Integrating Headedness Information into an Auto-generated Multilingual CCGbank for Improved Semantic Interpretation

Tu-Anh Tran, Yusuke Miyao
The University of Tokyo
7-3-1 Hongo, Bunkyo, Tokyo, Japan
{anh, yusuke}@is.s.u-tokyo.ac.jp

Abstract

Previously, we introduced a method to generate a multilingual Combinatory Categorial Grammar (CCG) treebank by converting from the Universal Dependencies (UD). However, the method only produces bare CCG derivations without any accompanying semantic representations, which makes it difficult to obtain satisfactory analyses for constructions that involve non-local dependencies, such as control/raising or relative clauses, and limits the general applicability of the treebank. In this work, we present an algorithm that adds semantic representations to existing CCG derivations, in the form of predicate-argument structures. Through hand-crafted rules, we enhance each CCG category with headedness information, with which both local and non-local dependencies can be properly projected. This information is extracted from various sources, including UD, Enhanced UD, and proposition banks. Evaluation of our projected dependencies on the English PropBank and the Universal PropBank 2.0 shows that they can capture most of the semantic dependencies in the target corpora. Further error analysis measures the effectiveness of our algorithm for each language tested, and reveals several issues with the previous method and source data.

Keywords: combinatory categorial grammar, treebank, multilingual

1. Introduction

Combinatory Categorial Grammar (Steedman, 2000) is a grammar formalism well-known for offering a transparent interface between syntax and semantics. In CCG, each word in a phrase receives a syntactic *category*, which can then be combined with other categories using predefined combinatory rules to form a complete syntactic derivation. A semantic representation of the phrase can be compositionally obtained by simply assigning a semantic representation to each lexical category and tracing through the syntactic derivation. This property of CCG makes it an attractive target for those looking to build interpretable Natural Language Processing (NLP) systems (Curran et al., 2007; Blodgett and Schneider, 2019; Stanojević et al., 2023).

We previously proposed a method (Tran and Miyao, 2022) to automatically generate a multilingual CCGbank by converting from the Universal Dependencies (Nivre et al., 2020). The purpose is to quickly and cheaply acquire a large dataset of CCG syntactic derivations in many languages, which could be useful for downstream NLP tasks. However, the generated CCGbank only contains bare syntactic derivations without accompanying semantic representations, making its applicability to other tasks limited. In addition, analyses for complex constructions that involve non-local dependencies, such as control/raising or relative clauses, cannot be easily obtained. While manual annotation of semantic representations has been done in the past for some individual languages (Bos et al., 2004; Martínez-Gómez et al., 2016), applying the same approach to a large multilingual CCGbank requires considerable expertise in multilingual semantics.

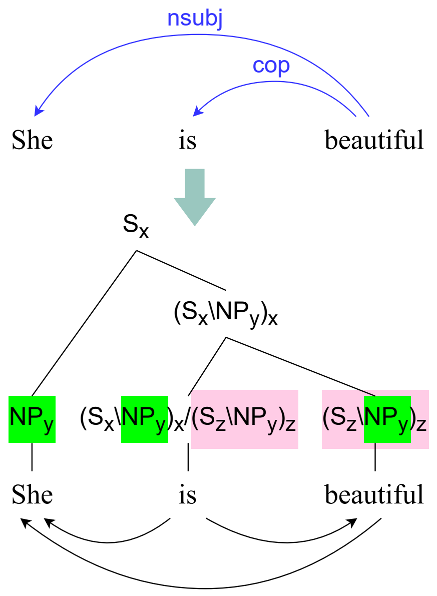

Figure 1: UD dependencies (blue arcs) and predicate-argument dependencies projected by our CCG categories (black arcs) for the sentence *She is beautiful*. Headedness information is marked using indices (x, y, z). Categories that share the same head word are marked with the same index. By examining the indices, one can identify that *She* is an argument of both *is* and *beautiful*, and *beautiful* is an argument of *is*.

In this work, we present an algorithm that adds semantic representations to the bare CCG categories obtained from UD conversion. Our representation of choice is predicate-argument structures. We mark each category with headedness information, based on which predicate-argument dependencies can be projected. This headedness information is derived from the word-word dependencies in UD and other complementary resources, including the Enhanced UD graphs, the English PropBank (Palmer et al., 2005), and the Universal PropBank 2.0 (Jindal et al., 2022). Since these complementary resources also include annotations for non-local dependencies, we are able to alleviate the problem of our previously proposed method (which we will henceforth refer to as the T&M method). The end result of our work is an improved multilingual CCGbank where every CCG derivation has a corresponding semantic representation.

An overview of our new algorithm[1] is illustrated in Figure 1:

- First, we binarize the input dependency tree and assign a bare category to each node in the binarized tree (this step is carried over from the T&M method).
- From top to bottom, *as the categories are being assigned*, we mark the head of each category with a variable using hand-crafted rules. The head variables are passed down through the tree, a reverse process of the typical unification process in CCG (Clark et al., 2002; Hockenmaier and Steedman, 2007).
- Once we hit the bottom of the tree, the head word that corresponds to each variable can be identified. By looking at the arguments of each functor category, we are then able to project predicate-argument dependencies based on this headedness information.

