SemEval 2014 Task 8: Broad-Coverage Semantic Dependency Parsing

Stephan Oepen[♣♠], Marco Kuhlmann[♡], Yusuke Miyao[♢], Daniel Zeman[◦], Dan Flickinger[•], Jan Hajič[◦], Angelina Ivanova[♣], and Yi Zhang[⋆]

[♣] University of Oslo, Department of Informatics

[♠] Potsdam University, Department of Linguistics

[♡] Linköping University, Department of Computer and Information Science

[♢] National Institute of Informatics, Tokyo

[◦] Charles University in Prague, Faculty of Mathematics and Physics, Institute of Formal and Applied Linguistics

[•] Stanford University, Center for the Study of Language and Information

[⋆] Nuance Communications Aachen GmbH

sdp-organizers@emmtee.net

Abstract

Task 8 at SemEval 2014 defines *Broad-Coverage Semantic Dependency Parsing* (SDP) as the problem of recovering sentence-internal predicate–argument relationships for *all content words*, i.e. the semantic structure constituting the relational core of sentence meaning. In this task description, we position the problem in comparison to other sub-tasks in computational language analysis, introduce the semantic dependency target representations used, reflect on high-level commonalities and differences between these representations, and summarize the task setup, participating systems, and main results.

1 Background and Motivation

Syntactic dependency parsing has seen great advances in the past decade, in part owing to relatively broad consensus on target representations, and in part reflecting the successful execution of a series of shared tasks at the annual Conference for Natural Language Learning (CoNLL; Buchholz & Marsi, 2006; Nivre et al., 2007; inter alios). From this very active research area accurate and efficient syntactic parsers have developed for a wide range of natural languages. However, the predominant data structure in dependency parsing to date are *trees*, in the formal sense that every node in the dependency graph is reachable from a distinguished root node by exactly one directed path.

Unfortunately, tree-oriented parsers are ill-suited for producing meaning representations, i.e. moving from the analysis of grammatical structure to sentence semantics. Even if syntactic parsing arguably can be limited to tree structures, this is not the case in semantic analysis, where a node will often be the argument of multiple predicates (i.e. have more than one incoming arc), and it will often be desirable to leave nodes corresponding to semantically vacuous word classes unattached (with no incoming arcs).

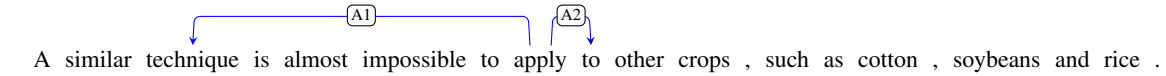
Thus, Task 8 at SemEval 2014, *Broad-Coverage Semantic Dependency Parsing* (SDP 2014),[1] seeks to stimulate the dependency parsing community to move towards more general graph processing, to thus enable a more direct analysis of *Who did What to Whom?* For English, there exist several independent annotations of sentence meaning over the venerable Wall Street Journal (WSJ) text of the Penn Treebank (PTB; Marcus et al., 1993). These resources constitute parallel semantic annotations over the same common text, but to date they have not been related to each other and, in fact, have hardly been applied for training and testing of data-driven parsers. In this task, we have used three different such target representations for bi-lexical semantic dependencies, as demonstrated in Figure 1 below for the WSJ sentence:

(1) A similar technique is almost impossible to apply to other crops, such as cotton, soybeans, and rice.

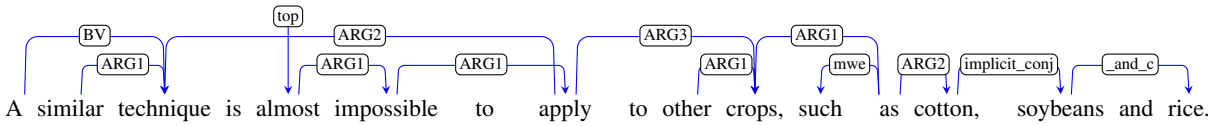
Semantically, *technique* arguably is dependent on the determiner (the quantificational locus), the modifier *similar*, and the predicate *apply*. Conversely, the predicative copula, infinitival *to*, and the vac-

[1] See http://alt.qcri.org/semeval2014/task8/ for further technical details, information on how to obtain the data, and official results.

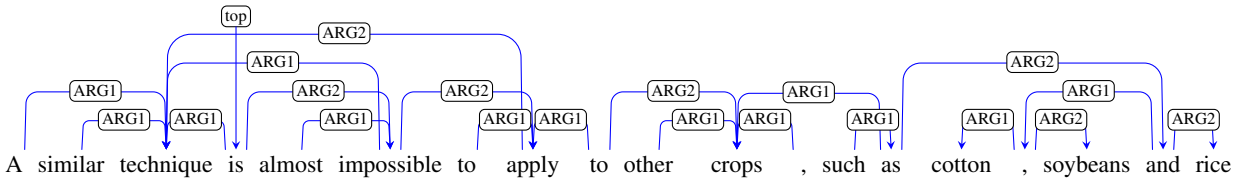
Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), pages 63–72, Dublin, Ireland, August 23-24, 2014.



