Deep Dependency Graph Conversion in English

Jinho D. Choi

Department of Mathematics and Computer Science
Emory University
jinho.choi@emory.edu

Abstract

This paper presents a method for the automatic conversion of constituency trees into deep dependency graphs consisting of primary, secondary, and semantic relations. Our work is distinguished from previous work concerning the generation of shallow dependency trees such that it generates dependency graphs incorporating deep structures in which relations stay consistent regardless of their surface positions, and derives relations between out-of-domain arguments, caused by syntactic variations such as open clause, relative clause, or coordination, and their predicates so the complete argument structures are represented for both verbal and non-verbal predicates. Our deep dependency graph conversion recovers important argument relations that would be missed by dependency tree conversion, and merges syntactic and semantic relations into one unified representation, which can reduce the bundle of developing another layer of annotation dedicated for predicate argument structures. Our graph conversion method is applied to six corpora in English and generated over 4.6M dependency graphs covering 20 different genres.[1]

1 Introduction

Several approaches have been proposed for the automatic conversion of constituency trees into dependency trees in English [8, 9, 12, 13, 22]. Multiple benefits are found by this type of conversion. First, there exists a large amount of corpora annotated with constituency trees in English such that by converting them into dependency trees, large data can be obtained for building robust dependency parsing models with a minimum of manual effort. Second, long-distance dependencies are represented by non-projective dependencies in dependency trees, which can be reliably found by the current state-of-the-art dependency parsers [10], whereas they are represented by empty categories in constituency trees and little to no constituency parsers produce them well [15]. Third, dependency trees are more suitable for representing flexible word order languages as well as colloquial writings such that they are often preferred to represent universal structures over constituency trees.

[1] All our resources are publicly available: https://github.com/emorynlp/ddr

Most of the previous work focuses on the generation of shallow dependency trees, which do not necessarily carry on the same dependency structures given different syntactic alternations even when they comprise similar semantics. For the following sentences, shallow dependency trees give different structures although the underlying semantics of these sentence pairs are the same:

John called Mary	vs.	*John made a call to Mary*
John gave Mary a book	vs.	*A book was given by John to Mary*

Furthermore, since dependency trees are bounded by tree properties, *single-root*, *connected*, *single-head*, and *acyclic*, they cannot represent any argument structure that would break these properties [25]. Such argument structures occur often, where an argument is shared by multiple predicates (e.g., open clauses, coordination) or it becomes the head of its predicate by the syntax (e.g., relative clauses). Preserving the tree properties allows the development of efficient parsing models [20, 30, 32, 41]; however, this ends up requiring the development of another model for finding the missing arguments (e.g., semantic role labeling), which can be more cumbersome than developing one graph parsing model that generates deep dependency graphs.

This paper presents a method that converts the Penn Treebank style constituency trees [27] into deep dependency graphs. Our dependency graphs are motivated by deep structures [11], where arguments take the same semantic roles regardless of their surface positions, and give complete predicate argument structures by utilizing function tags, empty categories, and unexplored features in coordination provided by the constituency trees. We believe that this work will be beneficial for those who need a large amount of dependency graphs with rich predicate argument structures, where predicates are abstracted away from their syntactic variations.

2 Related Work

Nivre [31] proposed a deterministic conversion method using head-finding and labeling rules for the conversion of constituency trees into dependency trees. Johansson and Nugues [22] improved this method by adding non-projective dependencies and semantic relations using empty categories and function tags; their representation had been used for the CoNLL'08-09 shared tasks [17, 37]. Choi and Palmer [8] extended this work by updating the head-finding rules for the recent Penn Treebank format and handling several complex structures such as small clauses or gapping relations. de Marneffe and Manning [12] suggested a separate conversion method that gives rich dependency labels, well-known as the Stanford typed dependencies. Choi and Palmer [9] improved this work by adding non-projective dependencies and secondary dependencies. de Marneffe et al. [13] introduced another conversion method aiming towards the Universal Dependencies [33], a project that attempts to develop an universal representation for multiple languages. Our work is distinguished from the previous work because they mostly target on the generation of tree structures whereas our main focus is on the generation of graph structures.

Our work was highly inspired by previous frameworks on lexicalized tree adjoining grammars (LTAG), combinatory categorial grammars (CCG), lexical functional grammars (LFG), and head-driven phrase structure grammars (HPSG). Xia [39] extracted LTAG from constituency trees by automatically deriving elementary trees with linguistic knowledge. Hockenmaier and Steedman [18] converted constituency trees into a corpus of CCG derivations by making several systematic changes in the constituency trees, knowns as CCGbank [19]. Cahill et al. [5] extracted LFG subcategorization frames and paths linking long distance dependencies from f-structures converted from constituency trees. Miyao et al. [29] extracted HPSG by deriving fine-grained lexical entries from constituency trees with heuristic annotations. Numerous statistical parsers have been developed from the corpora generated by these approaches where the generated structures can be viewed as direct acyclic graphs. All of the above approaches were based on the old bracketing guidelines from the Penn Treebank [26], whereas we followed the latest guidelines that made several structural as well as tagging changes. Our work is similar to Schuster and Manning [36] in a way that we both try to find the complete predicate argument structures by adding secondary dependencies to shallow dependency trees, but distinguished because their dependency relations are still sensitive to the surface positions whereas such syntactic alternations are abstracted away from our representation.

There exist several corpora consisting of deep dependency graphs. Kromann [24] introduced the Danish Dependency Treebank containing dependency graphs with long-distance dependencies, gapping relations, and anaphoric reference links. Al-Raheb et al. [1] created the DCU 250 Arabic Dependency Bank including manual annotation based on the theoretical framework of LFG. Yu et al. [42] generated the Enju Chinese Treebank (ECT) from the Penn Chinese Treebank [40] by developing a large-scale grammar based on HPSG. Flickinger et al. [14] introduced DeepBank derived from parsing results using linguistically precise HPSG and manual disambiguation. Hajič et al. [16] created the Prague Czech-English Dependency Treebank (PDT) consisting of parallel dependency graphs over the constituency trees in the Penn Treebank and their Czech translations. ECT, DeepBank, and PDT were used for the SemEval 2015 Task 18: *Broad-Coverage Semantic Dependency Parsing*. Candito et al. [7] introduced the Sequoia French Treebank that added a deep syntactic representation to the existing Sequia corpus [6].

Although not directly related, it is worth mentioning the existing corpora consisting of predicate argument structures. Baker et al. [3] introduced FrameNet based on frame semantics that gave manual annotation of lexical units and their semantic frames. Palmer et al. [34] created PropBank where each predicate was annotated with a sense and each sense came with its own argument structure. Meyers et al. [28] created NomBank providing annotation of nominal arguments in the Penn Treebank by fine-tuning the lexical entries. The original PropBank included only verbal predicates; Hwang et al. [21] extended PropBank with light verb constructions where eventive nouns associated with light verbs were also considered. Banarescu et al. [4] introduced Abstract Meaning Representation which was motivated by PropBank but richer in representation and more abstracting away from syntax.

3 Deep Dependency Graph

Our deep dependency graphs (DDG) preserve only two out of the four tree properties: *single-root* and *connected*. Two types of dependencies are used to represent DDG. The primary dependencies, represented by the top arcs in figures, form dependency trees similar to the ones introduced by the Universal Dependencies (UD) [33]. The secondary dependencies, represented by the bottom arcs in figures, form dependency graphs allowing multiple heads and cyclic relations. Separating these two types of dependencies enables to develop either tree or graph parsing models. Additionally, semantic roles extracted from function tags are annotated on the head nodes.[2]

3.1 Non-verbal Predicates

Copula Non-verbal predicates are mostly constructed by copulas. DDG considers both the prototypical copula (e.g., *be*) as well as semi-copulas (e.g., *become*, *remain*). Non-verbal predicates with copulas can be easily identified by checking the function tag `PRD` (secondary predicate) in constituency trees (Figure 1a). Unlike UD, the preposition becomes the head of a preposition phrase when it is a predicate in DDG (Figure 1b). This is to avoid multiple subjects per predicate, which would be caused by making a clause as the head of a prepositional phrase (Figure 1c).

