Towards a Universal Dependency Corpus for Old Saxon (Old Low German)

Christian Chiarcos[a], Janine Siewert[a,b]
[a]Applied Computational Linguistics (ACoLi), University of Augsburg, Germany
[b]Department of Digital Humanities, University of Helsinki, Finland
christian.chiarcos@uni-a.de, janine.siewert@helsinki.fi

Abstract

Among the West Germanic languages of the first millennium C.E. (Old English, Old Low Franconian/Old Dutch, Old High German, and – although much later – Old Frisian), Old Saxon occupies a special role both linguistically – in that it represents a middle ground in the dialect continuum between Old English at one extreme and Old High German on the other –, and in terms of material quality, in that it is attested with considerable amounts of coherent text (unlike Old Low Franconian) which is not only particularly old (unlike, especially, Old Frisian), but also original (i.e., not translated, a rarity in the attested Old English and Old High German material). It is thus a language central to the understanding of the emergence of several modern major languages, incl. English, Dutch and German, and has been studied intensely, albeit – so far – not in the context of the Universal Dependencies. This paper addresses this gap and describes the introduction of (a) a manually annotated test corpus of Old Saxon, (b) a highly reusable conversion pipeline for converting the Penn bracketing syntax of the Penn Historical Corpora (and the Old Saxon Heliand) to UD, and (c) the evaluation of the latter against the manual annotations.

Keywords: Universal Dependencies, Old Saxon (Old Low German), Penn bracketing syntax, UD conversion and evaluation

1. Introduction

The Old Saxon (Old Low German) language has been the historical language of the Saxons, a Germanic tribe settling primarily in North-Western Germany in the first millenium C.E., neighboring Scandinavians (the linguistic predecessors of Danish) to the north, Frisians to the north-west, the Frankish empire (the linguistic predecessors of modern Dutch and German) to the south and west and Slavs to the east. Linguistically, the languages of the Saxons, Frisians and Franks formed a dialect continuum that extended further onto the British Isles whose early Germanic inhabitants settlers also adopted a self-designation as Saxons when they began their emmigration after the Romans gave up the province of Britannia in 410 C.E. In a series of intense wars, the continental Saxons were conquered by the Franks in the 780s, and part of their political and social integration into the Frankish empire included their Christianization, giving rise to the Old Saxon literature, primarily attested in the *Heliand*, a 9th c. gospel harmony, the *Genesis*, a free, poetic retelling of the first book of the Bible, and in a number of smaller, fragmentary texts and glosses.

In this paper, we describe the creation of a new corpus of Old Saxon, syntactically annotated in accordance with Universal Dependency (UD) guidelines. For the most part, the corpus is, however, not manually annotated, but *converted*, or, more precisely, compiled from existing corpora with morphosyntactic and/or syntactic annotation. These annotations, applied to (different editions of) the same source text, were however, conducted in accordance with different linguistic traditions (primarily interlinear glossed text, or tier-based annotations, and phrase structure syntax), so that substantial efforts in into their harmonization and consolidation are required in order to provide a CoNLL-U representation that encompasses all sources of information. We thus refer to our corpus as *Consolidated Old Saxon* (ConOS) corpus. Its current release[1] consists of UD-compliant data automatically compiled from the respective source corpora and the accompanying conversion scripts, encompasses a total of 109,000 tokens, which, however, represent two *versions* of the same text. Beyond that, we provide UD ConOS, a manually annotated validation set of 2,000 tokens, published as part of the Universal Dependencies.[2] In addition to the first fit (chapter) of the Heliand text, it also provides annotations of a fragment of the Old Saxon Genesis.

At the core of the ConOS corpus is the *Heliand*. As the most extensive Old Saxon text, it occupies a central position in early West Germanic philology and its linguistic characteristics are crucial for reconstructing early syntactic change (Petrova and Solf, 2009; Lühr, 2025). Reflecting its importance,

[1]Available at https://github.com/nds-spraakverarbeiden/ConOS

[2]Available at https://github.com/UniversalDependencies/UD_Old_Saxon-ConOS

Ms. M:	that	inatorht	lico	tidi	gimanodun
Ms. C:	that	ina	torohtlico	tidi	gimanodun
	that	him	brilliant	times	remembered

'that he was remembered of great times'

Table 1: Differences in word segmentation

it has been annotated repeatedly in independent projects: the HeliPaD treebank (Walkden, 2016), following Penn-style constituency annotation; the Heliand DDD corpus (Referenzkorpus "Altdeutsch") with tier-based morphosyntactic and clausal annotation; and the Heliand B4 corpus (Linde, 2009), developed for diachronic information-structural research. These corpora differ in granularity. HeliPaD provides full phrase-structure parses using the Penn bracketing notation; DDD and B4 offer partial, non-recursive annotations (POS, clause boundaries, and – B4 only – nominal and prepositional phrases). The consolidation of their annotations and the creation of the ConOS corpus is primarily described in Chiarcos and Siewert (2026). The current paper focuses on the creation of the manually annotated UD ConOS corpus which represents the gold data used for evaluating the conversion and consolidation task, and which is released as a UD data set.