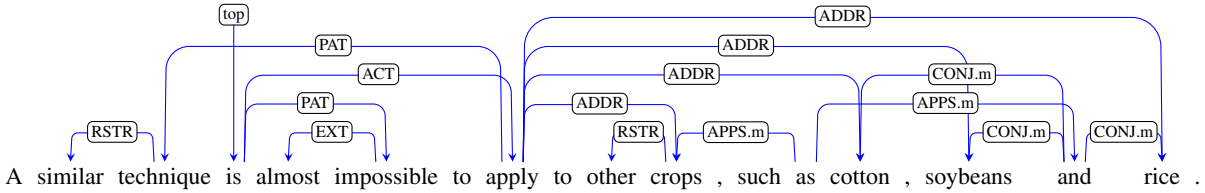
(a) Partial semantic dependencies in PropBank and NomBank.



(b) DELPH-IN Minimal Recursion Semantics–derived bi-lexical dependencies (DM).



(c) Enju Predicate–Argument Structures (PAS).



(d) Parts of the tectogrammatical layer of the Prague Czech-English Dependency Treebank (PCEDT).

Figure 1: Sample semantic dependency graphs for Example (1).

uous preposition marking the deep object of *apply* can be argued to not have a semantic contribution of their own. Besides calling for node reentrancies and partial connectivity, semantic dependency graphs may also exhibit higher degrees of non-projectivity than is typical of syntactic dependency trees.

In addition to its relation to syntactic dependency parsing, the task also has some overlap with Semantic Role Labeling (SRL; Gildea & Jurafsky, 2002). In much previous work, however, target representations typically draw on resources like PropBank and NomBank (Palmer et al., 2005; Meyers et al., 2004), which are limited to argument identification and labeling for verbal and nominal predicates. A plethora of semantic phenomena—for example negation and other scopal embedding, comparatives, possessives, various types of modification, and even conjunction—typically remain unanalyzed in SRL. Thus, its target representations are partial to a degree that can prohibit semantic downstream processing, for example inference-based techniques. In contrast, we require parsers to identify all semantic dependencies, i.e. compute a representation that integrates all content words in one structure. Another difference to common interpretations of SRL is that the SDP 2014 task definition does not encompass predicate disambiguation, a design decision in part owed to our goal to focus on parsing-oriented, i.e. structural, analysis, and in part to lacking consensus on sense inventories for all content words.

Finally, a third closely related area of much current interest is often dubbed ‘semantic parsing’, which Kate and Wong (2010) define as “the task of mapping natural language sentences into complete formal meaning representations which a computer can execute for some domain-specific application.” In contrast to most work in this tradition, our SDP target representations aim to be task- and domain-independent, though at least part of this generality comes at the expense of ‘completeness’ in the above sense; i.e. there are aspects of sentence meaning that arguably remain implicit.

2 Target Representations

We use three distinct target representations for semantic dependencies. As is evident in our running example (Figure 1), showing what are called the DM, PAS, and PCEDT semantic dependencies, there are contentful differences among these annotations, and there is of course not one obvious (or even objective) truth. In the following paragraphs,

we provide some background on the 'pedigree' and linguistic characterization of these representations.

DM: DELPH-IN MRS-Derived Bi-Lexical Dependencies These semantic dependency graphs originate in a manual re-annotation of Sections 00–21 of the WSJ Corpus with syntactico-semantic analyses derived from the LinGO English Resource Grammar (ERG; Flickinger, 2000). Among other layers of linguistic annotation, this resource—dubbed DeepBank by Flickinger et al. (2012)—includes underspecified logical-form meaning representations in the framework of Minimal Recursion Semantics (MRS; Copestake et al., 2005). Our DM target representations are derived through a two-step 'lossy' conversion of MRSs, first to variable-free Elementary Dependency Structures (EDS; Oepen & Lønning, 2006), then to 'pure' bi-lexical form—projecting some construction semantics onto word-to-word dependencies (Ivanova et al., 2012). In preparing our gold-standard DM graphs from DeepBank, the same conversion pipeline was used as in the system submission of Miyao et al. (2014). For this target representation, top nodes designate the highest-scoping (non-quantifier) predicate in the graph, e.g. the (scopal) degree adverb *almost* in Figure 1.[2]

PAS: Enju Predicate-Argument Structures The Enju parsing system is an HPSG-based parser for English.[3] The grammar and the disambiguation model of this parser are derived from the Enju HPSG treebank, which is automatically converted from the phrase structure and predicate–argument structure annotations of the PTB. The PAS data set is extracted from the WSJ portion of the Enju HPSG treebank. While the Enju treebank is annotated with full HPSG-style structures, only its predicate–argument structures are converted into the SDP data format for use in this task. Top nodes in this representation denote semantic heads. Again, the system description of Miyao et al. (2014) provides more technical detail on the conversion.

PCEDT: Prague Tectogrammatical Bi-Lexical Dependencies The Prague Czech-English Dependency Treebank (PCEDT; Hajič et al., 2012)[4] is a set of parallel dependency trees over the WSJ texts from the PTB, and their Czech translations. Similarly to other treebanks in the Prague family, there are two layers of syntactic annotation: *analytical* (a-trees) and *tectogrammatical* (t-trees). PCEDT bi-lexical dependencies in this task have been extracted from the t-trees. The specifics of the PCEDT representations are best observed in the procedure that converts the original PCEDT data to the SDP data format; see Miyao et al. (2014). Top nodes are derived from t-tree roots; i.e. they mostly correspond to main verbs. In case of coordinate clauses, there are multiple top nodes per sentence.

id	*form*	*lemma*	*pos*	*top*	*pred*	*arg1*	*arg2*
#20200002							
1	Ms.	Ms.	NNP	−	+	_	_
2	Haag	Haag	NNP	−	−	compound	ARG1
3	plays	play	VBZ	+	+	_	_
4	Elianti	Elianti	NNP	−	−	_	ARG2
5	.	.	.	−	−	_	_

Table 1: Tabular SDP data format (showing DM).

