On different approaches to syntactic analysis into bi-lexical dependencies: An empirical comparison of direct, PCFG-based, and HPSG-based parsers

Angelina Ivanova[1], *Stephan Oepen*[1], *Rebecca Dridan*[1], *Dan Flickinger*[2], *Lilja Øvrelid*[1], *and Emanuele Lapponi*[1]
[1] University of Oslo, Department of Informatics
[2] Stanford University, Center for the Study of Language and Information

ABSTRACT

We compare three different approaches to parsing into syntactic, bi-lexical dependencies for English: a 'direct' data-driven dependency parser, a statistical phrase structure parser, and a hybrid, 'deep' grammar-driven parser. The analyses from the latter two are post-converted to bi-lexical dependencies. Through this 'reduction' of all three approaches to syntactic dependency parsers, we determine empirically what performance can be obtained for a common set of dependency types for English; in- and out-of-domain experimentation ranges over diverse text types. In doing so, we observe what trade-offs apply along three dimensions: accuracy, efficiency, and resilience to domain variation. Our results suggest that the hand-built grammar in one of our parsers helps in both accuracy and cross-domain parsing performance. When evaluated extrinsically in two downstream tasks – negation resolution and semantic dependency parsing – these accuracy gains do sometimes but not always translate into improved end-to-end performance.

Keywords: syntactic dependency parsing, domain variation

1 MOTIVATION

Bi-lexical dependencies, i.e. binary head–argument relations holding exclusively between lexical units, are widely considered an attractive target representation for syntactic analysis. At the same time, Cer *et al.* (2010) and Foster *et al.* (2011), inter alios, have demonstrated that higher dependency accuracies can sometimes be obtained by parsing into a phrase structure representation first, and then reducing parse trees into bi-lexical dependencies.[1] Thus, if one is willing to accept pure syntactic dependencies as a viable interface (and evaluation) representation, an experimental setup like the one of Cer *et al.* (2010) allows the exact experimental comparison of quite different parsing approaches.[2] Existing such studies to date have predominantly focused on purely data-driven (or statistical) parsers, i.e. systems where linguistic knowledge is exclusively acquired through supervised machine learning from annotated training data. For English, the venerable Wall Street Journal (WSJ) portion of the Penn Treebank (PTB; Marcus *et al.* 1993) has been the most common source of training data for phrase structure and dependency parsers.

Two recent developments make it possible to broaden the range of parsing approaches that can be assessed empirically on the task of deriving bi-lexical syntactic dependencies. Flickinger *et al.* (2012) make available another annotation layer over the same WSJ text, 'deep' syntacto-semantic analyses in the linguistic framework of Head-Driven Phrase Structure Grammar (HPSG; Pollard and Sag 1994; Flickinger 2000). This resource, dubbed DeepBank, is available since late 2012. For the type of HPSG analyses recorded in DeepBank, Zhang and Wang (2009) and Ivanova *et al.* (2012) define a reduction into bi-lexical syntactic dependencies, which they call Derivation Tree-Derived Dependencies (DT). Through application of the converter of Ivanova *et al.* (2012) to DeepBank, we can thus obtain a DT-annotated version of the standard WSJ text, to train and test a data-driven dependency and

[1] This conversion from one representation of syntax to another is lossy, in the sense of discarding constituency information, hence we consider it a reduction in linguistic detail.

[2] In contrast, much earlier work on cross-framework comparison involved post-processing parser outputs in form *and* content, into a target representation for which gold-standard annotations were available. In Section 2 below, we argue that such conversion inevitably introduces blur into the comparison.

phrase structure parser, respectively, and to compare parsing results to a hybrid, grammar-driven HPSG parser. Furthermore, we can draw on a set of additional corpora annotated in the same HPSG format (and thus amenable to conversion for both phrase structure and dependency parsing), instantiating a comparatively diverse range of domains and genres (Oepen *et al.* 2004). Adding this data to our setup for additional cross-domain testing, we seek to document not only what trade-offs apply in terms of dependency accuracy vs. parser efficiency, but also how these trade-offs are affected by domain and genre variation, and more generally how resilient the different approaches are to variation in parser inputs.

2 RELATED WORK

Comparing between parsers from different frameworks has long been an area of active interest, ranging from the original PARSEVAL design (Black *et al.* 1991), to evaluation against 'formalism-independent' dependency banks (King *et al.* 2003; Briscoe and Carroll 2006), to dedicated workshops (Bos *et al.* 2008). Grammatical Relations (GRs; Briscoe and Carroll 2006) have been the target of a number of benchmarks, but they require a heuristic mapping from 'native' parser outputs to the target representations for evaluation, which makes results hard to interpret. Clark and Curran (2007) established an upper bound by running the mapping process on gold-standard data, to put into perspective the mapped results from their CCG parser proper. When Miyao *et al.* (2007) carried out the same experiment for a number of different parsers, they showed that the loss of accuracy due to the mapping process can swamp any actual parser differences. As long as heuristic conversion is required before evaluation, cross-framework comparison inevitably includes a level of fuzziness. An alternative approach is possible when there is enough data available in a particular representation to train a statistical parser, and any necessary conversion between representations is deterministic and hence doesn't introduce the same fuzziness. One example of this approach is demonstrated by Cer *et al.* (2010), who used Stanford Dependencies (de Marneffe and Manning 2008) to evaluate a range of statistical parsers. Since there is a deterministic process for converting between PTB phrase structure trees and Stanford Dependencies, they were able

to evaluate a large number of different parsers which can be trained on one or the other of these representations, using the standard PTB training and test data, without resorting to fuzzy mapping processes.

Fowler and Penn (2010) formally proved that a range of Combinatory Categorial Grammars (CCGs) are context-free. They trained the PCFG Berkeley parser on CCGBank, the CCG annotation of the PTB WSJ text (Hockenmaier and Steedman 2007), advancing the state of the art in terms of supertagging accuracy, PARSEVAL measures, and CCG dependency accuracy. They concluded that a specialized CCG parser is not necessarily more accurate than the general-purpose Berkeley parser; this study, however, fails to also take parser efficiency into account.

In related work for Dutch, Plank and van Noord (2010) suggest that, intuitively, one should expect that a grammar-driven system can be more resilient to domain shifts than a purely data-driven parser. In a contrastive study on parsing into Dutch syntactic dependencies, they substantiated this expectation by showing that their HPSG-based Alpino system performed better and was more resilient to domain variation than data-driven direct dependency parsers.

3 BACKGROUND: HPSG SYNTACTIC DEPENDENCIES

The dependency format we use is a deterministic conversion of HPSG derivation trees licensed by the English Resource Grammar (ERG; Flickinger 2000). Figure 1 of an ERG derivation tree, where labels of internal nodes name HPSG constructions (e.g. subject–head or head–complement: sb-hd_mc_c and hd-cmp_u_c, respectively; see Section 5.3.1 for more details on unary rules). Preterminals are labeled with fine-grained lexical categories – called ERG lexical types – that augment common parts of speech with additional information, for example argument structure or the distinction between count, mass, and proper nouns. In total, the ERG distinguishes about 250 construction types and 1000 lexical types.

ERG derivations are grounded in a formal theory of grammar that explicitly marks heads. For this reason mapping these trees onto projective bi-lexical dependencies is straightforward (Zhang and Wang 2009). Ivanova *et al.* (2012) coin the term DT for ERG Derivation Tree-Derived Dependencies, where they reduce the inventory of some 250

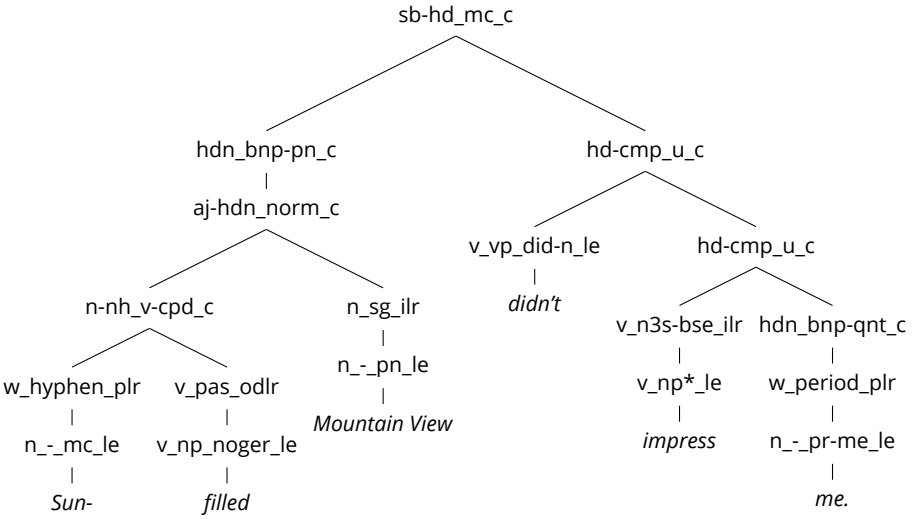


Figure 1: Sample HPSG derivation: construction identifiers label internal nodes, and lexical types the preterminals