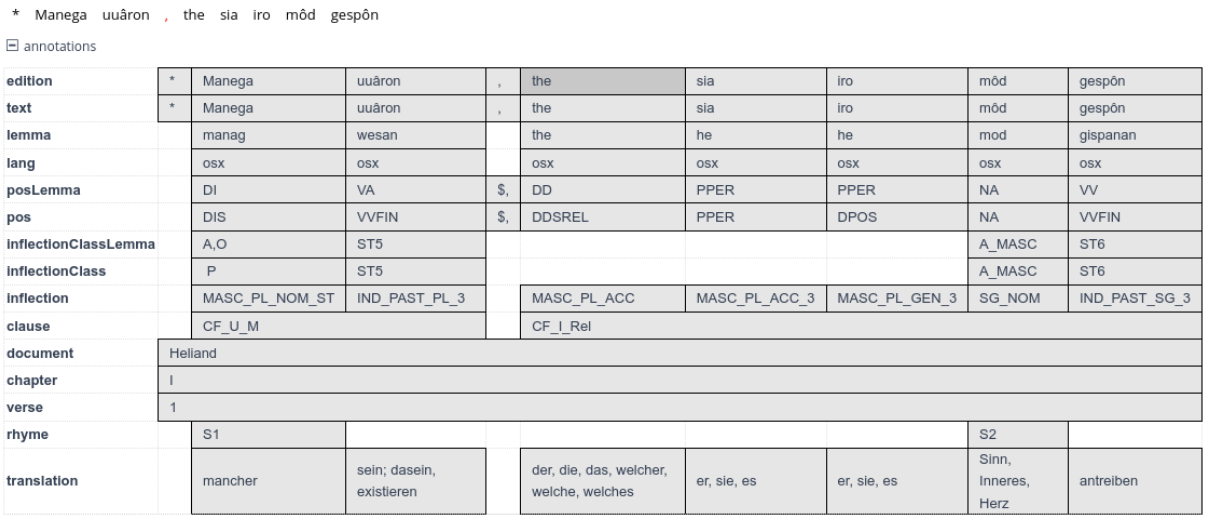
2. Old Saxon: Texts and Corpora

Old Saxon (OS) is attested in two major texts, Heliand and Genesis, which are both alliterative rhyme poems dated back to the 9th century. In addition, there is a small number of liturgical texts and tax lists, but limited in size, so that research on Old Saxon primarily focuses on Heliand and Genesis. With about 6,000 long verses, the Heliand is the more extensive, and so far, has remained the primary basis for corpus-based analyses. The OS Heliand was handed down in two major manuscripts, C (Ms. Cotton. Calig. A. VII, London, British Library) and M (Cgm 25, München, Bayerische Staatsbibliothek) as well as in four fragments. The text is divided into 72 chapters ('fits') indicated in the manuscripts by initial capital letters (Ms. M and C) and Roman numbers (Ms. C only). A major difference between the two manuscripts is in their word segmentation, as exemplified in Table 1. This will require special consideration when aligning the corpora in future extensions of our work.

The earliest syntactically annotated corpus of Heliand (Linde, 2009, Heliand-B4) was developed along with digital editions for a number of Old High German texts (Petrova et al., 2009). With only 3.500 tokens, it covered less than 10% of the text, but the corpus, in particular in the design of the corpora developed here inspired the subsequent DDD project: The Old German Reference Corpus (Referenzkorpus Altdeutsch, DDD) (Linde and Mittmann, 2013; Dipper et al., 2013) aimed to digitize and annotate all extant Old German texts, including the Old Saxon *Heliand*, the *Genesis* and a number of smaller texts (primarily tax registers and short religious texts) by combining automated pre-annotation from historical glossaries with manual refinement. For Heliand and Genesis, the glossary by Sehrt (1925) provides extensive lemma, morphological, and attestational information that was digitized by the project and linked with the text to derive morphosyntactic annotations in an automated fashion (Linde and Mittmann, 2013). These have been manually refined and extended using ELAN, a tool for multi-tier annotation in the tradition of interlinear glossed text (Bow et al., 2003, IGT), with detailed morphological annotation and labels for clause linkage, but without any annotation of phrase structure or grammatical relations within the clause. Figure 1 illustrates DDD annotations for the first verse of Heliand, visualized with ANNIS.[3]

The first corpus to introduce full syntactic annotations for Old Saxon was the *HeliPaD* (Heliand Parsed Database) (Walkden, 2016), annotated according to the standards of the Penn Corpora of Historical English (Taylor et al., 2003), employing explicit constituency structures, grammatical function labels, empty categories, and indices to represent long-distance dependencies. HeliDaD comprises 46,000 tokens, based on an older, authoritative edition (Sievers, 1878). The corpus is designed for the comparative study of syntax, in particular among the Penn Treebank family of parsed corpora. HeliPaD is the only corpus of Old Saxon to offer explicit constituency structures with recursive phrase structures and grammatical relations. For the (partial syntactic analysis of the) first line of the Heliand, Fig. 2 shows pecularities of this format: Phrases are marked by Lisp-style brackets, with the first element representing the category of the phrase, optionally followed by grammatical role annotations, and the following elements representing the respective child nodes. Words (tokens) are terminal nodes in this tree, with part-of-speech (`XPOS`) annotations as their phrase label and represented as a concatenation of word form and lemma, separated by `-`. The annotation also uses a considerable amount of empty elements, including traces (marked with `*`) and non-textual elements (marked with `CODE`). The category and relation inventory and the `XPOS` annotations are based on the Penn Treebank (Taylor et al., 2003), albeit with extensions for historical languages. This includes the annotation of morphosyntactic features attached to the part of speech and separated by ^, e.g., `N^N^SG` for a noun (`N`) in

[3] https://korpling.german.hu-berlin.de/annis/ddd#_q=dG9rCg&ql=aql&_c=RERELUFELUhlbGlhbmRfMS4y&cl=5&cr=5&s=0&l=10&_seg=ZWRpdGlvbg&



* Manega uuâron , the sia iro môd gespôn

annotations

edition	*	Manega	uuâron	,	the	sia	iro	môd	gespôn
text	*	Manega	uuâron	,	the	sia	iro	môd	gespôn
lemma		manag	wesan		the	he	he	mod	gispanan
lang		osx	osx		osx	osx	osx	osx	osx
posLemma		DI	VA	$,	DD	PPER	PPER	NA	VV
pos		DIS	VVFIN	$,	DDSREL	PPER	DPOS	NA	VVFIN
inflectionClassLemma		A,O	ST5					A_MASC	ST6
inflectionClass		P	ST5					A_MASC	ST6
inflection		MASC_PL_NOM_ST	IND_PAST_PL_3		MASC_PL_ACC	MASC_PL_ACC_3	MASC_PL_GEN_3	SG_NOM	IND_PAST_SG_3
clause		CF_U_M			CF_I_Rel				
document	Heliand								
chapter	I								
verse	1								
rhyme		S1						S2	
translation		mancher	sein; dasein, existieren		der, die, das, welcher, welche, welches	er, sie, es	er, sie, es	Sinn, Inneres, Herz	antreiben

Figure 1: Heliand DDD annotation, first verse, visualized with ANNIS4

```
( (IP-MAT
    (CODE <P_7>)
    ...
    (NP-SBJ *exp*)
    (NP-PRD (Q^N^PL Manega-manag)
            (CP-REL *ICH*-1))
    (BEDI^3^PL uuaron-wesan)
    (CODE <C>)
    (CP-REL-1
      (WNP-SBJ-2 0)
      (C the-the)
      (IP-SUB (NP-SBJ-RSP-2 (PRO^A^3^PL sia-he))
              (NP-OB1 (PRO$^N^3^SG iro-his)
                      (N^N^SG mod-mod))
              (GE+VBDI^3^SG gespon-spanan)
              (, ,-,)
              ... ))))
```

Figure 2: HeliPaD annotation, first verse, PTB bracketing format

nominative (^N) singular (^SG). A number of similar extensions exist, e.g., for historical English (Pintzuk and Taylor, 1997), historical German Light (2012); Sapp et al. (2024), Yiddish (Santorini, 1993) and Icelandic (Rögnvaldsson et al., 2011), but it is to be noted that neither of these are fully identical with each other, although the Old Saxon schema follows the model of the Old English corpora.

In the following sections, we describe the creation of a manually annotated gold standard for Old Saxon UD annotations, the conversion of HeliPaD to UD and the evaluation of this conversion.

3. Building the UD ConOS corpus

We manually created a UD-compliant test corpus for Old Saxon from the DDD corpus, that consists of two parts: (1) Heliand fit (chapter) 1, based on the text of Behaghel and Taeger (1984, further BT), and (2) Genesis fragment 2, BT edition.

