How useful are Enhanced Universal Dependencies for semantic interpretation?

Jamie Y. Findlay
University of Oslo / Oslo, NO
jamie.findlay@iln.uio.no

Dag T. T. Haug
University of Oslo / Oslo, NO
d.t.t.haug@ifikk.uio.no

Abstract

We discuss the role of enhanced Universal Dependencies (E-UD) in the task of deriving semantic predicate-argument structures from UD treebanks in a universal, non-language-specific way. We consider the usefulness of three kinds of E-UD annotation (controllers of `xcomp`s, propagation of outgoing dependencies in coordinations, and coreference in relative clauses) and assess some heuristics for automatically adding such enhancements. We conclude that one large obstacle both for deriving predicate-argument structures from UD treebanks and for the automatic enhancement of basic UD treebanks is the fact that UD does not represent empty elements such as pro-dropped arguments, and we suggest that devoting effort to this would often provide a better return on investment than spending resources on improving or adding E-UD annotations.

1 Introduction

One important, traditional application of syntactic analysis is to support the creation of meaning representations. In fact, formal semantics in the tradition of Montague (Montague, 1970; Heim and Kratzer, 1998) holds that syntactic structure together with lexical meaning *determines* sentence meaning. But even if we do not accept that strong view, it is clear that syntax informs and constrains meaning. In particular, this is true of predicate-argument structures, which we take to involve relating each entity referred to in a sentence to an appropriate eventuality either directly or indirectly (via a relation to another entity so connected). In many ways, this can be thought of as the semantic reflex of syntactic dependencies and we can definitely expect UD to support this task.[1]

That said, it is well known that the basic UD representation does not consistently provide all the information that is needed to generate correct predicate-argument structures. Some of the deficiencies are remedied in the enhanced UD (E-UD), but there is a tradeoff with coverage, as only 31 out of the 213 UD treebanks contain useful E-UD edges.[2] This tradeoff becomes especially important in the context of universal semantic parsing (Reddy et al., 2017), i.e. an attempt to produce semantic representations (in our case, predicate-argument structures), in a universal way, relying only on the UD syntax and without using language-specific (e.g. lexical) resources.

In this paper, we try to assess how much E-UD helps with this task by asking to what extent it can be replaced with language-independent heuristics based on the basic UD alone.[3] The answer can inform practical decisions on how much effort to put into the creation of enhanced dependencies, and to guide future decisions on the development of the (E-)UD annotation. Because our goal is to support universal semantic parsing, we do not consider heuristics that rely on language-specific knowledge or resources.

[1] The relations should also be *labelled*, giving rise to a task of translating UD grammatical functions to appropriate semantic roles, but we do not consider this task here. Also, note that UD annotation does not allow us to identify eventualities introduced by non-verbal predicates (e.g. action nouns), so we ignore those in this paper.

[2] The TuDeT treebanks merely copy the basic dependencies over into the E-UD, while the Akkadian treebank only contains a single E-UD edge.

[3] There are many existing systems for augmenting basic UD dependency trees, and several whose effectiveness has been reported in the literature (Nyblom et al., 2013; Schuster and Manning, 2016; Nivre et al., 2018; Bouma et al., 2020). However, some of these are language specific, and several rely on machine learning. Here we report on the effectiveness of simple, algorithmic heuristics based on linguistic generalisations, and we apply them to a broad range of languages.

To evaluate our heuristics, we used the actual E-UD annotation as gold data and measured how well the heuristics reproduce this annotation from the basic UD. However, as it turned out, this approach is problematic because the E-UD annotations are often quite poor or inconsistent, both between and inside treebanks, making it hard to assess when the heuristic is wrong and when the annotation is wrong. Nevertheless, such cases still yield useful annotation recommendations.

The UD documentation specifies six types of enhancement:[4] empty (null) nodes for elided predicates, propagation of incoming dependencies to conjuncts, propagation of outgoing dependencies from conjuncts, additional subject relations for control and raising constructions, coreference in relative clause constructions, and modifier labels that contain the preposition or other case-marking information.

We will not deal with ellipsis in this paper, since its proper treatment is arguably semantic rather than (purely) syntactic (Dalrymple et al., 1991). The final type of enhancement, modifier labels, is entirely predictable from the basic UD graph and will not be further studied here either. Propagation of incoming dependencies is almost entirely predictable, except in cases of unlike function coordination, (Przepiórkowski and Patejuk, 2018). Worse, it is in fact not useful for semantic interpretation, but actually complicates matters, since it leads to the second conjunct having two incoming edges (`conj` and the copied edge), only one of which should be semantically interpreted.

This leaves three types of enhancement that we will deal with in the rest of this paper: control/raising (Section 2), propagation of outgoing dependencies (Section 3) and relative clauses (Section 4). For each of these types of annotation, we provide a theoretical discussion of how the E-UD can aid in obtaining a predicate-argument structure, quantify how well we can predict the E-UD from the basic dependencies with language-independent heuristics, and suggest changes to annotation policies and practices that would make E-UD more useful.

2 Additional subject relations for control and raising constructions

This enhancement adds dependencies which indicate the subject controllers of `xcomp`s. It is worth noting that the UD guidelines state that, to qualify as an `xcomp`, a predicate must participate in *obligatory control* (Williams, 1980), where its missing subject has to be interpreted as identical to a specified argument of another predicate (usually in a higher clause).[5] This is a fairly narrow understanding of control, and explicitly excludes cases of optional or arbitrary control (see Landau (2013) for explanation of these terms), which ought instead to be annotated as `ccomp`s or `advcl`s, according to context. We can see this as essentially restricting the UD annotation to the *grammatically determined* instances of control, in keeping with UD's role as a syntactic annotation scheme, and leaving processes such as anaphor resolution to the semantics.