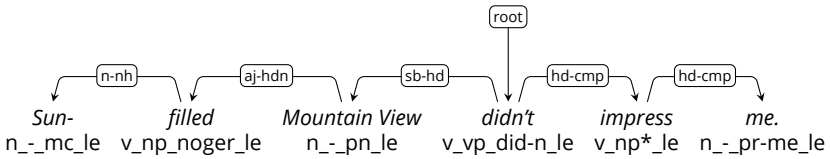


Figure 2: Sample DT bi-lexical dependencies: construction identifiers are generalized to major types (cutting at the first underscore)

ERG syntactic rules to 48 broad HPSG constructions.[3] The DT syntactic dependency tree for our running example is shown in Figure 2.

To better understand the nature of the DT scheme, Ivanova *et al.* (2012) offer a quantitative, structural comparison against two pre-existing dependency standards for English, viz. those from the CoNLL dependency parsing competitions (Nivre *et al.* 2007) and the 'basic' variant of Stanford Dependencies. They observe that the three dependency representations are broadly comparable in granularity and that there are substantial structural correspondences between the schemes.

[3] The ERG distinguishes main clause vs. subordinate subjects, for example, as seen in Figure 1. Ivanova *et al.* (2012) discard this and other grammar-internal contrasts by 'cutting' construction labels at the first underscore.

Measured as average Jaccard similarity over unlabeled dependencies, they observe the strongest correspondence between DT and CoNLL (at a Jaccard index of 0.49, compared to 0.32 for DT and Stanford, and 0.43 between CoNLL and Stanford).

Ivanova *et al.* (2013) complement the above discussed comparison of dependency schemes through an empirical assessment in terms of 'parsability', i.e. accuracy levels available for the different target representations when training and testing a range of state-of-the-art parsers on the same data sets. In their study, the dependency parser of Bohnet and Nivre (2012), henceforth B&N, consistently performs best for all schemes and output configurations. Furthermore, parsability differences between the representations are generally very small.

For a more exact comparison, we replicate their study and evaluate B&N for all three schemes when trained and tested on the same subset of PTB WSJ sentences that are available in DeepBank.[4] The results in Table 1 show that there are no interesting differences in performance of the Bohnet and Nivre (2012) parser across the DT, CoNLL, and Stanford Basic dependency schemes.

Table 1:
Parsability of three dependency schemes, measured as labeled attachment score (LAS) and unlabeled attachment score (UAS)

	LAS	UAS
CoNLL	90.53	93.56
Stanford	90.43	92.87
DT	90.48	92.77

Based on the observations from the above comparisons, we conjecture that DT is as suitable a target representation for parser comparison as any of the others. Furthermore, two linguistic factors add to the attractiveness of DT for our study: it is defined in terms of a formal (and implemented) theory of grammar; and it makes available more fine-grained lexical categories, ERG lexical types, than is common in PTB-derived dependency banks.

[4] For compatibility with much previous work, and to level the playing field for all three schemes, we opt for a slightly different setup for this comparison than in (most) subsequent experiments: here, we apply PTB-style tokenization, coarse-grained PTB parts of speech, and exclude punctuation from scoring.

4 DATA

Below we describe the construction and characteristics of the data sets we use in our parsing experiments, highlighting some of the relevant differences to the more widely-known Penn Treebank format.

4.1 *DeepBank*

DeepBank annotations were created by combining the native ERG parser PET (Callmeier 2002), with a discriminant-based tree selection tool (Carter 1997; Oepen *et al.* 2004), thus making it possible for annotators to navigate the large space of possible analyses efficiently, identify and validate the intended reading, and record its full HPSG analysis in the treebank. Owing to this setup, DeepBank version 1.0, as used presently, lacks analyses for some 15 percent of the WSJ sentences, for which either the ERG parser failed to suggest a set of candidates (within certain bounds on time and memory usage), or the annotators found none of the available parses acceptable.[5] Furthermore, DeepBank annotations to date only comprise the first 21 sections of the PTB WSJ corpus. Following the splits suggested by the DeepBank developers, we train on Sections 0–19, use Section 20 for tuning, and test against Section 21 (abbreviated as WSJ below).[6]

4.2 *Cross-domain test data*

Another benefit of the DT target representation is the availability of comparatively large and diverse samples of additional test data. The ERG Redwoods Treebank (Oepen *et al.* 2004) is similar in genealogy and format to DeepBank, comprising corpora from various domains and genres. Although Redwoods counts a total of some 400,000 annotated tokens, we only draw on it for additional *testing* data. In other words, we do not attempt parser re-training or adaptation against this additional data, but rather test our WSJ-trained parsers on out-of-domain samples from Redwoods. We report on four such test corpora,

[5] Thus, limitations in the ERG and PET effectively lead to the exclusion of a non-trivial percentage of sentences from our training and testing corpora. We discuss methodological ramifications of this setup to our study in Section 13 below.

[6] To 'protect' Section 21 as unseen test data, also for the ERG parser, this final section in Version 1.0 of DeepBank was not exposed to its developers until the grammar and disambiguation model were finalized and frozen for this release.

Table 2:
Sentence, token, and type counts for data sets

		Sentences	Tokens	Types
DeepBank	Train	33,783	661,451	56,582
	Tune	1,721	34,063	8,964
	WSJ	1,414	27, 515	7,668
Redwoods	CB	608	11,653	3,588
	SC	864	13,696	4,925
	VM	993	7,281	1,007
	WS	520	8,701	2,974

viz. (a) a software advocacy essay, *The Cathedral and the Bazaar* (CB); (b) a subset of the SemCor portion of the Brown Corpus (SC; Francis and Kučera 1982); (c) a collection of transcribed, task-oriented spoken dialogues (VM; Wahlster 2000); and (d) part of the Wikipedia-derived WeScience Corpus (WS; Ytrestøl *et al.* 2009). Table 2 provides exact sentence, token, and type counts for these data sets.

4.3 *Tokenization conventions*

A relevant peculiarity of the DeepBank and Redwoods annotations in this context is the ERG approach to tokenization. Three aspects in Figure 1 deviate from the widely used PTB conventions: (a) hyphens (and slashes) introduce token boundaries; (b) whitespace in multiword lexical units (like *ad hoc*, *of course*, or *Mountain View*) does not force token boundaries; and (c) punctuation marks are attached as 'pseudo-affixes' to adjacent words, reflecting the rules of standard orthography. Adolphs *et al.* (2008) offer some linguistic arguments for this approach to tokenization, but for our purposes it suffices to note that these differences to PTB tokenization may in part counter-balance each other in terms of overall parsing difficulty, but they do increase the types-per-tokens ratio somewhat. This property of the DeepBank annotations, arguably, makes English look somewhat similar to languages with moderate inflectional morphology. To take advantage of the fine-grained ERG lexical categories, most of our experiments assume ERG tokenization. In two calibration experiments, however, we also investigate the effects of tokenization differences on our parser comparison.

5 PARSERS

This section describes our three parsers, including the alternate configurations we use, and details of how they are trained and run.

5.1 *PET: native HPSG parsing*

The parser most commonly used with the ERG is called PET (Callmeier 2002), a highly engineered chart parser for unification grammars. PET constructs a complete parse forest, using subsumption-based ambiguity factoring (Oepen and Carroll 2000), and then extracts from the forest n-best lists of complete analyses according to a discriminative parse ranking model (Zhang *et al.* 2007). For our experiments, we employ ERG version 1212, train the parse ranker on Sections 00–19 of DeepBank, and otherwise use the default configuration (which corresponds to the environment used by the DeepBank and Redwoods developers), which is optimized for accuracy. This parser, performing exact inference, we will call ERG_a.

In recent work, Dridan (2013) has augmented ERG parsing with lattice-based sequence labeling over lexical types and lexical rules. Pruning the parse chart prior to forest construction yields greatly improved efficiency at a moderate accuracy loss. We include the best-performing configuration of Dridan (2013) in our experiments, a variant henceforth referred to as ERG_e.