Heliand is structured into a total of 72 'fits', or chapters. UD ConOS provides a manual annotation of the first fit, whose main witness is Ms. C (it is largely missing in Ms. M). For this manual annotation, we rely on an existing morphosyntactic annotation of the Heliand provided by the DDD corpus. This has been conducted independently from the HeliPaD, and is based on the alignment of the glossary of Sehrt (1925) that has been aligned with the source text of the BT edition. Although this text and the HeliPaD ultimately go back to the same manuscript, it is to be noted that the BT text and the HeliPaD text are not identical, but that the BT text is heavily amended, the orthography of all source manuscripts differ from each other and their normalization in BT, and modernizing punctuation of both texts has been created independently for both editions. The Heliand part was based on the B4 corpus, aligned with the DDD annotations for morphosyntax, lemmatization, German glosses, see Chiarcos and Siewert (2026).

The second major text attesting the Old Saxon language is the Old Saxon Genesis, a fragmentarily attested retelling of the first book of the Genesis, presumably written by the Heliand author and surviving in three Old Saxon fragments, and an Old English translation (or, as Old English and Old Saxon are often seen as mere dialects – transliteration).[4] With 1,145 tokens, the second of these fragments is the largest consecutive piece of text of Old Saxon outside the Heliand. The Genesis part of UD ConOS was directly taken from the DDD corpus.

We ground the UD ConOS annotations in the morphosyntactic annotations of the DDD corpus,

[4]Although the Old Saxon and the Old English fragments of the Genesis do not overlap, the Old Saxon source of the Old English text is indicated by a number of spelling errors, where Old Saxon forms are used in place of expected Old English forms.

which provides Heliand and Genesis along with all other major text fragments of the Old Saxon language, with annotations for morphosyntax, agreement features, clausal juncture, and German glosses.

For creating morphosyntactic annotations, we first converted the DDD corpus from its native ELAN format to a CoNLL-U representation: Using a generic ELAN converter, we identified the minimal segments as tokens and then transformed every tier to an independent column. With standard GNU tools (`cut`, `grep` and `sed`), these were transformed to (the `FORM`, `LEMMA`, `XPOS`) columns of CoNLL-U as well as to the `Trans` (translation) property we put into the `MISC` column to keep track of the German translations of the Old Saxon lemmas. For HeliPaD and B4 annotations, DDD annotations were automatically aligned with both corpora, see Chiarcos and Siewert (2026).

Actual annotation was performed with off-the-shelf spreadsheet software (LibreOffice), where we used Excel-style formulas to insert `ID`s and conditional formatting over the columns `ID` and `HEAD` to visualize attachment: from a color scheme ranging from green over yellow to red, the same color indicates the same number, as illustrated in Fig. 7 for the sentence *Sîđoda im thuo te seliđon, habda im sundea giuuaraht bittra an is bruođar;* 'He now went home, (after) he had done bitter sin against his brother' (translation by authors, see Appendix).

For the DDD tagset (Dipper et al., 2013), no UD mapping was known to exist at the time and was thus provided by the authors. For manual annotation, we used the standard filter function of modern spreadsheets to perform a manual mapping from DDD `XPOS` to `UPOS`, by filtering every (group of) UD-equivalent DDD tags and replacing them all together with a UPOS tag. It is to be noted that some choices seem to have been rather arbitrary. As an example, clause-initial *sô* is annotated as either complementizer (and thus `SCONJ`, as in Example 1) or adverb (and thus `ADV`, as in Example 2). Apparently, this is based on the German translation as *daß* 'that (complementizer)' or *so* 'so', but often without concrete contextual triggers that justify this differentiation.

(1)

Sô	*he*	*thô*	*thana*	*uuîrôc*
SCONJ	PRON	ADV	DET	NOUN
so/that	he	now	the-ACC.SG	incense

drôg
VERB
carry.PST.3SG

'As he took the incense' (translation: Scott, 1966) (DDD Heliand)

(2)

Sô	*he*	*ina*	*thô*	*gehungrean*
ADV	PRON	PRON	ADV	VERB
so/that	he	he.ACC	now	starve

lêt
VERB
let.PST.3SG

'As He let Himself hunger' (translation: Scott, 1966) (DDD Heliand)

For UD ConOS, we respected the existing DDD annotations, with `SCONJ` entailing a `mark` dependency and a `acl`, `advcl`, `ccomp` or `xcomp` head, and `ADV` entailing a `advmod` dependency without any implications for the dependency label of its head. Only in two cases, we observed (and corrected) apparent errors in the morphosyntactic annotation of DDD, both of which seem to arise from homonymy. As an example, *lêđo* 'suffering' (German *Leid*) was incorrectly annotated as adjective in the sentence in (3). Indeed, using the adjective *lêđ* 'disgusting' (German *leid*) in place of the homonymous noun appears to be grammatically possible, but would be contextually dispreferred: the sentence continues with '(the suffering) that the Antichrist throws all people into ruin', with the word *lêđo* acting as matrix *noun* for the relative clause.

(3)

Than	*hier*	*ôk*	*thie*	*lêđo*
ADV	ADV	ADV	DET	**NOUN**
then	here	also	the-NOM.SG	**suffering**

kumit
VERB
come.PRS3SG

'then here comes the suffering, too' (translation: authors) (DDD Genesis)

The filtering function was also used to annotate punctuations automatically (either modern punctuation inserted into the historical text or symbols indicating additional spaces or line breaks), in that these were filtered out, with `UPOS` and `DEP` set to `PUNCT` and `punct`, resp., and `HEAD` set to the `ID` of the preceding row (i.e., the preceding word).

Annotation was performed left-to-right (or, in the software, top-to-bottom) by first reading the full sentence and its German glosses, and then annotating the syntactic head of the word under consideration (if the head precedes the word) and its dependency label. If the word points to a head further down, it is not annotated yet. Instead, as soon as the head is annotated, we go back in the `HEAD` column and fill up the annotations of the preceding dependents with the `ID` of the head.

4. From Penn Bracketing Syntax to Universal Dependencies

We describe the conversion of Penn Bracketing annotations of the HeliPaD corpus to Universal Dependencies as this is evaluated against UD ConOS annotations in this paper. For the transformation of and subsequent consolidation of B4 and DDD annotations with the technologies described here see Chiarcos and Siewert (2026).

4.1. Technological prerequisites

Previous tools for converting the Penn bracketing syntax to UD have largely been ad-hoc solutions focusing exclusively on one particular resource or type of annotation, and because none of these clearly separated the conversion task from the mapping logic, every attempt at converting an unseen dataset had to be re-implemented from scratch, see Arnardóttir et al. (2020) for an overview (and yet another, corpus-specific solution).

