Negation in Universal Dependencies

Jamie Y. Findlay and **Dag T. T. Haug**
University of Oslo
j.y.findlay@iln.uio.no, d.t.t.haug@ifikk.uio.no

Abstract

In this paper we study the representation of negation in UD treebanks. We show that the existing annotations are often inconsistent with the guidelines and that there are ill-motivated differences in annotation of constructions across and even within languages. Moreover, we argue that even if the annotation of the two negation-related features (`Polarity=Neg` and `PronType=Neg`) were consistent, these two features would be inadequate for straightforwardly expressing the semantics of negation because they relate to the word level only and hence to form rather than meaning. We therefore propose to add two features, `Negated=+` and `DoubleNegated=+`, which directly encode when a predicate is *semantically* under negation, and thereby allow a straightforward semantic interpretation of a UD parse in terms of negation.

1 Introduction

Negation is a complex but ubiquitous phenomenon in natural language, involving constraints across the syntax-semantics interface. In this paper, we address the representation of negation in Universal Dependencies (UD: Nivre et al. 2020), point out inconsistencies in the current representation, and make a suggestion for how it can be improved, including an implementation that will create this representation for English UD.

Our ultimate motivation for this work is the use of UD in semantic parsing, i.e. the translation of UD syntactic annotations into semantic representations. In Section 2 we discuss this use case for UD and what it entails – especially for feature representation, which is much less standardised than dependency relations. Section 3 then gives a (simplified) overview of the challenges this task raises at the syntax-semantics interface. Negation is a wide field and we therefore choose to focus on some selected cases all involving sentential negation. Unfortunately, the existing UD protocol and its current implementations in the UD treebanks leave something to be desired when it comes to helping with this challenge. In Section 4 we show that the current annotation of negation is inconsistent across treebanks, and sometimes even within single treebanks. In Section 5 we show that even if the annotation was consistent, the word-level, morphology-dominated features currently used in the UD annotation scheme make it challenging to introduce negation compositionally in the semantics. In Section 6, we propose an addition to the UD annotation scheme whereby features are added to the predicate itself to indicate whether it is negated (or doubly negated). Using the Graph Rewriting formalism (GREW: Guillaume et al. 2012; Bonfante et al. 2018), we provide a ruleset which can add these features automatically to UD annotations for English. However, we suggest that these features should be included in the standard annotation scheme in future, in keeping with recent proposals to add phrase- or clause-level features to UD annotations (Savary et al. 2023; Weissweiler et al. 2024).

2 Motivation

There is a tradition of work using UD representations to create semantic representation (Reddy et al. 2017; Poelman et al. 2022; Findlay et al. 2023). Most of this work focuses on the information available in the tree structure and the labels, which clearly provide important clues to the structure of the semantic representations as well as to the semantic relations between different words. But features also have an important role to play, since they provide *language neutral* representations of certain aspects of lexical information, most often in function words. For example, in the absence of a featural representation of negation, a semantic parser will need language-specific lexical information about what words express negation and how. While high quality semantic parsing will likely require fine-grained language-specific information in any case, the process can clearly be simplified to the extent that certain lexically-encoded aspects of meaning can be read off the features in a language neutral way.

This is, however, not possible with the current UD features. Consider tense, for example. Despite the guidelines' claim that `Tense` "is a feature that specifies the time when the action took/takes/will take place",[1] the actual features that we find in UD reflect morphological rather than semantic facts. Part of the reason this is so is that features in UD are necessarily word-bound, and automatic annotation procedures tend not to take the surrounding context into account, so that properties which arise constructionally or periphrastically are not represented. As a concrete example, consider the value of the `Tense` feature on an English verb in the UD treebanks – this depends entirely on the verb's form, and not on its meaning, so *provided* gets `Tense=Past` regardless of whether it is used as a preterite (1), a (present) perfect (2), or a passive of any tense (3):

(1) They also bred small domesticated dogs which [. . .] provided their protein.
`(UD_English-GUM, GUM_textbook_history-37)`

(2) Mary has provided a preface.
`(UD_English-GUM, GUM_letter_arendt-33)`

(3) [T]he tourist card is normally provided on the flight.
`(UD_English-GUM, GUM_voyage_cuba-11)`

It is possible to compute tense values for the verbal complexes in (3), by inspecting other features such as `VerbForm` and `Voice`, but this cannot be done without detailed, language-specific knowledge about the English annotation. This approach is therefore not amenable to large-scale typological investigations, which is one of the oft-cited use cases for UD.