There are two types of `xcomp` discussed in the guidelines: classic raising or control structures such as *I promised to come*, and secondary predications where the predicative component is a core argument of the main verb, such as *She declared the cake beautiful*. Figure 1 gives example annotations for these two structures, with the enhancement adding the controller indicated below the string:

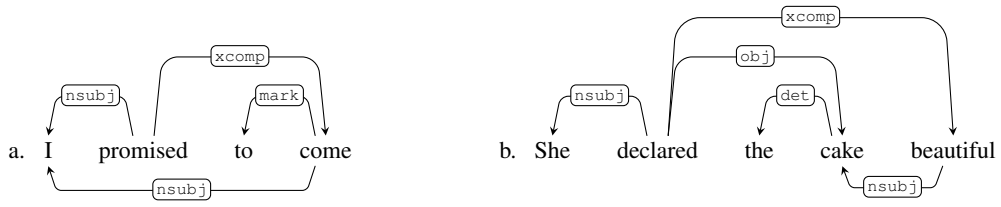

Figure 1: Examples of additional subject relations

In the basic annotation, there is no indication of the dependency between *come* and *I* (it is the speaker who will come), or between *beautiful* and *cake* (it is the cake which is declared to be beautiful); this is remedied in the enhanced representation.

[4] https://universaldependencies.org/u/overview/enhanced-syntax.html
[5] https://universaldependencies.org/u/dep/xcomp.html

Clearly, in order to obtain the correct predicate-argument structure for sentences like (0a) or (0b), these additional dependencies are necessary. And since these are not available in the basic UD tree, we do need the extra information added by the E-UD. What is more, the choice of external controller is, in principle, a lexical one – that is, it cannot be deterministically inferred from the syntactic structure of the sentence alone. For example, the a. and b. sentences in Figure 2 have identical basic UD trees (shown above the string), but the enhancements indicating the controller of the `xcomp` (shown below) differ, simply because the verbs in the main clause differ.

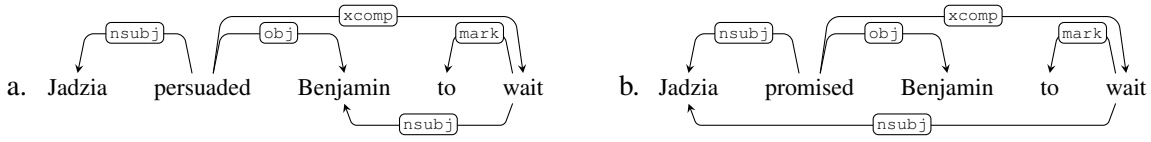

Figure 2: Lexically-determined control relations

Given this, an accurately annotated E-UD treebank would be particularly valuable for the purposes of semantic interpretation. However, only 22 of the UD treebanks contain this particular enhancement, and, as mentioned in the introduction, the quality of such annotations is not always high. So how successful can we be in adding such enhancements automatically to a basic UD annotation?

2.1 Heuristic

The linguistic generalisation we exploit here also appears in the UD guidelines' definition of `xcomp`: the `xcomp`'s subject is controlled "normally by the object of the next higher clause, if there is one, or else by the subject of the next higher clause". That is, our heuristic assumes that if the head of the `xcomp` has an object dependent (`obj`, `iobj`, or `ccomp`), then that will be the controller;[6] if there is no object, then the subject, if present, will be the controller. If neither is present, we check whether the next highest head is itself an `xcomp`; if so, we continue to search upwards until we find a subject or object, or are no longer in an `xcomp`. This recursive search accounts for embedded `xcomp`s, as in Figure 3:

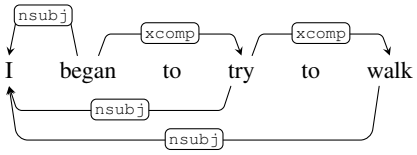

Figure 3: An embedded `xcomp` and its controller

Our heuristic is similar to the approach of Schuster and Manning (2016) and Nivre et al. (2018, 103), but with the addition of this recursive search in the case of embedded `xcomp`s.

2.2 Results

Comparing the output of this heuristic against the E-UD annotations present in the 22 treebanks under discussion, we obtain an average precision score of 72.49% (see Table 1 for details). This is perhaps not terribly impressive. However, there are important caveats to consider. Firstly, the Dutch treebanks represent clear outliers (with 37.00% and 30.92%), and their removal increases the average by several percentage points (to 76.34%). The issues with Dutch appear to be because of systematic annotation errors in these two treebanks, where a number of `xcomp`s do not have their controllers indicated, even though they are present in the string.[7]

[6]The relation `ccomp` is included as a kind of object here to account for examples like the following, from the English-GUM treebank, where the `ccomp` headed by *waking* is the `csubj` of the `xcomp` headed by *easier*: "It makes [waking up in the morning and getting out of bed at 6:00 a.m. when it's pitch black outside] [so much easier] when you're waking up really early" (`GUM_vlog_london-18`).

[7]Such problems are especially focussed around auxiliary-like predicates such as *lijken* 'seem', *liggen* 'lie'/'be', *blijven* 'remain', and *worden* 'become'/'be' (as passive auxiliary), whose `xcomp` complements very often do not have their controllers annotated. In many cases, it seemed to us that it might have been more more appropriate to annotate the complements of these verbs as the head of such constructions, and mark these verbs as `aux` instead, but we do not pursue this issue here.

Corpus name	Precision	Precision (controllers marked)	Recall
Albanian-TSA	66.67%	100.00%	66.67%
Belarusian-HSE	60.24%	76.70%	73.82%
Bulgarian-BTB	71.87%	98.49%	75.69%
Czech-CAC	67.00%	85.23%	78.12%
Czech-FicTree	63.03%	88.58%	78.83%
Czech-PDT	74.60%	87.89%	83.53%
Czech-PUD	55.36%	78.81%	75.61%
Dutch-Alpino	37.00%	92.97%	90.01%
Dutch-LassySmall	30.92%	94.57%	83.94%
English-EWT	93.84%	95.65%	90.76%
English-GUM	92.70%	99.46%	94.24%
English-GUMReddit	93.23%	99.20%	89.21%
English-PUD	92.00%	94.52%	88.09%
Finnish-TDT	56.89%	99.43%	60.29%
Italian-ISDT	76.94%	82.39%	80.28%
Latvian-LVTB	69.30%	95.03%	87.88%
Lithuanian-ALKSNIS	59.78%	93.79%	78.16%
Polish-LFG	95.26%	98.37%	94.41%
Slovak-SNK	68.81%	87.98%	84.03%
Swedish-PUD	87.62%	89.39%	84.29%
Swedish-Talbanken	86.31%	90.92%	86.13%
Ukrainian-IU	95.34%	98.50%	88.39%
AVERAGE	72.49%	92.18%	82.38%