The idea of adding headedness information to categories is not new (Clark et al., 2002; Hockenmaier and Steedman, 2007; Ambati et al., 2018; Tse and Curran, 2010), but has not been done for UD-based multilingual CCGbanks. Compared to the T&M method, which only considers basic UD relations, our new rules are more extensive.

To evaluate the quality of the projected predicate-argument structures, we perform experiments on the English PropBank and the Universal PropBank 2.0, and measure how well semantic dependencies in the proposition banks can be recovered. On average, we achieve 83.50% recall on 12 treebanks of 8 languages tested. Error analysis shows that the effectiveness of our algorithm is tied to the quality of the source treebanks, in addition to the accuracy of the bare categories. We point out possible areas of improvement for the original T&M algorithm and the annotations of the dependency treebanks.

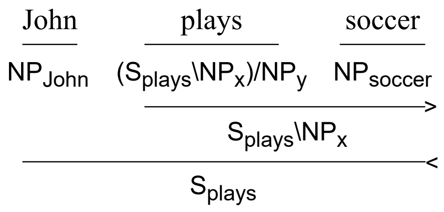

Figure 2: A CCG derivation for the sentence *John plays soccer*. Note that this is the traditional way of displaying a CCG derivation in literature. However, in this paper, we show the derivations in the form of tree-shape structures (Figure 1) to better illustrate our top-down approach.

2. Background

2.1. CCG Dependencies

In CCG, syntactic information is encoded in each *category* that is assigned to each constituent in a sentence. There are two types of categories: *atomic* categories and *functor* categories. Atomic categories, in the context of this work, include S (assigned to sentences) and NP (assigned to noun phrases). Functor categories can either be $X\backslash Y$ or X/Y, where X and Y are themselves categories. For example, in Figure 2, the transitive verb *plays* receives a functor category $(S\backslash NP)/NP$. The slashes indicate that *plays* takes an NP argument to the left (indicated by a backslash $\backslash$ before NP), and an NP argument to the right (indicated by a forward slash $/$ before the second NP), and returns a sentence S as a result.

CCG predicate-argument dependencies are projected from words with functor categories to their arguments following the derivation process. To produce these dependencies, we first need to mark the head word of each category with a variable (denoted by the indices in our figures for illustrative purposes). NP_{John} means that the head word of this NP is *John*. $(S_{plays}\backslash NP_x)/NP_y$ means that the final sentence is headed by *plays*, awaiting an NP headed by an unknown word x and another NP headed by an unknown word y. To identify x and y, we follow the derivation and apply head unification. For instance, when the category of *plays* combines with the category of *soccer* in the first step of the derivation in Figure 2, we may unify y and $soccer$, and thus establish a dependency from *plays* to *soccer*. Note that only two categories can combine at a time; as a result, a CCG derivation can be illustrated as a binary tree.

[1]Our implementation is available at `https://github.com/mynlp/ud2ccg-code`.

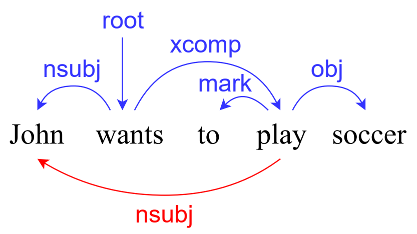

Figure 3: A UD tree for the sentence *John wants to play soccer*. Basic UD dependencies are shown in blue. Enhanced dependencies are in red.

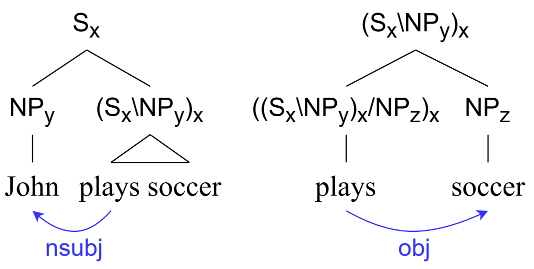

Figure 4: Head assignment rule for head-argument relations.

2.2. UD Dependencies

UD annotates word-word syntactic dependencies following a cross-linguistically consistent guideline, with treebanks available for over 100 languages. Some UD trees (Figure 3) may contain extra annotations to denote non-local dependencies (e.g., through control/raising, coordination, or relative clauses). These annotations are called *Enhanced Dependencies* (EUD). Basic UD dependencies form a tree structure; however, this is not a requirement for EUD dependencies.

2.3. UD to CCGbank

In this section, we give a brief overview of our previously proposed algorithm that converts a UD tree to a CCG derivation.

There are three main steps:

- First, the basic UD tree is binarized so that each node has at most two children. Since UD does not limit the number of dependents a word may have, this step is done to match the typical binary structure of a CCG derivation.
- Second, we assign a category to each node in the binarized tree using hand-crafted rules. The result of this step is similar to the binary tree in Figure 1, but without head indices.
- Finally, we correct problematic categories found in the second step with another set of rules.

We mostly follow the same process for assigning syntactic categories to create a final CCG derivation, and will not go into detail in this paper. What we will focus on is our new rules for adding headedness information to these categories. We explain our new method in Section 3.