It would be possible to solve this problem "compositionally" by defining universal resolution rules for sets of features coming from both the lexical verb and the auxiliary. A more direct route is to add annotations of *constructions* on top of UD, as argued for by Weissweiler et al. (2024). Here we will take a slightly different route, by allowing head features to represent features of the whole construction. This is largely equivalent to constructional annotations for constructions that have a clear head, which is the case for both verb-auxiliary constructions and verb-negation constructions.[2] Adding features to the head also has a practical advantage by making it possible to directly compare constructional and lexical expressions (although one could of course just admit single word constructions).

Our approach also aligns with the goals of the UniDive shared task on morpho-syntactic parsing (MSP).[3] Following UD, MSP makes a sharp divide between lexical words and function words. The latter are always dependents in UD and can therefore be ignored in the tree structure. Instead they contribute features to their heads. So, in the MSP annotation of *will not go*, only *go* is a node in the tree; *will* and *not* contribute `Fut` and `Neg` to the features of *go*.

Although we do not care about the tree structures and what counts as a node here, we adopt the same principle as MSP that function words can contribute features to their head. This move is in fact what makes it possible for us (and MSP) to take advantage of a much-discussed feature of UD, that function words are always dependents. The main argument for this approach is that it makes annotations more similar across languages that use analytic and synthetic means of expression. As the UD guidelines put it,[4] "Preferring content words as heads maximizes parallelism between languages because content words vary less than function words between languages. In particular, one commonly finds the same grammatical relation being expressed by morphology in some languages or constructions and by function words in other languages or constructions [. . .]". However, the primacy of lexical words only makes the *tree structures* similar, not the *features*, unless we let dependents contribute features to their head.

Consider the example discussed by de Marneffe et al. (2021: 264) shown in Figure 1. The tree topology is identical modulo function words, but the feature representation is not: in Swedish, the subject NP bears both a `Number=Plur` and a

[1]See https://universaldependencies.org/u/feat/Tense.html.

[2]Although UD disagrees with most linguistic analyses in taking the auxiliary to be a dependent of the verb, both approaches agree that the structure is headed.

[3]https://www.aclweb.org/portal/content/1st-shared-task-morphosyntactic-parsing

[4]https://universaldependencies.org/u/overview/syntax.html

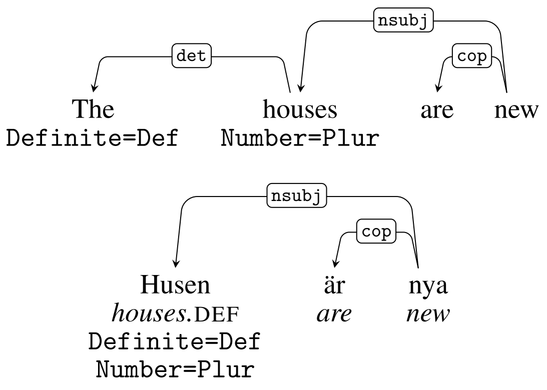


Figure 1: Definiteness in English and Swedish

`Definite=Def` feature, whereas the English subject NP only has the number feature. This means the UD representation cannot be used to study plural, definite noun phrases across languages without some further post-processing. In this case, the rule that is needed to unify the two representations is simple (move the definite feature from the functional dependent to its lexical head), but as we have seen for tense, and as we will see for negation in this paper, this is not always the case. While the percolation of features from functional dependents to lexical heads, as made explicit in MSP, can be seen as a sort of default, the rules of feature construction can be quite involved and often cannot be stated without deep knowledge of the language in question, which is why it should be done by annotators, and not by end users of UD treebanks.

3 The challenges of negation

With all this said, our goal is to explore to what extent the *morphosyntactic* representation of negation in UD treebanks allows for an adequate *semantic* representation of negation in a uniform manner across languages, and to propose amendments to the UD treatment of negation that would facilitate this.

We do not want to commit to a particular meaning representation, so in the following we discuss the inference patterns that negated sentences give rise to, largely following the presentation in Huddleston and Pullum (2002: ch. 9). We do not explicitly discuss the role of tense, but assume, following Partee (1984), that sentences are interpreted relative to some particular time interval that is anaphorically determined in the context, and that this interval is held constant from premises to conclusions in inferencing.