Table 1: Performance of the heuristic used for adding external subjects

This points to a wider problem: in cases where the treebank in question contains an error, the precision score of the heuristic will suffer even when it is doing the right thing, linguistically speaking. There are cases where this occurs because of errors in the E-UD annotation, as with the Dutch examples, where controller annotations are omitted even when the controllers are present – here the heuristic often does a better job than the treebanks as annotated. There are also other cases where the quality of the *basic* UD annotations is the problem, and can mislead the heuristic: for example, where the treebank is right not to include a controller, but should not therefore have used an `xcomp` annotation in the first place. For instance, the following sentence shows an example of arbitrary control from the English-GUM treebank, which should have *chased* as a `ccomp` dependent of *what*, not an `xcomp`, as annotated: "Do you know what it's like to be chased by the Ghost of Failure while staring through Victory's door?" (`GUM_interview_messina-36`). If it were annotated in this way, the heuristic would (correctly) not look for a controller, and so would not (incorrectly) guess that it was the `nsubj` of *what*, namely *it*, and thereby hurt its precision score.

In both these cases, the only way to comprehensively determine to what extent the heuristic performs better than, or is unfairly misled by, the existing annotations would be through manual inspection. This is obviously time consuming, and also requires knowledge of many different languages, and so we have not been able to carry out such verification on a large scale. However, a sample analysis of 100 random errors from both the Dutch-Alpino and English-GUM treebanks indicates that our suspicions are borne out. Table 2 shows the sources of errors: overwhelmingly, the fault is with the annotation rather than with the heuristic. Most commonly this is because a controller is not annotated in the E-UD annotation when it should be (93/95 of the E-UD errors are of this kind in the Dutch corpus, 67/76 in the English). In all but one of these cases for each treebank, our heuristic correctly identifies the controller.

Corpus	Basic UD	E-UD	Heuristic	Not an error
Dutch-Alpino	3	95	1	2
English-GUM	25	74	1	1

Table 2: Sources of error in sample of 100 sentences from two corpora (numbers don't sum to 100 because the 2 heuristic errors also involved E-UD errors)

Problems with the basic UD annotation constitute a sizeable minority in the English corpus, though they are rarer in the Dutch corpus. The majority of these are cases where the word which bears the `xcomp` dependency should have been annotated differently: either the construction in question involves non-obligatory control, so the dependency label should be `ccomp`;[8] or the dependent is a modifier not an argument (e.g. a purpose clause), so the label should be `advcl`;[9] or it is a secondary pred-

[8] As in the example mentioned earlier on this page.

[9] As in e.g. `GUM_news_asylum-7` from the English-GUM corpus: *Basya also believes the asylum seekers* [...] *may have left the boat on purpose to be* ***rescued*** *to* ***avoid*** *being sent away from Indonesia waters*, where putative `xcomps` are in boldface.

ication which is not a core dependent of the head, so it should be an acl.[10] The heuristic itself makes one error in each sample, and in both cases the E-UD annotation is also incorrect (because it does not include a controller at all).

Since the majority of errors we found in this sample analysis were due to the omission of controllers in the E-UD annotation, Table 1 also gives a precision score where the denominator is the number of guesses where the xcomp in question actually has a controller marked, rather than simply the total number of guesses, as a stand-in for a more thoroughgoing error analysis. Under these conditions, performance improves dramatically, to an average of 92.16% (and the Dutch outliers fall into line too). Of course, this may be concealing errors where the heuristic guesses a controller when one is genuinely not present; but such situations should be rare, given the definition of xcomp, and so we believe these figures are a fairer representation of the performance of this heuristic.

2.3 Discussion

There are of course cases where our heuristic will actually fail: with *promise*-type verbs as described at the start of this section, for example, since we assume that if the control verb has an object it must be the controller. Some languages might pose their own challenges too: for example, Nivre et al. (2018, 104) mention the fact that Italian allows (dative) obl controllers. This is not something we could easily incorporate into the heuristic as it stands, since the relation obl is used for verbal adjuncts as well as arguments. Judicious use of subtypes could help here, but we cannot guarantee that such subtypes would be present in a basic UD treebank.

We saw at the start of this section that the label xcomp is intended to be used for (grammatically governed) obligatory control. *Ceteris paribus*, we would therefore expect all xcomps to have controllers indicated in the E-UD annotations. However, as we saw above, this is not the case in the existing treebanks – the majority of 'errors' from our heuristic were cases where the heuristic identifies the correct controller but the treebank simply doesn't indicate one at all. One might therefore suggest that E-UD validation should include a check to ensure that all xcomps are properly controlled. However, this would not in fact be workable, because of the problem of implicit arguments. In some cases, the controller of an xcomp corresponds to an argument which is not realised in the string. This is especially pronounced in so-called *pro-drop* languages, where arguments of a verb (which ones will depend on the language) need not be realised overtly, with their referents being inferred either through context or through morphological marking on the verb itself. This is a problem for E-UD annotations of control because where an unexpressed argument of a higher predicate is the controller of an xcomp, it clearly cannot bear any relation to the xcomp in the enhanced representation (since it can bear no relations at all, not corresponding to a node in the string). In fact, this situation is not limited to pro-drop languages, and can also occur in a language like English, for example in a relative clause with no overt relative pronoun like *The man I told to leave* . . . , where the gap is the controller of the xcomp *leave*.

Having no controller marked on E-UD representations of certain xcomps is problematic for two reasons. Firstly, we lose linguistic information: we cannot capture the fact that control predicates enforce exactly the same kind of obligatory coreference between arguments when one of them is implicit as when it is explicit, because in the former case there is simply no node to be shared. There is no linguistic difference here at the relevant level of abstraction, but the annotation suggests there is. This representational divergence is undesirable from the point of view of UD's *universal* goals, and also makes downstream tasks such as semantic interpretation that much more difficult.