In our approach, we aim to provide a more reusable alternative and suggest the application of Fintan, the Flexible Integrated Annotation eNgineering platform (Fäth et al., 2020) that, on the one hand, allows to consume standard formats from NLP, corpus linguistics and computational lexicography, including all flavours of CoNLL-TSV (incl. CoNLL-U). These are then, sentence by sentence, transformed to RDF graphs, such that graph rewriting rules can be applied, using the W3C standard SPARQL, a language for querying and transforming RDF graphs. These rules represent the transformation logic and are stored in separate files. Fintan supports parallelization and iteration of such transformations to ensure fast and efficient processing, but as the transformation logic is *declarative*, i.e., decoupled from the conversion between source and target formats, these transformations can be executed in any technical environment that supports RDF data, say, off-the-shelf databases such as Apache Fuseki[5] or Neo4J,[6] in-memory solutions such as TARQL[7] or programming libraries such as RDFLib for Python[8] or Apache Jena for Java and C++.[9]

With SPARQL 1.1, every transformation is a sequence of retrieval (`WHERE`) and update (`INSERT`/`DELETE`) operations, with a number of optimizations for matching complex graph patterns. In particular, this includes SPARQL property paths, i.e., the possibility to perform transitive search (repeatedly following edges of the same kind), inversion (follow an edge in the opposite direction), sequences (follow one or more edges of one type, then another), or disjunctions (follow one kind of edge or another).

Moreover, because transformation are formulated in a series of relatively fine-grained operations, these operations can be re-used in constellations in which similar data structures are to be modified, say, for the conversion from trees to dependencies, regardless of whether the source data was provided in a CoNLL IOB notation, in Penn bracketing syntax, an XML tree or an array of nested JSON dictionaries. All of these are supported by Fintan and input and output formats, but here, we specifically use the CoNLL-RDF customization of Fintan (Chiarcos and Fäth, 2017) for reading and writing CoNLL (and CoNLL-U) data, and the CoNLL-RDF vocabulary (Chiarcos et al., 2021), as summarized in Fig. 3. CoNLL(-TSV) formats represent sentences as blocks of tab-separated rows, one token per line, optionally preceded by comment metadata. Columns encode annotations such as form, lemma, POS, morphological features, dependency relations or task-specific labels. CoNLL-RDF maps each token to a `nif:Word` node, linked via `nif:nextWord`, with column values represented as literal properties in the `conll:` namespace. Some columns receive dedicated treatment: `HEAD`, for example, is resolved into explicit `conll:HEAD` relations that link tokens (`nif:Word`s) with each other (for syntactic dependencies) or the sentence (for roots). The CoNLL-RDF tree extensions (Chiarcos and Glaser, 2020) extend the basic CoNLL-RDF model with the POWLA vocabulary to represent hierarchical structures such as phrase-structure trees. This extension allows Penn Treebank-style parses to be encoded as additional nodes linked via `powla:hasParent` and `powla:next`. Figure 4 shows a fragment in which a token is connected to a phrase node, which in turn participates in a larger tree. Thus, any kind of annotation that can be represented in a TSV format can be represented in a unified RDF graph, which can then can be easily traversed and manipulated with SPARQL, even if the data includes multiple layers of diverse styles of syntax annotation, be it phrase structure trees or dependency syntax.

4.2. HeliPaD conversion

We first converted the native HeliPaD format into the conventional CoNLL representation of the Penn bracketing format, illustrated in Fig. 5, with four columns that we label `WORD`, `POS`, `LEMMA` and `PARSE`. Using the CoNLL-RDF tree extensions, this was parsed into the structure illustrated in Fig. 6

Figure 6 illustrates the conversion process for

[5] https://jena.apache.org/documentation/fuseki2/

[6] https://neo4j.com/blog/developer/rdflib-neo4j-rdf-integration-neo4j/

[7] https://tarql.github.io/

[8] https://rdflib.readthedocs.io/en/stable/

[9] https://jena.apache.org

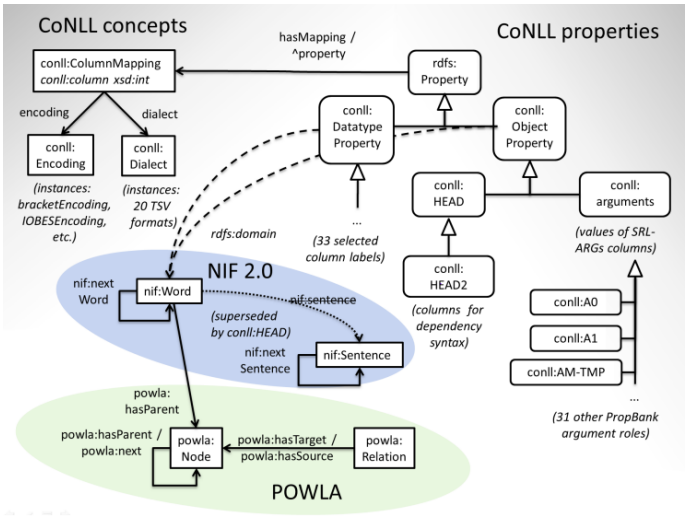


Figure 3: CoNLL-RDF Data Model (Chiarcos et al., 2021)

```
    # CoNLL-RDF conversion with CoNLL-
RDF tree extension
    :s1_0    a                nif:Sentence .
    :s1_7    a                nif:Word ;
             conll:HEAD       :s1_0 ;
             conll:WORD       "Manega" .
             conll:POS        "Q^N^PL" ;
             conll:LEMMA      "manag" ;
             nif:nextWord     :s1_8 ;
             powla:hasParent  :bPARSE_4 .
    :bPARSE_4  a              conll:PARSE ;
             conll:CAT        "NP" ;
             conll:ROLE       "PRD" ;
             powla:hasParent  :bPARSE_2 ;
             powla:next       :bPARSE_5 .
```

Figure 4: CoNLL-RDF fragment for the first word of HeliPaD, with tree extensions

```
<P_7>    CODE          _        ((IP-MAT *
...
*exp*    _             _        (NP-SBJ  *)
Manega   Q^N^PL        manag    (NP-PRD *
*ICH*-1  _             _        (CP-REL  *))
uuaron   BEDI^3^PL     wesan     *
<C>      CODE          _         *
0        _             _        (CP-REL-1(WNP-SBJ-2  *)
the      C             the       *
sia      PRO^A^3^PL    he       (IP-SUB(NP-SBJ-RSP-2 *)
iro      PRO$^N^3^SG   his      (NP-OB1 *
mod      N^N^SG        mod       *)
gespon   GE+VBDI^3^SG  spanan    *
,        ,             ,         *
...
```

Figure 5: Initial CoNLL conversion of HeliPaD

the first verse of HeliPaD. Above is a (slightly simplified) visualization of the CoNLL-RDF input graph of words (green, with values of `conll:WORD`, `conll:POS`, and `conll:LEMMA` properties), phrases (yellow, with values of `conll:CAT` and `ROLE` and the sentence (blue), connected by `nif:nextWord` (⟶), `powla:hasParent` (↑), `powla:next` (between siblings in the parse tree, hidden) and `conll:HEAD` (between every word and the sentence, hidden).