Let us first consider the simple negated transitive clause in (4):

(4) Jason did not eat the porridge.

We take it that this sentence licenses an inference to the disjunction of (5a–c):

(5) a. No eating event took place.
b. The Agent of the eating event was not Jason.
c. The Theme of the eating event was not the porridge.

This is a fairly weak semantics for negation, but it is all that can be attributed to the linguistic information alone. Exactly which of the disjuncts turn(s) out to be true will be determined by context and world knowledge.

Notice that the negation scopes over both the verb's event description and the semantic roles associated with the event. We take it that *clausal* negation (as opposed to so-called *constituent* negation) always negates these parts of the sentence: that is what it means for negation to be clausal. However, clausal negation may or may not scope over adjuncts.

Adjunct scope Consider the following extended variant of (4).

(6) Jason didn't eat the porridge because he had already eaten.

World knowledge tells us that in (6), the most plausible reading is that one or more of the disjuncts that we saw in (5) is true, and the *because*-clause gives the reason why this is so. That is, negation does not scope over the adjunct.

But now consider the extension in (7)

(7) Jason didn't eat the porridge because he was hungry.

Here, world knowledge tells us that it is likely none of the disjuncts in (5) is true; instead only the causal relation between John's hunger and his eating is denied. In other words, he ate the porridge, but not because he was hungry.

The linguistic structure itself is insufficient to distinguish between these two cases, and this is once again a situation where context and world knowledge will guide interpretation.

Negative indefinites Another challenging phenomeon is the fact that, in addition to marking negation on the verb, many languages allow negation to be expressed with so-called "negative indefinites" (Haspelmath 1997: ch. 8), which combine sentence negation and an indefinite (existential) quantifier. Hence (8a) can be paraphrased as (8b).

(8) a. Jason ate nothing.
    b. It was not the case that Jason ate something.

An adequate representation of negation in these cases must be able to account for this equivalence, i.e. that the negation is clausal in spite of being marked only on the object quantifier. In other words, we want *nothing* to contribute a clausal negation, represented on the clausal head, while at the same time filling the object position. This is a direct challenge to the the strict division adopted by MSP, where function words contribute features to their heads and lexical words are nodes in the tree: words like *nothing* need to be able to do both.

4 Negation in the UD treebanks

In this section we explore the variable representations of negation in the UD treebanks (in the 2.15 release: Zeman et al. 2024). It will be seen that there are unfortunate inconsistencies within and across languages, even taking into account typological variation. Ideally, we would like the UD parse to provide us with sufficient information to produce a schematic meaning representation of the sentence, with at least the right logical structure – even though specific predicate names or thematic relations will ultimately have to be retrieved from a lexicon (see e.g. Reddy et al. 2017 or Findlay et al. 2023 for proposals along these lines). However, inconsistencies and lacunae in the annotations make this more challenging than it needs to be, and negation proves to be a case in point.

UD provides two relevant features for representing negation: `Polarity=Neg` and `PronType=Neg`. According to the guidelines,[5] the former is used for "negating particles and words which inflect for polarity (verbs, adjectives, etc.)", including English *not*, *nor*, and *no* as an interjection, while the latter is used for negative pronouns such as English *nobody* or *nothing*, along with other negative proforms such as *never*, and negative determiners such as *no* (as in *no books*).

When it comes to straightforward clausal negation, the treebanks are largely uniform in correctly employing `Polarity=Neg` alone on the negator, whether that be the predicate itself, inflected with a negative affix, as in e.g. Persian, or a dependent of the negated predicate, as in e.g. Norwegian – see Figure 2

There are exceptions even with this most basic kind of negative construction, however. The `UD_Italian-ISDT` treebank, for example, uses `PronType=Neg` for this purpose instead, contrary to the annotation guidelines, and only uses `Polarity=Neg` for the discourse particle/interjection *no* – see Figure 3. This inconsistency between treebanks is an issue for semantic interpretation, since there is no single feature which is reliably associated with clausal negation.