Secondly, the process of enhancing basic UD annotations, and of verifying that enhancement, is made more difficult. If the controller of an xcomp was always present in the string, the heuristic discussed above could be applied without missing implicit controllers. It would also be easier to verify enhancements or conduct error analysis, since any xcomp missing a subject edge in the E-UD could automatically be flagged as an error.

For these reasons, we agree with Patejuk and Przepiórkowski (2018, 216ff.) that including an 'empty' node in the basic UD representation for an implicit or gapped argument would be a major improvement

[10] As in e.g. cgn_exs\68 from the Dutch-Alpino corpus: *hij kwam **dronken** thuis* 'he came home drunk'.

to the expressivity and utility of the basic UD tree, and without adding too great a burden to the annotation task. This would ensure that all xcomps can have their controllers indicated in the enhanced UD annotations, thus harmonising the aforementioned differences across languages and constructions.[11]

3 Propagation of outgoing dependencies from conjuncts in coordinations

It is difficult to achieve a linguistically adequate annotation of coordination structures in dependency frameworks, as is widely accepted and extensively discussed in Popel et al. (2013). One particular problem is the distinction between modifiers that are private to one conjunct and those that apply to all, which is crucial to the creation of correct semantic representations: in *young capercaillies and grouses* we need to know whether *young* applies only to the capercaillies or also to the grouses, and in *shaved and brushed the cat* we need to know whether only the brushing or also the shaving applied to the cat.

If the conjunction is the head, as in some dependency annotation schemes, it provides an attachment point different from the individual conjuncts, and this can be used to make the annotation unambiguous.[12] But in schemes like UD where one of the conjuncts (the first, in the case of UD) is selected as the head, the problem becomes severe. In the basic annotation, shared dependents are attached to the first conjunct and so cannot in principle be distinguished from private dependents of the first conjunct. This underspecification is resolved in the enhanced dependencies, where dependents are attached to all the heads they belong to, so these are crucial for generating the correct predicate argument structure.

3.1 Heuristics

Disambiguation of shared dependencies potentially relies on very detailed contextual and encyclopedic knowledge. To correctly resolve the cases above, we need to know whether the context makes it likely that we are speaking about young grouses and whether it is normal to shave cats. This is way beyond the reach of language-independent heuristics. But in some cases we can make informed guesses.

First, valency information helps. UD does not directly express valency, but in situations where the potentially shared dependent bears a core or a functional relation (nsubj, obj, iobj, csubj, ccomp, xcomp, expl, aux, det, case, mark, cop) and the conjunct already has its own instantiation of that core dependent, we can be quite confident that the dependent is *not* shared. For the purposes of this heuristic, we can count nsubj and csubj as the same relation.[13] The resulting heuristic is purely negative, but can still be useful in restricting other, positive heuristics.

Second, because shared dependents are always attached to the first conjunct in the basic UD, the alternative, private dependent analysis looks quite different, depending on whether it is one where the dependent is private to the first or a later conjunct. Figure 4 shows the first case; the basic dependencies look the same for the private and the shared analysis, and we simply add an edge in the E-UD to express the shared conjunct. Figure 5 shows the second case, where the alternative analysis has the dependent being private to the second (or a later) conjunct. Here the alternative analyses differ also in the basic dependencies, because the dependent is attached to the first conjunct if it is shared, but otherwise to the second. This means that Figure 5a is unambiguous, and needs no enhancement.[14] The basic UD of Figure 5b, on the other hand, is ambiguous as long as we only consider the unordered tree, which is identical to that of Figure 4. But the word order disambiguates this case, since it shows that we only need to decide between a shared analysis and a second conjunct-only analysis, and the latter would look different already in the basic UD.

For English objects this is a foolproof heuristic: an object following the second verb cannot be private to the first conjunct. We speculate that this might also hold true in languages with freer word order, if

[11] Note that we are not proposing that the subjects of xcomps themselves be represented in the basic UD annotation, although this would also be possible. It would simply change the nature of the enhancement process: instead of adding a subject edge to the xcomp, we would have to connect the now already existing subject to its antecedent, perhaps with a ref dependency, as used in the E-UD analysis of relative clauses.

[12] This is done, for example, in the source treebank of Polish-PDB and Polish-PUD; see Wróblewska (2018).

[13] For languages like English, it would be tempting to treat expl as a subject relation, but this would hurt performance in treebanks where expl is used e.g. for "detransitivizing" object reflexive clitics; see Bouma et al. (2018).

[14] Notice that adding an object edge from *shaved* to *cat* in the E-UD would be an annotation error as the shared dependent should be attached to the first conjunct in the basic UD.

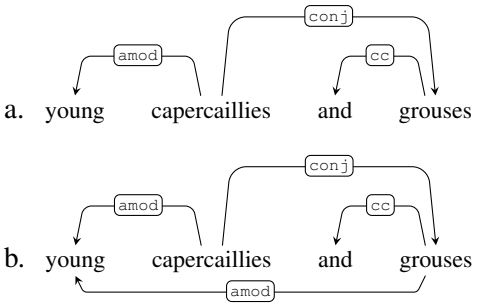

Figure 4: Coordination where both analyses have identical basic UD trees

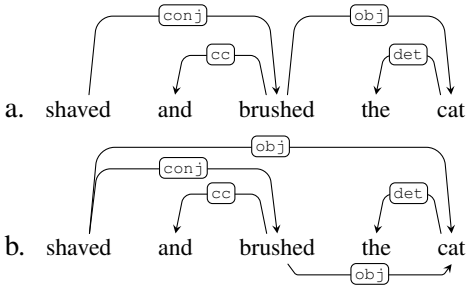

Figure 5: Coordination where each analysis has a different basic UD tree

nothing else because of a tendency to follow Behaghel's first "law": what goes together semantically goes together in the word order (Behaghel, 1932, 4–7). More generally, it is tempting to assume that if the potentially shared dependent belongs linearly to the second conjunct, but is annotated as a dependent of the the first conjunct, we are in the situation illustrated in Figure 5b; i.e. the dependent is shared and should be propagated. For the purposes of this heuristic, we take "belongs linearly to the second conjunct" to mean "occurs to the right of the leftmost word in the subgraph of the second conjunct".[15] Notice that we do not require that the dependent occurs to the right of the *head* of the second conjunct as it does in Figure 5b. If a language allows a word order like *shaved and the cat brushed*, *the cat* will count as belonging linearly in the second conjunct, and if it is annotated as a dependent of *shaved*, this will trigger a shared analysis. We will refer to dependents that are linearly in the second conjunct but are annotated as dependents of the first conjunct as *distant dependents*.