Light verb construction Non-verbal predicates can also be constructed by light-verbs, which are not annotated in constituency trees but they are in PropBank [21]. A set of light verbs $L = \{make, take, have, do, give, keep\}$, a set of 2,474 eventive nouns $N = \{call, development, violation, \ldots\}$, and a map $M \in |L| \times |N| \rightarrow |P| = \{(give, call) \rightarrow to, (make, development) \rightarrow of, \ldots\}$ of prepositions indicating the objects of the nominal predicates are collected from PropBank. Given a verb $v \in L$ with the direct object $n \in N$, v is considered a light verb and the preposition phrase that immediately follows n and contains the preposition $p \leftarrow M(v,n)$ is considered the object of n in DDG (Figure 2b). This lexicon-based approach yields about 2.5 times more light verb constructions than PropBank annotation; further assessment of this pseudo annotation should be performed, which we will explore in the future.

3.2 Deep Arguments

Dative Indirect objects as well as preposition phrases whose semantic roles are the same as the indirect objects are considered datives. A nominal phrase is identified as an indirect object if it is followed by another nominal phrase representing the direct object (Figure 3a). A preposition phrase is considered a dative if it has either the function tag `DTV` (dative; Figure 3b) or `BNF` (benefactive; Figure 3c). Whether or not all benefactives should be considered datives is opened to a discussion; we plan to analyze this by using large unstructured data such as Wikipedia to measure the likelihood of dative constructions for each verb.

[2] All figures are provided together at the end of this paper.

Expletive Both the existential *there* and the extrapositional *it* in the subject position are considered expletives. The existential *there* can be identified by checking the part-of-speech tag `EX`. The extrapositional *it* is indicated by the empty category `*EXP*-d` in constituency trees, where `d` is the index to the referent clause (Figure 4c). When there exists an expletive, DDG labels the referent as the subject of the main predicate (Figures 4a and 4d) such that it is consistently represented regardless of the syntactic alternations, whereas it is not the case in UD (Figures 4b and 4e).

Passive construction Arguments in passive constructions are recognized as they would be in active constructions. The NP-movement for a passive construction is indicated by the empty category `*-d` in the constituency tree, where `d` is the index to the antecedent (Figures 5a and 5b). However, the NP-movement for a reduced passive construction is indicated by the empty category `*` with no index provided for the antecedent (Figure 5c). To find the antecedents in reduced passive constructions, we use the heuristic provided by NLP4J, an open source NLP toolkit, which gives over 99% agreement to the manual annotation of this kind in PropBank.[3] In Figure 5, *John*, *Mary*, and *book*, are the subject (`nsbj`), the dative (`dat`), and the object (`obj`) of the predicate *give*, regardless of their syntactic variations in the active, passive, and reduced passive constructions, which can be achieved by deriving dependency relations from the empty categories. Note that the object relation in Figure 5c would cause a cyclic relation among primary dependencies such that it is represented by the secondary dependency in DDG.

Small clause A small clause is a declarative clause that consists of a subject and a secondary predicate, identified by the function tags `SBJ` and `PRD`, respectively. There are two kinds of small clauses found in constituency trees, one with an internal subject and the other with an external subject. Figure 6 shows examples of small clauses with internal subjects. In this case, *John* is consistently recognized as the subject of the adjectival predicate *smart* in the declarative clause (Figure 6a), the small clause (Figure 6b), and the small clause in the passive construction (Figure 6c). The subject relation in Figure 6c causes the non-projective dependency, which adds another complexity to DDG; nonetheless, making *John* as the subject of *consider* instead of *smart* as in UD would yield different relations between active (Figure 7a) and passive (Figure 7b) constructions, which is against the main objective of DDG.

Unlike the case of a small clause with the internal subject, a small clause with the external subject contains the empty category `*PRO*-d` where `d` is the index to the external subject. In this case, the external subject takes two separate semantic roles, one from its matrix verb and the other from the secondary predicate in the small clause. In Figure 8, *John* is consistently recognized as the object of the verbal predicate *call* and the subject of the nominal predicate *baptist* for both the active (Figure 8a) and the passive (Figure 8b) constructions in DDG, whereas it is not the case in UD. The subject relation between *John* and *baptist* is preserved by the secondary dependency to avoid multiple heads among the primary dependencies.

[3]This heuristic is currently used to pseudo annotate these links in PropBank, labeled as `LINK-PSV`.

Open clause An open clause is a clause with the external subject indicated by the empty category `*PRO*-d` (see the description above). Figure 9 shows examples of open clauses. The external subjects are represented by the secondary dependencies to avoid multiple heads. Notice that the head of the open clause, *teach*, in Figure 9b is assigned with the semantic role `prp` (purpose) extracted from the function tag `PRP`, which gives a more fine-grained relation to this type (Section 3.4).

Relative clause The NP-movement for the relativizer in a relative clause is noted by the empty category `*T*-d` in the constituency tree. Each relativizer is assigned with the dependency relation before its NP-movement and labeled as `r-*`, indicating that there exists a referent to this relativizer that should be assigned with the same relation. In Figure 10a, the relativizer *who* becomes the subject of the predicate *smart* so it is labeled as `r-nsbj`, implying that there exists the referent *John* that should be considered the real subject of *smart*. Similarly in Figure 10b, the relativizer *who* becomes the dative of the predicate *buy* so labeled as `r-dat`, implying that there exists *John* who is the real dative of *buy*. These referent relations are represented by the secondary dependencies to avoid cyclic relations. The constituency trees do not provide such referent information; we again use the heuristic provided by NLP4J,[4] which has been used to pseudo generate such annotation in PropBank, `LINK-SLC`.

Coordination Arguments in coordination structures are shared across predicates. These arguments can be identified in constituency trees; they are either the siblings of the coordinated verbs (e.g., *the book* and *last year* in Figure 11a) or the siblings of the verb phrases that are the ancestors of these verbs (e.g., *John* in Figure 11a). When the coordination is not on the same level, right node raising is used, which can be identified by the empty category `*RNR*-d`. In Figure 11b, *John* is coordinated across the verb phrase including *value* and the preposition phrase including *for*. Unlike the coordinated verbs in Figure 11a that are siblings, these are not siblings so need to be coordinated through right node raising. The coordinated arguments are represented by the secondary dependencies to avoid multiple heads.

3.3 Auxiliaries

Modal adjective Modal adjectives are connected with the class of modal verbs such as *can*, *may*, or *should* that are used with non-modal verbs to express possibility, permission, intention, etc:

able	915	ready	105	prepared	32	due	24	glad	21
likely	235	happy	69	eager	30	sure	24	unwilling	20
willing	173	about	49	free	30	determined	22	busy	18
unable	165	reluctant	44	unlikely	28	afraid	22	qualified	16

Table 1: Top-20 modal adjectives and their counts from the corpora in Table 3.

[4] `https://github.com/emorynlp/nlp4j`

An adjective a_m is considered a modal if 1)it is a non-verbal predicate (i.e., if it belong to an adjective phrase with the function tag PRD), 2)it is followed by a clause whose subject is an empty category e, and 3)the antecedent of e is the subject of a_m. In Figure 12, *able* and *about* are considered modal adjectives because they are followed by the clauses whose subjects are linked to the subjects of the adjectival predicates, *John*. Modal adjectives together with modal verbs give another level of abstraction in DDG.

Raising verb Distinguished from most of the previous work, raising verbs modify the "raised" verbs in DDG. A verb is considered a raising verb if 1)it is followed by a clause whose subject is the empty category *-d, and 2)the antecedent of the empty category is the subject of the raise verb. In Figure 13, the raising verbs *go*, *have*, and *keep* are followed by the clauses whose subjects are the empty categories *-1, *-2, and *-3, which all link to the same subject as the raised verb, *study*.

have	1,846	begin	825	stop	379	keep	158	prove	89
go	1,461	seem	787	be	322	use	157	turn	67
continue	1,210	appear	714	fail	233	get	136	happen	38
need	1,038	start	546	tend	168	ought	91	expect	38

Table 2: Top-20 raising verbs and their counts from the corpora in Table 3.

3.4 Semantic Roles

As shown in Figure 9a, semantic roles are extracted from certain function tags and added to the terminal heads of the phrases that include such function tags. The function tags used to extract semantic roles are: DIR: directional, EXT: extent, LOC: locative, MNR: manner, PRP: purpose, and TMP: temporal.