A different challenge comes from treebanks such as `UD_English-EWT`, which make no use of `Polarity=Neg` whatsoever, such that clausal negation is simply not marked in the syntactic analysis. In fact, only 224 of the 296 treebanks in the 2.15 release (76%) use the `Polarity=Neg` feature, and only 99 (33%) use the `PronType=Neg` feature. Overall, 62 treebanks (21%) use neither feature, meaning over a fifth of the treebanks do not represent negation of any kind, even though negation is a phenomenon present in every natural language (Zeijlstra 2020: 426).[6]

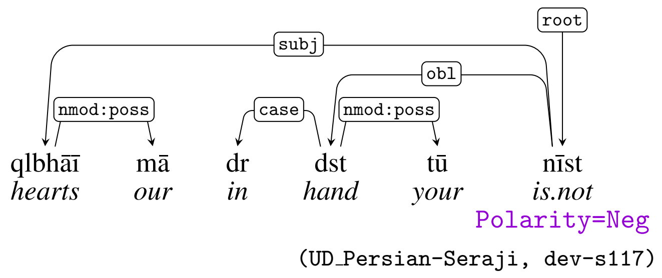
One area of much more cross-treebank consistency is in the treatment of negative conjunctions like *neither* and *nor* or their translation equivalents. Unfortunately, this is because treebanks consistently *fail* to annotate these words with any feature that indicates a negative polarity. This is once again contrary to the annotation guidelines – which explicitly mention English *nor* as meriting the feature `Polarity=Neg`, for example – and is also plainly unsatisfactory from the standpoint of interpretation: (9a) is truth-conditionally equivalent to (9b), and therefore just as clearly negated, but only the latter will contain a `Polarity=Neg` as things stand:

(9) a. Neither did Jonathan.
    b. Jonathan didn't either.

In other areas, inconsistencies abound. Neg-

[5]See https://universaldependencies.org/u/feat/PronType.html#Neg and https://universaldependencies.org/u/feat/Polarity.html.

[6]The script used to calculate these figures is available at https://github.com/findlayjy/negation-in-UD/.



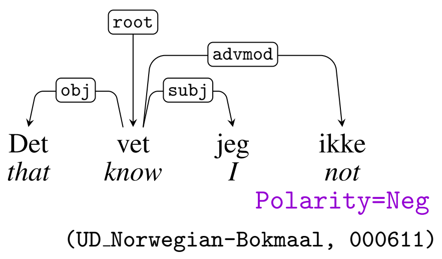


Figure 2: UD trees for two negated sentences

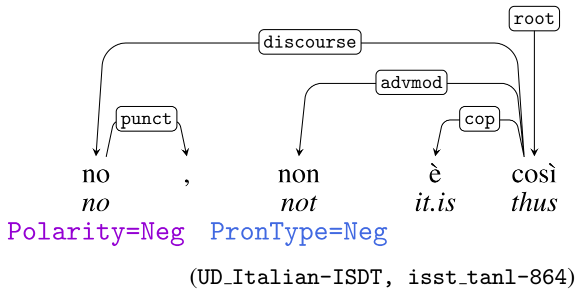


Figure 3: The two features in UD_Italian-ISDT

ative determiners, for example – the equivalent of English *no* (as in *We have no bananas*) – ought to be annotated with a PronType=Neg feature according to the UD guidelines. And some treebanks do just this, such as UD_Persian-Seraji. But we also find such words annotated with only a Polarity=Neg feature, as in UD_Norwegian-Bokmaal, or with neither feature, as in UD_Turkish-BOUN. Part of the challenge here comes from the fact that in negative concord languages these determiners can appear under negation, such that they have the (negative polarity item) meaning 'any' (as in *We do not have any bananas*). But, as noted above, treebanks generally choose a single annotation for a particular word regardless of context, and this therefore leads to inaccuracies in both directions. For example, Turkish *hiç* can have a directly negative meaning but is consistently annotated as if it cannot, while the Persian cognate *hīč* is explicitly marked as negative but can equally well have the NPI meaning.