3 Graph Representation

The SDP target representations can be characterized as labeled, directed graphs. Formally, a *semantic dependency graph* for a sentence $x = x_1, \ldots, x_n$ is a structure $G = (V, E, \ell_V, \ell_E)$ where $V = \{1, \ldots, n\}$ is a set of *nodes* (which are in one-to-one correspondence with the tokens of the sentence); $E \subseteq V \times V$ is a set of *edges*; and ℓ_V and ℓ_E are mappings that assign *labels* (from some finite alphabet) to nodes and edges, respectively. More specifically for this task, the label $\ell_V(i)$ of a node i is a tuple consisting of four components: its word form, lemma, part of speech, and a Boolean flag indicating whether the corresponding token represents a *top* predicate for the specific sentence. The label $\ell_E(i \rightarrow j)$ of an edge $i \rightarrow j$ is a semantic relation that holds between i and j. The exact definition of what constitutes a top node and what semantic relations are available differs among our three target representations, but note that top nodes can have incoming edges.

All data provided for the task uses a column-based file format (dubbed the *SDP data format*) similar to the one of the 2009 CoNLL Shared Task (Hajič et al., 2009). As in that task, we assume gold-standard sentence and token segmentation. For ease of reference, each sentence is prefixed by a line with just a unique identifier, using the scheme *2SSDDIII*, with a constant leading 2, two-digit *section* code, two-digit *document* code (within each

[2]Note, however, that non-scopal adverbs act as mere intersective modifiers, e.g. *loudly* is a predicate in DM, but the main verb provides the top node in structures like *Abrams sang loudly.*

[3]See `http://kmcs.nii.ac.jp/enju/`.

[4]See `http://ufal.mff.cuni.cz/pcedt2.0/`.

section), and three-digit *item* number (within each document). For example, identifier `20200002` denotes the second sentence in the first file of PTB Section 02, the classic *Ms. Haag plays Elianti.* The annotation of this sentence is shown in Table 1.

With one exception, our fields (i.e. columns in the tab-separated matrix) are a subset of the CoNLL 2009 inventory: (1) `id`, (2) `form`, (3) `lemma`, and (4) `pos` characterize the current token, with token identifiers starting from 1 within each sentence. Besides the lemma and part-of-speech information, in the closed track of our task, there is no explicit analysis of syntax. Across the three target representations in the task, fields (1) and (2) are aligned and uniform, i.e. all representations annotate exactly the same text. On the other hand, fields (3) and (4) are representation-specific, i.e. there are different conventions for lemmatization, and part-of-speech assignments can vary (but all representations use the same PTB inventory of PoS tags).

The bi-lexical semantic dependency graph over tokens is represented by two or more columns starting with the obligatory, binary-valued fields (5) `top` and (6) `pred`. A positive value in the `top` column indicates that the node corresponding to this token is a *top* node (see Section 2 below). The `pred` column is a simplification of the corresponding field in earlier tasks, indicating whether or not this token represents a predicate, i.e. a node with outgoing dependency edges. With these minor differences to the CoNLL tradition, our file format can represent general, directed graphs, with designated top nodes. For example, there can be singleton nodes not connected to other parts of the graph, and in principle there can be multiple tops, or a non-predicate top node.

To designate predicate–argument relations, there are as many additional columns as there are predicates in the graph (i.e. tokens marked + in the `pred` column); these additional columns are called (7) `arg1`, (8) `arg2`, etc. These colums contain argument roles relative to the i-th predicate, i.e. a non-empty value in column `arg1` indicates that the current token is an argument of the (linearly) first predicate in the sentence. In this format, graph reentrancies will lead to a token receiving argument roles for multiple predicates (i.e. non-empty $\texttt{arg}_i$ values in the same row). All tokens of the same sentence must always have all argument columns filled in, even on non-predicate words; in other words, all lines making up one block of tokens will have the same number n of fields, but n can differ across sentences, depending on the count of graph nodes.

		DM	PAS	PCEDT
(1)	**# labels**	51	42	68
(2)	**% singletons**	22.62	4.49	35.79
(3)	**# edge density**	0.96	1.02	0.99
(4)	**%$_g$ trees**	2.35	1.30	56.58
(5)	**%$_g$ projective**	3.05	1.71	53.29
(6)	**%$_g$ fragmented**	6.71	0.23	0.56
(7)	**%$_n$ reentrancies**	27.35	29.40	9.27
(8)	**%$_g$ topless**	0.28	0.02	0.00
(9)	**# top nodes**	0.9972	0.9998	1.1237
(10)	**%$_n$ non-top roots**	44.71	55.92	4.36

Table 2: Contrastive high-level graph statistics.