Finally, what if the potentially shared dependent belongs linearly to the first conjunct? In general, it is very hard to guess whether a dependent should be shared in this situation. Still, the strong universal tendency for verbs to always require a subject suggests that we can assume that the subject relations `nsubj` and `csubj` are always propagated to conjuncts that do not themselves have such a dependent. This may fail in cases where the second conjunct is an impersonal verb or in cases of pro-drop, as we will see in the error analysis.

In sum, this yields the following heuristic (**Heuristic 1**): never propagate a core dependent to a conjunct which has its own instantiation of that dependent, but otherwise **a)** always propagate subject relations and **b)** always propagate distant dependents. We will see that many treebanks have automatic enhancements which in many cases only propagate distant objects. For purposes of comparison, we therefore also include the results of a restricted version of Heuristic 1, which only propagates distant objects (**Heuristic 2**): never propagate a core dependent to a conjunct which has its own instantiation of that dependent, but otherwise always propagate subject relations and always propagate distant objects. This second heuristic has no linguistic motivation but merely aims to replicate automatic enhancements.

3.2 Results

Table 3 shows the performance of our two heuristics on the UD treebanks that have enhancements for propagation of outgoing dependencies.[16] For Heuristic 1, we report its overall performance, but also the performance of its two component parts.[17]

The treebanks clearly fall into two groups: for some treebanks (Bulgarian-BTB, English-EWT, English-PUD, Italian-ISDT, Swedish-PUD, Swedish-Talbanken) recall is over 95% for both heuristics and Heuristic 2 also achieves a precision of close to 100% (except in the case of Bulgarian-BTB). These are treebanks where the the propagation has been added by a heuristic very similar to our Heuristic 2 and, as such, the data are of little interest for assessing how well our heuristics can replicate gold stan-

[15] In the typical case, the leftmost word in the subgraph of the second conjunct will be the conjunction, which is a `cc` dependent of the head of the second conjunct, but other cases are possible e.g. when the conjunction is a clitic.

[16] We ignore Belarusian-HSE because there are only 25 scattered instances of propagation.

[17] Notice that there is some overlap between the components, as distant subjects will be propagated by both parts. Therefore, the recall of the whole heuristic will often be lower than the sum of the recalls of the parts.

	1: subj + dist		1a: all and only subj		1b: dist only		2: subj + dist obj	
	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
Arabic-PADT	61.8	77.5	27.1	14.4	87.1	64.1	27.7	14.9
Bulgarian-BTB	40.2	100.0	63.6	100.0	0.2	0.2	62.2	100.0
Czech-CAC	81.7	50.0	64.9	18.5	95.3	33.0	66.7	20.2
Czech-FicTree	69.7	46.6	66.6	36.9	82.7	10.1	66.8	37.3
Czech-PDT	69.6	56.0	54.9	25.9	89.9	30.8	56.2	27.5
Dutch-Alpino	50.9	48.3	59.7	32.9	38.5	15.6	59.7	33.4
Dutch-LassySmall	50.8	48.3	51.6	32.7	49.7	16.0	51.6	32.7
English-EWT	61.1	100.0	98.7	93.4	10.9	7.7	98.1	99.1
English-GUM	59.6	83.7	97.9	78.2	10.0	6.1	97.6	83.3
English-GUMReddit	69.8	80.6	100.0	78.3	11.8	4.7	99.0	80.6
English-PUD	63.7	100.0	98.9	93.0	12.5	8.0	99.0	99.0
Finnish-TDT	84.5	38.9	84.3	26.1	85.2	13.2	84.8	27.2
Italian-ISDT	63.5	97.2	92.4	93.5	10.1	5.4	92.6	96.7
Latvian-LVTB	83.2	38.2	79.5	28.1	95.9	10.7	80.5	29.8
Lithuanian-ALKSNIS	59.9	36.1	48.3	19.3	77.4	18.2	51.7	22.1
Polish-LFG	69.5	31.9	67.9	29.6	100.0	2.9	68.2	30.0
Polish-PDB	81.0	34.9	72.8	21.4	98.6	13.8	74.1	22.8
Polish-PUD	87.9	33.4	81.8	20.7	100.0	13.4	82.7	22.0
Slovak-SNK	53.0	58.8	40.6	34.7	92.8	25.8	42.6	37.6
Swedish-PUD	63.6	100.0	100.0	93.8	11.3	7.3	100.0	100.0
Swedish-Talbanken	72.0	100.0	99.1	88.8	24.2	12.2	99.2	96.9
Ukrainian-IU	26.4	48.4	31.5	37.8	17.1	11.2	31.2	38.7

Table 3: Performance of propagation heuristics (bold-face = gold standard propagation enhancements)

dard annotation. English-GUM and English-GUMReddit also belong to this group, but the recall is lower because these treebanks also do some propagation of auxiliary verbs.

The other treebanks are more interesting. These are treebanks that arguably have genuine "gold standard" propagation of dependents. The Finnish and the Ukrainian treebanks have manually annotated enhanced graphs; but in the case of the Ukrainian treebank, the README reports that the annotation of propagated dependencies is only 40% complete, so we will disregard this treebank. The other treebanks have been converted from formats where shared dependencies were deterministically expressed (either Prague-style annotation, dependency schemes with the conjunction as the head, hybrid phrase structure/ dependency formats, or LFG). In principle, this was the case also with the Dutch treebanks, but here we discovered a number of conversion errors in the annotation.

