shows a preference in favour of the UD scheme (three statistically significant differences in favour of UD and zero in favour of SUD), learnability of Polish PDB appears to be dependent on the parser used (as discussed earlier), Polish LFG and Slovenian do not show any significant preferences, while Croatian, Russian (strongly), Slovak and Serbian present higher learnability in the SUD scheme – all observed statistically significant differences are in favour of SUD. On the other hand, Romance languages do not show strong preferences for either of the schemes. In total, five statistically significant preferences were found in this language group, out of which three are in favour of UD and two in favour of SUD. In the case of Baltic (Latvian and Lithuanian) and Finno-Ugric (Estonian), all of the statistically significant differences prefer SUD scheme (14 in total). However, since only two Baltic treebanks and only one Finno-Ugric treebank were included in the study, we refrain from drawing any conclusions about these language groups; a more comprehensive study is needed.

To what extent do these results reflect headedness decisions of the two schemes, i.e., preference for content heads in UD and for functional heads in SUD? It is important to note that, unlike in the previous experiments, differences between the two annotation schemes do not only concern headedness, but also the repertoire and meanings of dependency labels. The number of basic dependency labels (as defined in §2.3) is consistently smaller in the case of SUD than in the case of UD, which may favourably bias parsers towards SUD. For instance, the English GUM treebank has 48 and 43 different labels in UD and SUD respectively, and applying the label processing described in §2.3 results in further reduction to 36 and 25 different labels, respectively, making the task of parsers easier in the case of SUD than in the case of UD. (These predictions are confirmed by the results concerning Label Entropy, reported in §3.2 below.) Hence, the results reported in this section cannot at this stage be interpreted as showing – but are compatible with the claim – that 'functional headedness' tends to be more learnable than 'content headedness'; further experiments are needed to confirm or deny such a claim.

3.2 Quantitative syntactic properties

In an attempt to find which quantifiable syntactic properties of treebanks may impact the differences in parsing performance, five different metrics were calculated (see Table 2 on the next page). Some of these properties differ substantially between UD and SUD. Two notable examples are Average Dependency Length (*ADL*) and Average Token Depth (*ATD*) – properties which are inversely related to each other. *ADL* is calculated so that the length of a dependency between neighbouring tokens is equal to one, and each intervening token increases it by one. *ATD* is calculated by only taking non-root tokens into consideration; immediate children of root have depth equal to one, and each intervening token in the path from a node to root adds one to the depth of the token. SUD is characterised by deeper trees (with higher *ATD*), and UD – by flatter trees and longer dependency arcs (i.e., higher *ADL*). SUD treebanks have, without exception, higher *ATD*, and lower *ADL* than their UD counterparts.

These differences may be important, as there is growing evidence that natural languages tend to minimise dependency lengths (see, e.g., Temperley and Gildea, 2018 and references therein). Nevertheless, as shown in Table 3, differences between UD and SUD in *ADL* and *ATD* are not significantly correlated with differences between UD and SUD in terms of UAS or LAS.

In addition, two entropy-based measures were calculated: Arc Direction Entropy (*ADE*), and Label Entropy (*LE*). *ADE* is used to quantify the rigidity of word order in a given corpus, i.e., given two tokens connected by a dependency arc, the label of the relation, and the UPOS tags of the tokens, how much certainty can we have about the linear ordering of these tokens (head-initial vs. head-final). Arguably, the more consistent word order is, the easier the task of parsing becomes. As expected, English treebanks have the lowest *ADE*, whereas free-order languages such as Polish show higher entropy.