4 Data Sets

All three target representations are annotations of the same text, Sections 00–21 of the WSJ Corpus. For this task, we have synchronized these resources at the sentence and tokenization levels and excluded from the SDP 2014 training and testing data any sentences for which (a) one or more of the treebanks lacked a gold-standard analysis; (b) a one-to-one alignment of tokens could not be established across all three representations; or (c) at least one of the graphs was cyclic. Of the 43,746 sentences in these 22 first sections of WSJ text, DeepBank lacks analyses for close to 15 %, and the Enju Treebank has gaps for a little more than four percent. Some 500 sentences show tokenization mismatches, most owing to DeepBank correcting PTB idiosyncrasies like ⟨`G.m.b, H.`⟩, ⟨`S.p, A.`⟩, and ⟨`U.S., .`⟩, and introducing a few new ones (Fares et al., 2013). Finally, 232 of the graphs obtained through the above conversions were cyclic. In total, we were left with 34,004 sentences (or 745,543 tokens) as training data (Sections 00–20), and 1348 testing sentences (29,808 tokens), from Section 21.

Quantitative Comparison As a first attempt at contrasting our three target representations, Table 2 shows some high-level statistics of the graphs comprising the training data.[5] In terms of distinctions

[5]These statistics are obtained using the 'official' SDP toolkit. We refer to nodes that have neither incoming nor outgoing edges and are not marked as top nodes as *singletons*; these nodes are ignored in subsequent statistics, e.g. when determining the proportion of edges per node (3) or the percentages of rooted trees (4) and fragmented graphs (6). The notation '%$_n$' denotes (non-singleton) node percentages, and '%$_g$' percentages over all graphs. We consider a *root* node any (non-singleton) node that has no incoming edges; *reentrant* nodes have at least two incoming edges. Following Sagae and Tsujii (2008), we consider a graph *projective* when there are no crossing edges (in a left-to-right rendering of nodes) and no roots are 'covered', i.e. for any root j there is no edge $i \rightarrow k$

	Directed			Undirected		
	DM	PAS	PCEDT	DM	PAS	PCEDT
DM	–	.6425	.2612	–	.6719	.5675
PAS	.6688	–	.2963	.6993	–	.5490
PCEDT	.2636	.2963	–	.5743	.5630	–

Table 3: Pairwise F_1 similarities, including punctuation (upper right diagonals) or not (lower left).

drawn in dependency labels (1), there are clear differences between the representations, with PCEDT appearing linguistically most fine-grained, and PAS showing the smallest label inventory. Unattached singleton nodes (2) in our setup correspond to tokens analyzed as semantically vacuous, which (as seen in Figure 1) include most punctuation marks in PCEDT and DM, but not PAS. Furthermore, PCEDT (unlike the other two) analyzes some high-frequency determiners as semantically vacuous. Conversely, PAS on average has more edges per (non-singleton) nodes than the other two (3), which likely reflects its approach to the analysis of functional words (see below).

Judging from both the percentage of actual trees (4), the proportions of projective graphs (5), and the proportions of reentrant nodes (7), PCEDT is much more 'tree-oriented' than the other two, which at least in part reflects its approach to the analysis of modifiers and determiners (again, see below). We view the small percentages of graphs without at least one top node (8) and of graphs with at least two non-singleton components that are not interconnected (6) as tentative indicators of general well-formedness. Intuitively, there should always be a 'top' predicate, and the whole graph should 'hang together'. Only DM exhibits non-trivial (if small) degrees of topless and fragmented graphs, and these may indicate imperfections in the DeepBank annotations or room for improvement in the conversion from full MRSs to bi-lexical dependencies, but possibly also exceptions to our intuitions about semantic dependency graphs.

Finally, in Table 3 we seek to quantify pairwise structural similarity between the three representations in terms of unlabeled dependency F_1 (dubbed UF in Section 5 below). We provide four variants of this metric, (a) taking into account the directionality of edges or not and (b) including edges involving punctuation marks or not. On this view, DM and PAS are structurally much closer to each other than either of the two is to PCEDT, even more so when discarding punctuation. While relaxing the comparison to ignore edge directionality also increases similarity scores for this pair, the effect is much more pronounced when comparing either to PCEDT. This suggests that directionality of semantic dependencies is a major source of diversion between DM and PAS on the one hand, and PCEDT on the other hand.

Linguistic Comparison Among other aspects, Ivanova et al. (2012) categorize a range of syntactic and semantic dependency annotation schemes according to the role that functional elements take. In Figure 1 and the discussion of Table 2 above, we already observed that PAS differs from the other representations in integrating into the graph auxiliaries, the infinitival marker, the case-marking preposition introducing the argument of *apply* (*to*), and most punctuation marks;[6] while these (and other functional elements, e.g. complementizers) are analyzed as semantically vacuous in DM and PCEDT, they function as predicates in PAS, though do not always serve as 'local' top nodes (i.e. the semantic head of the corresponding sub-graph): For example, the infinitival marker in Figure 1 takes the verb as its argument, but the 'upstairs' predicate *impossible* links directly to the verb, rather than to the infinitival marker as an intermediate.

At the same time, DM and PAS pattern alike in their approach to modifiers, e.g. attributive adjectives, adverbs, and prepositional phrases. Unlike in PCEDT (or common syntactic dependency schemes), these are analyzed as semantic predicates and, thus, contribute to higher degrees of node reentrancy and non-top (structural) roots. Roughly the same holds for determiners, but here our PCEDT projection of Prague tectogrammatical trees onto bi-lexical dependencies leaves 'vanilla' articles (like *a* and *the*) as singleton nodes.