3. Method

The main goals of our approach are to (1) preserve the content-head nature of UD dependencies, and (2) preserve the non-local dependencies specified by EUD. Compared to other dependency grammars, UD prefers content words as heads instead of function words to maximize parallelism among annotations of different languages. Converting back to function-head dependencies would risk cross-linguistic consistency. For this reason, our headedness rules may diverge from other works in CCG in several constructions, as explained below.

The rules are applied to subtrees of a binarized dependency tree in a top-down, recursive manner. Assuming the head variable of a result category is already assigned in the preceding subtree, our rules assign head variables to the functor and argument categories. Once we reach the bottom of the tree, the head word that corresponds to each head variable is revealed, allowing us to project appropriate CCG dependencies.

3.1. General cases

Rule for head-argument relations (`nsubj`, `csubj`, `obj`, `iobj`, `xcomp`, `ccomp`, `expl`) We:

- set the head of the functor category to be the same as the head of the result category,
- initialize a new head variable for the argument category.

An example is given in Figure 4. Assuming that the bare categories are assigned using the T&M algorithm, we first set *x* as the head variable of *S*, and *y* as the head variable of *NP*. The functor category of *plays soccer* can be deduced given the result *S* and the argument *NP*. As we reach the bottom of the tree, we are able to identify *y* as *John*, and *x* as *plays* (the head word of an entire lexical category is the word assigned that category). With this rule, we can see that *plays* is the head of itself, the verb phrase *plays soccer*, and the whole sentence *John plays soccer*. Given the functor category of *plays*, we can project a dependency from *plays* to *John* (and similarly from *plays* to *soccer*).

Rule for head-modifier relations (the rest of UD relations, with the exception of `cop`, `conj`, `cc`, and `punct`) We:

- initialize a new head variable for the functor category,

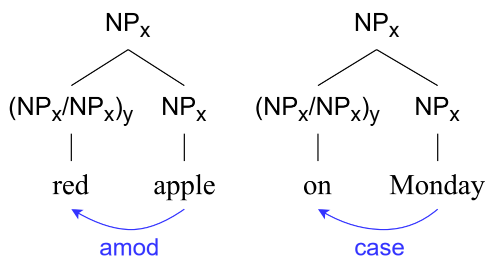

Figure 5: Head assignment rule for head-modifier relations.

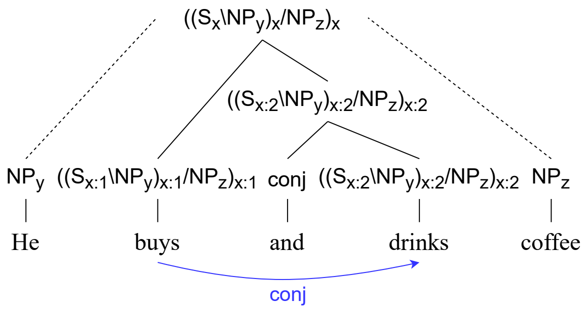

Figure 6: Head assignment rule for coordination.

- set the head of the argument category to be the same as the head of the result category.

Note that function words such as case markers are also included in the list of head-modifier relations. This allows the content word to be the head in cases like adpositional phrases (Figure 5), which diverges from the common analyses in English.

Rule for coordination Both conjuncts should share the same category, and thus share the same head structure within the category. By doing this, dependencies can be mirrored between the conjuncts. To distinguish between the two conjuncts, sub-indices are used. An advantage of this approach is that in cases such as verb coordination, it is very clear from the derivation that both verbs share the same subject and object (Figure 6).

Rule for punctuation marks We do not project dependencies from punctuation marks. A punctuation mark's category is initialized with a new head variable.

3.2. Special cases

Rule for copulas Even though copulas are function words in UD, we treat them in the same way as other head-argument relations. As a result, a copula should be the head of a CCG predicate-argument dependency. This treatment is consistent with that of the English CCGbank (Hockenmaier and Steedman, 2007), as well as other PropBanks

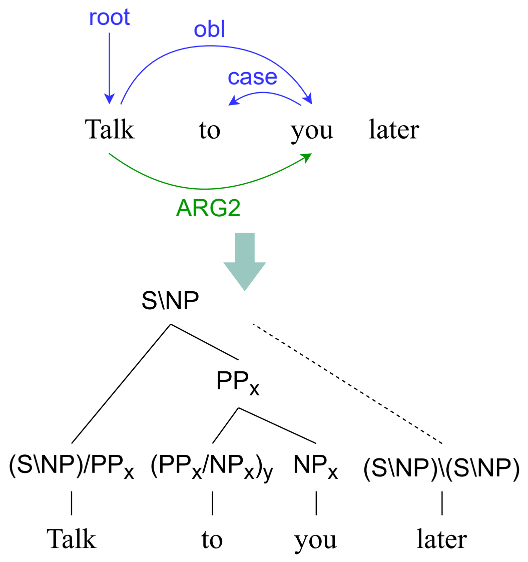

Figure 7: Head assignment rule for adpositional phrases. UD dependencies are shown in blue, and PropBank dependencies are in green. Some head variables are omitted for conciseness.