Unlike the other parsers in our study, PET internally operates over an ambiguous token lattice, and there is no easy interface to feed the parser pre-tokenized inputs. We approximate the effects of gold-standard tokenization by requesting from the parser a 2000-best list, which we filter for the top-ranked analysis whose leaves match the treebank tokenization. This approach is imperfect, as in some cases no token-compatible analysis may be on the n-best list, especially so in the ERG_e setup (where lexical items may have been pruned by the sequence labeling model). When this happens, we fall back to the top-ranked analysis and adjust our evaluation metrics to robustly deal with tokenization mismatches (see Section 6).

5.2 *B&N: direct dependency parsing*

The parser of Bohnet and Nivre (2012), henceforth B&N, is a transition-based *dependency parser* with joint tagger that implements global

learning and a beam search for non-projective labeled dependency parsing. This parser consistently outperforms pipeline systems (such as the Malt and MST parsers) both in terms of tagging and parsing accuracy for typologically diverse languages such as Chinese, English, and German. We apply B&N mostly 'out-of-the-box', training on the DT conversion of DeepBank Sections 00–19, and running the parser with an increased beam size of 80.

5.3 *Berkeley: PCFG parsing*

The Berkeley parser (Petrov *et al.* 2006), henceforth just Berkeley, is a generative, unlexicalized *phrase structure* parser that automatically derives a smoothed latent-variable PCFG from the treebank and refines the grammar by a split–merge procedure. The parser achieves state-of-the-art performance on various standard benchmarks.

Formally, the HPSG analyses in the DeepBank and Redwoods treebanks transcend the class of context-free grammars. Nevertheless, one can pragmatically look at an ERG derivation as if it were a context-free phrase structure tree. On this view, standard, off-the-shelf PCFG parsing techniques are applicable to the ERG treebanks. Zhang and Krieger (2011) explore this space experimentally, combining the ERG, Redwoods (but not DeepBank), and massive collections of automatically parsed text. Their study, however, does not consider parser efficiency.[7] In contrast, our goal is to reflect on practical trade-offs along multiple dimensions. We therefore focus on Berkeley, as one of the currently best-performing (and relatively efficient) PCFG engines. We train the parser on the derivation trees and then, for comparison to the other parsers in terms of DT dependency accuracy, we apply the converter of Ivanova *et al.* (2012) to Berkeley outputs. For technical reasons, however, the optional mapping from ERG to PTB tokenization is not applicable in this setup, and hence our experiments involving Berkeley are limited to ERG tokens and fine-grained lexical categories.

5.3.1 Tuning

Table 3 summarizes a series of experiments, seeking to tune the Berkeley parser for maximum accuracy on our development set, DeepBank

[7] Their best PCFG results are only a few points F_1 below the full HPSG parser, using very large PCFGs and exact inference; in this set-up, parsing times in fact exceed those of the native HPSG parser.

Table 3: Tagging accuracy, PARSEVAL F_1, and dependency accuracy for Berkeley on WSJ development data

	Unary Rules Removed			
Labels	Long		Short	
Cycles	5	6	5	6
Gaps	3	3	**0**	**0**
TA	88.46	87.65	89.16	88.46
F_1	74.53	73.72	75.15	73.56
LAS	83.96	83.20	80.49	79.56
UAS	87.12	86.54	87.95	87.15

	Unary Rules Preserved					
Labels	Long		Short		Mixed	
Cycles	5	6	5	6	5	6
Gaps	2	5	**0**	**0**	11	19
TA	90.96	90.62	91.11	**91.62**	90.93	90.94
F_1	76.39	75.66	79.81	**80.33**	76.70	76.74
LAS	86.26	85.90	82.50	83.15	**86.72**	86.16
UAS	89.34	88.92	89.80	**90.34**	89.42	88.84

Section 20. Due to its ability to internally rewrite node labels, this parser should be expected to adapt well also to ERG derivations. Compared to the phrase structure annotations in the PTB, there are two structural differences evident in Figure 1. First, the inventories of phrasal and lexical labels are larger, at around 250 and 1000, respectively, compared to only about two dozen phrasal categories and 45 parts of speech in the PTB. Second, ERG derivations contain more unary (non-branching) rules, recording for example morphological variation or syntacto-semantic category changes.[8]

We experiment with preserving unary rules in ERG derivations or removing them (as they make no difference to the final DT analysis); we further run experiments using the native ('long') ERG construc-

[8] Examples of morphological rules in Figure 1 include v_pas_odlr and v_n3s-bse_ilr, for passive-participle and non-third person singular or base inflection, respectively. Also, there are two instances of bare noun phrase formation: hdn_bnp-pn_c and hdn_bnp-qnt_c.

tion identifiers, their generalizations to 'short' labels as used in DT, and a variant with long labels for unary and short ones for branching rules ('mixed'). We report results for training with five or six split–merge cycles, where fewer iterations generally show inferior accuracy, and larger values lead to more parse failures ('gaps' in Table 3). There are some noticeable trade-offs across tagging accuracy, dependency accuracy, and coverage, without a single best performer along all three dimensions. As our primary interest across parsers is dependency accuracy, we select the configuration with unary rules and long labels, trained with five split–merge cycles, which seems to afford near-premium LAS at near-perfect coverage.[9]

6 EVALUATION

Standard evaluation metrics in dependency parsing are labeled and unlabeled attachment scores (LAS, UAS; implemented by the CoNLL eval.pl scorer). These measure the percentage of tokens which are correctly attached to their head token and, for LAS, have the right dependency label. As assignment of lexical categories is a core part of syntactic analysis, we complement LAS and UAS with tagging accuracy scores (TA), where appropriate. However, in our work there are two complications to consider when using eval.pl. First, some of our parsers occasionally fail to return any analysis, notably Berkeley and ERG$_e$. For these inputs, our evaluation re-inserts the missing tokens in the parser output, padding with dummy 'placeholder' heads and dependency labels.

Second, a more difficult issue is caused by occasional tokenization mismatches in ERG parses, as discussed above. Since eval.pl identifies tokens by their position in the sentence, any difference of tokenization will lead to invalid results. One option would be to treat all system outputs with token mismatches as parse failures, but this over-penalizes, as potentially correct dependencies among corresponding tokens are also removed from the parser output. For this reason, we modify the evaluation of dependency accuracy to use character offsets, instead of consecutive identifiers, to encode token identities. This way, tokenization mismatches local to some sub-segment of the input will not 'throw

[9] A welcome side-effect of this choice is that we end up using native ERG derivations without modifications.

off' token correspondences in other parts of the string.[10] We will refer to this character-based variant of the standard CoNLL metrics as LAS_c and UAS_c.

7 IN-DOMAIN PARSING RESULTS

Our first cross-paradigm comparison of the three parsers is against the WSJ in-domain test data, as summarized in Table 4. There are substantive differences between parsers both in terms of coverage, speed, and accuracy. Berkeley fails to return an analysis for one input, whereas ERG_e cannot parse 13 sentences (close to one percent of the test set); just as the 44 inputs where parser output deviates in tokenization from the treebank, this is likely an effect of the lexical pruning applied in this setup. At an average of one second per input, Berkeley is the fastest of our parsers; ERG_a is exactly one order of magnitude slower. However, the lexical pruning of Dridan (2013) in ERG_e leads to a speed-up of almost a factor of six, making this variant of PET perform comparable to B&N. The strongest differences, however, we observe in tagging and dependency accuracies. The two data-driven parsers perform very similarly (at close to 93% TA and around 86.7% LAS); the two ERG parsers are comparable too, but at accuracy levels that are four to six points higher in both TA and LAS. Compared to ERG_a, the faster ERG_e variant performs very slightly worse – which likely reflects penalization for missing coverage and token mismatches – but it nevertheless delivers much higher accuracy than the data-driven parsers.

	Gaps	Time	TA_c	LAS_c	UAS_c
Berkeley	1+0	**1.0**	92.9	86.65	89.86
B&N	**0+0**	1.7	92.9	86.76	89.65
ERG_a	**0+0**	10	**97.8**	**92.87**	**93.95**
ERG_e	13+44	1.8	96.4	91.60	92.72

Table 4:
Parse failures and token mismatches ('gaps'), efficiency, and tagging and dependency accuracy on WSJ

[10] Where tokenization is identical for the gold and system outputs, the score given by this generalized metric is exactly the same as that of eval.pl. Unless indicated otherwise, punctuation marks are included in scoring.