The analysis of coordination is distinct in the three representations, as also evident in Figure 1. By design, DM opts for what is often called the Mel'čukian analysis of coordinate structures (Mel'čuk, 1988), with a chain of dependencies rooted at the first conjunct (which is thus considered the head, 'standing in' for the structure at large); in the DM approach, coordinating conjunctions are not integrated with the graph but rather contribute different types of dependencies. In PAS, the final coordinating conjunction is the head of the

such that $i < j < k$.

[6] In all formats, punctuation marks like dashes, colons, and sometimes commas can be contentful, i.e. at times occur as both predicates, arguments, and top nodes.

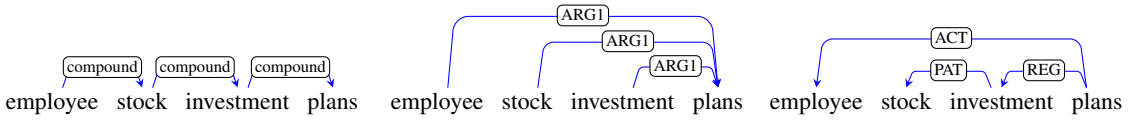


Figure 2: Analysis of nominal compounding in DM, PAS, and PCEDT, respectively .

structure and each coordinating conjunction (or intervening punctuation mark that acts like one) is a two-place predicate, taking left and right conjuncts as its arguments. Conversely, in PCEDT the last coordinating conjunction takes all conjuncts as its arguments (in case there is no overt conjunction, a punctuation mark is used instead); additional conjunctions or punctuation marks are not connected to the graph.[7]

A linguistic difference between our representations that highlights variable granularities of analysis and, relatedly, diverging views on the scope of the problem can be observed in Figure 2. Much noun phrase–internal structure is not made explicit in the PTB, and the Enju Treebank from which our PAS representation derives predates the bracketing work of Vadas and Curran (2007). In the four-way nominal compounding example of Figure 2, thus, PAS arrives at a strictly left-branching tree, and there is no attempt at interpreting semantic roles among the members of the compound either; PCEDT, on the other hand, annotates both the *actual* compound-internal bracketing and the assignment of roles, e.g. making *stock* the `PAT`(ient) of *investment*. In this spirit, the PCEDT annotations could be directly paraphrased along the lines of *plans by employees for investment in stocks.* In a middle position between the other two, DM disambiguates the bracketing but, by design, merely assigns an underspecified, construction-specific dependency type; its `compound` dependency, then, is to be interpreted as the most general type of dependency that can hold between the elements of this construction (i.e. to a first approximation either an argument role or a relation parallel to a preposition, as in the above paraphrase). The DM and PCEDT annotations of this specific example happen to diverge in their bracketing decisions, where the DM analysis corresponds to *[...] investments in stock for employees*, i.e. grouping the concept *employee stock* (in contrast to 'common stock').

Without context and expert knowledge, these decisions are hard to call, and indeed there has been much previous work seeking to identify and annotate the relations that hold between members of a nominal compound (see Nakov, 2013, for a recent overview). To what degree the bracketing and role disambiguation in this example are determined by the linguistic signal (rather than by context and world knowledge, say) can be debated, and thus the observed differences among our representations in this example relate to the classic contrast between 'sentence' (or 'conventional') meaning, on the one hand, and 'speaker' (or 'occasion') meaning, on the other hand (Quine, 1960; Grice, 1968). In turn, we acknowledge different plausible points of view about which level of semantic representation should be the target representation for data-driven *parsing* (i.e. structural analysis guided by the grammatical system), and which refinements like the above could be construed as part of a subsequent task of *interpretation*.

5 Task Setup

Training data for the task, providing all columns in the file format sketched in Section 3 above, together with a first version of the SDP toolkit—including graph input, basic statistics, and scoring—were released to candidate participants in early December 2013. In mid-January, a minor update to the training data and optional syntactic 'companion' analyses (see below) were provided, and in early February the description and evaluation of a simple baseline system (using tree approximations and the parser of Bohnet, 2010). Towards the end of March, an input-only version of the test data was released, with just columns (1) to (4) pre-filled; participants then had one week to run their systems on these inputs, fill in columns (5), (6), and upwards, and submit their results (from up to two different runs) for scoring. Upon completion of the testing phase, we have shared the gold-standard test data, official scores, and system results for all submissions with participants and are currently preparing all data for general release through the Linguistic Data Consortium.

[7] As detailed by Miyao et al. (2014), individual conjuncts can be (and usually are) arguments of other predicates, whereas the topmost conjunction only has incoming edges in nested coordinate structures. Similarly, a 'shared' modifier of the coordinate structure as a whole would take as its argument the local top node of the coordination in DM or PAS (i.e. the first conjunct or final conjunction, respectively), whereas it would depend as an argument on all conjuncts in PCEDT.