For the other, gold standard treebanks, the precision of Heuristic 1 ranges from 53.0% on Slovak to 84.5% on Finnish, while recall ranges from 31.9% on Polish to 77.5% on Arabic. The propagation of distant dependents (1b) is a very sound heuristic in Polish (100% precision in two of the treebanks) and does quite well in Arabic, Czech, Finnish, Latvian, and Slovak (precision in the mid eighties or higher), but fares less well in Lithuanian (precision 77.4%). It naturally achieves very little recall on its own except in Arabic where it catches 64.1% of propagations. By contrast, subject propagation has a surprisingly low precision.

To understand better the behaviour of the heuristics, we performed manual error analysis of the 100 first precision errors[18] in the Lithuanian treebank, 50 errors in the propagation of subjects and 50 errors in the propagation of distant dependents. Table 4 shows the results.

As we see, annotation errors are by far the most common cause of precision errors by our heuristics. In 13 of 50 cases, the subject propagation rule adds a shared subject edge that should in fact have been there in the E-UD. In 19 cases, the error is in the basic UD leading to a misannotated structure, often involving a `csubj` that the heuristic propagates but which in fact should not be there at all. The actual linguistic errors are fewer (18), but of course more interesting. A characteristic of Lithuanian is the frequent use of impersonal verbs and those account for 12 cases where the heuristic propagates the subject of the first conjunct to the second conjunct which does not in fact take a subject at all. Many other kinds of subject shift can be detected with a simple feature check: the second verb is often in the first or second person.

[18]It makes little sense to explore the recall errors since we already know that the heuristics only cover a small proportion of possible shared conjunct structures.

But in this case both verbs are third person and so, for the basic UD to be unambiguous, we would need a feature `VerbType=Impersonal`.

	Impers. verb	Subj. shift	Basic UD	E-UD
1a) Subject	12	6	19	13
1b) Dist. dep.	—	—	48	2

Table 4: Sources of propagation errors in Lithuanian-ALKSNIS

For the propagation of distant dependents, the error analysis is more depressing in that all errors are in fact due to the annotation. It should be stressed that many of these involve "technical" relations such as `dep` and `flat` that are used in suprising ways. If we instead consider only the propagation of distant objects and obliques (106 cases), there is only a single precision error.

3.3 Insights

Word order turns out to be reasonably reliable as an indication of a shared dependency and most errors are due to misannotations. However, the coverage of this heuristic is quite limited. Subject propagation achieves much higher coverage, but its precision is low. Here too, the majority of errors are due to annotation errors. Some of these could be avoided with simple feature checking but, at least in the Lithuanian error sample, this would be much more useful if impersonal verbs were marked with a special feature.

4 Relative clauses

Relative clauses are clausal dependents of nouns, and hence bear the relation `acl` in the UD annotation. Semantically speaking, they represent unsaturated predicates, containing a gap which is either unrealised in the syntax or appears as a pronoun (relative or resumptive), which can either be *in situ* or displaced. They restrict the reference of the noun which heads the `acl` relation. For example, in interpreting the sentences *boys who Mary gave flowers* and *boys who gave Mary flowers*, we intersect the set of boys with the set of individuals that are respectively the goal or the agent of some giving event in the past.

The first condition for correct interpretation of relative clauses is therefore that we know there is a gap. The `acl` relation does not by itself provide this information, as it is also used for other clausal dependents of nouns (*a way to get my discount*, *the fact that nobody cares*). However the subtype `acl:relcl` is widely used in UD treebanks, and in this section we only consider this data.

4.1 Heuristic

Given that we know there is a gap, the next step in constructing the correct predicate-argument struture is the identification of the gap. This can sometimes, but not always, be done on the basis of the basic UD; by contrast, a proper E-UD annotation will always identify the gap. In fact, the E-UD representation is not so much an enhancement of the basic UD as a different theoretical perspective on relative clauses. The two analyses are shown in Figure 6:

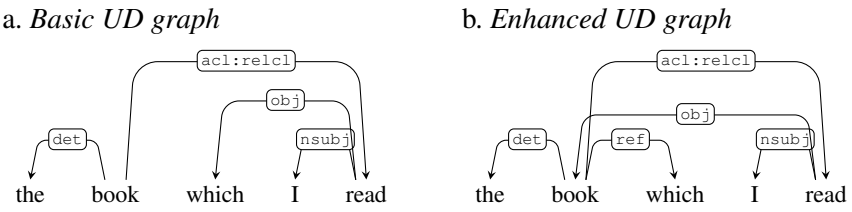

Figure 6: Two analyses of relative clauses

Figure 6a is what Falk (2010) calls a mediated analysis, i.e. one where the connection between the head of the relative clause and the gap inside the relative clause is mediated anaphorically by the relative pronoun. As a consequence, this connection is not represented directly in the syntax. By contrast, Figure 6b illustrates an unmediated analysis, where the head directly contracts a syntactic relation with the relative clause verb. Consequently, the graph contains a cycle, and the enhanced graph is not merely a

straight augmentation of the basic graph. In addition, the relative pronoun becomes a `ref` dependent of the head. This is suggestive of the mediated analysis, but actually adds no information.

In the basic UD, the gap is only retrievable to the extent that the relative clause contains an identifiable relativizer, usually carrying the feature `PronType=Rel`. In such cases, it is straightforward to translate between the two analyses, and both would serve equally well as the basis for semantic interpretation.[19]

The case which distinguishes the approaches is the one where there is no relative pronoun; an example is shown in Figure 7. Here the E-UD graph has an argument dependency which is missing in the basic UD graph.

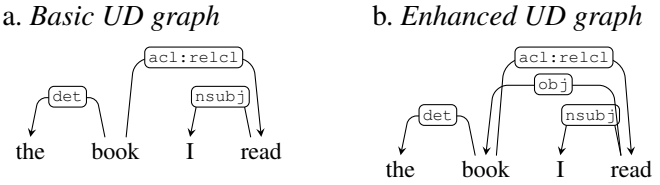

Figure 7: UD and E-UD analyses of a relative clause without a relative pronoun

The same problem may or may not arise in relative clauses introduced by a complementizer. For example, there is consensus in the grammatical literature that the word *that* in the variant *the book that I read* is a complementizer and not a pronoun filling the object position of *read* (Huddleston and Pullum, 2002, 1056f.). Nevertheless, the English treebanks consistenly treat it as a relative pronoun, allowing the relation of the gap to be expressed even in the basic dependencies. The Swedish treebanks do the same for *som*, while the related *som* in the Norwegian treebank is treated as a complementizer, thus prioritising giving the correct part of speech tag over expressing the gap that is needed for semantic interpretation in the basic UD (the Norwegian treebanks have no E-UD).