8 CROSS-DOMAIN PARSING RESULTS

To gauge the resilience of the different systems to domain and genre variation, we apply the same set of parsers – without re-training or other adaptation – to the additional Redwoods test data. Table 5 summarizes coverage and accuracy results across the four diverse samples. Again, Berkeley and B&N pattern alike, with Berkeley slightly ahead in terms of dependency accuracy, but penalized on two of the test sets for parse failures. LAS for the two data-driven parsers ranges between 74% and 81%, up to 12 points below their WSJ performance. Though large, accuracy drops on a similar scale have been observed repeatedly for purely statistical systems when moving out of the WSJ domain without adaptation (Gildea 2001; Nivre *et al.* 2007). In contrast, ERG_e performance is more similar to WSJ results, with a maximum LAS drop of less than two points.[11] For Wikipedia text (WS; previously unseendata for the ERG, just as for the other two), for example, both tagging and dependency accuracies are around ten points higher, an error reduction of more than 50%. From these results, it is evident that the general linguistic knowledge available in ERG parsing makes it far more resilient to variation in domain and text type.

9 ERROR ANALYSIS

The ERG parsers outperform the two data-driven parsers on the WSJ and cross-domain data. Through in-depth error analysis, we seek to identify parser-specific properties that can explain the observed differences. In the following, we look at (a) the accuracy of individual dependency types, (b) dependency accuracy relative to (predicted and gold) dependency length, and (c) the distribution of LAS over different lexical categories.

Among the different dependency types, we observe that the notion of an adjunct is difficult for all three parsers. One of the hardest

[11] It must be noted that, unlike the WSJ test data, some of these cross-domain data sets have been used in ERG development throughout the years, notably VM and CB, and thus the grammar is likely to have particularly good linguistic coverage of this data. Conversely, SC has hardly had a role in grammar engineering so far, and WS is genuinely unseen (for the ERG and Redwoods release used here), i.e. treebankers were first exposed to it once the grammar and parser were frozen.

		Gaps	TA_c	LAS_c	UAS_c
CB	Berkeley	1+0	87.1	78.13	83.14
	B&N	0+0	87.06	77.70	82.96
	ERG_a	0+4	96.3	90.77	92.47
	ERG_e	8+8	95.3	90.02	91.58
SC	Berkeley	1+0	87.2	79.81	85.10
	B&N	0+0	85.9	78.08	83.21
	ERG_a	0+0	96.1	90.84	92.65
	ERG_e	11+7	94.9	89.49	91.26
VM	Berkeley	7+0	84.0	74.40	83.38
	B&N	0+0	83.1	75.28	82.86
	ERG_a	0+40	94.3	90.44	92.27
	ERG_e	11+42	94.4	90.18	91.75
WS	Berkeley	7+0	87.7	80.31	85.11
	B&N	0+0	88.4	80.63	85.24
	ERG_a	0+0	97.5	91.33	92.48
	ERG_e	4+12	96.9	90.64	91.76

Table 5:
Cross-domain coverage gaps (parse failures and token mismatches) and tagging and dependency accuracies

dependency labels across domains is hdn-aj (post-adjunction to a nominal head), the relation employed for relative clauses and prepositional phrases attaching to a nominal head. The most common error for this relation is verbal attachment.

It has been noted that dependency parsers may exhibit systematic performance differences with respect to dependency length (i.e. the distance between a head and its argument; McDonald and Nivre 2007). In our experiments, we find that the parsers perform comparably on longer dependency arcs (upwards of fifteen words), with ERG_a constantly showing the highest accuracy, and Berkeley holding a slight edge over B&N as dependency length increases.

In Figure 3, one can eyeball frequency and accuracy levels per lexical category on WSJ. The cross-domain picture is similar to the in-domain one, but the difference between accuracy for PET and the data-driven parsers on adjectives (aj), adverbs (av), and conjunctions (c) is more pronounced on the out-of-domain data. Determiners (d) and complimentizers (cm) are similar, while conjunctions (c) and various types of prepositions (p and pp) are the most difficult for all three parsers.

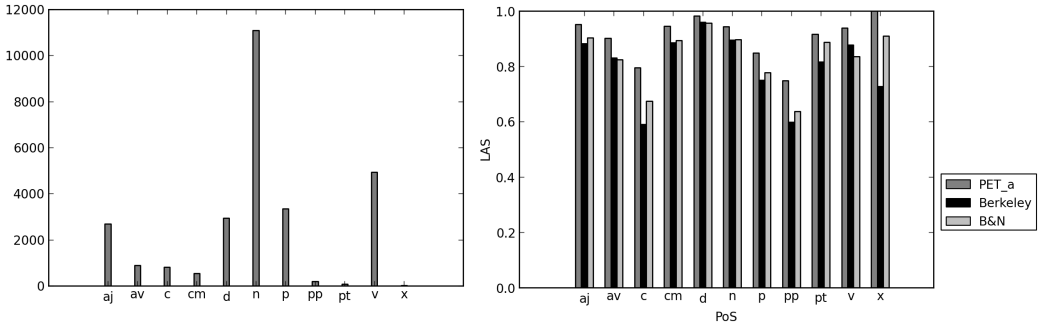


Figure 3: WSJ per-category frequency (left) and dependency accuracies (right) on coarse lexical head categories: adjective, adverb, conjunction, complementizer, determiner, noun, preposition, lexical prepositional phrase, punctuation, verb, and others

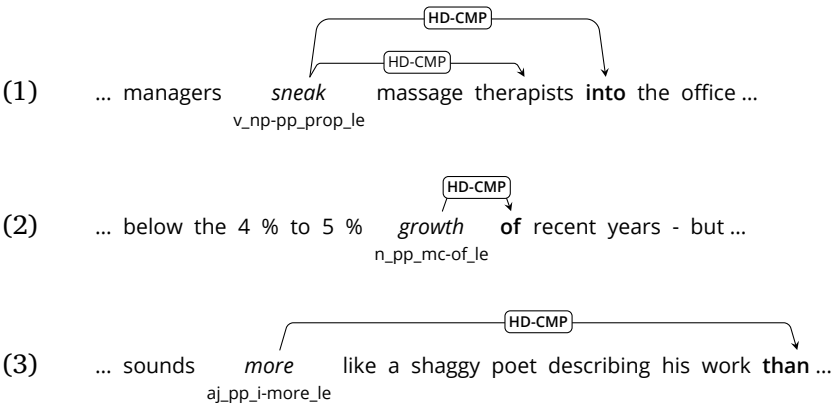
It is unsurprising that the DT analysis of coordination is challenging. Schwartz *et al.* (2012) show that choosing conjunctions as heads in coordinate structures is harder to parse for direct dependency parsers (while this analysis also is linguistically more expressive). Our results confirm this effect also for the PCFG parser and (though to a lesser degree) for ERG_a. At the same time, conjunctions are among the lexical categories for which ERG_a most clearly outperforms the other parsers. Berkeley and B&N exhibit LAS error rates of around 35–41% for conjunctions, whereas the ERG_a error rate is below 20%. For many of the coordinate structures parsed correctly by ERG_a but not the other two, we find that attachment to root constitutes the most frequent error type – indicating that clausal coordination is particularly difficult for the data-driven parsers.

The attachment of prepositions constitutes a notorious difficulty in syntactic analysis. Unlike 'standard' PoS tag sets, ERG lexical types provide a more fine-grained analysis of prepositions, for example recognizing a lexicalized PP like *in full*, or making explicit the distinction between semantically contentful vs. vacuous prepositions. In our error analysis, we find that parser performance varies a lot across the various prepositional sub-types. For some prepositions, all parsers perform comparatively well; e.g. p_np_ptcl-of_le, for semantically vacuous *of*, ranks among the twenty most accurate lexical categories across the board. Other types of prepositions are among the categories exhibiting the highest error rates, e.g. p_np_i_le for 'common' prepositions, taking

an NP argument and projecting intersective modifier semantics. Even so, Figure 3 shows that the attachment of prepositions (p and pp) is an area where ERG$_a$ excels most markedly. Three frequent prepositional lexical types that show the largest ERG$_a$ advantages are p_np_ptcl-of_le (*history of Linux*), p_np_ptcl_le (*look for peace*), and p_np_i_le (*talk about friends*).

Looking more closely at inputs where the parsers disagree, they largely involve (usages of) prepositions which are lexically selected for by their head. ERG lexical rules, which function as lexical types in DT, encode valency information in the form of an ordered sequence of complements for the given type. For example, v_np-pp_prop_le is a verb that requires two complements: a noun phrase and a prepositional phrase (see example (1)).