		DM				PAS				PCEDT			
	$\overline{\text{LF}}$	LP	LR	LF	LM	LP	LR	LF	LM	LP	LR	LF	LM
Peking	85.91	90.27	88.54	89.40	26.71	93.44	90.69	92.04	38.13	78.75	73.96	76.28	11.05
Priberam	85.24	88.82	87.35	88.08	22.40	91.95	89.92	90.93	32.64	78.80	74.70	76.70	09.42
Copenhagen-Malmö	80.77	84.78	84.04	84.41	20.33	87.69	88.37	88.03	10.16	71.15	68.65	69.88	08.01
Potsdam	77.34	79.36	79.34	79.35	07.57	88.15	81.60	84.75	06.53	69.68	66.25	67.92	05.19
Alpage	76.76	79.42	77.24	78.32	09.72	85.65	82.71	84.16	17.95	70.53	65.28	67.81	06.82
Linköping	72.20	78.54	78.05	78.29	06.08	76.16	75.55	75.85	01.19	60.66	64.35	62.45	04.01

		DM				PAS				PCEDT			
	$\overline{\text{LF}}$	LP	LR	LF	LM	LP	LR	LF	LM	LP	LR	LF	LM
Priberam	86.27	90.23	88.11	89.16	26.85	92.56	90.97	91.76	37.83	80.14	75.79	77.90	10.68
CMU	82.42	84.46	83.48	83.97	08.75	90.78	88.51	89.63	26.04	76.81	70.72	73.64	07.12
Turku	80.49	80.94	82.14	81.53	08.23	87.33	87.76	87.54	17.21	72.42	72.37	72.40	06.82
Potsdam	78.60	81.32	80.91	81.11	09.05	89.41	82.61	85.88	07.49	70.35	67.33	68.80	05.42
Alpage	78.54	83.46	79.55	81.46	10.76	87.23	82.82	84.97	15.43	70.98	67.51	69.20	06.60
In-House	75.89	92.58	92.34	92.46	48.07	92.09	92.02	92.06	43.84	40.89	45.67	43.15	00.30

Table 4: Results of the closed (top) and open tracks (bottom). For each system, the second column ($\overline{\text{LF}}$) indicates the averaged LF score across all target representations), which was used to rank the systems.

Evaluation Systems participating in the task were evaluated based on the accuracy with which they can produce semantic dependency graphs for previously unseen text, measured relative to the gold-standard testing data. The key measures for this evaluation were labeled and unlabeled precision and recall with respect to predicted dependencies (predicate–role–argument triples) and labeled and unlabeled exact match with respect to complete graphs. In both contexts, identification of the top node(s) of a graph was considered as the identification of additional, 'virtual' dependencies from an artificial root node (at position 0). Below we abbreviate these metrics as (a) labeled precision, recall, and F_1: LP, LR, LF; (b) unlabeled precision, recall, and F_1: UP, UR, UF; and (c) labeled and unlabeled exact match: LM, UM.

The 'official' ranking of participating systems, in both the closed and the open tracks, is determined based on the arithmetic mean of the labeled dependency F_1 scores (i.e. the geometric mean of labeled precision and labeled recall) on the three target representations (DM, PAS, and PCEDT). Thus, to be considered for the final ranking, a system had to submit semantic dependencies for all three target representations.

Closed vs. Open Tracks The task was subdivided into a *closed* track and an *open* track, where systems in the closed track could only be trained on the gold-standard semantic dependencies distributed for the task. Systems in the open track, on the other hand, could use additional resources, such as a syntactic parser, for example—provided that they make sure to not use any tools or resources that encompass knowledge of the gold-standard syntactic or semantic analyses of the SDP 2014 test data, i.e. were directly or indirectly trained or otherwise derived from WSJ Section 21.

This restriction implies that typical off-the-shelf syntactic parsers had to be re-trained, as many data-driven parsers for English include this section of the PTB in their default training data. To simplify participation in the open track, the organizers prepared ready-to-use 'companion' syntactic analyses, sentence- and token-aligned to the SDP data, in two formats, viz. PTB-style phrase structure trees obtained from the parser of Petrov et al. (2006) and Stanford Basic syntactic dependencies (de Marneffe et al., 2006) produced by the parser of Bohnet and Nivre (2012).

6 Submissions and Results

From 36 teams who had registered for the task, test runs were submitted for nine systems. Each team submitted one or two test runs per track. In total, there were ten runs submitted to the closed track and nine runs to the open track. Three teams submitted to both the closed and the open track. The main results are summarized and ranked in Table 4. The ranking is based on the average LF score across all three target representations, which is given in the $\overline{\text{LF}}$ column. In cases where a team submitted two runs to a track, only the highest-ranked score is included in the table.

Team	Track	Approach	Resources
Linköping	C	extension of Eisner's algorithm for DAGs, edge-factored structured perceptron	—
Potsdam	C & O	graph-to-tree transformation, Mate	companion
Priberam	C & O	model with second-order features, decoding with dual decomposition, MIRA	companion
Turku	O	cascade of SVM classifiers (dependency recognition, label classification, top recognition)	companion, syntactic n-grams, word2vec
Alpage	C & O	transition-based parsing for DAGs, logistic regression, structured perceptron	companion, Brown clusters
Peking	C	transition-based parsing for DAGs, graph-to-tree transformation, parser ensemble	—
CMU	O	edge classification by logistic regression, edge-factored structured SVM	companion
Copenhagen-Malmö	C	graph-to-tree transformation, Mate	—
In-House	O	existing parsers developed by the organizers	grammars

Table 5: Overview of submitted systems, high-level approaches, and additional resources used (if any).