(Palmer et al., 2005; Jindal et al., 2022). In addition, because a copula links a subject to a nonverbal predicate, we co-index the NP argument of the result category and the NP argument of the argument category (if it exists) so that they share the same head. An example is shown in Figure 1. This enables a dependency from the predicative adjective *beautiful* to its subject *She* to be projected.

Rule for adpositional phrases A drawback of our previous work is that we were not able to design a category assignment rule for adpositional phrases, due to the lack of argument-adjunct distinction in UD. We attempt to handle this problem by relying on annotations from the Universal PropBank 2.0 (UP 2.0) and the English PropBank. UP 2.0 provides semantic role labels for treebanks of 23 languages in UD, while the English PropBank provides annotations for the English Web Treebank (EWT), so we are able to extract necessary semantic dependencies for our rule. Specifically, we consider an oblique modifier (`obl`) or a nominal modifier (`nmod`) to be an argument (rather than an adjunct) if (1) it is annotated as a core argument (ARG0-ARG5) in the proposition banks, (2) it has a case marker (`case`) dependent, and (3) the case marker has a part-of-speech tag `ADP` (for adpositions) in UD. Consequently, we set the category of the modifier to be PP, and the head assignment rule follows that of other head-modifier relations (Figure 7).

Rule for non-local dependencies specified by EUD EUD adds extra annotations for several cases of non-local dependencies. In this work, we design head assignment rules for control/raising and relative clauses based on these enhancements. One type of enhancement that we do not deal with is related to ellipsis, since ellipsis is not handled by the original T&M algorithm, and also cannot be analyzed with standard CCG rules (Steedman, 2000).

For control/raising, if there exists an *NP* argument in the category of the clausal complement, and if there exists a core EUD dependency (e.g., `nsubj`, `obj`, `iobj`) from the head of the clausal complement, we mark the head of the *NP* argument to be the same as the head of the dependent of the EUD dependency. Similarly for relative clauses, if there exists an *NP* argument in the category of the relative clause, and if there exists a core EUD dependency from the head of the relative clause, we also mark the head of the *NP* argument to be the same as the head of the dependent of the EUD dependency.

For example, in the case of control/raising (top figure), the *NP* category of *John* shares the same head variable with the *NP* argument of *to play soccer* (and *play soccer*).

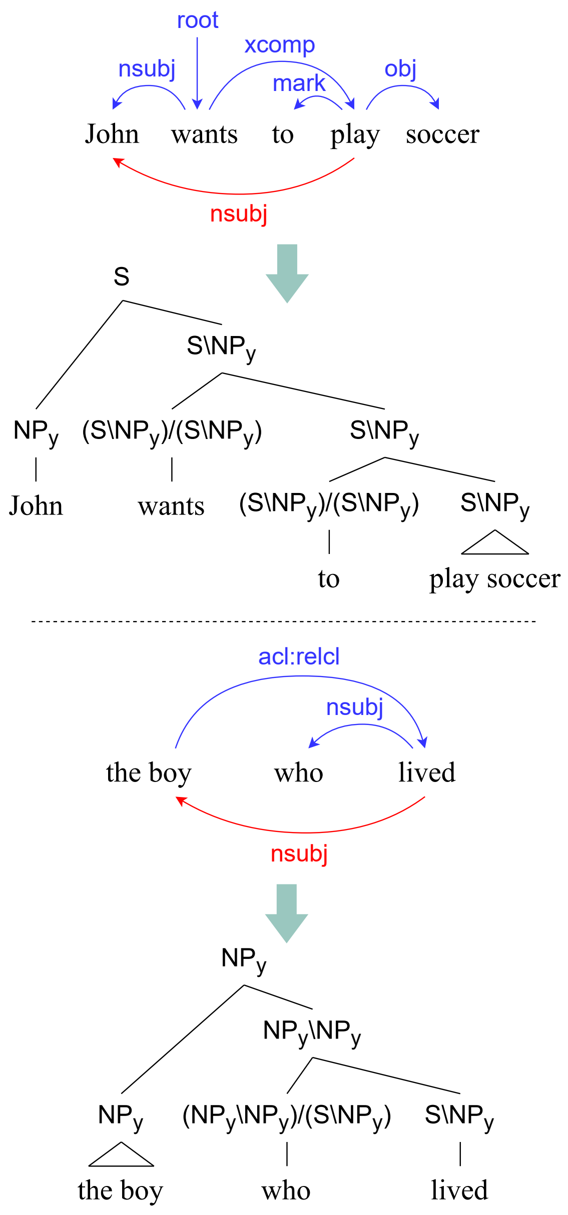

Figure 8: Head assignment rule for non-local dependencies specified by EUD. UD dependencies are shown in blue; EUD dependencies are in red. The top figure shows an example of our handling of subject control, while the bottom figure shows an example of our rule applied to a relative clause.

4. Experiments

4.1. Experimental setup

Data To verify the quality of the predicate-argument dependencies extracted from our multilingual CCGbank, we compare them against the semantic dependencies in UP 2.0. As previously mentioned, UP is built upon the same text corpora as UD, enabling our comparison. For conversion, we select UD treebanks that (1) contain EUD annotations, and (2) have a corresponding UP 2.0 corpus. For English in particular, we compare with the EWT section of the English PropBank, since UP 2.0 does not provide an English corpus. The version of UD we use is v2.9 to align with UP 2.0.