	ADL			*ATD*			*ADE*			*LE*			*NPROJ*		
	UD	SUD	Δ	UD	SUD	Δ	UD	SUD	Δ	UD	SUD	Δ	UD	SUD	Δ
bg-btb	2.07	1.74	0.33	1.75	1.93	−0.18	0.19	0.20	−0.00	2.84	2.26	0.58	0.03	0.09	−0.05
cs-fictree	2.10	1.88	0.22	1.64	1.79	−0.15	0.32	0.32	−0.00	2.91	2.32	0.60	0.11	0.22	−0.11
de-gsd	2.76	2.43	0.34	1.79	1.99	−0.20	0.24	0.26	−0.02	2.80	2.32	0.48	0.10	0.20	−0.10
en-gum	2.32	1.95	0.38	1.83	2.06	−0.23	0.13	0.13	−0.00	2.95	2.34	0.61	0.05	0.09	−0.03
es-ancora	2.46	2.10	0.36	2.14	2.41	−0.27	0.17	0.17	−0.00	2.82	2.17	0.65	0.05	0.22	−0.17
et-edt	2.16	1.96	0.19	1.67	1.80	−0.13	0.35	0.36	−0.00	2.90	2.21	0.70	0.03	0.12	−0.09
hr-set	2.39	2.07	0.32	1.95	2.19	−0.24	0.21	0.24	−0.03	2.97	2.29	0.68	0.08	0.35	−0.27
it-isdt	2.18	1.86	0.32	1.91	2.14	−0.23	0.18	0.18	−0.00	2.73	2.20	0.53	0.01	0.08	−0.07
it-vit	2.36	2.00	0.36	2.04	2.30	−0.26	0.18	0.18	0.00	2.72	2.19	0.54	0.03	0.12	−0.09
la-llct	2.73	2.62	0.11	2.01	2.12	−0.11	0.31	0.31	−0.00	2.85	2.30	0.55	0.29	0.32	−0.03
lt-alksnis	2.29	2.10	0.18	1.93	2.05	−0.11	0.30	0.28	0.02	2.73	1.99	0.74	0.12	0.14	−0.03
lv-lvtb	2.15	1.93	0.22	1.77	1.90	−0.12	0.29	0.29	−0.00	2.95	2.27	0.69	0.07	0.11	−0.05
pl-lfg	1.67	1.55	0.12	1.42	1.49	−0.07	0.36	0.37	−0.00	2.73	2.16	0.58	0.01	0.07	−0.06
pl-pdb	2.05	1.83	0.21	1.82	1.97	−0.15	0.30	0.30	−0.00	2.93	2.24	0.69	0.06	0.16	−0.10
ro-nonstandard	2.41	2.08	0.33	1.83	2.08	−0.25	0.29	0.28	0.01	2.98	2.35	0.63	0.06	0.33	−0.28
ru-gsd	2.09	1.91	0.18	1.92	2.05	−0.13	0.20	0.21	−0.01	2.74	2.13	0.61	0.06	0.08	−0.02
sk-snk	1.83	1.67	0.16	1.55	1.66	−0.11	0.31	0.31	−0.00	2.82	2.21	0.61	0.03	0.12	−0.09
sl-ssj	2.34	1.99	0.34	1.80	2.03	−0.23	0.24	0.26	−0.01	2.89	2.15	0.74	0.12	0.28	−0.16
sr-set	2.35	2.02	0.33	1.96	2.22	−0.26	0.18	0.21	−0.03	2.93	2.26	0.67	0.03	0.27	−0.24
sv-lines	2.26	1.90	0.36	1.76	1.97	−0.21	0.22	0.21	0.01	2.93	2.28	0.66	0.05	0.11	−0.05
sv-talbanken	2.27	1.92	0.34	1.76	1.95	−0.20	0.22	0.21	0.01	2.93	2.24	0.68	0.03	0.08	−0.05

Table 2: Quantitative syntactic properties of the treebanks used in the experiment: average dependency length *ADL*, average token depth *ATD*, arc direction entropy *ADE*, label entropy *LE*, percentage of non-projective trees *NPROJ* Columns marked with Δ represent differences between UD and SUD; in green if a given figure is higher for UD, and red otherwise

On the other hand, Label Entropy is the entropy of the frequency distribution of dependency labels across the treebank. It is calculated by iterating over all tokens in the treebank and counting their dependency labels. This frequency distribution is treated as a probability distribution and used for calculating entropy. *LE* was introduced because the SUD scheme has substantially smaller sets of labels for dependency relations, and *LE* offers a more informed way of assessing the baseline difficulty of a label scheme than the mere cardinality of the labelset. SUD versions of the same treebanks are in all cases characterized by lower *LE*. Both measures were calculated using the dependency label transformations defined in §2.3, i.e., with the exception of some SUD labels, all labels were split after a colon.

We were not able to confirm the correlation between differences in learnability and differences in *ADE* reported in Gulordava and Merlo (2016) (on the basis of artificially created data) and in Rehbein et al. (2017). Most probably, this is because of the very small differences in *ADE* between SUD and UD, much lower than in the experiments cited in these two papers. In fact, the differences are so insignificant that we would prefer to be cautious in interpreting the one statistically significant, positive correlation that concerns COMBO parser. Following the results from the papers cited above, one would expect to obtain a negative correlation between ADE and learnability (i.e. higher ADE leads to lower learnability). The opposite is the case here. This result is puzzling; it is possible that the correlation is in fact spurious.