In the closed track, the average LF scores across target representations range from 85.91 to 72.20. Comparing the results for different target representations, the average LF scores across systems are 85.96 for PAS, 82.97 for DM, and 70.17 for PCEDT. The scores for labeled exact match show a much larger variation across both target representations and systems.[8]

In the open track, we see very similar trends. The average LF scores across target representations range from 86.27 to 75.89 and the corresponding scores across systems are 88.64 for PAS, 84.95 for DM, and 67.52 for PCEDT. While these scores are consistently higher than in the closed track, the differences are small. In fact, for each of the three teams that submitted to both tracks (Alpage, Potsdam, and Priberam) improvements due to the use of additional resources in the open track do not exceed two points LF.

7 Overview of Approaches

Table 5 shows a summary of the systems that submitted final results. Most of the systems took a strategy to use some algorithm to process (restricted types of) graph structures, and apply machine learning like structured perceptrons. The methods for processing graph structures are classified into three types. One is to transform graphs into trees in the preprocessing stage, and apply conventional dependency parsing systems (e.g. Mate; Bohnet, 2010) to the converted trees. Some systems simply output the result of dependency parsing (which means they inherently lose some dependencies), while the others apply post-processing to recover non-tree structures. The second strategy is to use a parsing algorithm that can directly generate graph structures (in the spirit of Sagae & Tsujii, 2008; Titov et al., 2009). In many cases such algorithms generate restricted types of graph structures, but these restrictions appear feasible for our target representations. The last approach is more machine learning–oriented; they apply classifiers or scoring methods (e.g. edge-factored scores), and find the highest-scoring structures by some decoding method.

It is difficult to tell which approach is the best; actually, the top three systems in the closed and open tracks selected very different approaches. A possible conclusion is that exploiting existing systems or techniques for dependency parsing was successful; for example, Peking built an ensemble of existing transition-based and graph-based dependency parsers, and Priberam extended an existing dependency parser. As we indicated in the task description, a novel feature of this task is that we have to compute graph structures, and cannot assume well-known properties like projectivity and lack of reentrancies. However, many of the participants found that our representations are mostly tree-like, and this fact motivated them to apply methods that have been well studied in the field of syntactic dependency parsing.

Finally, we observe that three teams participated in both the closed and open tracks, and all of them reported that adding external resources improved accuracy by a little more than one point. Systems with (only) open submissions extensively use syntactic features (e.g. dependency paths) from external resources, and they are shown effective even

[8] Please see the task web page at the address indicated above for full labeled and unlabeled scores.

with simple machine learning models. Pre-existing, tree-oriented dependency parsers are relatively effective, especially when combined with graph-to-tree transformation. Comparing across our three target representations, system scores show a tendency PAS > DM > PCEDT, which can be taken as a tentative indicator of relative levels of 'parsability'. As suggested in Section 4, this variation most likely correlates at least in part with diverging design decisions, e.g. the inclusion of relatively local and deterministic dependencies involving function words in PAS, or the decision to annotate contextually determined speaker meaning (rather than 'mere' sentence meaning) in at least some constructions in PCEDT.

8 Conclusions and Outlook

We have described the motivation, design, and outcomes of the SDP 2014 task on semantic dependency parsing, i.e. retrieving bi-lexical predicate–argument relations between all content words within an English sentence. We have converted to a common format three existing annotations (DM, PAS, and PCEDT) over the same text and have put this to use for the first time in training and testing data-driven semantic dependency parsers. Building on strong community interest already to date and our belief that graph-oriented dependency parsing will further gain importance in the years to come, we are preparing a similar (slightly modified) task for SemEval 2015. Candidate modifications and extensions will include cross-domain testing and evaluation at the level of 'complete' predications (in contrast to more lenient per-dependency F_1 used this year). As optional new sub-tasks, we plan on offering cross-linguistic variation and predicate (i.e. semantic frame) disambiguation for at least some of the target representations. To further probe the role of syntax in the recovery of semantic dependency relations, we will make available to participants a wider selection of syntactic analyses, as well as add a third (idealized) 'gold' track, where syntactic dependencies are provided directly from available syntactic annotations of the underlying treebanks.

Acknowledgements

We are grateful to Željko Agić and Bernd Bohnet for consultation and assistance in preparing our baseline and companion parses, to the Linguistic Data Consortium (LDC) for support in distributing the SDP data to participants, as well as to Emily M. Bender and two anonymous reviewers for feedback on this manuscript. Data preparation was supported through access to the ABEL high-performance computing facilities at the University of Oslo, and we acknowledge the Scientific Computing staff at UiO, the Norwegian Metacenter for Computational Science, and the Norwegian tax payers. Part of this work has been supported by the infrastructural funding by the Ministry of Education, Youth and Sports of the Czech Republic (CEP ID LM2010013).

References

Bohnet, B. (2010). Top accuracy and fast dependency parsing is not a contradiction. In *Proceedings of the 23rd International Conference on Computational Linguistics* (p. 89 – 97). Beijing, China.

Bohnet, B., & Nivre, J. (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* (p. 1455 – 1465). Jeju Island, Korea.