Evaluation metric Comparing CCG dependencies and UP dependencies is not a straightforward task, due to the differences in head selection. For example, a determiner or a preposition may be the head in a CCG dependency projected by its functor category, but this is not usually the case in UP. For this reason, we apply the following rules to facilitate a better comparison:

- Reverse the direction of dependencies projected from modifiers: In CCG, a modifier usually receives a functor category $X|X$, which makes it the head of the argument that it modifies. We reverse the role so that the modifier now becomes a dependent of its argument, resembling the design of UP.
- Remove predicates that are not in UP/PropBank: Since CCG parses tend to generate a lot more dependencies than what is available in UP (e.g., dependencies projected from nouns), and since UP's predicate identification is noncomprehensive (Jindal et al., 2022), we ignore predicates that are not relevant to the comparison.

The metric we employ is an unlabelled *within-constituent* metric (Lewis et al., 2015), which considers a CCG dependency to be correct if its de-

Treebank (dev)	CR	Baseline			Baseline + cop			Baseline + cop + ap			Baseline + cop + ap + eud		
		R	P	F1	R	P	F1	R	P	F1	R	P	F1
Czech-CAC	91.38	78.30	59.89	67.87	78.78	59.93	68.08	78.72	60.81	68.61	82.35	61.49	70.40
Czech-FicTree	88.92	82.43	58.17	68.21	82.56	58.15	68.24	82.98	59.89	69.57	85.07	60.18	70.49
Czech-PDT	87.77	77.39	59.97	67.57	77.75	59.87	67.65	78.14	60.99	68.51	81.60	61.60	70.20
Dutch-Alpino	90.53	83.08	57.61	68.04	83.24	57.55	68.06	83.78	58.44	68.85	85.84	58.76	69.77
Dutch-LassySmall	90.38	80.41	55.00	65.33	80.71	55.12	65.51	81.01	55.77	66.06	82.19	55.95	66.58
English-EWT	97.75	83.89	67.94	75.07	86.05	68.15	76.06	85.98	68.53	76.27	89.79	69.24	78.19
Finnish-TDT	93.11	82.38	54.86	65.86	82.49	54.75	65.82	82.59	54.87	65.94	83.94	55.22	66.62
Italian-ISDT	97.52	69.55	46.76	55.92	69.71	46.78	55.99	76.27	51.49	61.47	80.00	52.15	63.14
Polish-LFG	99.66	86.49	71.91	78.53	86.49	71.91	78.53	86.49	72.67	78.98	88.17	73.06	79.91
Polish-PDB	93.36	82.99	60.56	70.02	83.08	60.51	70.02	83.02	61.65	70.76	83.02	61.66	70.76
Tamil-TTB	100.00	74.29	46.98	57.56	74.29	46.98	57.56	75.00	47.32	58.02	75.00	47.32	58.02
Ukrainian-IU	91.07	79.64	64.42	71.23	79.74	64.42	71.27	80.08	65.23	71.89	85.02	65.88	74.24
Macro average	93.45	80.07	58.67	67.60	80.41	58.68	67.73	81.17	59.81	68.74	83.50	60.21	69.86
Macro average Δ					+0.34	+0.01	+0.13	+0.76	+1.13	+1.01	+2.33	+0.40	+1.12

Table 1: PAS extraction results on the development set of 11 treebanks of 7 languages in UP 2.0, as well as the EWT section of the English PropBank. From left to right: (1) treebank name, (2) conversion rate, (3) baseline results, (4) results after adding our copula rule, (5) results after adding copula and adpositional phrase rules, (6) results after adding copula, adpositional phrase, and EUD-based co-indexation rules.

pendent falls within a corresponding UP argument span. We do not consider semantic role labels because they are not defined in CCG. We evaluate only completely converted CCG derivations.

Results On average, our CCG derivations can recover 83.50% of the semantic dependencies in UP 2.0 (Table 1). While the annotations in UP 2.0 are not perfect, they give us an approximation of the quality of our obtained predicate-argument structures. We conduct an ablation study to evaluate the effectiveness of each special rule we introduced, as well as analyzing how well extra annotations help the conversion. The third column of Table 1 shows the baseline results without any co-indexation rules or rules for adpositional phrases. The last column shows the results after all rules are applied. Some treebanks, such as Czech treebanks, English-EWT, Italian-ISDT, or Ukrainian-IU, benefit more from the added rules than others.

4.2. Discussion

In general, we observe steady improvements in semantic dependency coverage at each step for every language tested. Precision tends to be low since many CCG dependencies are not considered semantic dependencies in UP, such as those involving function words as dependents (e.g., case markers, subordinating conjunctions, adpositions), or those where light verbs are predicates (explored later in Section 4.3). However, recall is high for most languages, signifying that our CCG derivations do properly encode meaningful semantic dependencies, both local and non-local.