Treatment of negative indefinites also varies substantially, even within a single language. We take UD_English-GUM as a case study. Table 1 shows the spread of features used across various negative words in the treebank in comparison with what the UD guidelines would lead us to expect. As can be seen, only three of the nine n-words are annotated as the guidelines would suggest. Among the quantifiers/indefinites, only *nobody* is annotated "correctly"; the others all bear an unexpected Polarity=Neg feature. But what is also striking is that only two of the four quantifiers bear the expected PronType=Neg feature. Perhaps *nobody* and *nothing* are distinguished from *nowhere* and *never* in UD_English-GUM because the former are pronouns while the latter are broader proforms (e.g. they can replace PPs), but the guidelines explicitly list all four as examples of words that should carry the PronType=Neg feature. However, we can understand the UD_English-GUM creators' decision here, since this use of PronType=Neg is somewhat at odds with the higher-level, more general guidelines around the feature PronType, which say that, as the name would suggest, it "typically" applies to pronouns and other pronominal items.[7] (Of course, this is presumably one of the "atypical" times, where the more specific guidelines should supersede the broader ones.)

On the other hand, we do not understand the decision to distinguish *nobody* from *nothing* by the former's lack of a Polarity=Neg feature; it is not clear to us what linguistic distinction this could be intended to capture.

In sum, the existing UD treebanks handle negation in an inconsistent manner, both across and within treebanks. Because of this, neither of the two negation-related features can be given a consistent semantic interpretation. This is problematic for our goal of providing a skeletal semantic representation of the logical structure of a sentence, since negation plays an important role in determining this structure.

5 Insufficiency of the annotation scheme

Lack of annotation consistency makes universal interpretation directly from the features impossible. Certainly, clarifying the intended usage of both

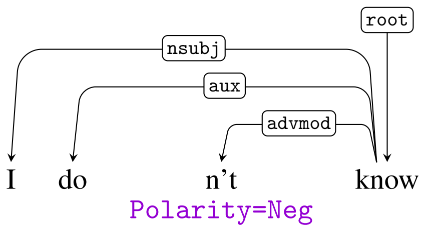
[7] https://universaldependencies.org/u/feat/PronType.html

	Polarity=Neg		PronType=Neg	
	GUM	UD-G	GUM	UD-G
not	✓	✓	–	–
no$_{disc}$	✓	✓	–	–
neither	–	✓	–	–
nor	–	✓	–	–
no$_{det}$	✓	–	✓	✓
nobody	–	–	✓	✓
nothing	✓	–	✓	✓
nowhere	✓	–	–	✓
never	✓	–	–	✓

Table 1: Assignment of negative features to n-words in the English GUM corpus in comparison with the UD guidelines (UD-G)

Polarity=Neg and PronType=Neg and achieving greater coverage and consistency across the treebanks would help in this regard. Nevertheless, we believe even treebanks perfectly annotated according to the existing guidelines would be inadequate as the basis for semantic interpretation when it comes to negation. Let us see why.

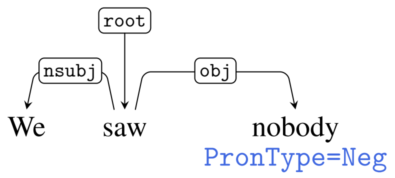
The most elementary clausal negation is associated with the feature Polarity=Neg.

(10)


(English-GUM, GUM_conversation_blacksmithing-26)

We might therefore suggest that this feature triggers the introduction of negation in the semantics, so that (10) has the interpretation described in Section 3: there is no 'knowing' situation with the speaker as the Experiencer.

What about the other feature, PronType=Neg? In languages like English or Norwegian, which lack negative concord, this also sometimes needs to introduce clausal negation into the semantics:

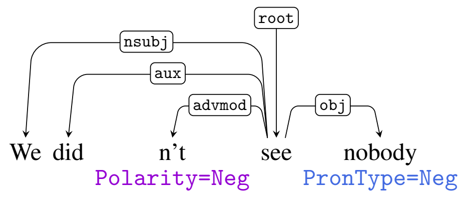
(11)


The sentence in (11) is, at least truth-conditionally, equivalent to (12), which has an explicit clausal negation:[8]

(12) We didn't see anybody.

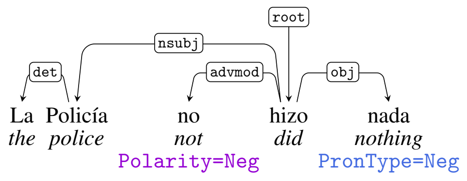
So it seems reasonable to assume that PronType=Neg introduces such a meaning in (11) as well.