4 Analysis

4.1 Corpora

Six corpora that consist of the Penn Treebank style constituency trees are used to generate deep dependency graphs: OntoNotes (Weischedel et al. [38]), the English Web Treebank (Web; Petrov and McDonald [35]), QuestionBank (Judge et al. [23]), and the MiPACQ|Sharp|Thyme corpora (Albright et al. [2]). All together, these corpora cover 20 different genres including formal, colloquial, conversational, and clinical documents, providing enough diversities to our dependency representation.

	OntoNotes	**Web**	**Question**	**MiPACQ**	**Sharp**	**Thyme**
SC	138,566	16,622	4,000	19,141	50,725	88,893
WC	2,620,495	254,830	38,188	269,178	499,834	936,166

Table 3: Distributions of six corpora used to generate deep dependency graphs. SC: sentence count, WC: word count.

4.2 Primary vs. Secondary Dependencies

Table 4 shows the distributions of the primary and secondary dependencies generated by our deep dependency graph conversion. At a glimpse, the portion of the secondary dependencies over the entire primary dependencies seems rather small (about 2.3%). However, when only the core arguments (`*sbj`, `obj`, `dat`, `comp`) and the adverbials (`adv*`, `neg`, `ppmod`) are considered, where the secondary dependencies are mostly focused on, the portion increases to 8.4%, which is more significant. Few of the secondary dependencies are generated for unexpected relations such as `acl`, `appo`, and `attr`; from our analysis, we found that those were mostly caused by annotation errors in constituency trees.

4.3 Syntactic vs. Semantic Dependencies

Table 5 shows the confusion matrix between the syntactic dependencies in Table 4 and the semantic roles in Section 3.4. As expected, the adverbials followed by the clausal complements (`comp`) take the most portion of the semantic dependencies. A surprising number of semantic roles are assigned to the root; from our analysis, we found that those were mostly caused by non-verbal predicates implying either locative or temporal information. It is possible to use these semantic dependencies in place of the syntactic dependencies, which will increase the number of labels, but will allow to develop a graph parser that handles both syntactic and semantic dependencies without developing complex joint inference models.

5 Conclusion

We present a conversion method that automatically transforms constituency trees into deep dependency graphs. Our graphs consist of three types of relations, primary dependencies, secondary dependencies, and semantic roles, which can be processed separately or together to produce one unified dependency representation. The primary dependencies form dependency trees that can be generated by any non-projective dependency parser. The secondary dependencies together with the primary dependencies form deep dependency graphs. The semantic roles together with the syntactic dependencies form rich predicate argument structures. Our conversion method is applied to large corpora (over 4.6 times larger than the original Penn Treebank), which provides big data with much diversities. We plan to further extend this approach to more semantically-oriented dependency graphs by utilizing existing lexicons such as PropBank and VerbNet.

Acknowledgments

We gratefully acknowledge the support of the Kindi research grant. A special thank is due to professor Martha Palmer at the University of Colorado Boulder, who had encouraged the author to develop this representation during his Ph.D. program.

Type	Label	Description	Primary	Secondary
Subject	csbj	Clausal subject	5,291	123
	expl	Expletive	10,808	0
	nsbj	Nominal subject	298,418	71,383
Object	comp	Clausal complement	86,884	105
	dat	Dative	6,763	87
	obj	(Direct or preposition) object	205,149	20,785
Auxiliary	aux	Auxiliary verb	148,829	0
	cop	Copula	81,661	0
	lv	Light verb	7,655	0
	modal	Modal (verb or adjective)	49,259	0
	raise	Raising verb	10,598	0
Nominal and Quantifier	acl	Clausal modifier of nominal	24,791	7
	appo	Apposition	32,460	17
	attr	Attribute	352,939	14
	det	Determiner	334,784	0
	num	Numeric modifier	95,957	0
	poss	Possessive modifier	62,489	0
	relcl	Relative clause	35,371	0
Adverbial	adv	Adverbial	156,473	7,736
	advcl	Adverbial clause	49,503	1,750
	advnp	Adverbial noun phrase	73,026	480
	neg	Negation	26,373	1,037
	ppmod	Preposition phrase	371,927	4,471
Particle	case	Case marker	420,045	0
	mark	Clausal marker	47,286	0
	prt	Verb particle	13,078	0
Coordination	cc	Coordinating conjunction	131,622	0
	conj	Conjunct	137,128	0
Miscellaneous	com	Compound word	270,326	0
	dep	Unclassified dependency	39,101	0
	disc	Discourse element	14,834	0
	meta	Meta element	19,228	0
	p	Punctuation or symbol	647,505	0
	prn	Parenthetical notation	6,973	0
	root	Root	318,694	0
	voc	Vocative	2,303	0
Referential	r-adv	Referential adv	2,220	0
	r-advcl	Referential advcl	2	0
	r-advnp	Referential advnp	16	0
	r-attr	Referential attr	1	0
	r-comp	Referential comp	1	0
	r-dat	Referential dat	13	0
	r-nsbj	Referential nsbj	17,523	0
	r-obj	Referential obj	1,975	0
	r-ppmod	Referential ppmod	1,409	0
Total			4,618,691	107,995

Table 4: Distributions of the primary and the secondary dependencies for each label. The last two columns show the frequency counts of the primary and the secondary dependencies across all corpora in Table 3, respectively.

	clr	dir	ext	loc	mnr	prp	tmp	Total
csbj	3	0	0	32	4	0	4	43
expl	0	0	0	4	0	0	0	4
nsbj	99	4	0	8	4	1	6	122
comp	5,736	10	0	779	39	89	166	6,819
dat	3	0	0	1	0	0	0	4
obj	546	9	0	22	3	5	6	591
acl	7	0	0	45	1	4	16	73
appo	15	3	0	6	1	0	7	32
attr	0	0	0	17	0	0	5	22
num	3	0	2	0	0	0	44	49
relcl	14	9	2	437	9	14	31	516
adv	1,614	3,448	304	6,725	9,800	1,210	32,172	55,273
advcl	38	24	2	820	648	13,174	10,155	24,861
advnp	0	92	1,113	3,852	441	19	27,678	33,195
neg	0	1	0	1	0	1	1,597	1,600
ppmod	37,502	11,280	531	47,195	8,192	7,492	34,687	146,879
case	164	65	2	67	3	36	87	424
conj	80	31	8	382	48	47	141	737
com	0	2	0	14	4	0	148	168
dep	46	3	0	47	5	8	31	140
disc	0	0	0	0	1	0	1	2
meta	16	11	0	68	14	20	96	225
prn	1	1	0	25	2	4	21	54
root	181	44	1	2,519	95	399	2,682	5,921
r-adv	0	8	2	1,176	43	12	891	2,132
r-advcl	0	0	0	1	0	0	1	2
r-advnp	0	3	1	0	2	1	8	15
r-comp	0	0	0	1	0	0	0	1
r-nsbj	5	0	0	0	0	0	0	5
r-ppmod	140	15	2	273	48	44	95	617
Total	46,213	15,063	1,970	64,517	19,407	22,580	110,776	280,526

Table 5: Confusion matrix between the syntactic and the semantic dependencies. Each cell shows the frequency counts of their overlaps across all corpora in Table 3.

References

[1] Yafa Al-Raheb, Amine Akrout, Josef van Genabith, and Joseph Dichy. DCU 250 Arabic Dependency Bank: An LFG Gold Standard Resource for the Arabic Penn Treebank. In *The Challenge of Arabic for NLP/MT at the British Computer Society*, pages 105–116, 2006.

[2] Daniel Albright, Arrick Lanfranchi, Anwen Fredriksen, William F. Styler, Colin Warner, Jena D. Hwang, Jinho D. Choi, Dmitriy Dligach, Rodney D. Nielsen, James Martin, Wayne Ward, Martha Palmer, and Guergana K. Savova. Towards comprehensive syntactic and semantic annotations of the clinical narrative. *Journal of the American Medical Informatics Association*, 20(5): 922–930, 2013.

[3] Collin F. Baker, Charles J. Fillmore, and John B. Lowe. The Berkeley FrameNet Project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics*, 1998.