We analyze parse errors on prepositional complements for heads of various lexical types, including the most frequent verbs, nouns, and adjectives, illustrated by (1), (2), and (3). Example (1) depicts the analysis of the argument structure of such a verb (*sneak*) with a noun phrase and a prepositional phrase. Both B&N and Berkeley incorrectly define the head of the phrase *into the office* as the noun *therapists*, while ERG$_a$ delivers the parse tree that corresponds to the gold standard. In example (2) ERG$_a$ correctly identifies *growth* as the head of the prepositional phrase *of recent years* while B&N attaches *of* to the cardinal *4* and Berkeley to the conjunction *but* with erroneous dependency labels. In example (3), ERG$_a$ correctly analyzes the prepositional complement, and B&N and Berkeley predict the proper label, but wrongly assign attachment to the noun *work* and verb *sounds*, respectively.



In most cases the lexical category of the head explicitly shows the requirement of a prepositional complement; taking advantage of this rule, ERG$_a$ consistently outperforms other parsers in- and cross-domain as depicted in Table 6 which shows the number of total and correct analyses of prepositional complement structures.

Table 6: Number of total and correct analyses of prepositional complement structures

domain	total	ERG$_a$	Berkeley	B&N
WSJ	940	**905**	778	799
CB	469	**446**	348	354
SC	602	**553**	471	454
VM	164	**142**	113	119
WS	372	**361**	293	289

Most prepositions in isolation are ambiguous lexical items. However, it appears that lexical information about the argument structure of heads encoded in the grammar allows ERG$_a$ to analyze these prepositions (in context) much more accurately.

10 SANITY: PTB TOKENIZATION AND POS TAGS

Up to this point, we have applied the two data-driven parsers in a setup that one might consider somewhat 'off-road'; although our experiments are on English, they involve unusual tokenization and lexical categories. For example, the ERG treatment of punctuation as 'pseudo-affixes' increases vocabulary size, which PET may be better equipped to handle due to its integrated treatment of morphological variation. In these experiments, we seek to isolate the effects of tokenization conventions and granularity of lexical categories, taking advantage of optional output flexibility in the DT converter of Ivanova *et al.* (2012).[12] Table 7 confirms that tokenization does make a difference. In combination with fine-grained lexical categories still, B&N obtains LAS gains of two to three points, compared to smaller gains (around or below one point) for ERG$_e$.[13] However, in this setup our

[12] As mapping from ERG derivations into PTB-style tokens and PoS tags is applied when converting to bi-lexical dependencies, we cannot easily include Berkeley in these final experiments.

[13] When converting to PTB-style tokenization, punctuation marks are always attached low in the DT scheme, to the immediately preceding or following to-

two earlier observations still hold true: ERG$_e$ is substantially more accurate within the WSJ domain and far more resilient to domain and genre variation. When we simplify the syntactic analysis task and train and test B&N on coarse-grained PTB PoS tags only, in-domain differences between the two parsers are further reduced (to 0.8 points), but ERG$_e$ still delivers an error reduction of ten percent compared to B&N. The picture in the cross-domain comparison is not qualitatively different, also in this simpler parsing task, with ERG$_e$ maintaining accuracy levels comparable to WSJ, while B&N accuracies degrade markedly.

		Gaps	Lexical Types		PTB PoS Tags	
			LAS$_c$	UAS$_c$	LAS$_c$	UAS$_c$
WSJ	B&N	0+0	88.78	91.52	91.56	93.63
	ERG$_e$	13+9	92.38	93.53	92.38	93.53
CB	B&N	0+0	81.56	86.18	84.54	88.53
	ERG$_e$	8+4	90.77	92.21	90.77	92.21
SC	B&N	0+0	81.69	86.11	85.17	88.85
	ERG$_e$	11+0	90.13	91.86	90.13	91.86
VM	B&N	0+0	77.00	83.73	82.76	88.11
	ERG$_e$	10+0	91.55	93.08	91.55	93.08
WS	B&N	0+0	82.09	86.17	84.59	88.41
	ERG$_e$	4+0	91.61	92.62	91.61	92.62

Table 7:
Coverage and dependency accuracies with PTB tokenization and either detailed or coarse lexical categories

11 FIRST EXTRINSIC EVALUATION: NEGATION SCOPE RESOLUTION

One reason for using a representation format like DT is to make it easy to use parsing results in a downstream application, since parsing is rarely the final goal. In order to test the suitability of DT and also explore the effects that improved parser accuracy may have in such a downstream application, we embed our different parsing setups in an extrinsic evaluation scenario.

Elming *et al.* (2013) try a number of different tasks to explore the effects of different dependency representation formats. They find the

ken, effectively adding a large group of 'easy' dependencies. However, results of evaluation excluding punctuation tokens are not qualitatively different.

negation resolution task (Morante and Blanco 2012) to be the most sensitive to changes in dependency format, and so we chose that as our first external task.

11.1 *Negation resolution task*

Negation resolution (NR) is the task of determining negation cues, i.e. morphemes, words or a combination of words that express negation (for example, *un-*, *no*, *by no means*), scopes of negation, i.e. parts of a sentence that are affected by a negation cue, and negated events, i.e. semantically negated eventualities inside scopes in factual sentences. We employ the NR system of Lapponi *et al.* (2012a), one of the best performing systems of the 2012 Computational Semantics (*SEM) Shared Task (Morante and Blanco 2012) which uses a CRF model for scope resolution relying on dependency features. The dataset for the 2012 *SEM shared task comprises negation annotated stories of Conan Doyle: a training set of 3644 sentences, a development set of 787 sentences, and a test set of 1089 sentences. One example from the training set is presented in (4) below. The cue is typeset in small caps, its scope italicized, and the negated event underlined.

(4) *I* therefore spent the day at my club and *did* NOT <u>return</u> *to Baker Street until evening.*

Note that this negation scope is discontinuous, which shows that proximity to a negation cue is not always a good strategy to assign tokens to scopes; here the subject (*I*) at the beginning of the sentence is a part of the clause negated by the cue in the coordinate sentence and should be retrieved.

For the evaluation we use software developed by the 2012 *SEM Shared Task organizers which reports the following metrics (Morante and Blanco 2012):

Cues Cue F_1-measure.

Scopes Scope-level F_1-measure.

Negated F_1-measure over negated events, computed independently from cues and from scopes

Global Global F_1-measure of negation: the three elements of the negation – cue, scope, and negated event – all have to be correctly identified (strict match)

11.2 *Format comparison*

Table 8 presents evaluation of performance of the NR system relying on dependency features from the analyses of the B&N parser with the three dependency formats we tested in Section 3: CoNLL, Stanford Basic, and DT dependencies. As Elming *et al.* (2013) saw, we get quite a range of performance across the three formats, particularly considering Table 1 showed that intrinsic parse accuracy is basically identical.

	CoNLL	Stanford	DT
Scopes	79.57	**81.69**	80.43
Negated	**75.96**	71.15	73.33
Global	**65.89**	63.78	**65.89**

Table 8:
Performance of the NR system with gold cues on the Conan Doyle development set for three dependency formats using the B&N parser

Table 9 reproduces the numbers Elming *et al.* (2013) reported, using dependency formats that varied more than ours do. While these numbers are not directly comparable to our work due to some differences in the data sets for training parsing models, DT is well within their range of variation, and as such, seems a reasonable format for the task.

	Yamada	CoNLL-07	EWT	LTH
Scopes	**81.27**	80.43	78.70	79.57
Negated	76.19	72.90	73.15	**76.24**
Global	**67.94**	63.24	61.60	64.31

Table 9:
Performance of the NR system with gold cues on the Conan Doyle development set for different dependency formats using the Mate parser, reproduced from Elming *et al.*

11.3 *Parser comparison*

To see if the intrinsic parser accuracy differences we saw earlier translate to better negation resolution, we use the PET and B&N parsers to produce DT dependencies for our NR system.

Intrinsic parser evaluation on the 91 manually annotated sentences taken from the story *Wisteria Lodge*, a subset of Conan Doyle development data, is shown in Table 10. Since negation resolution system uses PTB tokenization with PTB PoS tags, we again cannot include Berkeley in this comparison. The Conan Doyle domain is genuinely new for the ERG as it was not available before the release of

Table 10: Parse failures and token mismatches ('gaps'), and tagging and dependency accuracy on the subset of the Conan Doyle development data

	Gaps	TA_c	LAS_c	UAS_c
B&N	0+0	92.24	83.92	87.92
ERG_a	0+0	**96.36**	**92.54**	**93.84**
ERG_e	0+3	94.21	89.22	90.57

version 1212, used in the present work. Consistent with our expectations, results are similar to the cross-domain evaluation in Table 7.