Another aspect of tree structure which is significant in this context, is the proportion of non-projective arcs (and consequently non-projective trees) in the treebanks. As shown in the last column of Table 2, marked with *NPROJ*, SUD is characterised by a consistently larger degree of non-projective trees. On average, conversion into SUD increases the percentage ratio of non-projective trees in a treebank 3.52 times (up to 10.32 times in the case of Polish LFG). This is consistent with previous experiments in the domain; e.g., Kohita et al. (2017) report that, after applying syntactic-like transformations, the ratio of non-projective arcs in the training sets increased by 10 percentage points on average. Non-projective dependency structures are notoriously hard to parse for humans, and so one might expect a similar effect in computational settings. However, modern parsers are able to handle non-projective trees; they offer particular transition systems (e.g., UDPipe) or hyperparameters (e.g., Mate) that can be manipulated in order to better fit treebanks with a certain degree of non-

		ADL		*ATD*		*ADE*		*LE*		*NPROJ*	
parser	measure	cor	p	cor	p	cor	p	cor	p	cor	p
UDPipe	UAS	0.05	0.819	−0.27	0.238	−0.06	0.797	**−0.46**	0.035	**−0.50**	0.022
	LAS	−0.35	0.121	0.27	0.240	0.13	0.577	−0.28	0.222	−0.29	0.196
Mate	UAS	−0.09	0.699	−0.20	0.377	−0.13	0.581	**−0.53**	0.015	−0.31	0.176
	LAS	−0.29	0.202	0.18	0.445	0.10	0.653	−0.28	0.220	−0.12	0.597
COMBO	UAS	0.19	0.408	−0.26	0.254	0.04	0.850	**−0.60**	0.005	−0.21	0.364
	LAS	−0.18	0.425	0.29	0.196	**0.51**	0.021	−0.24	0.301	0.11	0.637
UUParser (transition)	UAS	0.12	0.613	−0.41	0.067	−0.18	0.422	−0.38	0.090	−0.43	0.051
	LAS	−0.12	0.613	−0.08	0.745	0.09	0.682	−0.15	0.502	−0.32	0.153
UUparser (graph)	UAS	0.36	0.107	−0.36	0.105	0.10	0.661	**−0.62**	0.003	−0.27	0.242
	LAS	−0.07	0.750	0.17	0.459	0.32	0.153	−0.36	0.106	−0.07	0.758

Table 3: Correlations (cor) between 1) learnability difference between UD and SUD and 2) differences in values of various corpus measures: average dependency length *ADL*, average token depth *ATD*, arc direction entropy *ADE*, label entropy *LE*, percentage of non-projective trees *NPROJ*. Statistically significant ($p < 0.05$) values are in bold

projectivity. Correlations between the difference in the percentage of non-projective trees between UD and SUD treebanks and learnability scores are presented in the last column of Table 3. Only one statistically significant, correlation: -0.50 can be observed in the case of UDPipe, with respect to UAS score.

4 Conclusions

While some initial work suggested clear relation between learnability of dependency parsing and the 'semantic' or 'syntactic' approach to headedness, with 'syntactic' annotations usually reported as more learnable, the experiments often had a very limited scope: they concerned one language, or just one or a very small number of constructions, or just one or two parsers. More extensive experiments, performed on a number of languages, taking into account a handful of constructions and a few parsers, such as those reported in Rehbein et al. (2017), showed that this relation between learnability and different approaches to headedness, even though imperfect, in general favours syntactic-like approaches, but also suggested a more stable correlation between learnability and other corpus characteristics (such as *ADE*). All these experiments were performed *in vitro*, on the basis of dependency corpora with one or just a few constructions reanalysed for the purpose of the experiments.

In contrast, the current paper presents the results of comparing two full-fledged annotation schemes *in vivo*: the 'semantic' UD and the 'syntactic' SUD. The experiments confirm that it cannot be claimed that more 'syntactic' approaches to annotation uniformly lead to better learnability: this depends on particular languages (rather than on language families) and on particular parsers. However, corpora annotated according to the SUD scheme tend to be more learnable, especially, by the graph-based parsers utilised in the experiments.

As to correlations between corpus characteristics and learnability, the experiments show a clear correlation between Label Entropy and parsers' performance (especially, in terms of UAS), which suggests that SUD may take advantage of its smaller set of labels or lower order variability of labels, or both. Also, correlation was found between the difference in the percentage of non-projective trees between both schemes and learnability in the case of UDPipe (again, in terms of UAS). This may suggest the inability of this parser to effectively deal with higher degrees of non-projectivity, even though some hyperparameter tuning was implemented to deal with this issue. On the other hand, the results do not confirm the recent hypothesis that learnability of the two kinds of annotations is negatively correlated with arc direction entropy. In the same vein, we have not found statistically significant correlations between parser performance, and average dependency length.