Rule for copulas Recall on English-EWT increases by 2.16% with our copula rule, while other treebanks gain less significant improvements. The reason could be the relative lack of predicative adjectives in UP 2.0 compared to the English PropBank, possibly either a characteristic of UP 2.0 treebanks, or a by-product of the imperfect annotation projection method used to create the data. Specifically, predicative adjectives account for 4.34% of all predicates in the development set of English-EWT, but only 0.28% in UP 2.0 (ranging from 0% in four treebanks to 1.10% in Czech-PDT).

Rule for adpositional phrases Because of the reversal of dependencies from modifiers, adpositional phrases will always be dependents, whether they are arguments or adjuncts, which explains the minimal changes after we add our rule to handle them. Beyond the current evaluation setting, having the correct categories for adpositional phrases is always beneficial. The changes we see here are mostly indirect; since the categories of the adpositional phrases change from being a functor type to an atomic type, the categories of surrounding elements (e.g., modifiers and arguments of the adpositional phrases) also change. The result for Italian-ISDT is an outlier; in many sentences, we find that the preposition is not part of the span that covers the prepositional phrase. This lowers the results of *Baseline* and *Baseline + cop*[2].

Rule for non-local dependencies denoted by EUD We see the biggest improvement in recall after adding co-indexation rules for non-local dependencies. Some treebanks such as those in Czech, English, Italian, and Ukrainian improve noticeably, while other treebanks such as Polish-PDB and Tamil-TTB stay mostly the same. We hypothesize this is due to the limited quantity and quality of the current EUD annotations (Findlay and Haug, 2021). Polish-PDB and Tamil-TTB treebanks add few to no extra dependencies for control/raising or relative clauses.

Core arguments vs. modifier arguments Breaking down the recall results into two groups, core argument (ARG*) recall and modifier argument (ARGM-*) recall, we can better see where the improvements come from (Table 2). On average, core arguments receive 4.97% improvement compared to the baseline, with Italian-ISDT (+14.44%), English-EWT (+8.73%), and Ukrainian-IU (+7.80%) leading the way. These results are expected, considering the focus of our co-indexation rules on core UD dependency relations (e.g., `nsubj`, `obj`, `iobj`).

4.3. Error analysis

To better understand the sources of errors, we performed manual analysis on the converted English-EWT CCGbank. Table 3 shows the recall errors from the first 100 sentences in the development set. We classified the errors into three classes: (1) errors from the conversion rules, (2) errors from the input sources (UD/EUD/UP), and (3) non-errors that stemmed from the discrepancy between CCG and UP on how to analyze certain constructions. The results reveal a problematic component of the original T&M algorithm, and indicate the importance of having high-quality source treebanks for conversion.

Treebank (dev)	Baseline		Base. + cop + ap + eud	
	Core R	Modifier R	Core R	Modifier R
Czech-CAC	79.18	76.10	85.22	75.26
Czech-FicTree	85.24	73.99	88.92	73.47
Czech-PDT	80.09	70.66	86.22	70.12
Dutch-Alpino	82.58	84.28	86.57	84.10
Dutch-LassySmall	80.79	79.70	83.37	79.91
English-EWT	83.34	85.05	92.07	84.91
Finnish-TDT	83.87	78.64	86.05	78.64
Italian-ISDT	72.72	60.91	87.16	60.52
Polish-LFG	88.25	82.59	91.02	81.88
Polish-PDB	84.84	78.53	85.13	77.93
Tamil-TTB	73.20	76.74	74.23	76.74
Ukrainian-IU	81.56	75.28	89.36	75.12
Macro average	81.31	76.87	86.28	76.55
Macro average Δ			+4.97	-0.32

Table 2: Recall results for core arguments and modifier arguments in UP 2.0.

Error	Count
from conversion rules (15.9%)	
- adverbial clause modifier	4
- coordination	2
- subject-verb inversion	1
from input sources (47.7%)	
- dubious annotation	11
- relative clause	8
- control/raising	2
from CCGbank/UP discrepancy (36.4%)	
- light verb	6
- adjectival modifier	3
- modifier of predicative adjective	3
- adverbial modifier	2
- reference role	2

Table 3: Recall errors from the first 100 sentences of the English-EWT treebank's development set.

[2]For example, in the phrase *colti da improvviso ottimismo* ("caught by sudden optimism"), the preposition *da* ("by") is not annotated as part of the prepositional phrase *da improvviso ottimismo* ("by sudden optimism") in UP 2.0. With the *Baseline* method, the functor category of the preposition projects a dependency to the predicate. When this dependency gets reversed by our pre-evaluation processing, the preposition becomes a dependent. As a result, the dependency is not counted as correct, because the preposition is not part of the argument span in Italian-ISDT.

Errors from conversion rules (15.9%) We identified six errors, four for adverbial clause modifiers and two for coordination, that were caused by incorrect binarization of dependency trees. In the four cases of adverbial clause modifiers, the binarization method put them in sentential modifier positions, while they were actually verb phrase modifiers. In the two cases of coordination, the binarization method failed to identify the correct conjuncts. For example, in *he founded and he is the spiritual leader of Hamas*, the conjuncts were incorrectly identified as *he founded* and *he is the spiritual leader of Hamas*. Binarization is an important first step of the conversion algorithm; the resulting binarized trees also represent the constituency structures of the input sentences. The T&M binarization method follows a rigid *obliqueness hierarchy* (Reddy et al., 2017) that may not hold true for all dependency trees. Either a more fine-grained obliqueness hierarchy or a more flexible approach to binarization is necessary to resolve these cases.