How can we guess the position of the gap if it is not present in the basic UD? Rule-based parsers typically use valency information to identify the missing argument, but this information is not present in the UD tree. We therefore rely on cross-linguistic tendencies as to what arguments are most accessible to relativisation, the so-called *Accessibility Hierarchy* (Keenan and Comrie, 1977), to determine which dependency is missing. The hierarchy is given in its original formulation in Figure 8a and translated to UD relations in Figure 8b.[20] Notice that we ignore genitives and objects of comparison, as they are rare and would not in any case be direct dependents of the `acl:relcl` verb, necessitating a further search.[21]

a. subject < direct object < indirect object < oblique < genitive < object of comparison
b. `subj` < `obj` < `iobj` < `obl`

Figure 8: The Accessibility Hierarchy and its translation into UD

The idea behind the heuristic, then, is to scan the dependents of the verb that bears the `acl:relcl` relation and assume that the gap bears the highest relation on the Keenan-Comrie hierarchy that is not present in the basic UD dependencies.

4.2 Results

Unfortunately, it turned out to be hard to evaluate this heuristic. First, few treebanks contain useful enhanced dependencies for relative clauses. 27 of the treebanks with E-UD do not have enhancements for relative clauses, or only have them when there is an overt relative pronoun, allowing them to be generated automatically but adding no new information. This leaves only 11 treebanks for our evaluation.

[19]Notice, however, that the E-UD can be interpreted directly from the graph, while the interpretation of the basic UD annotation relies on a lexical feature. Lexical features are less standardised than other parts of UD. We return to this point below.

[20]For the purposes of checking existing relations, we collapse `nsubj` and `csubj` into a single `subj` relation, since no predicate will have both. If this relation is missing, we assume it is `nsubj`, since relative clauses modifying clausal heads are rare, especially in the case of restrictive relative clauses, which is what we are considering here.

[21]We also ignore the possibility of 'long-distance relativization' as in *the book you asked Mary to look for*, which are also rare and necessitate a search for the correct attachment point.

Of these, Swedish-Talbanken, English-PUD, Italian-ISDT, English-EWT, Swedish-PUD and Estonian-EWT contain less than 10 instances of non-predictable enhanced dependencies for relative clauses. All these treebanks contain a large number of predictable enhanced dependencies and it seems that the scattered non-predictable E-UD edges are due to accidental omission of the feature `PronType=Rel` on the relative pronoun. Furthermore, the only non-predictable relative clauses in the Dutch treebanks are introduced by the relativizer *waar*, which bears the grammatical relation of the gap, but does not have the `PronType=Rel` feature. In such cases, the heuristic is doubly misled: first, it applies where it should not, because there is no `PronType=Rel` feature present in the clause, and next, it wrongly assumes that the grammatical function corresponding to the gap is actually filled. For example, in (8a), if *which* does not bear the the `PronType=Rel` feature, the heuristic will assume that we are in a pronoun-less relative clause and that the object position is filled, so that the gap is therefore `iobj`.[22]

Disregarding the treebanks where the only "informative" E-UD edges for relative clauses are due to accidental omission of the `PronType=Rel` feature, we have only three treebanks with non-trivial E-UD edges: Tamil-TTB, Ukrainian-IU and Belarusian-HSE. This indicates that the current E-UD annotation policy for relative clauses has not been very successful, as most treebanks either do not use it, or generate it only in the cases where it can be done automatically from the basic UD. We suspect the reason for this may be that it embodies a different theoretical perspective on relative clauses and therefore seems like an alternative analysis rather than a more informative one, even if it does it some cases contain more information.

Be that as it may, the results of applying our heuristic to the three treebanks that have non-trivial E-UD edges for relative clauses are shown in Table 5, and as we can see, they are decidedly mixed. We get good results on Tamil and Ukrainian and abysmal results on Belarusian. As it turns out, in 97.0% of the errors in Belarusian-HSE, the correct relation is `advmod`. The Keenan-Comrie hierarchy never predicts this, as it specifically addresses relativization on nominal positions.[23] But many languages use the equivalents of *where* and *when* to introduce clauses expressing location and time, and in many treebanks these are analysed as relative clauses. English is a case in point, but in the English treebanks these words are generally given the `PronType=Rel` feature (despite not being pronouns), hence making the gapped relation transparent. The Belarusian treebank, by contrast, does not add this feature.

	Belarusian-HSE	Tamil-TTB	Ukrainian-IU
success	1	368	65
failure	202	28	4

Table 5: Evalutation of heuristic for relative clauses

4.3 Discussion

The E-UD representation is crucial for a correct semantic analysis of relative clauses where the gap cannot be identified by the `PronType` feature. However, very few treebanks contain such enhanced dependencies; in practice, the enhanced dependencies are only generated when they can be unambiguously derived from the basic UD. This suggests that here too a limited use of empty nodes could be beneficial in allowing for the expression of the gap in the basic UD, even when there is no overt relative pronoun. Figure 9 shows what the annotation would look like.

This would make it possible to consistently give an interpretable annotation of relative clauses in the basic UD, and render the enhanced version superfluous.

[22] Another problem is that many treebanks use the the multivalue feature `PronType=Int,Rel` (reflecting the interrogative/relative ambiguity that is common in Indo-European and beyond), although the UD guidelines specify that these should be used sparingly and only when one cannot decide between the two features. If the clause itself is marked as `acl:relcl`, it is of course clear that the *wh*-word that introduces it is a relativizer and not an interrogative, so `PronType=Rel` should have been used. However, precisely for that reason, it seems safe to interpret `PronType=Int,Rel` as indicating a relative pronoun in this context. Our experience suggests that it is even safe to interpret the wrong tag `PronType=Int` as meaning `PronType=Rel` inside an `acl:relcl` subtree.