Future work should seek to dissociate the effect of more learnable dependency labels from that of different approaches to headedness; to this end experiments should be performed on corpora with trees typologically just like those in UD and SUD, but with label schemes modified so that Label Entropy is not correlated with learnability. Also, it

would be interesting to relate learnability of particular schemes by particular parsers to their inherent dependency displacement bias (cf. Anderson and Gómez-Rodríguez 2020). Clearly, many more experiments of this sort are needed to establish exact factors influencing learnability of dependency parsers.

References

Mark Anderson and Carlos Gómez-Rodríguez. 2020. Inherent dependency displacement bias of transition-based algorithms. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5147–5155, Marseille, France. European Language Resources Association.

Bernd Bohnet. 2010. Very high accuracy and fast dependency parsing is not a contradiction. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010)*, pages 89–97, Beijing.

Miryam de Lhoneux and Joakim Nivre. 2016. Should have, would have, could have. Investigating verb group representations for parsing with Universal Dependencies. In *Proceedings of the Workshop on Multilingual and Cross-lingual Methods in NLP*, pages 10–19, San Diego, California. Association for Computational Linguistics.

Miryam de Lhoneux, Sara Stymne, and Joakim Nivre. 2017. Arc-hybrid non-projective dependency parsing with a static-dynamic oracle. In *Proceedings of the The 15th International Conference on Parsing Technologies (IWPT).*, Pisa, Italy.

Kim Gerdes, Bruno Guillaume, Sylvain Kahane, and Guy Perrier. 2018. SUD or Surface-Syntactic Universal Dependencies: An annotation scheme near-isomorphic to UD. In *Proceedings of the Second Workshop on Universal Dependencies (UDW 2018)*, pages 66–74. Association for Computational Linguistics.

Kim Gerdes, Bruno Guillaume, Sylvain Kahane, and Guy Perrier. 2019. Improving Surface-syntactic Universal Dependencies (SUD): surface-syntactic relations and deep syntactic features. In *TLT 2019 - 18th International Workshop on Treebanks and Linguistic Theories*, Paris, France.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning word vectors for 157 languages. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Kristina Gulordava and Paola Merlo. 2016. Multilingual dependency parsing evaluation: A large-scale analysis of word order properties using artificial data. *Transactions of the Association for Computational Linguistics*, 4:343–356.

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

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional LSTM feature representations. *TACL*, 4:313–327.

Angelika Kirilin and Yannick Versley. 2015. What is hard in Universal Dependency parsing? In *Proceedings of the Sixth Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 31–38.

Ryosuke Kohita, Hiroshi Noji, and Yuji Matsumoto. 2017. Multilingual back-and-forth conversion between content and function head for easy dependency parsing. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 1–7, Valencia, Spain. Association for Computational Linguistics.

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 44th Annual Meeting of the Association for Computational Linguistics*, pages 257–264, Sydney, Australia.

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 for Computational Linguistics*, pages 968–975, Prague.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal Dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation, LREC 2016*, pages 1659–1666, Portorož, Slovenia. European Language Resources Association (ELRA), European Language Resources Association (ELRA).

Ines Rehbein, Julius Steen, Bich-Ngoc Do, and Anette Frank. 2017. Universal Dependencies are hard to parse – or are they? In *Proceedings of the Fourth International Conference on Dependency Linguistics (DepLing 2017)*, pages 218–228, Pisa, Italy.

Rudolf Rosa. 2015. Multi-source cross-lingual delexicalized parser transfer: Prague or Stanford? In *Proceedings of the Third International Conference on Dependency Linguistics (DepLing 2015)*, pages 281–290, Uppsala.

Piotr Rybak and Alina Wróblewska. 2018. Semi-supervised neural system for tagging, parsing and

lemmatization. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 45–54, Brussels, Belgium. Association for Computational Linguistics.

Roy Schwartz, Omri Abend, and Ari Rappoport. 2012. Learnability-based syntactic annotation design. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012)*, pages 2405–2421, Mumbai, India.

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, Uppsala.

Milan Straka and Jana Straková. 2017. Tokenizing, POS tagging, lemmatizing and parsing UD 2.0 with UDPipe. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 88–99, Vancouver, Canada. Association for Computational Linguistics.

David Temperley and Daniel Gildea. 2018. Minimizing syntactic dependency lengths: Typological/cognitive universal? *Annual Review of Linguistics*, 4:67–80.

Guillaume Wisniewski and Ophélie Lacroix. 2017. A systematic comparison of syntactic representations of dependency parsing. In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*, pages 146–152, Gothenburg, Sweden. Association for Computational Linguistics.

Amir Zeldes. 2017. The GUM corpus: Creating multilayer resources in the classroom. *Language Resources and Evaluation*, 51(3):581–612.