Errors from input sources (47.7%) Dubious or insufficient annotations account for the largest number of errors that we found. For instance, in one sentence, *But getting past who should get them, is who has them, and who is really close*, the dependencies between the pronouns *who* and their verbs were misannotated as `acl:relcl`, leading to incorrect CCG categories for almost all constituents in the sentence. Another large group of errors came from insufficient EUD annotations for relative clauses and control/raising (e.g., a missing arc between *freedom* and *believe* in *freedom Westerners believe in*). Since we rely on EUD annotations for co-indexation and identification of extracted arguments, we are unable to produce correct semantic dependencies for these cases. In general, we find that UD tends to be conservative with their EUD annotations; however, we expect further improvements in future revisions as it continues to develop.

Errors from CCGbank/UP discrepancy (36.4%) The most common case in this class was light verb constructions. For example, in *The case against Iran has a feeling of Déjà vu*, dependencies headed by the light verb *has* were shifted to the noun *feeling* in the English PropBank, but our CCG analysis did not make this movement and kept *has* as the main predicate. Another case involved adjectival modifiers such as *text-based* in *text-based ads*, where we simply treated them as normal modifiers, but the PropBank marked the base verbs (e.g., *based*) as predicates. Overall, there exist several similar cases that we do not consider to be errors, but rather discrepancies in semantic analyses between the two formalisms.

5. Related Work

Marking CCG categories with headedness information has been done in the past for several individual languages, such as English (Clark et al., 2002; Hockenmaier and Steedman, 2007), Hindi (Ambati et al., 2018), and Chinese (Tse and Curran, 2010). The process is performed manually, under the assumption that all lexical categories of the same type are indexed in the same manner. In contrast, the same lexical category in our work may be indexed differently in different situations, depending on the input dependency trees. The advantage of our approach is apparent in cases where words with different semantic roles receive the same category, such as control. Our method would be able to produce differently indexed categories for subject control verbs and object control verbs, given correct EUD annotations. However, in the English CCGbank for example, these two types of verbs receive the same indexed category.

Outside of predicate-argument structures, some works also provide semantic representations for CCG categories in the form of first-order or higher-order logic representations (Bos et al., 2004; Mineshima et al., 2015), and Discourse Representation Structures (Bos, 2009) for English. Again, these works cannot be easily replicated due to the large scale of the generated multilingual CCGbank. Evang and Bos (2016) propose a method that creates multilingual CCG parses with semantic interpretations through annotation projection, but their method requires a parallel corpus. Abzianidze et al. (2017) build upon the work of Evang and Bos to later create the Parallel Meaning Bank.

In the context of providing semantic interpretation for UD, Reddy et al. (2017) introduce a method to convert UD trees to logical forms, and Y. Findlay et al. (2023) convert UD trees to Discourse Representation Structures (Kamp et al., 2010). We favor CCG since syntactic information in UD can be preserved.

Last but not least, Evang (2020) presents an algorithm to extract dependency trees from CCG derivations, which is related to our PAS extraction step. However, the algorithm cannot correctly handle non-local dependencies.

6. Conclusion

In this work, we proposed a method to enhance bare CCG categories generated from UD with headedness information, allowing local and non-local predicate-argument dependencies to be projected. Consequently, each CCG derivation has a corresponding semantic representation in the form of a predicate-argument structure, improving the usefulness of the current multilingual CCGbank. Our method takes advantage of the word-word dependencies in UD, EUD, and the proposition banks to extract this headedness information. Error analysis shows that the effectiveness of our algorithm depends on the quality of the source treebanks and the bare CCG derivations. Fine-tuning the hand-crafted rules of the T&M algorithm, or increasing the quantity and accuracy of existing dependency annotations would greatly benefit our produced semantic representations.

7. Limitations

We have not solved some existing problems with the original conversion algorithm, such as the inability to convert trees with crossing dependencies, trees with `orphan` dependencies, or trees involving argument extraction without explicit traces.

Relying on EUD and UP for conversion rules also limits the number of treebanks we can work on, as well as the number of linguistic constructions we can design rules for. In addition, the quality of our converted CCGbank is basically tied to the quality of the source corpora. As shown earlier, the quality of EUD and UP annotations can be inconsistent, since many of them are automatically generated, leading to unprojected semantic dependencies. Nonetheless, as these resources are still being actively developed, we hope to see better results in the future.

Finally, manual error analysis was done only on the English-EWT treebank due to our limited language expertise. At the same time, since the English PropBank was manually annotated rather than automatically converted from UD like other UP corpora, it may give us more accurate insights and minimize possible errors when used as a target for comparison.

8. Acknowledgements

This work was supported by JST, CREST Grant Number JPMJCR2114, and JSPS KAKENHI Grant Number JP23KJ0573, Japan.