What is more, when both features are present, we get a double-negation reading (in non-negative concord varieties of English, at least):

(13)


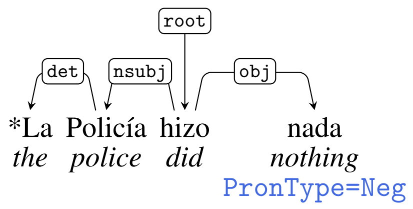
It therefore seems perfectly well-motivated to have both negation-related features introduce a clausal negation meaning.

Of course, when we come to negative concord (NC) languages, things change. Despite having two negative words, (14) does not express double, but only single negation – 'The police did nothing':

(14)


(UD_Spanish-AnCora, 3LB-CAST-n1-2-s2)

What is more, the PronType=Neg word cannot express clausal negation alone, at least in this position:

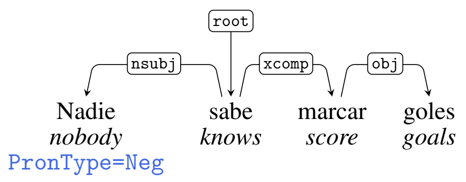
(15)


[8]Indeed, in languages like Norwegian, there is a systematic alternation betwen these two types of sentence: the form with an explicit negator is required with compound tenses, while both forms are compatible with simplex tenses:

(i) a. Han har ikke sett noe.
he has not seen something.
'He has not seen anything./He has seen nothing.'
b. Han så { ikke noe | ingenting }.
He saw not something nothing
'He didn't see anything./He saw nothing.'

We could, therefore, parametrise our semantic interpretation rule so that PronType=Neg introduces a negative meaning in non-negative concord languages, but does not do so in negative concord languages.

Unfortunately, this is not the whole story. Negative quantifiers in NC languages *do* sometimes introduce clausal negation into the semantics, when they are in subject position; (16) means 'Nobody knows how to score goals', i.e. 'It is not the case that somebody knows how to score goals.':

(16)


(UD_Spanish-AnCora, 3LB-CAST-d2-3-s1)

So we cannot simply parametrise the PronType=Neg feature and 'switch off' its semantic contribution in NC languages, since in fact it is sometimes still needed.

6 Proposal

Negation is in a similar bind to tense, as discussed above (in Section 2): the current annotations really relate to form alone, which means that, for example, no accommodation is made for (a) the difference between NC and non-NC languages, and (b) the contrast between different uses of n-words within a single language. Our solution is to introduce a new feature, Negated=+, which is annotated directly on a predicate which is *semantically* under negation, regardless of any morphosyntactic concerns. In the next section we explain how this can be interpreted, but for now we can observe that such a move also contributes to an increased universality in representations, since it establishes a clearer parallel between synthetic and analytic expressions of negation: in Figure 4, for example, we see that the English and Czech verbs both bear the Negated=+ feature, even though the existing Polarity=Neg feature appears in different places in the two languages.

In addition to the Negated=+ feature, we also add a feature DoubleNegated=+, for sentences like *We didn't see nobody* in non-NC varieties, where this truly expresses a double negation. An alternative would be to simply omit the Negated=+ feature in this case, on the basis that the semantics is equivalent to that of positive *We saw some-*

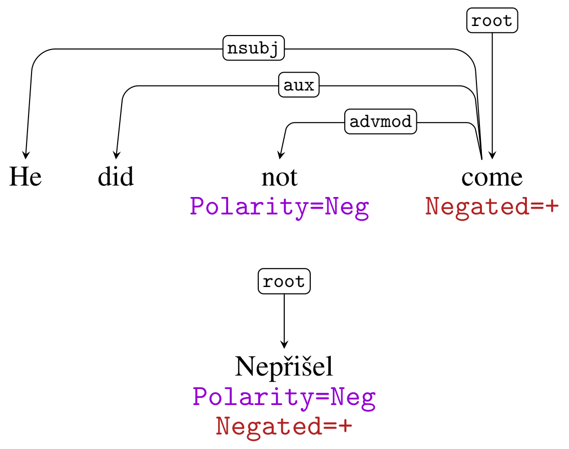


Figure 4: An English sentence and its Czech translation; the verb bears the Negated=+ feature in both cases, even though Polarity=Neg appears in different places in the two languages

body, but we choose to include more information rather than make the decision that double negations should be interpreted exactly the same way as un-negated sentences – after all, they are clearly pragmatically marked, and we prefer to leave open the space of analytical possibilities. Przepiórkowski and Patejuk (2015: 327–328) suggest that double negation actually consists of one instance of clausal negation and one of constituent negation, and that multiple instances of a single type are not in fact possible. This may be so, but UD does not always enable us to make the distinction between these two types of negation explicit in the parse, so we use the feature DoubleNegated=+ as a proxy for a more sophisticated analysis.