[4] Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. Abstract Meaning Representation for Sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, LAW-ID'13, pages 178–186, 2013.

[5] Aoife Cahill, Michael Burke, Ruth O'Donovan, Josef van Genabith, and Andy Way. Long-distance Dependency Resolution in Automatically Acquired Wide-coverage PCFG-based LFG Approximations. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, ACL'04, 2004.

[6] Marie Candito and Djamé Seddah. The sequoia corpus : Syntactic annotation and use for a parser lexical domain adaptation method in french. In *Proceedings of the Joint Conference JEP-TALN-RECITAL*, pages 321–334, 2012.

[7] Marie Candito, Guy Perrier, Bruno Guillaume, Corentin Ribeyre, Karën Fort, Djamé Seddah, and Eric De La Clergerie. Deep Syntax Annotation of the Sequoia French Treebank. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation*, LREC'14, 2014.

[8] Jinho D. Choi and Martha Palmer. Robust Constituent-to-Dependency Conversion for Multiple Corpora in English. In *Proceedings of the 9th International Workshop on Treebanks and Linguistic Theories*, TLT'10, 2010.

[9] Jinho D. Choi and Martha Palmer. Guidelines for the Clear Style Constituent to Dependency Conversion. Technical Report 01-12, University of Colorado Boulder, 2012.

[10] Jinho D. Choi, Amanda Stent, and Joel Tetreault. It depends: Dependency parser comparison using a web-based evaluation tool. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*, ACL'15, pages 387–396, Beijing, China, 2015.

[11] Noam Chomsky. *Lectures in Government and Binding*. Dordrecht, Foris, 1981.

[12] Marie-Catherine de Marneffe and Christopher D. Manning. The Stanford typed dependencies representation. In *Proceedings of the COLING workshop on Cross-Framework and Cross-Domain Parser Evaluation*, 2008.

[13] Marie-Catherine de Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D. Manning. Universal Stanford dependencies: A cross-linguistic typology. In *Proceedings of the 9th International Conference on Language Resources and Evaluation*, LREC'14, pages 4585–4592, 2014.

[14] Daniel Flickinger, Yi Zhang, and Valia Kordoni. DeepBank: A Dynamically Annotated Treebank of the Wall Street Journal. In *Proceedings of the 11th International Workshop on Treebanks and Linguistic Theories. International Workshop on Treebanks and Linguistic Theories*, TLT'12, pages 85–96, 2012.

[15] Ryan Gabbard, Mitchell Marcus, and Seth Kulick. Fully parsing the penn treebank. In *Proceedings of the Conference on Human Language Technology - North American chapter of the Association for Computational Linguistics*, NAACL'06, pages 184–191, 2006.

[16] 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. Announcing Prague Czech-English Dependency Treebank 2.0. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation*, LREC'12, 2012.

[17] Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. The CoNLL-2009 Shared Task: Syntactic and Semantic Dependencies in Multiple Languages. In *Proceedings of the 13th Conference on Computational Natural Language Learning: Shared Task*, CoNLL'09, pages 1–18, 2009.

[18] Julia Hockenmaier and Mark Steedman. Acquiring Compact Lexicalized Grammars from a Cleaner Treebank. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation*, LREC'02, 2002.

[19] Julia Hockenmaier and Mark Steedman. CCGbank: A Corpus of CCG Derivations and Dependency Structures Extracted from the Penn Treebank. *Computational Linguistics*, 33(3):355–396, 2007.

[20] Liang Huang and Kenji Sagae. Dynamic Programming for Linear-Time Incremental Parsing. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL'10, 2010.

[21] Jena D. Hwang, Archna Bhatia, Clare Bonial, Aous Mansouri, Ashwini Vaidya, Nianwen Xue, and Martha Palmer. PropBank Annotation of Multilingual Light Verb Constructions. In *Proceedings of ACL workshop on Linguistic Annotation*, LAW'10, pages 82–90, 2010.

[22] Richard Johansson and Pierre Nugues. Extended Constituent-to-dependency Conversion for English. In *Proceedings of the 16th Nordic Conference of Computational Linguistics*, NODALIDA'07, 2007.

[23] John Judge, Aoife Cahill, and Josef van Genabith. QuestionBank: Creating a Corpus of Parse-Annotated Questions. In *Proceedings of the 44th Annual Meeting of the Association for Computational Linguistics*, ACL'06, pages 497–504, 2006.

[24] Matthias T. Kromann. The Danish Dependency Treebank and the underlying linguistic theory. In *Proceedings of the Second Workshop on Treebanks and Linguistic Theories*, TLT'03, 2003.

[25] Sandra Kübler, Ryan T. McDonald, and Joakim Nivre. *Dependency Parsing*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers, 2009.

[26] Mitchell Marcus, Grace Kim, Mary Ann Marcinkiewicz, Robert Macintyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. The Penn Treebank: Annotating Predicate Argument Structure. In *ARPA Human Language Technology Workshop*, pages 114–119, 1994.

[27] Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993.

[28] Adam Meyers, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. Annotating Noun Argument Structure for NomBank. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, LREC'04, 2004.

[29] Yusuke Miyao, Takashi Ninomiya, and Jun'ichi Tsujii. Corpus-Oriented Grammar Development for Acquiring a Head-driven Phrase Structure Grammar from the Penn Treebank. In *Proceedings of the 1st International Joint Conference on Natural Language Processing*, IJCNLP'04, pages 684–693, 2005.

[30] Joakim Nivre. An Efficient Algorithm for Projective Dependency Parsing. In *Proceedings of the 8th International Workshop on Parsing Technologies*, IWPT'03, pages 149–160, 2003.

[31] Joakim Nivre. *Inductive Dependency Parsing*. Springer, 2006.

[32] Joakim Nivre, Johan Hall, Jens Nilsson, Gülşen Eryiğit, and Svetoslav Marinov. Labeled Pseudo-Projective Dependency Parsing with Support Vector Machines. In *Proceedings of the 10th Conference on Computational Natural Language Learning*, CoNLL'06, pages 221–225, 2006.

[33] Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. Universal Dependencies v1: A Multilingual Treebank Collection. In *Proceedings of the 10th International Conference on Language Resources and Evaluation*, LREC'16, pages 23–28, 2016.

[34] Martha Palmer, Daniel Gildea, and Paul Kingsbury. The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational Linguistics*, 31(1): 71–106, 2005.

[35] Slav Petrov and Ryan McDonald. Overview of the 2012 Shared Task on Parsing the Web. In *Proceedings of the 1st Workshop on Syntactic Analysis of Non-Canonical Language*, SANCL, 2012.

[36] Sebastian Schuster and Christopher D. Manning. Enhanced English Universal Dependencies: An Improved Representation for Natural Language Understanding Tasks. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation*, LREC'16, 2016.

[37] Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. The CoNLL-2008 Shared Task on Joint Parsing of Syntactic and Semantic Dependencies. In *Proceedings of the 12th Conference on Computational Natural Language Learning: Shared Task*, CoNLL'08, pages 59–177, 2008.

[38] Ralph Weischedel, Eduard Hovy, Martha Palmer, Mitch Marcus, Robert Belvin, Sameer Pradhan, Lance Ramshaw, and Nianwen Xue. OntoNotes: A Large Training Corpus for Enhanced Processing. In Joseph Olive, Caitlin Christianson, and John McCary, editors, *Handbook of Natural Language Processing and Machine Translation*. Springer, 2011.

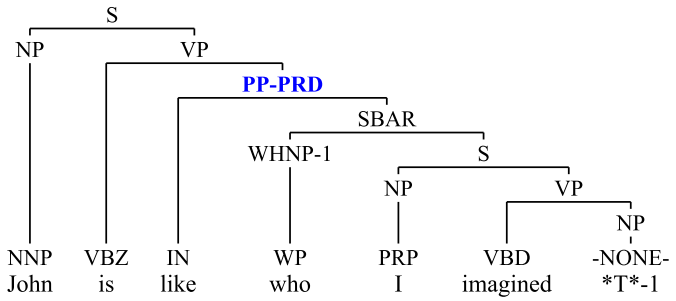
[39] Fei Xia. Extracting Tree Adjoining Grammars from Bracketed Corpora. In *In Proceedings of the Fifth Natural Language Processing Pacific Rim Symposium*, 1999.