The lower part of Fig. 6 shows results of the graph rewriting operations with a number of tokens dropped (empty tokens, illustrated by gaps) and additional properties (`conll:UPOS` and `conll:EDGE`) derived for the others. In the lower part, the `nif:nextWord` property is adjusted to connect adjacent tokens, but skipped in the diagram to facilitate readability. Further, the `conll:HEAD` property of all tokens has been adjusted to point to their respective syntactic head.

Note that even after rewriting, the original POWLA data structures remain in the RDF graph. However, the CoNLL-RDF/Fintan writer is asked to produce a table of (`conll:`)`ID` (automatically generated from the number of preceding `nif:nextWord` properties for every token), `conll:WORD` (UD `FORM`), `conll:LEMMA`, `UPOS`, `conll:POS` (UD `XPOS`), `conll:FEATS` (not available, thus filled up with _), (`conll:`)`HEAD` (automatically generated from the `ID` of the `conll:HEAD`), `conll:EDGE` (UD `DEP`) and two empty columns, with words ordered according to the `nif:nextWord` property. Any other property, be it from POWLA, CoNLL-RDF or user-defined name spaces, is quietly lost, and the result effectively corresponds to CoNLL-U.

As the figure illustrates, complicated rewriting rules are required, which can be conveniently implemented in SPARQL. As a first step, we simplify the label inventories (e.g., we infer `conll:CAT IP` for every element with `conll:CAT IPX` – and keep both values), and we detach `nif:Word`s that represent punctuation and `CODE` elements. Then, we identify the location of the syntactic head (`temp:HEAD`) for every phrase type, using a total of 12 rules implemented by 21 SPARQL update operations, e.g.,

1. for a non-branching `powla:Node`, the head is

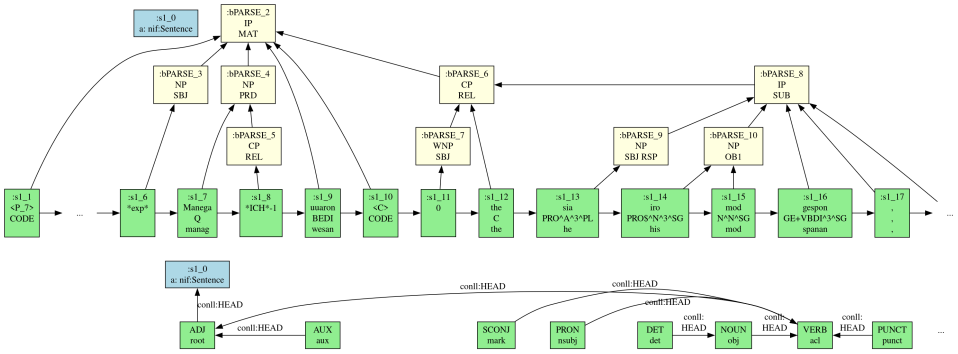


Figure 6: HeliPaD transformation (CoNLL-RDF: PTB-based input graph and derived UD data structures)

the child

2. the head of a node with `conll:CAT` `CONJP` is the first conjunct, the `conll:CAT` of the head is copied to the parent
3. the head of a `PP` node is the first child with `conll:CAT` `NP`
4. the head of any `PP` node without head annotation is the first child not carrying `conll:POS` `P`, etc.

Unfortunately, the relatively large number of rules is unavoidable due to the idiosyncrasies of both PTB- and UD-styles of syntactic annotation. A number of additional SPARQL updates are used in preprocessing to simplify the label inventories.

The listing below illustrates an example for the SPARQL updates employed, where we identify the head-carrying child for every `CP` node as its first child node with `conll:CAT` `IP`:

```
INSERT {                          # set a value for
  ?node temp:HEAD ?child;         # temp:HEAD
        rdfs:comment "(g)"        # with rule (g)
} WHERE {                         # as follows:
  ?node conll:CAT "CP".           # for every CP node
  MINUS { ?node temp:HEAD [] }    # without HEAD
  ?child powla:hasParent ?node;   # use a child node
         conll:CAT "IP".          # with category IP
  MINUS {                         # if there is no
        [ conll:CAT "IP" ]        # IP node that
          powla:next+ ?child }    # precedes it
};
```

As the comments underline, this directly translates to human-readable instructions (and vice versa). Also note that in addition to the transformation, we also document all applied operations by adding a `rdfs:comment` that identifies the responsible rule to facilitate subsequent debugging.

As a result of these rules (and a final fall-back rule to set the `temp:HEAD` to the first child node that contains at least one non-empty token), every `powla:Node` contains exactly one `temp:HEAD`. The listing below contains the SPARQL Update that uses the transitive closures of the `temp:HEAD` property (`temp:HEAD*` and `temp:HEAD+`) to derive the `conll:HEAD` property that directly connects individual words rather than phrases:

```
INSERT {
 ?w conll:HEAD ?head.
} WHERE {
    ?head a nif:Word; (^temp:HEAD)+ ?phrase.
 ?child powla:hasParent ?phrase.
 MINUS { ?phrase temp:HEAD ?child }
 ?child terms:HEAD* ?w.
 ?w a nif:Word.
};
```

For every `?phrase`, this update defines the `?head` as a `nif:Word` on the transitive closure of `temp:HEAD` (`(temp:HEAD)+`). For every `?child` node of `?phrase` that is not (`MINUS`) the `temp:HEAD` of `?phrase`, retrieve its syntactic head `?w` (a `nif:Word` identified along the `temp:HEAD*` axis) and set the `conll:HEAD` of `?w` to `?head`. Whereas `temp:HEAD` connected phrasal nodes with their dependents, `conll:HEAD` now is a property pointing from one word to another.

Similarly (but in a separate update), `?w` is assigned *multiple* values for `conll:EDGE`: `conll:ROLE` of `?dependent`, `conll:CAT` of `?dependent` and the concatenation of `conll:CAT` (or `conll:POS`) of `?dependent` and `conll:CAT` of `?phrase`. These are later mapped to UD dependency labels.

After the establishment of `conll:HEAD` properties, i.e., (unlabelled) dependency relations, we now attach any detached word (code or punctuation) to the head of the minimal spanning relation, resp., the overall syntactic root. Then, we perform a rule-based mapping for parts of speech and `conll:EDGE` annotations to UD standards, based on a manual inspection of existing `conll:POS` and `conll:EDGE` values.

Finally, we remove empty tokens, induce the new `conll:ID` values by counting the number of preceding `nif:nextWord` properties for every word and write a TSV-separated table with the properties that represent the equivalents of the standard CoNLL-U columns. Fintan serializes this table based on the `nif:nextWord` sequence, and except for some greater level of flexibility in the CoNLL comments that every sentence comes with (and that also preserves all `rdf:comment` properties attached to the `nif:Sentence`), the result corre-

sponds to the CoNLL-U format. The next sentence is then serialized after an empty line, etc.