While B&N has complete coverage on the full Conan Doyle corpus, Table 11 shows that both of our PET variants sometimes fail to produce an analysis, especially the ERG_e variant due to excessive pruning. In addition, PET does not always land on the gold-standard tokenization as the parsing process starts from the raw text. Due to this, we fall back on the B&N parser for the sentences that lack syntactic analysis in the negation resolution experiments with PET; e.g. for the experiments with ERG_a the training set consists of 89.24% PET analyses and 10.76% analyses from B&N.

Table 11: PET coverage on Conan Doyle and alignment with 'gold' tokenization

	ERG_a			ERG_e		
	Train	Dev	Test	Train	Dev	Test
% Coverage	89.96	91.11	87.42	81.64	83.99	79.98
% Alignment	89.24	91.11	86.23	80.98	83.86	78.88

Tables 12 and 13 show the results of the NR system on the development and test sets, respectively. The results from the original system using the Malt parser and Stanford Basic dependencies are shown for comparison Lapponi *et al.* (2012b). Somewhat surprisingly, the reasonably large differences in parser accuracy seen in Table 7 are not reflected in the task performance. Statistical significance testing using the paired, two-tailed formulation of the sign test shows that none of the performance differences are actually significant.

Table 12: Performance of the negation resolution system on the development set with gold cues

	ERG_a	ERG_e	B&N	Malt
Scopes	80.00	**80.85**	80.43	80.00
Negated	**75.73**	73.33	73.33	80.55
Global	64.31	63.24	**65.89**	66.41

	ERG$_a$	ERG$_e$	B&N	Malt
Cues	91.31	91.31	91.31	91.31
Scopes	73.52	74.83	**75.40**	72.39
Negated	**61.29**	60.95	60.44	61.79
Global	53.73	**55.53**	55.17	54.82

Table 13:
Performance of the negation resolution system on the test set with predicted cues

It is possible that this negation resolution task is not sensitive enough to parser performance to be a useful extrinsic parser evaluation. There is a reasonable body of previous work (Mollá and Hutchinson 2003; Miyao *et al.* 2008; Miwa *et al.* 2010) that has shown that many tasks such as answer extraction, protein-protein interaction (PPI) extraction, and event extraction are relatively insensitive to parser accuracy. It is possible that negation resolution, at least in this particular setup, is another such task.

12 SECOND EXTRINSIC EVALUATION: SEMANTIC DEPENDENCY PARSING

As another downstream application for extrinsic evaluation, we explore the task of Broad-Coverage Semantic Dependency Parsing (SDP; Oepen *et al.* 2014, 2015), which was part of the 2014 and 2015 Semantic Evaluation Exercises (SemEval). We re-train and evaluate the best-performing system from the SDP 2014 open track, called Priberam (Martins and Almeida 2014), which is based on a feature-rich model that takes advantage of the information from the syntactic dependency parser. For this second extrinsic evaluation, we test whether syntactic dependency features provided by the grammar-based system facilitate more accurate semantic parsing than features delivered by the data-driven tools.

12.1 *Broad-coverage semantic dependency parsing*

Oepen *et al.* (2014) define semantic dependency parsing (SDP) as the problem of recovering sentence-internal predicate-argument relationships for all content words. Thus, target representations are semantic in nature (rather than directly representing grammatical structure), and in contrast to syntactic parsing the SDP semantic dependencies

are general (directed and acyclic) graphs rather than trees, and need not span input tokens that are analyzed as semantically vacuous.

The SDP 2014 data consists of Sections 0–21 of the WSJ Corpus annotated with three target representations called DM, PAS, and PCEDT (which are all aligned at the sentence and token levels). DM is the result of a two-step reduction of the underspecified logical-form meaning representations produced by the ERG to pure bi-lexical dependencies (Oepen and Lønning 2006; Ivanova *et al.* 2012), as exemplified for our running example in Figure 4. PAS dependencies are predicate–argument structures produced by the Enju system, a statistical HPSG parser obtained by learning from a conversion of the (full) PTB annotations (Miyao and Tsujii 2008). PCEDT dependencies, in turn, are extracted from the tectogrammatical analysis layer of the Prague Czech–English Dependency Treebank (Hajič *et al.* 2012).

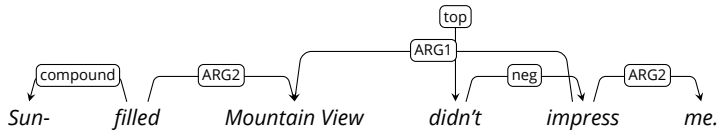


Figure 4: DM bi-lexical semantic dependencies for our running example

The task is organized into two tracks: systems in the *closed* track were trained only on the data distributed by the task organizers while the systems in the *open* track could use additional resources. We are, therefore, only interested in the latter track. In the open track of the SDP 2014 task, participants had been offered syntactic 'companion' files with Stanford dependencies produced by the parser of Bohnet and Nivre (2012). Evaluation measures are labeled precision (LP), labeled recall (LR), labeled F_1 (LF), and labeled exact match (LM) with respect to predicted $\langle predicate, role, argument \rangle$ triples.

The Priberam system (Martins and Almeida 2014), which relies on a model with second-order features and decoding with dual decomposition, was ranked first in the SDP 2014 open track, and achieved the second highest score in the closed track. By virtue of syntactic features extracted from the output of a syntactic dependency parser, Priberam attained an improvement of around 1% in LF for all three dependency formats. We have chosen this system for extrinsic evaluation.

12.2 *Parser comparison*

We compare the quality of syntactic features produced by ERG$_a$, ERG$_e$ and B&N for the semantic parsing with Priberam. Using these three parsers we prepare alternate companion files containing DT bi-lexical dependencies. Of the 1348 sentences in the SDP test set, ERG$_a$ and ERG$_e$ fail to deliver analysis for 11 and 24 sentences, respectively; thus, we 'borrow' the missing analyses from B&N outputs, much like we did in Section 11 above.

Tables 14, 15, and 16 present SDP results for DM, PAS, and PCEDT, respectively. For comparison, we include results of Priberam from the SDP 2014 task with the original companion file generated by task organizers. Compared to the original SDP 2014 results, moving from Stanford to DT dependencies (derived by B&N in both cases) appears to only have a small effect on semantic dependency parsing. Our re-trained version of Priberam with the DT syntactic companion performs marginally below the published SDP 2014 results for

	ERG$_a$	ERG$_e$	B&N	SDP 2014
LP	**90.88**	90.77	88.96	90.23
LR	**89.86**	89.67	88.10	88.11
LF	**90.37**	90.22	88.53	89.16
LM	**32.42**	32.64	29.75	26.85

Table 14:
SDP open track results on DM

	ERG$_a$	ERG$_e$	B&N	SDP 2014
LP	92.04	**92.19**	91.91	92.56
LR	89.67	**89.89**	89.63	90.97
LF	90.84	**91.03**	90.75	91.76
LM	31.38	30.93	**32.86**	37.83

Table 15:
SDP open track results on PAS

	ERG$_a$	ERG$_e$	B&N	SDP 2014
LP	79.62	**79.94**	79.42	80.14
LR	75.67	**75.82**	75.45	75.79
LF	77.59	**77.82**	77.38	77.90
LM	11.05	**11.20**	10.98	10.68

Table 16:
SDP open track results on PCEDT

DM and PCEDT, whereas for PAS there is a more pronounced drop in semantic dependency LF when replacing Stanford with DT dependencies. Different syntactic accuracy levels of our three DT parsers, on the other hand, actually do project into downstream differences in semantic dependency quality: For all three target representations, the ERG parsers yield higher semantic dependency LF than B&N. The differences are comparatively small for the PAS and PCEDT targets, but for DM there is a large benefit in deriving the (more accurate) DT syntactic companion from ERG_a rather than from B&N. Seeing as DM and DT both originate from DeepBank, while PAS as well as Stanford dependencies originate from the PTB, our results suggest that it is beneficial for the semantic dependency parsing task to rely on 'correlated' syntactic dependency features: The overall best-performing SDP pipeline for the DM target representation uses DT dependencies (from ERG_a), but the best PAS results are obtained with Stanford syntactic dependencies (from B&N).