[40] Nianwen Xue, Fei Xia, Fu-Dong Chiou, and Martha Palmer. The penn chinese treebank: Phrase structure annotation of a large corpus. *Natural Language Engineering*, 11(2):207–238, 2005.

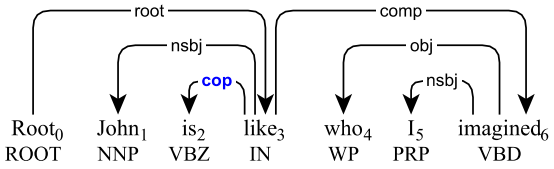
[41] Hiroyasu Yamada and Yuji Matsumoto. Statistical dependency analysis with support vector machine. In *Proceedings of the 8th International Workshop on Parsing Technologies*, IWPT'03, pages 195–206, 2003.

[42] Kun Yu, Miyao Yusuke, Xiangli Wang, Takuya Matsuzaki, and Junichi Tsujii. Semi-automatically Developing Chinese HPSG Grammar from the Penn Chinese Treebank for Deep Parsing. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING'10, pages 1417–1425, 2010.

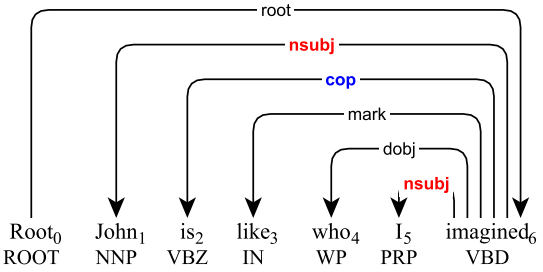
Figures



(a) Penn Treebank (PTB).

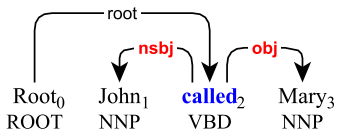


(b) Deep Dependency Graph (DDG).

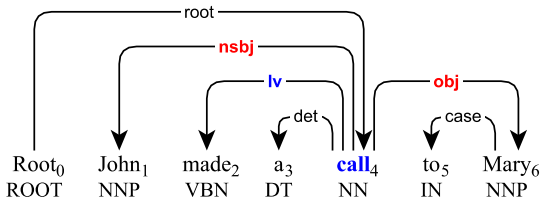


(c) Universal Dependencies (UD).

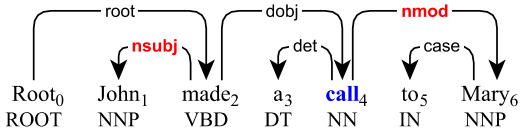
Figure 1: Examples of non-verbal predicates constructed by copulas (`cop`), which are identified by the function tag `PRD` in PTB (Figure 1a). The preposition becomes the head of a preposition phrase when it is a predicate in DDG (Figure 1b), whereas it is not the case in UD (Figure 1c) such that the verbal predicate *imagine* ends up having two nominal subjects (`nsubj`).



(a) Deep Dependency Graph (DDG).

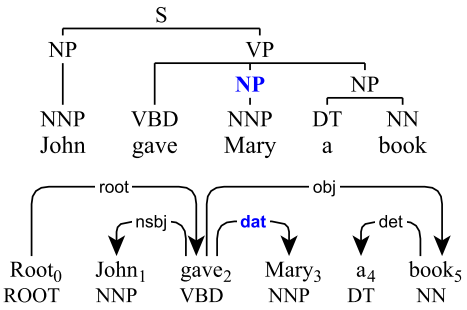


(b) Deep Dependency Graph (DDG).

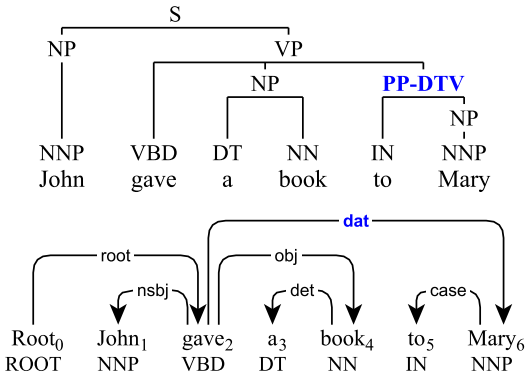


(c) Universal Dependencies (UD).

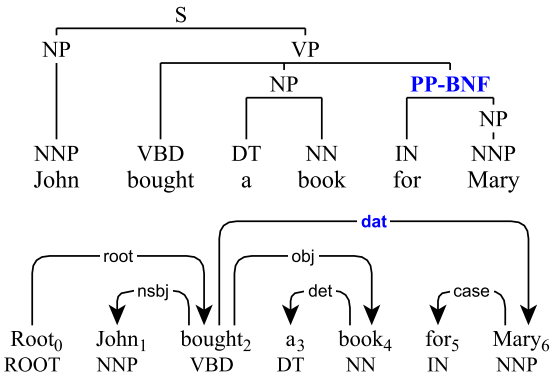
Figure 2: Examples of non-verbal predicates constructed by light verbs (`lv`). Compared to the one without a light verb construction (Figure 2a), the relations between the predicate *call* and its arguments *John* and *Mary* stay the same in DDG with the light verb construction (Figure 2b), whereas it is not the case in UD (Figure 2c).



(a) Penn Treebank (PTB; top) and Deep Dependency Graph (DDG; bottom).

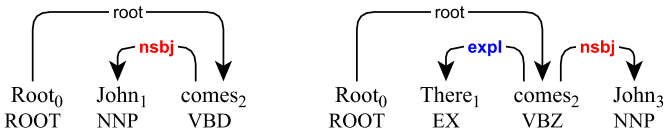


(b) Penn Treebank (PTB; top) and Deep Dependency Graph (DDG; bottom).

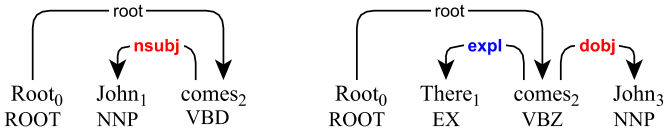


(c) Penn Treebank (PTB; top) and Deep Dependency Graph (DDG; bottom).

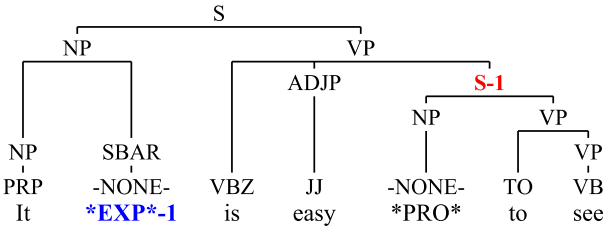
Figure 3: Examples of datives. Indirect objects (Figure 3a), preposition phrases with the function tag `DTV` (dative; Figure 3b), and preposition phrases with the function tag `BNF` (benefactive; Figure 3c) are considered datives (`dat`).



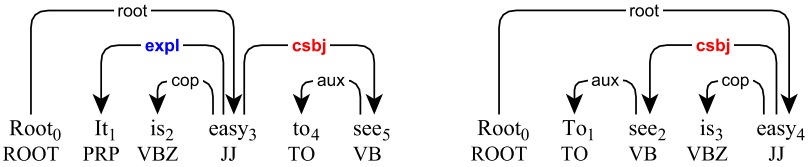
(a) Deep Dependency Graph (DDG).



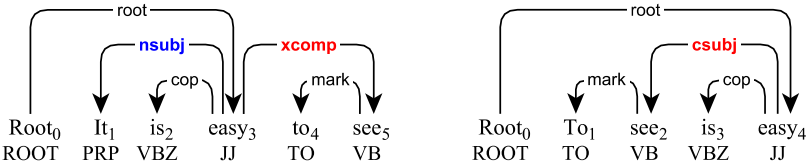
(b) Universal Dependencies (UD).



(c) Penn Treebank (PTB).