The CoNLL-RDF module of Fintan provides a number of convenient visualizations that were helpful in developing this converter. This includes a direct export of 'canonically formatted' CoNLL-RDF, where words are presented in their original order and in a single line, similar to the common JSON-L format, but as valid Turtle/RDF (or, optionally, with syntax highlighting). This also includes a GraphViz export of results and intermediate steps, as well as an ASCII visualization of dependency trees.

5. Evaluation

For evaluating the HeliPaD conversion, we operate with the Heliand subset of our gold data, only. The Genesis subset is reserved for future evaluations of parsers against unseed data. A problem is, however, that tokenization and spelling of the DDD Heliand differs from that of the HeliPaD. This is compensated by CoNLL-Merge (Chiarcos and Schenk, 2018) that can be used to align the text of different witnesses, versions or editions in CoNLL (TSV one-word-per-line) formats. CoNLL-Merge provides several strategies for the automated resolution of tokenzation, encoding or spelling conflicts (keep both, force source tokenization onto target annotations, split into maximum common segments, align by Levenshtein distance), as well as for merging their annotations by putting them into additional columns, following the annotations of the base text (i.e., our gold annotation). We use the default, so that tokens are aligned either because of string identity or positional correspondence . Otherwise, both alignments are kept, but we use GNU tools (`grep`) to filter out mismatches and evaluate alignable words, only. The merged CoNLL file then consists of 20 columns, but as the tokenization (and thus, the `ID`s) of both annotations differ, the HeliPaD `ID` and `HEAD` annotations are first replaced by values from the gold annotation.

With only about 2000 tokens overall, both fragments are relatively small, but nevertheless allow us to evaluate along three different dimensions, i.e., (1) the quality of the transformation rules (over Heliand fit 1), (2) the quality of the HeliPaD parser to replicate HeliPaD-UD annotations (over Heliand fit 1), and, (3) the quality of the HeliPaD parser over unseen text (over Genesis fragment 2). Whereas (2) and (3) are left as topics for future research, the UD ConOS data allows for some preliminary insights into structural differences between (the CoNLL-U edition of) HeliPaD and manual ConOS annotations. As the textual basis of HeliPaD and DDD Heliand is different, only 678 tokens remain for evaluation, with an `UPOS` accuracy of 96.9% (657/678). Recurring errors include confusions between DDD `PRON` and HeliPaD `DET` (4, in two of these cases, the HeliPaD combination of UPOS and dependency was invalid) and `ADJ` (3, for *manag* 'many' and *gihwilik* 'every'), DDD `VERB` and HeliPaD `AUX` (3, for *hebbian* 'to have, to own'), and DDD `VERB` and HeliPaD `ADJ` (3, for participles). We achieve an unlabelled attachment score (UAS) of 76.8% (521/678), and a labelled attachment score (LAS) of 58.6% (397/678).

The UAS seems fair, the LAS numbers are rather low – but also, achieved over a very small data set. In the future, we would like to provide additional fits (Heliand chapters) to substantiate our findings and to assess a number of necessary corrections we observed during evaluation: Upon closer inspection we found that the LAS score may be due to different conceptualizations rather than linguistically different analyses. Checking 124 cases with correct attachment, but divergent labels, we found that 24.1% (30/124) were for DDD `nmod:gen` against HeliPaD `nmod`, and, analoguously, 9.7% (12/124) for DDD `det:poss` against HeliPaD `det`. We found 13 cases of DDD `cc` against HeliPaD `mark` (10.5%, 13/124) which indicate a certain degree of uncertainty also found in the literature with regard to the identification of (subordinate) clauses and their functions, and, if you will, a different school of thought rather than a fundamental disagreement. It is to be noted, however, that these divergent analyses pertain to exactly three words: *endi* 'and', *eftha* 'or' and *ak* 'but'. We would suggest to generally label them as `cc` based on their lemma. For 12 cases (9.7%), we identified an actual error of the HeliPaD conversion, where the traditional UD dependency `iobj` was used for dative arguments rather than the UD v.2 dependency `obl` currently recommended for related languages (esp., modern German). Eliminating these sources of error would lift the LAS to 68.4% (464/678), and, thus, to a more reasonable level.

The UD ConOS currently only comprises the manually annotated subset of the larger ConOS corpus. Both datasets are published under open licenses (see introduction), but the (UD conversion of the) HeliPaD corpus is currently not integrated with the UD ConOS corpus. Unless we can provide complete manual verification or raise the level of agreement between converted and manually annotated Old Saxon data to substantially higher levels, it remains to be published as a complementary, independent, data set, but cross-referenced with UD ConOS in their respective GitHub repositories and their documentation.

6. Results and Perspectives

The UD ConOS corpus is the first UD corpus for Old Saxon, which, although being small-scale, comes

with the prospect to integrate the full HeliPaD corpus, and the Old Saxon parts of DDD into UD. As these conversions have, however, not been manually verified, their UD conversions reside in the primary ConOS repository, separated from UD ConOS. We described the rule-based transformation of the Old Saxon HeliPaD corpus to UD and its evaluation against UD ConOS, with notable contributions on a technical level:

- the first *declarative*, tool-independent technique for mapping Penn Bracketing syntax schemas to UD,
- using Fintan and SPARQL to *disentangle* format conversion (Fintan/CoNLL-RDF) from handling linguistic data structures (SPARQL), facilitating the application of the same transformation logic against different backend technologies (say, an RDF database), and the reusability of transformation operations for different formats (say, XML- or JSON-based formats for phrase structure syntax), and
- evaluation of dependency syntax annotations *across divergent tokenizations* by means of CoNLL-Merge.

Recent years have seen a considerable number of syntactically annotated corpora of older stages of continental West Germanic (Sapp et al., 2024; Dipper et al., 2024; Haiber, 2024), but, aside from the Walkden (2016) and Linde and Mittmann (2013), these focus on the 2nd millennium C.E. As for the older stages of Germanic, so far, only (Old) Icelandic, Gothic and Old English are covered by UD, but we are not aware of any data available for the corresponding historical stages of German (Old High German), Low German/Low Saxon (Old Saxon) and Dutch (Old Low Franconian). Both Icelandic and the original English UD corpora data are based on a conversion of Penn Treebank-style corpora (Arnardóttir et al., 2020),[10] so that, technologically, converting yet another Penn Historical Treebank to UD is novel to a moderate extent, at best. Still, the use of SPARQL and Fintan/CoNLL-RDF technology may be noteworthy, and even more so in combination with CoNLL-Merge: The declarative rules do not only facilitate re-usability for similar conversion tasks, but also, both technologies in conjunction allow to merge multiple CoNLL(-U) files from different sources and to aggregate over them in order to produce richer, consolidated, and more consistent annotations. To some extent, this has been illustrated in previous research, already, but for very different tasks: Chiarcos and Fäth (2019) combined CoNLL-U files from the Universal Dependencies with Skel (anonymized CoNLL) files from the Universal Propositions on the basis of their (largely shared) parts-of-speech annotation, and then used SPARQL to decompose both types of annotation and to aggregate them into a complex, conjoint semantic-syntactic data structure as postulated by Role and Reference Grammar Van Valin (2014, RRG). Similarly, Chiarcos et al. (2022) described how Fintan/CoNLL-RDF and SPARQL can be used to query over syntax annotations from 12 corpora representing the entire body of syntactically annotated historical German available at the time and covering vastly diverse types of annotation, ranging from shallow, tier-based annotations (such as in DDD and Heliand-B4) over phrase structure annotations (such as in the Early New High German treebank of Light, 2012) to dependency syntax (Scheible et al., 2011; Salomoni, 2021).