[23] Note that even if we added `advmod` to the bottom of our prediction hierarchy, our heuristic would only add it to verbs that already have `subj`, `obj`, `iobj` and `obl` dependents.

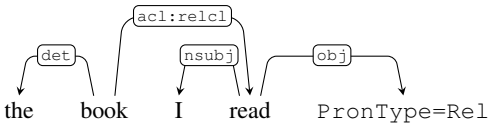

Figure 9: Null relative pronoun annotated in basic UD

5 Conclusion

Overall, it is clear that some enhancements of basic UD annotations are necessary in order to derive correct predicate-argument structures. E-UD does offer these, but there are two important limitations. The first is coverage: only 31 treebanks have any (useful) E-UD annotations, and even fewer contain all six subtypes identified by the UD guidelines. The small size of this selection is further compounded by it being more typologically restricted than the impressively global spread of UD: those treebanks with E-UD are much more European and much less diverse (of the 31 treebanks, 4 are English, 4 are Czech, 3 are Polish, . . .). The second limitation is quality. In our investigations, we found that the E-UD annotations were inconsistent at best, and often the result of limited automatic processes with minimal manual verification – although we have only quantified these shortcomings in a very preliminary way.

One solution to these limitations would be to invest time and resources into improving the quality of existing E-UD annotations. For the propagation of outgoing dependents in coordinations, this may in fact be the only feasible solution. For control and relative clauses, however, we suggest another approach. As we noted above, the addition of empty nodes for certain phenomena in the basic UD annotation would allow for the automatic generation of an improved E-UD annotation, or even make it redundant, since the basic UD would now contain the missing information already. Noting that the existence of empty nodes has already been sanctioned in the E-UD treatment of ellipsis, we suggest that generalising this to other phenomena in the basic UD annotation could be much more worthwhile than annotating enhanced edges independently.

Acknowledgements

This work was supported by the Research Council of Norway, grant number 300495 “Universal Natural Language Understanding”. We are grateful to the three reviewers for their feedback on the previous version and to Alicia Cleary-Venables for help with the Dutch data.

References

Otto Behaghel. 1932. *Deutsche Syntax IV*. Carl Winter, Heidelberg.

Gosse Bouma, Jan Hajic, Dag Haug, Joakim Nivre, Per Erik Solberg, and Lilja Øvrelid. 2018. Expletives in Universal Dependency treebanks. In *Proceedings of the Second Workshop on Universal Dependencies (UDW 2018)*, pages 18–26, Brussels, Belgium, November. Association for Computational Linguistics.

Gosse Bouma, Yuji Matsumoto, Stephan Oepen, Kenji Sagae, Djamé Seddah, Weiwei Sun, Anders Søgaard, Reut Tsarfaty, and Dan Zeman, editors. 2020. *Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies*, Online, July. Association for Computational Linguistics.

Mary Dalrymple, Stuart M. Shieber, and Fernando C. N. Pereira. 1991. Ellipsis and higher-order unification. *Linguistics and Philosophy*, 14(4):399–452.

Yehuda N. Falk. 2010. An unmediated analysis of relative clauses. In Miriam Butt and Tracy Holloway King, editors, *Proceedings of the LFG10 Conference*, pages 207–227. CSLI Publications.

Irene Heim and Angelika Kratzer. 1998. *Semantics in Generative Grammar*. Number 13 in Blackwell Textbooks in Linguistics. Blackwell, Oxford.

Rodney D. Huddleston and Geoffrey K. Pullum. 2002. *The Cambridge Grammar of the English Language*. Cambridge University Press, April.

Edward L. Keenan and Bernard Comrie. 1977. Noun phrase accessibility and Universal Grammar. *Linguistic Inquiry*, 8(1):63–99.

Idan Landau. 2013. *Control in generative grammar: a research companion*. Cambridge University Press, Cambridge, GB.

Richard Montague. 1970. Universal grammar. *Theoria*, 36(3):373–398.

Joakim Nivre, Paola Marongiu, Filip Ginter, Jenna Kanerva, Simonetta Montemagni, Sebastian Schuster, and Maria Simi. 2018. Enhancing Universal Dependency treebanks: A case study. In *Proceedings of the Second Workshop on Universal Dependencies (UDW 2018)*, pages 102–107, Brussels, Belgium, November. Association for Computational Linguistics.

Jenna Nyblom, Samuel Kohonen, Katri Haverinen, Tapio Salakoski, and Filip Ginter. 2013. Predicting conjunct propagation and other extended Stanford dependencies. In *Proceedings of the Second International Conference on Dependency Linguistics (DepLing 2013)*, pages 252–261, Prague, Czech Republic, August. Charles University in Prague, Matfyzpress, Prague, Czech Republic.

Agnieszka Patejuk and Adam Przepiórkowski. 2018. *From Lexical Functional Grammar to Enhanced Universal Dependencies: linguistically informed treebanks of Polish*. Institute of Computer Science Polish Academy of Sciences, Warsaw.

Martin Popel, David Mareček, Jan Štěpánek, Daniel Zeman, and Zdeněk Žabokrtský. 2013. Coordination structures in dependency treebanks. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 517–527, Sofia, Bulgaria, August. Association for Computational Linguistics.

Adam Przepiórkowski and Agnieszka Patejuk. 2018. Arguments and adjuncts in Universal Dependencies. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3837–3852, Santa Fe, New Mexico, USA, August. Association for Computational Linguistics.

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, September. Association folr Computational Linguistics.

Sebastian Schuster and Christopher D. Manning. 2016. 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)*, pages 2371–2378, Portorož, Slovenia, May. European Language Resources Association (ELRA).

Edwin Williams. 1980. Predication. *Linguistic Inquiry*, 11(1):203–238.

Alina Wróblewska. 2018. Extended and enhanced polish dependency bank in universal dependencies format. In Marie-Catherine de Marneffe, Teresa Lynn, and Sebastian Schuster, editors, *Proceedings of the Second Workshop on Universal Dependencies (UDW 2018)*, pages 173–182. Association for Computational Linguistics.