In conclusion, the results of this second extrinsic evaluation suggest that semantic dependency parsing is more sensitive to syntactic parser performance than negation resolution, especially when taking into account that the maximum in-domain difference between ERG_e and B&N observed in Table 7 is 0.82% in LAS_c when using PTB tokenization and PTB PoS tags (as is also the case in the SDP 2014 task).

13 DISCUSSION AND CONCLUSION

Our experiments have sought to contrast state-of-the-art representatives from three parsing paradigms on the task of producing bi-lexical syntactic dependencies for English. For the HPSG-derived DT scheme, we find that hybrid, grammar-driven parsing yields superior accuracy, both in- and in particular cross-domain, at processing times comparable to the currently best direct dependency parser; the grammar-driven parser in our experiments, however, fails to parse a small percentage of inputs in naturally occurring text. These results corroborate the Dutch findings of Plank and van Noord (2010) for English, where more training data is available, and in comparison to more advanced data-driven parsers. Extrinsic evaluation on semantic dependency parsing correlates with the results of the in-domain intrinsic evaluation. However, we do not find that this superior accuracy is reflected in superior ac-

curacy in the negation resolution task. In most of this work, we have focussed exclusively on parser inputs represented in the DeepBank and Redwoods treebanks, ignoring 15 percent of the original running text, for which the ERG and PET do not make available a gold-standard analysis. While a parser with partial coverage can be useful in some contexts, obviously the data-driven parsers must be credited for providing a syntactic analysis of (almost) all inputs. However, the ERG coverage gap can be straightforwardly addressed by falling back to another parser when necessary, as we did in our extrinsic evaluations. Such a system combination should yield better tagging and dependency accuracies than the data-driven parsers by themselves, especially so in an open-domain setup. A secondary finding from our experiments is that PCFG parsing with Berkeley and conversion to DT dependencies yields equivalent or mildly more accurate analyses, at much greater efficiency. In future work, it would be interesting to include in this comparison other PCFG parsers and linear-time, transition-based dependency parsers, but a tentative generalization over our findings to date is that linguistically richer representations enable more accurate parsing. It would also be informative to try a wider variety of downstream tasks to see which are sensitive to parser accuracy, as opposed to dependency representation.

ACKNOWLEDGMENTS

We are grateful to our colleagues Emily M. Bender, Bernd Bohnet, Francis Bond, Rui Wang, Yi Zhang, and André Martins for many helpful discussions, suggestions, and assistance in running third-party tools, as well as to our three anonymous reviewers for insightful comments. This work was in part funded by the Norwegian Research Council through its WeSearch project. Large-scale experimentation is made possible through access to the ABEL high-performance computing facilities at the University of Oslo, and we are grateful to the Scientific Computing staff at UiO, as well as to the Norwegian Metacenter for Computational Science, and the Norwegian tax payer.

REFERENCES

Peter ADOLPHS, Stephan OEPEN, Ulrich CALLMEIER, Berthold CRYSMANN, Dan FLICKINGER, and Bernd KIEFER (2008), Some Fine Points of Hybrid Natural Language Parsing, in *Proceedings of the 6th International Conference on Language Resources and Evaluation*, Marrakech, Morocco.

Ezra BLACK, Steve ABNEY, Dan FLICKINGER, Claudia GDANIEC, Ralph GRISHMAN, Phil HARRISON, Don HINDLE, Robert INGRIA, Fred JELINEK, Judith KLAVANS, Mark LIBERMAN, Mitch MARCUS, S. ROUKOS, Beatrice SANTORINI, and Tomek STRZALKOWSKI (1991), A Procedure for Quantitatively Comparing the Syntactic Coverage of English Grammars, in *Proceedings of the Workshop on Speech and Natural Language*, pp. 306–311, Pacific Grove, USA.

Bernd BOHNET and Joakim NIVRE (2012), A Transition-Based System for Joint Part-of-Speech Tagging and Labeled Non-Projective Dependency Parsing, in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Conference on Natural Language Learning*, pp. 1455–1465, Jeju Island, Korea.

Johan BOS, Edward BRISCOE, Aoife CAHILL, John CARROLL, Stephen CLARK, Ann COPESTAKE, Dan FLICKINGER, Josef VAN GENABITH, Julia HOCKENMAIER, Aravind JOSHI, Ronald KAPLAN, Tracy Holloway KING, Sandra KUEBLER, Dekang LIN, Jan Tore LØNNING, Christopher MANNING, Yusuke MIYAO, Joakim NIVRE, Stephan OEPEN, Kenji SAGAE, Nianwen XUE, and Yi ZHANG, editors (2008), *Workshop on Cross-Framework and Cross-Domain Parser Evaluation*, Manchester, UK.

Ted BRISCOE and John CARROLL (2006), Evaluating the Accuracy of an Unlexicalised Statistical Parser on the PARC DepBank, in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Meeting of the Association for Computational Linguistics*, pp. 41–48, Sydney, Australia.

Ulrich CALLMEIER (2002), Preprocessing and Encoding Techniques in PET, in Stephan OEPEN, Daniel FLICKINGER, J. TSUJII, and Hans USZKOREIT, editors, *Collaborative Language Engineering. A Case Study in Efficient Grammar-Based Processing*, pp. 127–140, CSLI Publications, Stanford, CA.

David CARTER (1997), The TreeBanker. A Tool for Supervised Training of Parsed Corpora, in *Proceedings of the Workshop on Computational Environments for Grammar Development and Linguistic Engineering*, pp. 9–15, Madrid, Spain.

Daniel CER, Marie-Catherine DE MARNEFFE, Dan JURAFSKY, and Chris MANNING (2010), Parsing to Stanford Dependencies. Trade-Offs between Speed and Accuracy, in *Proceedings of the 7th International Conference on Language Resources and Evaluation*, pp. 1628–1632, Valletta, Malta.

Stephen CLARK and James R. CURRAN (2007), Formalism-Independent Parser Evaluation with CCG and DepBank, in *Proceedings of the 45th Meeting of the Association for Computational Linguistics*, pp. 248–255, Prague, Czech Republic.

Marie-Catherine DE MARNEFFE and Christopher D. MANNING (2008), The Stanford Typed Dependencies Representation, in *Proceedings of the COLING Workshop on Cross-Framework and Cross-Domain Parser Evaluation*, pp. 1–8, Manchester, UK.

Rebecca DRIDAN (2013), Ubertagging. Joint Segmentation and Supertagging for English, in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1–10, Seattle, WA, USA.

Jacob ELMING, Anders JOHANNSEN, Sigrid KLERKE, Emanuele LAPPONI, Hector MARTINEZ, and Anders SØGAARD (2013), Down-Stream Effects of Tree-to-Dependency Conversions, in *Proceedings of Human Language Technologies: The 2013 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 617–626, Atlanta, GA, USA.

Dan FLICKINGER (2000), On Building a More Efficient Grammar by Exploiting Types, *Natural Language Engineering*, 6 (1):15–28.

Dan FLICKINGER, Yi ZHANG, and Valia KORDONI (2012), DeepBank. A Dynamically Annotated Treebank of the Wall Street Journal, in *Proceedings of the 11th International Workshop on Treebanks and Linguistic Theories*, pp. 85–96, Edições Colibri, Lisbon, Portugal.

Jennifer FOSTER, Ozlem CETINOGLU, Joachim WAGNER, Joseph LE ROUX, Joakim NIVRE, Deirdre HOGAN, and Josef VAN GENABITH (2011), From News to Comment. Resources and Benchmarks for Parsing the Language of Web 2.0, in *Proceedings of the 2011 International Joint Conference on Natural Language Processing*, pp. 893–901, Chiang Mai, Thailand.

Timothy A. D. FOWLER and Gerald PENN (2010), Accurate Context-Free Parsing with Combinatory Categorial Grammar, in *Proceedings of the 48th Meeting of the Association for Computational Linguistics*, pp. 335–344, Uppsala, Sweden.

W. Nelson FRANCIS and Henry KUČERA (1982), *Frequency Analysis of English Usage. Lexicon and Grammar*, Houghton Mifflin, New York, USA.

Daniel GILDEA (2001), Corpus Variation and Parser Performance, in *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, pp. 167–202, Pittsburgh, USA.