For languages in which the same text has been annotated multiple times independently (as for Old Saxon) or is attested in different manuscripts (i.e., *everything* published before the advent of the printing press, as books had to be copied by hand and scribes left traces of their languages in the text, so that manuscripts diverged substantially over time), this can now be used to consolidate these texts and their annotations in a systematic, rule-based fashion. This is not to suggest that rule-based methods are to be preferred over machine learning, but that, say, postprocessing annotations produced by machine learning or other techniques can benefit from a principled, systematic and reproducible approach. For the specific case of the Heliand, this even seems necessary, as the existing annotations are complementary in the sense that each of the known corpora provides something unique (e.g., DDD provides more fine-grained clause linking than HeliPaD) and they complement each other in that they allow us to spot errors in the respective other corpora. By combining these independent sources of information, complemented with machine-learning methods to 'fill up the gaps', more knowledge from legacy resources can be preserved in UD, for Old Saxon and beyond.

Because we inherit parts our morphosyntactic annotation and the text itself from existing source corpora, we adhere to their licensing conditions. For UD ConOS, this is an Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0) license with attribution to Zeige et al. (2021) for text and `XPOS` morphosyntax, and to the current paper for `UPOS`, `HEAD` and `DEP` annotations. For the Heliand subset of UD ConOS, we additionally require attribution to Svetlana et al. (2009).

[10] Initial UD data has actually been created from such data, using the StanfordNLP Universal Dependencies converter still available from `https://nlp.stanford.edu/software/stanford-dependencies.shtml`.

7. Bibliographical References

Þórunn Arnardóttir, Hinrik Hafsteinsson, Einar Freyr Sigurðsson, Kristín Bjarnadóttir, Anton Karl Ingason, Hildur Jónsdóttir, and Steinþór Steingrímsson. 2020. A Universal Dependencies conversion pipeline for a Penn-format constituency treebank. In *Proceedings of the Fourth Workshop on Universal Dependencies (UDW-2020)*, pages 16–25, Barcelona, Spain (online).

Otto Behaghel and Burkhard Taeger. 1984. *Heliand und Genesis*, 9 edition. Max Niemeyer Verlag, Tübingen.

Cathy Bow, Baden Hughes, and Steven Bird. 2003. Towards a general model of interlinear text. In *Proceedings of E-MELD Workshop 2003: Digitizing and annotating texts and field recordings*, pages 11–13, East Lansing, Michigan.

Christian Chiarcos and Christian Fäth. 2017. CoNLL-RDF: Linked corpora done in an NLP-friendly way. In *International Conference on Language, Data and Knowledge*, pages 74–88. Springer.

Christian Chiarcos and Christian Fäth. 2019. Graph-based annotation engineering: towards a gold corpus for Role and Reference Grammar. In *2nd Conference on Language, Data and Knowledge (LDK 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Christian Chiarcos, Christian Fäth, and Maxim Ionov. 2022. Querying a dozen corpora and a thousand years with Fintan. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference (LREC-2022)*, pages 4011–4021, Marseille, France.

Christian Chiarcos and Luis Glaser. 2020. A Tree Extension for CoNLL-RDF. In *Proceedings of the 12th Language Resources and Evaluation Conference (LREC-2020)*, pages 7161–7169, Marseille, France (online).

Christian Chiarcos, Maxim Ionov, Luis Glaser, and Christian Fäth. 2021. An Ontology for CoNLL-RDF: Formal Data Structures for TSV Formats in Language Technology. In *3rd Conference on Language, Data and Knowledge (LDK 2021)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik.

Christian Chiarcos and Niko Schenk. 2018. The ACoLi CoNLL libraries: Beyond tab-separated values. In *In Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan.

Christian Chiarcos and Janine Siewert. 2026. Consolidating syntactically annotated corpora with LLOD technology. An experiment in the Old Saxon Heliand. In *Tenth Workshop on Linked Data in Linguistics (LDL-2026)*, Palma de Mallorca, Spain. Co-located with LREC 2026.

Stefanie Dipper, Karin Donhauser, Thomas Klein, Sonja Linde, Stefan Müller, and Klaus-Peter Wegera. 2013. HiTS: Ein Tagset für historische Sprachstufen des Deutschen. *Journal for Language Technology and Computational Linguistics*, 28(1):85–137.

Stefanie Dipper, Cora Haiber, Anna Maria Schröter, Alexandra Wiemann, and Maike Brinkschulte. 2024. Universal Dependencies: Extensions for modern and historical German. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 17101–17111.

Christian Fäth, Christian Chiarcos, Björn Ebbrecht, and Maxim Ionov. 2020. Fintan - Flexible, integrated transformation and annotation engineering. In *Proceedings of the 12th Language Resources and Evaluation Conference (LREC-2020)*, pages 7212–7221, Marseille, France (online).

Cora Haiber. 2024. A Crosslingual Approach to Dependency Parsing for Middle High German. In *Proceedings of the 20th Conference on Natural Language Processing (KONVENS 2024)*, pages 23–31, Vienna, Austria.

Caitlin Light. 2012. The information structure of subject extraposition in Early New High German. *University of Pennsylvania Working Papers in Linguistics*, 18(1):20.

Sonja Linde. 2009. Aspects of word order and information structure in Old Saxon. In Roland Hinterhölzl and Svetlana Petrova, editors, *Information structure and language change: New approaches to word order variation in Germanic*, pages 367–389. Walter de Gruyter.

Sonja Linde and Roland Mittmann. 2013. Old German Reference Corpus: Digitizing the knowledge of the 19th century. In Paul Bennett, Martin Durrell, Silke Scheible, and Richard J Whitt, editors, *New Methods in Historical Corpora*, pages 235–246. Narr Francke Attempto Verlag, Tübingen.

Rosemarie Lühr. 2025. Reflexivität im Altsächsischen. In Norbert Kössinger, editor, *Altsächsisch: Beiträge zur altniederdeutschen Sprache, Literatur und Kultur*. Walter de Gruyter.

Svetlana Petrova and Michael Solf. 2009. On the methods of information-structural analysis in historical texts: A case study on Old High German. In Roland Hinterhölzl and Svetlana Petrova, editors, *Information structure and language change: New approaches to word order variation in Germanic*, pages 121–203. Walter de Gruyter.