9. Bibliographical References

Lasha Abzianidze, Johannes Bjerva, Kilian Evang, Hessel Haagsma, Rik van Noord, Pierre Ludmann, Duc-Duy Nguyen, and Johan Bos. 2017. The Parallel Meaning Bank: Towards a multilingual corpus of translations annotated with compositional meaning representations. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 242–247, Valencia, Spain. Association for Computational Linguistics.

Bharat Ram Ambati, Tejaswini Deoskar, and Mark Steedman. 2018. Hindi CCGbank: CCG Treebank from the Hindi Dependency Treebank. *Language Resources and Evaluation*, 52:67–100.

Austin Blodgett and Nathan Schneider. 2019. An improved approach for semantic graph composition with CCG. In *Proceedings of the 13th International Conference on Computational Semantics - Long Papers*, pages 55–70, Gothenburg, Sweden. Association for Computational Linguistics.

Johan Bos. 2009. Towards a large-scale formal semantic lexicon for text processing. In *From Form to Meaning: Processing Texts Automatically. Proceedings of the Biennal GSCL Conference*, pages 3–14.

Johan Bos, Stephen Clark, Mark Steedman, James R. Curran, and Julia Hockenmaier. 2004. Wide-coverage semantic representations from a CCG parser. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 1240–1246, Geneva, Switzerland. COLING.

Stephen Clark, Julia Hockenmaier, and Mark Steedman. 2002. Building deep dependency structures using a wide-coverage CCG parser. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 327–334, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

James Curran, Stephen Clark, and Johan Bos. 2007. Linguistically motivated large-scale NLP with C&C and boxer. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 33–36, Prague, Czech Republic. Association for Computational Linguistics.

Kilian Evang. 2020. Configurable dependency tree extraction from ccg derivations. In *Proceedings of the Fourth Workshop on Universal Dependencies (UDW 2020)*, pages 87–93.

Kilian Evang and Johan Bos. 2016. Cross-lingual learning of an open-domain semantic parser. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 579–588, Osaka, Japan. The COLING 2016 Organizing Committee.

Jamie Y. Findlay and Dag T. T. Haug. 2021. How useful are enhanced Universal Dependencies for semantic interpretation? In *Proceedings of the Sixth International Conference on Dependency Linguistics (Depling, SyntaxFest 2021)*, pages 22–34, Sofia, Bulgaria. Association for Computational Linguistics.

Julia Hockenmaier and Mark Steedman. 2007. CCGbank: A corpus of CCG derivations and dependency structures extracted from the Penn Treebank. *Computational Linguistics*, 33(3):355–396.

Ishan Jindal, Alexandre Rademaker, Michał Ulewicz, Ha Linh, Huyen Nguyen, Khoi-Nguyen Tran, Huaiyu Zhu, and Yunyao Li. 2022. Universal Proposition Bank 2.0. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 1700–1711, Marseille, France. European Language Resources Association.

Hans Kamp, Josef Van Genabith, and Uwe Reyle. 2010. Discourse representation theory. In *Handbook of Philosophical Logic: Volume 15*, pages 125–394. Springer.

Mike Lewis, Luheng He, and Luke Zettlemoyer. 2015. Joint A* CCG parsing and semantic role labelling. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1444–1454, Lisbon, Portugal. Association for Computational Linguistics.

Pascual Martínez-Gómez, Koji Mineshima, Yusuke Miyao, and Daisuke Bekki. 2016. ccg2lambda: A compositional semantics system. In *Proceedings of ACL-2016 System Demonstrations*, pages 85–90, Berlin, Germany. Association for Computational Linguistics.

Koji Mineshima, Pascual Martínez-Gómez, Yusuke Miyao, and Daisuke Bekki. 2015. Higher-order logical inference with compositional semantics. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2055–2061, Lisbon, Portugal. Association for Computational Linguistics.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 4034–4043, Marseille, France. European Language Resources Association.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Siva Reddy, Oscar Täckström, Slav Petrov, Mark Steedman, and Mirella Lapata. 2017. Universal semantic parsing. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 89–101, Copenhagen, Denmark. Association for Computational Linguistics.

Miloš Stanojević, Jonathan R Brennan, Donald Dunagan, Mark Steedman, and John T Hale. 2023. Modeling structure-building in the brain with ccg parsing and large language models. *Cognitive science*, 47(7):e13312.

Mark Steedman. 2000. *The Syntactic Process*. MIT Press, Cambridge, MA.

Tu-Anh Tran and Yusuke Miyao. 2022. Development of a multilingual CCG treebank via Universal Dependencies conversion. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 5220–5233, Marseille, France. European Language Resources Association.

Daniel Tse and James R. Curran. 2010. Chinese CCGbank: extracting CCG derivations from the Penn Chinese treebank. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1083–1091, Beijing, China. Coling 2010 Organizing Committee.

Jamie Y. Findlay, Saeedeh Salimifar, Ahmet Yıldırım, and Dag T. T. Haug. 2023. Rule-based semantic interpretation for Universal Dependencies. In *Proceedings of the Sixth Workshop on Universal Dependencies (UDW, GURT/SyntaxFest 2023)*, pages 47–57, Washington, D.C. Association for Computational Linguistics.