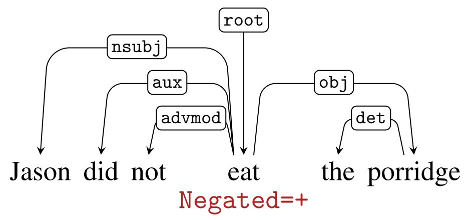
For future annotation projects, we would advocate for adding these annotations from the start, and preferably manually, in order to capture complex interactions or edge cases. But given a little language-specific information about different negative lemmata, they can also be added automatically, and we have prepared a script to do just this for the English treebanks, using the Graph Rewriting formalism (GREW: Guillaume et al. 2012; Bonfante et al. 2018).[9]

The Negated=+ and DoubleNegated=+ features allow for a straightforward translation from features to semantics: the presence of each of these features adds a new instance of negation to the semantics, one which scopes over the sub-tree rooted by the node bearing the annotation. This is a fairly

[9]This is available at https://github.com/findlayjy/negation-in-UD.

simplistic semantics for negation, but has the virtue of being able to be read directly off the UD tree. It will give rise to an interpretation which is consistent with but usually be weaker than the fully contextually-resolved meaning, as discussed in Section 3.

As mentioned above, a clausal negation like (17) entails the disjunction in (18):

(17)


(18) a. No eating event took place.
b. The Agent of the eating event was not Jason.
c. The Theme of the eating event was not the porridge.

We can rephrase this disjunction of negations as a conjunction under a single negation (via De Morgan's law):

(19) It was not the case that
a. an eating event took place, and
b. the Agent of the eating event was Jason, and
c. the Theme of the eating event was the porridge.

It is clear to see how this interpretation follows from the addition of the `Negated=+` feature, since these conjuncts correspond to the semantic contributions of the verb itself, the `nsubj` dependency, and the `obj` dependency, respectively, i.e. the subtree rooted in the *eat* node which hosts the feature.

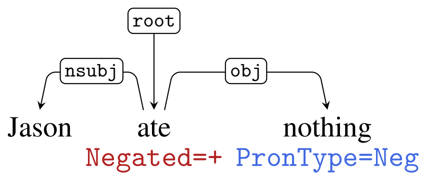
Of course, context may well provide us with the necessary information to determine a stronger meaning: perhaps there *was* a porridge-eating event, but Jason was not the Agent, so that only the second conjunct is actually false. Nevertheless, the UD parse will never give us sufficient information to determine this, so it is best to provide a weak but at least consistent semantics: even if it is true that a porridge-eating event took place with Isabella as the Agent, this does not diminish the truth of (19), since the *conjunction* of statements remains false.

We apply exactly the same logic to more obviously ambiguous sentences like (20):

(20) Jason didn't eat the porridge because he was hungry.

Here the most plausible interpretation, given our world knowledge, is that it is only the Cause relation between Jason's hunger and his eating the porridge which is denied. But this state of affairs is still consistent with the claim that there was no porridge-eating event with Jason as Agent and his hunger as Cause. Since the syntax alone does not tell us which or how many of these conjuncts are true or false, it is reasonable for our syntax-driven interpretation to provide only a weak semantics, allowing the context to narrow it down if possible. It is much easier to see how context might add additional information that narrows the space of possible interpretations, for example, than how it could broaden an already too-narrow interpretation.

An advantage of adding our annotation to the verb itself rather than the explicitly negative element, which may be a dependent, is that often the verb itself should be under the scope of negation, even when the negation is introduced lower down in the tree. Sentence (21) is a case in point:

(21)


The sentence in (21) should most plausibly not mean there was an eating event with no Theme, for example, but rather that there was no eating event (with some Theme) at all. This requires us to introduce the negation meaning at the verb, rather than just at the quantifier *nothing*, which is where the only negation-related feature would be found in the vanilla annotation.