Jan HAJIČ, Eva HAJIČOVÁ, Jarmila PANEVOVÁ, Petr SGALL, Ondřej BOJAR, Silvie CINKOVÁ, Eva FUČÍKOVÁ, Marie MIKULOVÁ, Petr PAJAS, Jan POPELKA, Jiří SEMECKÝ, Jana ŠINDLEROVÁ, Jan ŠTĚPÁNEK, Josef TOMAN, Zdeňka URešOVÁ, and Zdeněk ŽABOKRTSKÝ (2012), Announcing Prague Czech-English Dependency Treebank 2.0, in *Proceedings of the 8th International*

Conference on Language Resources and Evaluation, pp. 3153–3160, Istanbul, Turkey.

Julia HOCKENMAIER and Mark STEEDMAN (2007), CCGbank. A Corpus of CCG Derivations and Dependency Structures Extracted from the Penn Treebank, *Computational Linguistics*, 33:355–396.

Angelina IVANOVA, Stephan OEPEN, and Lilja ØVRELID (2013), Survey on Parsing Three Dependency Representations for English, in *Proceedings of the 51th Meeting of the Association for Computational Linguistics*, pp. 31–37, Sofia, Bulgaria.

Angelina IVANOVA, Stephan OEPEN, Lilja ØVRELID, and Dan FLICKINGER (2012), Who Did What to Whom? A Contrastive Study of Syntacto-Semantic Dependencies, in *Proceedings of the Sixth Linguistic Annotation Workshop*, pp. 2–11, Jeju, Republic of Korea.

Tracy Holloway KING, Richard CROUCH, Stefan RIEZLER, Mary DALRYMPLE, and Ronald M. KAPLAN (2003), The PARC 700 Dependency Bank, in *Proceedings of the 4th International Workshop on Linguistically Interpreted Corpora*, pp. 1–8, Budapest, Hungary.

Emanuele LAPPONI, Jonathon READ, and Lilja ØVRELID (2012a), Representing and Resolving Negation for Sentiment Analysis, in *Proceedings of the 2012 ICDM Workshop on Sentiment Elicitation from Natural Text for Information Retrieval and Extraction*, Brussels, Belgium.

Emanuele LAPPONI, Erik VELLDAL, Lilja ØVRELID, and Jonathon READ (2012b), UiO2: Sequence-Labeling Negation Using Dependency Features, in *Proceedings of the 1st Joint Conference on Lexical and Computational Semantics*, pp. 319–327, Montréal, Canada.

Mitchell MARCUS, Beatrice SANTORINI, and Mary Ann MARCINKIEWICZ (1993), Building a Large Annotated Corpora of English. The Penn Treebank, *Computational Linguistics*, 19:313–330.

T. André F. MARTINS and C. Mariana S. ALMEIDA (2014), Priberam: A Turbo Semantic Parser with Second Order Features, in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pp. 471–476, Association for Computational Linguistics, Dublin, Ireland.

Ryan T. 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 Conference on Natural Language Learning*, pp. 122–131, Prague, Czech Republic.

Makoto MIWA, Sampo PYYSALO, Tadayoshi HARA, and Jun'ichi TSUJII (2010), Evaluating Dependency Representations for Event Extraction, in *Proceedings of the 23rd International Conference on Computational Linguistics*, pp. 779–787.

Yusuke MIYAO, Rune SÆTRE, Kenji SAGAE, Takuya MATSUZAKI, and Jun'ichi TSUJII (2008), Task-Oriented Evaluation of Syntactic Parsers and Their Representations, in *Proceedings of the 46th Meeting of the Association for Computational Linguistics*, pp. 46–54, Columbus, OH, USA.

Yusuke MIYAO, Kenji SAGAE, and Jun'ichi TSUJII (2007), Towards Framework-Independent Evaluation of Deep Linguistic Parsers, in *Proceedings of the 2007 Workshop on Grammar Engineering across Frameworks*, pp. 238–258, Palo Alto, California.

Yusuke MIYAO and Jun'ichi TSUJII (2008), Feature Forest Models for Probabilistic HPSG Parsing, *Computational Linguistics*, 34(1):35–80.

Diego MOLLÁ and Ben HUTCHINSON (2003), Intrinsic Versus Extrinsic Evaluations of Parsing Systems, in *Proceedings of the EACL 2003 Workshop on Evaluation Initiatives in Natural Language Processing: Are Evaluation Methods, Metrics and Resources Reusable?*, pp. 43–50, Budapest, Hungary.

Roser MORANTE and Eduardo BLANCO (2012), *SEM 2012 Shared Task. Resolving the Scope and Focus of Negation, in *Proceedings of the 1st Joint Conference on Lexical and Computational Semantics*, pp. 265–274, Montréal, Canada.

Joakim NIVRE, Johan HALL, Sandra KÜBLER, Ryan MCDONALD, Jens NILSSON, Sebastian RIEDEL, and Deniz YURET (2007), The CoNLL 2007 Shared Task on Dependency Parsing, in *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Conference on Natural Language Learning*, pp. 915–932, Prague, Czech Republic.

Stephan OEPEN and John CARROLL (2000), Ambiguity Packing in Constraint-Based Parsing. Practical Results, in *Proceedings of the 1st Meeting of the North American Chapter of the Association for Computational Linguistics*, pp. 162–169, Seattle, WA, USA.

Stephan OEPEN, Daniel FLICKINGER, Kristina TOUTANOVA, and Christopher D. MANNING (2004), LinGO Redwoods. A Rich and Dynamic Treebank for HPSG, *Research on Language and Computation*, 2(4):575–596.

Stephan OEPEN, Marco KUHLMANN, Yusuke MIYAO, Daniel ZEMAN, Silvie CINKOVA, Dan FLICKINGER, Jan HAJIC, and Zdenka URESOVA (2015), SemEval 2015 Task 18: Broad-Coverage Semantic Dependency Parsing, in *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pp. 915–926, Denver, CO, USA.

Stephan OEPEN, Marco KUHLMANN, Yusuke MIYAO, Daniel ZEMAN, Dan FLICKINGER, Jan HAJIČ, Angelina IVANOVA, and Yi ZHANG (2014), SemEval 2014 Task 8. Broad-Coverage Semantic Dependency Parsing, in *Proceedings of the 8th International Workshop on Semantic Evaluation*, Dublin, Ireland.

Stephan OEPEN and Jan Tore LØNNING (2006), Discriminant-Based MRS Banking, in *Proceedings of the 5th International Conference on Language Resources and Evaluation*, pp. 1250–1255, Genoa, Italy.

Slav PETROV, Leon BARRETT, Romain THIBAUX, and Dan KLEIN (2006), Learning Accurate, Compact, and Interpretable Tree Annotation, in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Meeting of the Association for Computational Linguistics*, pp. 433–440, Sydney, Australia.

Barbara PLANK and Gertjan VAN NOORD (2010), Grammar-Driven versus Data-Driven. Which Parsing System is more Affected by Domain Shifts?, in *Proceedings of the 2010 Workshop on NLP and Linguistics: Finding the Common Ground*, pp. 25–33, Association for Computational Linguistics, Uppsala, Sweden.

Carl POLLARD and Ivan A. SAG (1994), *Head-Driven Phrase Structure Grammar*, Studies in Contemporary Linguistics, The University of Chicago Press, Chicago, USA.

Roy SCHWARTZ, Omri ABEND, and Ari RAPPOPORT (2012), Learnability-Based Syntactic Annotation Design, in *Proceedings of the 24th International Conference on Computational Linguistics*, Mumbai, India.

Wolfgang WAHLSTER, editor (2000), *Verbmobil. Foundations of Speech-to-Speech Translation*, Springer, Berlin, Germany.

Gisle YTRESTØL, Stephan OEPEN, and Dan FLICKINGER (2009), Extracting and Annotating Wikipedia Sub-Domains, in *Proceedings of the 7th International Workshop on Treebanks and Linguistic Theories*, pp. 185–197, Groningen, The Netherlands.

Yi ZHANG and Hans-Ulrich KRIEGER (2011), Large-Scale Corpus-Driven PCFG Approximation of an HPSG, in *Proceedings of the 12th International Conference on Parsing Technologies*, pp. 198–208, Dublin, Ireland.

Yi ZHANG, Stephan OEPEN, and John CARROLL (2007), Efficiency in Unification-Based N-Best Parsing, in *Proceedings of the 10th International Conference on Parsing Technologies*, pp. 48–59, Prague, Czech Republic.

Yi ZHANG and Rui WANG (2009), Cross-Domain Dependency Parsing Using a Deep Linguistic Grammar, in *Proceedings of the 47th Meeting of the Association for Computational Linguistics*, pp. 378–386, Suntec, Singapore.