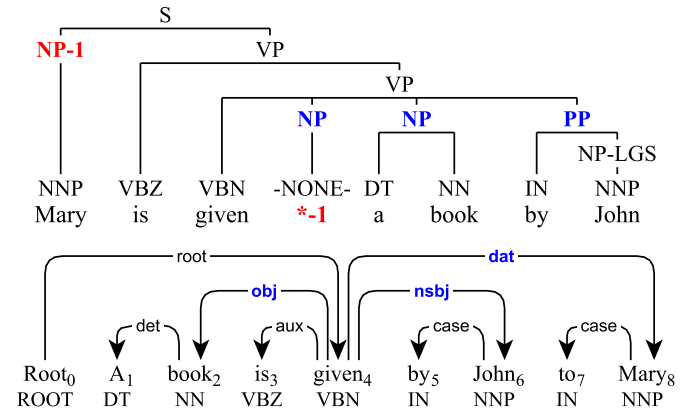


(d) Deep Dependency Graph (DDG).

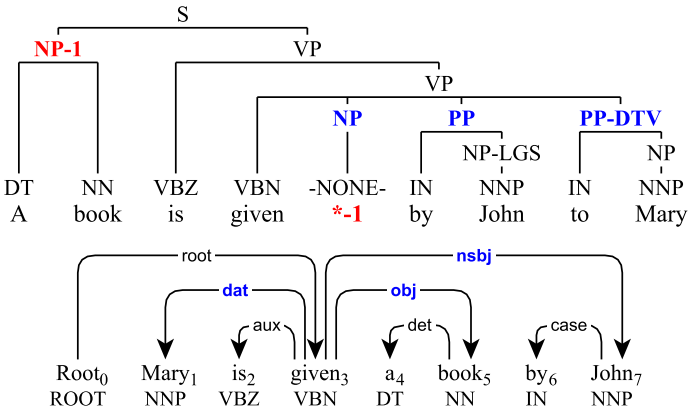


(e) Universal Dependencies (UD).

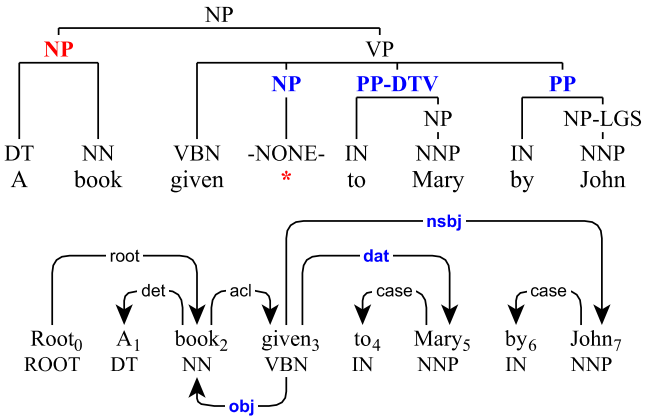
Figure 4: Examples of expletives (`expl`) where the subject relations are consistently represented with the existential *there* and the extrapositional *it* in DDG (Figures 4a and 4d) regardless of their syntactic alternations, whereas it is not the case in UD (Figures 4b and 4e). The extrapositional *it* and its referent clause can be identified by the function tag `*EXP*` (Figure 4c).



(a) Penn Treebank (PTB; top) and Deep Dependency Graph (DDG; bottom).

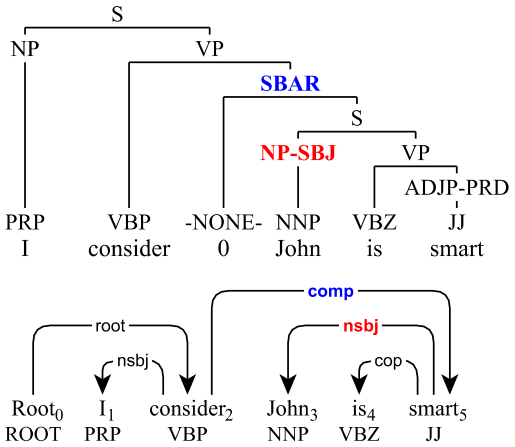


(b) Penn Treebank (PTB; top) and Deep Dependency Graph (DDG; bottom).

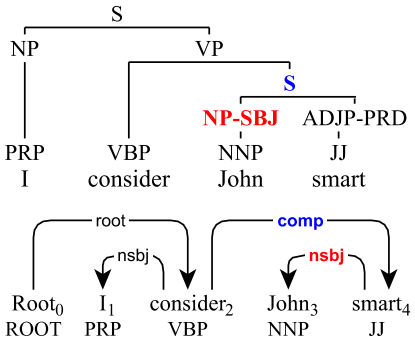


(c) Penn Treebank (PTB; top) and Deep Dependency Graph (DDG; bottom).

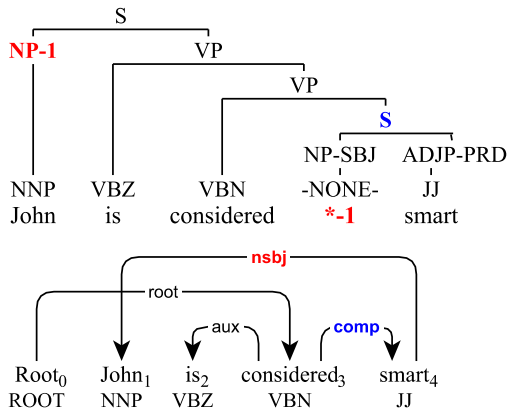
Figure 5: Examples of passive constructions where the relations between the predicate *give* and its arguments, *John*, *Mary*, and *book*, stay the same as the ones in the active construction (Figure 3a). The object in the reduced passive construction, *book*, is represented by the secondary dependency in Figure 5c to avoid the cyclic relation among the primary dependencies.



(a) Penn Treebank (PTB; top) and Deep Dependency Graph (DDG; bottom).

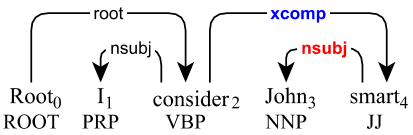


(b) Penn Treebank (PTB; top) and Deep Dependency Graph (DDG; bottom).

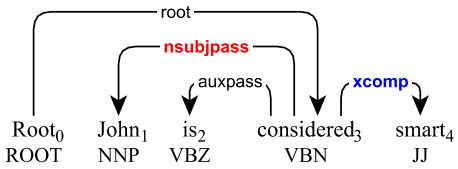


(c) Penn Treebank (PTB; top) and Deep Dependency Graph (DDG; bottom).

Figure 6: Examples of small clauses where *John* is consistently recognized as the subject of the adjectival predicate *smart* in the declarative clause (Figure 6a), the small clause (Figure 6b), and the small clause in the passive construction (Figure 6c). The subject relation in Figure 6c causes the non-projective dependency, which can be handled well by most recent dependency parsers.

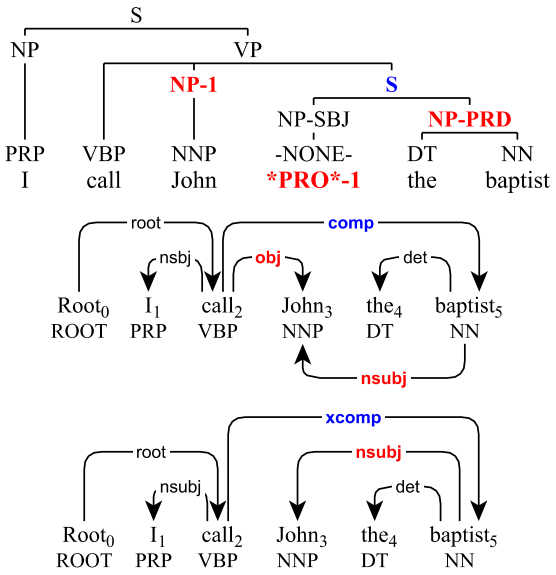


(a) Universal Dependencies (UD).