Svetlana Petrova, Michael Solf, Julia Ritz, Christian Chiarcos, and Amir Zeldes. 2009. Building and using a richly annotated interlinear diachronic corpus: The case of Old High German Tatian. In *Traitement Automatique des Langues, Volume 50, Numéro 2: Langues anciennes [Ancient Languages]*, pages 47–71.

Susan Pintzuk and Ann Taylor. 1997. Annotating the Helsinki Corpus: The Brooklyn-Geneva-Amsterdam-Helsinki Parsed Corpus of Old English and the Penn-Helsinki Parsed Corpus of Middle English. In *Tracing the Trail of Time: Proceedings from the Second Diachronic Corpora Workshop, New College, University of Toronto, Toronto, May 1995*, pages 91 – 104. Brill, Leiden, Niederlande.

Eiríkur Rögnvaldsson, Anton Karl Ingason, and Einar Freyr Sigurðsson. 2011. Coping with Variation in the Icelandic Parsed Historical Corpus (IcePaHC). *Language Variation Infrastructure*, 3:97–112.

Alessio Salomoni. 2021. *A UD Literary Treebank for German*. Ph.D. thesis, Università degli studi di Bergamo, Italy.

Beatrice Santorini. 1993. The rate of phrase structure change in the history of Yiddish. *Language Variation & Change*, 5(3).

Christopher D. Sapp, Elliott Evans, Rex Sprouse, and Daniel Dakota. 2024. Introducing a Parsed Corpus of Historical High German. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 9224–9233, Torino, Italia. ELRA and ICCL.

Silke Scheible, Richard J Whitt, Martin Durrell, and Paul Bennett. 2011. A gold standard corpus of Early Modern German. In *Proceedings of the 5th Linguistic Annotation Workshop (LAW V), held in conjunction with ACL-HLT 2011*, pages 124–128, Portland, Oregon, USA.

Mariana Scott. 1966. *The Heliand: Translated from the Old Saxon*. University of North Carolina Press, Chapel Hill.

Edward Henry Sehrt. 1925. *Vollständiges Wörterbuch zum Heliand und zur altsächsischen Genesis*. Vandenhoeck & Ruprecht.

Eduard Sievers. 1878. *Heliand*. Buchhandlung des Waisenhauses, Halle.

Petrova Svetlana, Donhauser Karin, and Linde Sonja. 2009. Heliand (Version 1.0). Humboldt-Universität zu Berlin.

Ann Taylor, Mitchell Marcus, and Beatrice Santorini. 2003. The Penn Treebank: An overview. In Anne Abeillé, editor, *Treebanks: Building and using parsed corpora*, pages 5–22. Kluwer Academic Publishers, Dordrecht/Boston/London.

Robert D Van Valin. 2014. Role and reference grammar. In *The Routledge Handbook of Syntax*, pages 579–603. Routledge.

George Walkden. 2016. The HeliPaD: a parsed corpus of Old Saxon. *International Journal of Corpus Linguistics*, 21(4):559–571.

Lars Erik Zeige, Gohar Schnelle, Martin Klotz, Karin Donhauser, Jost Gippert, and Rosemarie Lühr. 2021. Deutsch Diachron Digital - Referenzkorpus Altdeutsch (Version 1.2). Humboldt-Universität zu Berlin.

A. Appendix: Annotating Universal Dependencies with Spreadsheets

As accompanying material, Fig. 7 illustrates the annotation of UD-style syntax with the help of off-the-shelf spreadsheet software. We used LibreOffice 25.2.6.2 for this purpose, but also conducted successful tests with Microsoft Excel, Gnumeric[11] and Google Spreadsheets.[12] All of these tools support formulas and custom filters (marked by the drop-down menu button in the header row) that we used to perform global replacement and overall sanity checks (checking for typos). How both work together can be easily illustrated by how we implemented the annotation of punctuation characters: We first filtered for punctuation signs in column A (`FORM`), in the first result row (in this case, row 6), we then set column D (`UPOS`) to `PUNCT`, column H (`DEP`) to `punct`, and column G (`HEAD`) to the formula `=A6-1` (i.e., the preceding token = value of `ID` minus 1). This was then copied to all other result rows matched by the filter. As many punctuation signs in our texts do generally not serve a grammatical function, but are used to indicate verse boundaries or line breaks, they can then be filtered out accordingly, so that only lines without punctuation signs remain visible and these do not detract attention. As for the annotation of `UPOS` and `DEP`, this is supported by autocompletion. (As we observed on other corpora, spreadsheet software is less appropriate for free-text annotation as often needed for `FEAT`, `XPOS`, `LEMMA` and `DEPS`.)

	A	B	C	D	E	F	G	H	I	J
1	ID	FORM	LEMMA	UPOS	XPOS		HEAD	DEP		MISC
2	1	Sîðoda	siðon	VERB	VVFIN.IND_PAST_SG_3	_	0	root	_	Gloss=gehen
3	2	im	he	PRON	PRF.MASC_SG_DAT_3	_	1	obl	_	Gloss=er
4	3	thuo	do	ADV	ADV	_	1	advmod	_	Gloss=nun
5	4	te	te	ADP	APPR	_	5	case	_	Gloss=zu
6	5	seliðon	seliða	NOUN	NA.PL_DAT	_	1	obl	_	Gloss=Haus
7	6	,	,	PUNCT	$,	_	5	punct	_	_
8	7	habda	hebbian	AUX	VAFIN.IND_PAST_SG_3	_	10	aux	_	Gloss=haben
9	8	im	he	PRON	PRF.MASC_SG_DAT_3	_	10	obl	_	Gloss=er
10	9	sundea	sundea	NOUN	NA.SG_ACC	_	10	obj	_	Gloss=Übeltat
11	10	giuuaraht	giwirkian	VERB	VVPP	_	1	advcl	_	Gloss=tun
12	11	bittra	bittar	ADJ	ADJN.POS_FEM_SG_ACC_ST	_	10	xpos	_	Gloss=bitter
13	12	an	an	ADP	APPR	_	14	case	_	Gloss=an
14	13	is	he	DET	DPOS.MASC_SG_GEN_3	_	14	det:poss	_	Gloss=er
15	14	bruoðar	broðar	NOUN	NA.SG_DAT	_	11	obl	_	Gloss=Bruder
16	15	;	;	PUNCT	$.	_	14	punct	_	_

Figure 7: Annotating Universal Dependencies with spreadsheet software. First sentence of the Old Saxon Genesis (fragment 2)

Another important feature is the conditional formatting functionality supported by off-the-shelf spreadsheet software, where a colors are applied to columns A (`ID`) and G (`HEAD`), with small values in read, medium values in yellow and large values in green. The color of the `HEAD` column thus provides a visual key for the location of the head word in the current sentence.

[11] https://gnome.pages.gitlab.gnome.org/gnumeric-web/
[12] https://docs.google.com/spreadsheets