Buchholz, S., & Marsi, E. (2006). CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the 10th Conference on Natural Language Learning* (p. 149 – 164). New York, NY, USA.

Copestake, A., Flickinger, D., Pollard, C., & Sag, I. A. (2005). Minimal Recursion Semantics. An introduction. *Research on Language and Computation*, *3*(4), 281 – 332.

de Marneffe, M.-C., MacCartney, B., & Manning, C. D. (2006). Generating typed dependency parses from phrase structure parses. In *Proceedings of the 5th International Conference on Language Resources and Evaluation* (p. 449 – 454). Genoa, Italy.

Fares, M., Oepen, S., & Zhang, Y. (2013). Machine learning for high-quality tokenization. Replicating variable tokenization schemes. In *Computational linguistics and intelligent text processing* (p. 231 – 244). Springer.

Flickinger, D. (2000). On building a more efficient grammar by exploiting types. *Natural Language Engineering*, *6 (1)*, 15 – 28.

Flickinger, D., Zhang, Y., & Kordoni, V. (2012). DeepBank. A dynamically annotated treebank of the Wall Street Journal. In *Proceedings of the 11th International Workshop on Treebanks and Linguistic Theories* (p. 85 – 96). Lisbon, Portugal: Edições Colibri.

Gildea, D., & Jurafsky, D. (2002). Automatic labeling of semantic roles. *Computational Linguistics*, *28*,

245 – 288.

Grice, H. P. (1968). Utterer's meaning, sentence-meaning, and word-meaning. *Foundations of Language*, *4*(3), 225 – 242.

Hajič, J., Ciaramita, M., Johansson, R., Kawahara, D., Martí, M. A., Màrquez, L., ... Zhang, Y. (2009). The CoNLL-2009 Shared Task. syntactic and semantic dependencies in multiple languages. In *Proceedings of the 13th Conference on Natural Language Learning* (p. 1 – 18). Boulder, CO, USA.

Hajič, J., Hajičová, E., Panevová, J., Sgall, P., Bojar, O., Cinková, S., ... Žabokrtský, Z. (2012). Announcing Prague Czech-English Dependency Treebank 2.0. In *Proceedings of the 8th International Conference on Language Resources and Evaluation* (p. 3153 – 3160). Istanbul, Turkey.

Ivanova, A., Oepen, S., Øvrelid, L., & Flickinger, D. (2012). Who did what to whom? A contrastive study of syntacto-semantic dependencies. In *Proceedings of the Sixth Linguistic Annotation Workshop* (p. 2 – 11). Jeju, Republic of Korea.

Kate, R. J., & Wong, Y. W. (2010). Semantic parsing. The task, the state of the art and the future. In *Tutorial abstracts of the 20th Meeting of the Association for Computational Linguistics* (p. 6). Uppsala, Sweden.

Marcus, M., Santorini, B., & Marcinkiewicz, M. A. (1993). Building a large annotated corpora of English: The Penn Treebank. *Computational Linguistics*, *19*, 313 – 330.

Mel'čuk, I. (1988). *Dependency syntax. Theory and practice*. Albany, NY, USA: SUNY Press.

Meyers, A., Reeves, R., Macleod, C., Szekely, R., Zielinska, V., Young, B., & Grishman, R. (2004). Annotating noun argument structure for NomBank. In *Proceedings of the 4th International Conference on Language Resources and Evaluation* (p. 803 – 806). Lisbon, Portugal.

Miyao, Y., Oepen, S., & Zeman, D. (2014). In-house: An ensemble of pre-existing off-the-shelf parsers. In *Proceedings of the 8th International Workshop on Semantic Evaluation.* Dublin, Ireland.

Nakov, P. (2013). On the interpretation of noun compounds: Syntax, semantics, and entailment. *Natural Language Engineering*, *19*(3), 291 – 330.

Nivre, J., Hall, J., Kübler, S., McDonald, R., Nilsson, J., Riedel, S., & Yuret, D. (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* (p. 915 – 932). Prague, Czech Republic.

Oepen, S., & Lønning, J. T. (2006). Discriminant-based MRS banking. In *Proceedings of the 5th International Conference on Language Resources and Evaluation* (p. 1250 – 1255). Genoa, Italy.

Palmer, M., Gildea, D., & Kingsbury, P. (2005). The Proposition Bank. A corpus annotated with semantic roles. *Computational Linguistics*, *31*(1), 71 – 106.

Petrov, S., Barrett, L., Thibaux, R., & Klein, D. (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* (p. 433 – 440). Sydney, Australia.

Quine, W. V. O. (1960). *Word and object*. Cambridge, MA, USA: MIT press.

Sagae, K., & Tsujii, J. (2008). Shift-reduce dependency DAG parsing. In *Proceedings of the 22nd International Conference on Computational Linguistics* (p. 753 – 760). Manchester, UK.

Titov, I., Henderson, J., Merlo, P., & Musillo, G. (2009). Online graph planarisation for synchronous parsing of semantic and syntactic dependencies. In *Proceedings of the 21st International Joint Conference on Artifical Intelligence* (p. 1562 – 1567).

Vadas, D., & Curran, J. (2007). Adding Noun Phrase Structure to the Penn Treebank. In *Proceedings of the 45th Meeting of the Association for Computational Linguistics* (p. 240 – 247). Prague, Czech Republic.