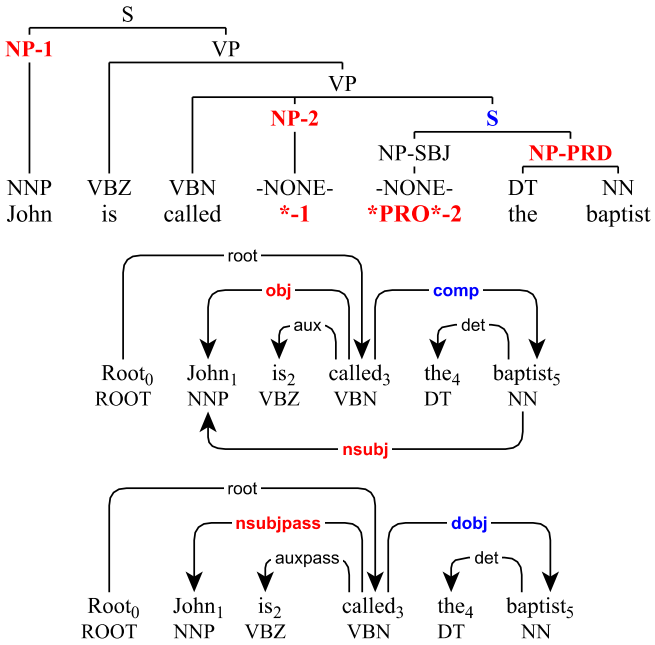


(b) Universal Dependencies (UD).

Figure 7: Examples of small clauses with internal subjects in UD where *John* is recognized as the subject of the adjectival predicate *smart* in the active construction (Figure 7a) but not in the passive construction (Figure 7b).

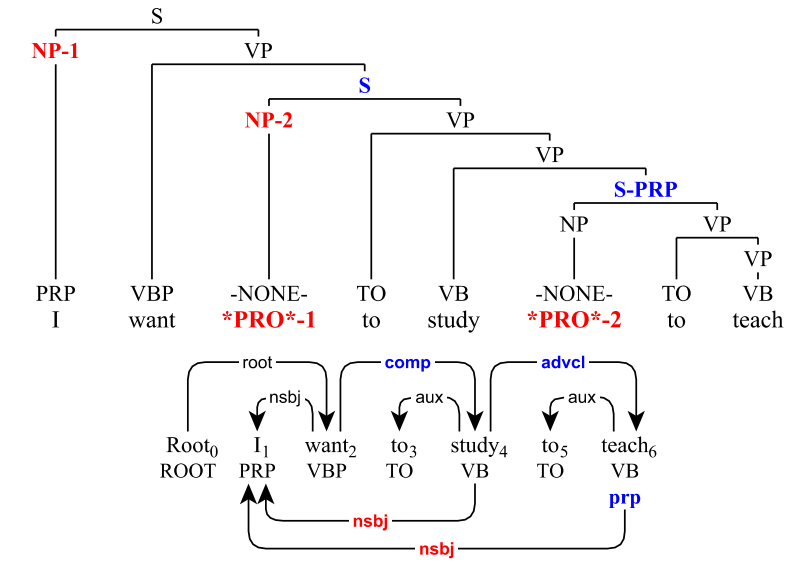


(a) Penn Treebank (top), Deep Dependency Graph (middle), and Universal Dependencies (bottom).

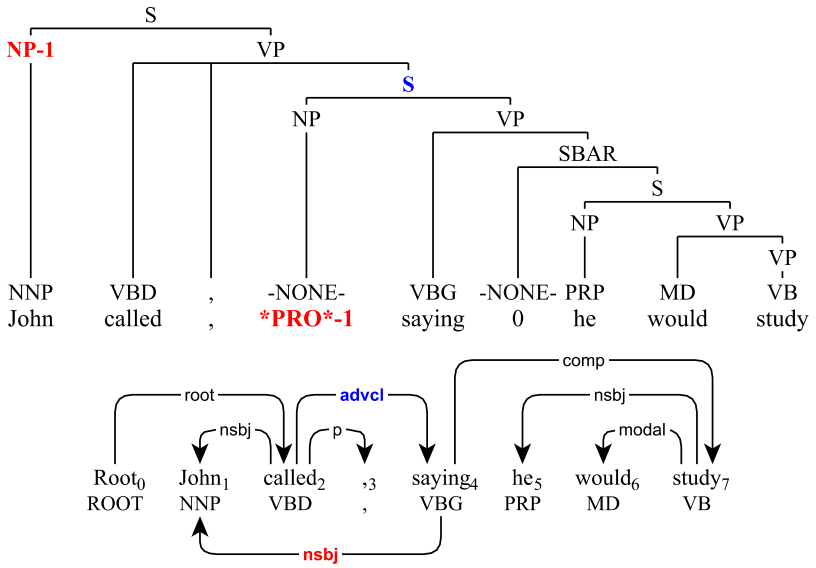


(b) Penn Treebank (top), Deep Dependency Graph (middle), and Universal Dependencies (bottom).

Figure 8: Examples of small clauses with external subjects where *John* is consistently recognized as the object of the verbal predicate *call* and the subject of the nominal predicate *baptist* for both the active (Figure 8a) and the passive (Figure 8b) constructions in DDG, whereas it is not the case in UD. The subject relation between *John* and *baptist* is preserved by the secondary dependency.

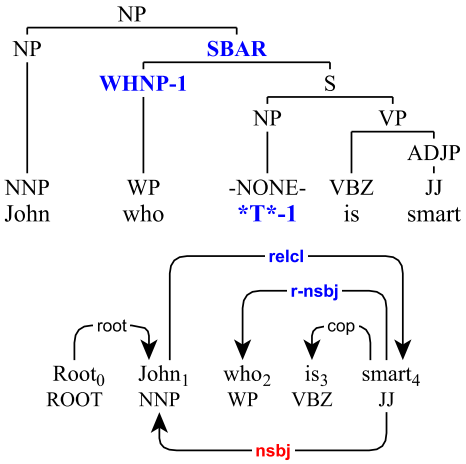


(a) Penn Treebank (PTB; top) and Deep Dependency Graph (DDG; bottom).

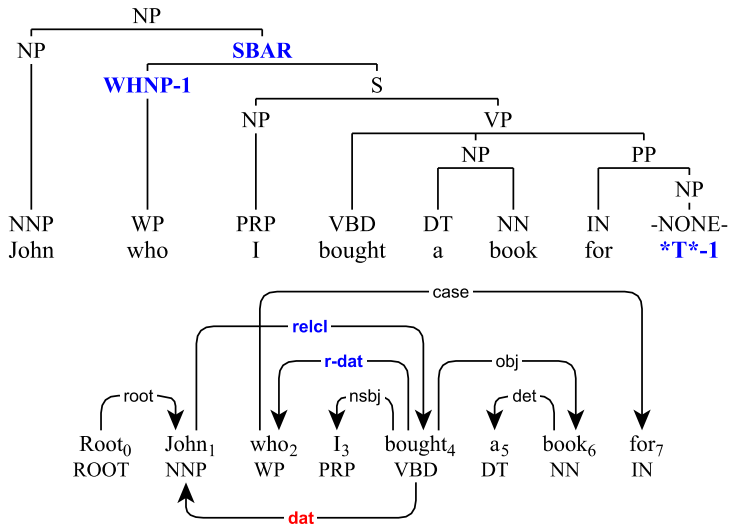


(b) Penn Treebank (PTB; top) and Deep Dependency Graph (DDG; bottom).

Figure 9: Examples of open clauses where the external subjects are indicated by the secondary dependencies to avoid multiple heads among the primary dependencies. Notice that the head of the open clause, *teach*, is also assigned with the semantic role `prp` (purpose) extracted from the function tag `PRP`.