Notice that this implies that a negative quantifier like *nothing* makes the same contribution to the compositional semantics as a positive quantifier like *something*. This means that we can treat NC and non-NC languages the same in this regard; the only difference is a syntactic one, namely when it is that negative quantifiers induce the presence of a `Negated=+` or `DoubleNegated=+` feature on their governor. In a language like English, they always will; in a language like Spanish they will do so only when in subject position.

7 Conclusion

UD's features are restricted to the word level, and this often means that annotators base their decisions on form instead of meaning. This is problematic if one wants to derive a semantic representation from a UD one, since many features which 'ought' to directly express semantic attributes do not. Negation is a case in point, which we have focussed on here. The existing treebanks apply the annotation guidelines incompletely and inconsistently for the two negation-related features (`Polarity=Neg` and `PronType=Neg`), but even if they did not, these two features would be inadequate for straightforwardly expressing the semantics of negation. We propose the addition of two features, `Negated=+` and `DoubleNegated=+`, which directly encode when a predicate is *semantically* under negation. If these are added as part of the annotation process, they will enable a straightforward semantic interpretation of a UD parse in terms of negation, at least for the cases we discussed here, namely basic clausal negation, adjunct scope, and negative quantifiers.

Acknowledgements

We would like to thank Saeedeh Salimifar and Ahmet Yıldırım for useful discussions of Persian and Turkish data.

References

Guillaume Bonfante, Bruno Guillaume, and Guy Perrier. 2018. *Application of Graph Rewriting to Natural Language Processing*. Number 1 in Logic, Linguistics and Computer Science set. ISTE & Wiley, London.

Marie-Catherine de Marneffe, Christopher D. Manning, Joakim Nivre, and Daniel Zeman. 2021. Universal Dependencies. *Computational Linguistics*, 47(2):255–308.

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.

Bruno Guillaume, Guillame Bonfante, Paul Masson, Mathieu Morey, and Guy Perrier. 2012. Grew : un outil de réécriture de graphes pour le TAL (Grew: a graph rewriting tool for NLP) [in French]. In *Proceedings of the Joint Conference JEP-TALN-RECITAL 2012, volume 5: Software Demonstrations*, pages 1–2, Grenoble, France. ATALA/AFCP.

Martin Haspelmath. 1997. *Indefinite pronouns*. Oxford Studies in Typology and Linguistic Theory. Clarendon Press, Oxford.

Rodney Huddleston and Geoffrey K. Pullum. 2002. *The Cambridge grammar of the English language*. Cambridge University Press, Cambridge.

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 Twelfth Language Resources and Evaluation Conference*, pages 4034–4043, Marseille, France. European Language Resources Association.

Barbara H. Partee. 1984. Nominal and temporal anaphora. *Linguistics and Philosophy*, 7(3):243–286.

Wessel Poelman, Rik van Noord, and Johan Bos. 2022. Transparent semantic parsing with Universal Dependencies using graph transformations. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4186–4192, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Adam Przepiórkowski and Agnieszka Patejuk. 2015. Two representations of negation in LFG: evidence from Polish. In *Proceedings of the LFG15 Conference*, pages 322–336, Stanford, CA. CSLI Publications.

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.

Agata Savary, Sara Stymne, Verginica Barbu Mititelu, Nathan Schneider, Carlos Ramisch, and Joakim Nivre. 2023. PARSEME meets Universal Dependencies: getting on the same page in representing multiword expressions. *Northern European Journal of Language Technology*, 9(1).

Leonie Weissweiler, Nina Böbel, Kirian Guiller, Santiago Herrera, Wesley Samuel Scivetti, Arthur Lorenzi, Nurit Melnik, Archna Bhatia, Hinrich Schütze, Lori Levin, Amir Zeldes, Joakim Nivre, William Croft, and Nathan Schneider. 2024. UCxn: Typologically-informed annotation of constructions atop Universal Dependencies. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 16919–16932, Torino, Italia. ELRA and ICCL.

Hedde Zeijlstra. 2020. Negative quantifiers. In Viviane Déprez and M. Teresa Espinal, editor, *The Oxford handbook of negation*, pages 426–440. Oxford University Press.

Daniel Zeman et al. 2024. Universal Dependencies 2.15. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.