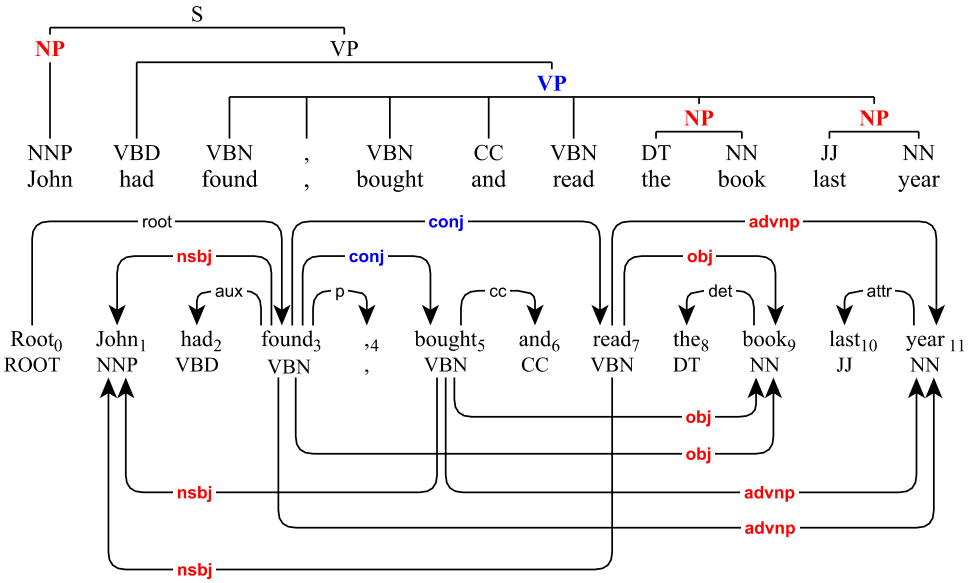


(a) Penn Treebank (PTB; top) and Deep Dependency Graph (DDG; bottom).

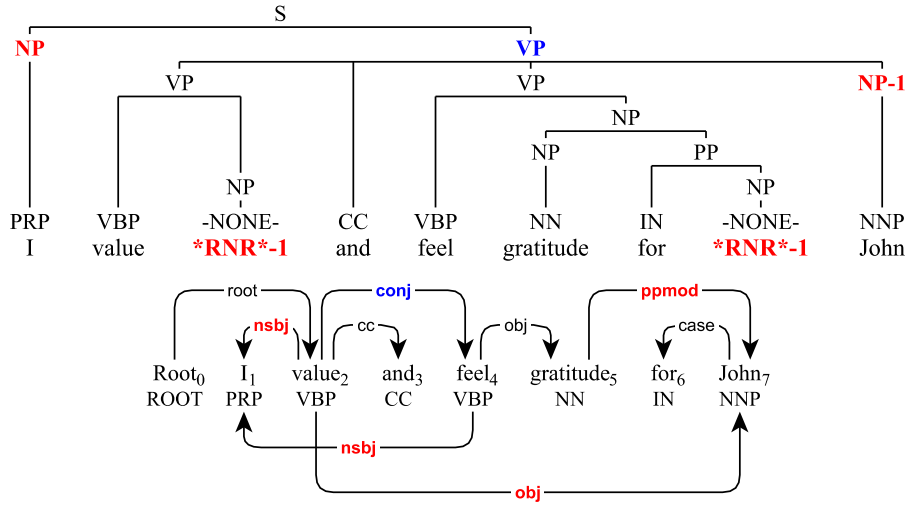


(b) Penn Treebank (PTB; top) and Deep Dependency Graph (DDG; bottom).

Figure 10: Examples of relative clauses (`relcl`) where the relativizers are assigned with the dependency relations (`r-*`) from their original positions indicated by the empty category `*T*-d`. Referent relations to these relativizers are represented by the secondary dependencies to avoid cyclic relations among the primary dependencies.

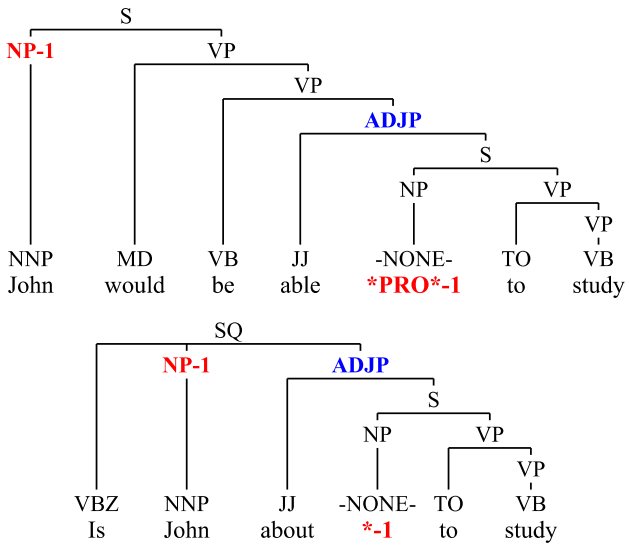


(a) Penn Treebank (PTB; top) and Deep Dependency Graph (DDG; bottom).

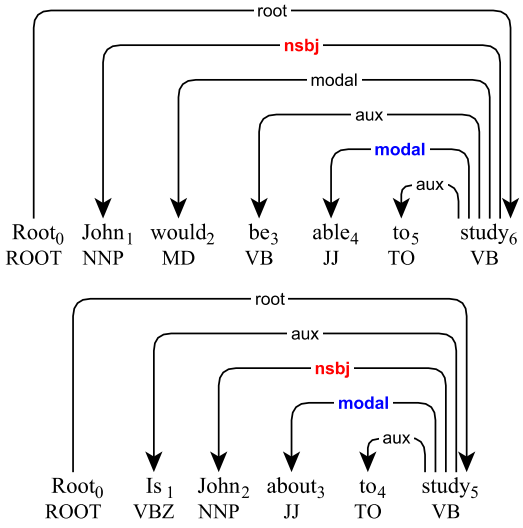


(b) Penn Treebank (PTB; top) and Deep Dependency Graph (DDG; bottom).

Figure 11: Examples of coordinated structures (`conj`) with (Figure 11a) and without (Figure 11b) right node raising, indicated by the empty category `*RNR*-d`. The arguments in the coordinations are represented by the secondary dependencies.

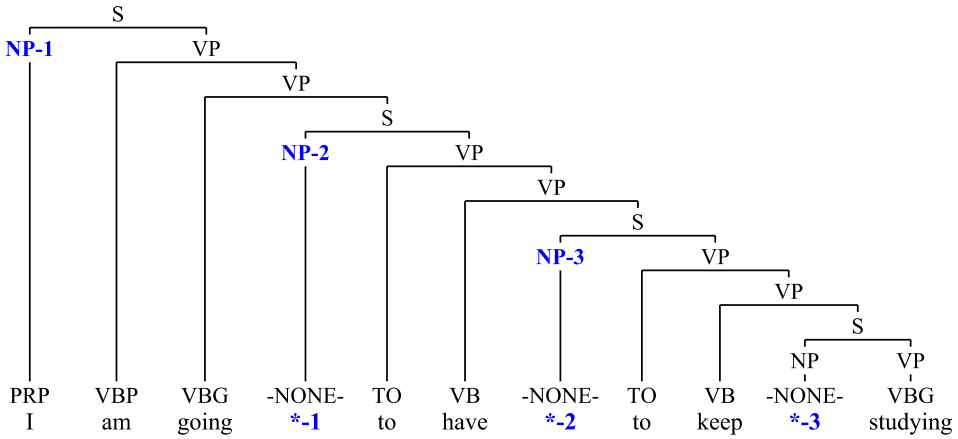


(a) Penn Treebank (PTB; top) and Deep Dependency Graph (DDG; bottom).

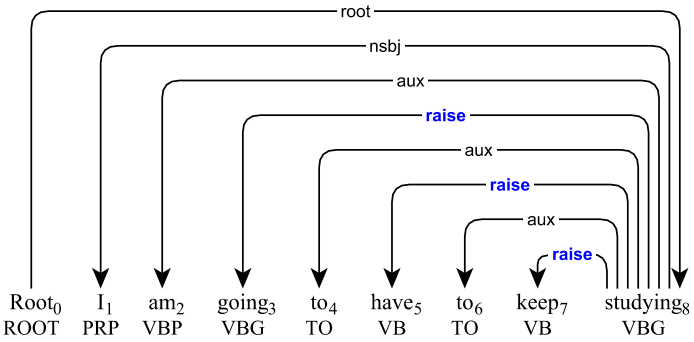


(b) Penn Treebank (PTB; top) and Deep Dependency Graph (DDG; bottom).

Figure 12: Examples of modal adjectives, followed by the clauses whose subjects are linked to the subjects of the adjectival predicates, *able* and *about* in Figures 12a and 12b, respectively.



(a) Penn Treebank (PTB).



(b) Deep Dependency Graph (DDG).

Figure 13: Examples of raising verbs, followed by the clauses whose subjects are the empty categories `*-d` linking to the subject of the raised verb, *study*.