LDK 2023

Crowdsourcing OLiA Annotation Models the Indirect Way

Christian Chiarcos
Applied Computational Linguistics
University of Augsburg, Germany
christian.chiarcos@philhist.uni-augsburg.de

Abstract

The paper describes a technology to complement established documentation workflows in two linguistic community projects with the possibility to automatically create OLiA Annotation Models, i.e., formal, ontological representations of their annotation schemas. For this purpose, we provide a domain-specific extractor that consumes MediaWiki wikitext, extracts sections headers and tables and produces an OWL2/DL ontology as a result. This ontology can be further processed with standard technology as established in the context of the Linguistic Linked Open Data (LLOD) community. The main contribution we provide effectively eliminates the entry barrier into LLOD technology and OLiA for two potential user communities, and that this setup can be trivially adopted to any comparable community project – as long as it uses Wiki technology and Wiki lists for documenting tags and abbreviations.

1 Background and Motivation

The Ontologies of Linguistic Annotation (Chiarcos, 2008; Chiarcos and Sukhareva, 2015, OLiA) serve as a central hub for linguistic annotation terminology on the web of data, and they constitute a formative element of the Linguistic Linked Open Data (LLOD) cloud in that they provide machine-readable semantics for linguistic annotations. These ontologies define reference concepts and relations that can be used to annotate linguistic data in a standardized way, making it easier to share and compare data across different languages and domains.

Applications of OLiA include the mapping of tags from one annotation schema to their closest counterparts in another schema (Chiarcos and Ionov, 2021), to perform cross-corpora queries across different corpora (Chiarcos and GÃ-tze, 2007), to aggregate information across heterogeneous tagsets in ensemble combination architectures (Chiarcos, 2010) or in multi-source annotation projection (Sukhareva and Chiarcos, 2016). Being based on RDF technologies, all of this can be achieved on-the-fly by identifying the shortest paths between different OLiA ontologies by means of a W3C-standardized query language (SPARQL). As schemas differ in their granularity, this mapping is not free of information loss, but its dynamic aspects sets OLiA apart from other attempts to establish interoperability between between different annotation schemas such as EAGLES (Calzolari and Monachini, 1996) or the Universal Dependencies (De Marneffe et al., 2021), in that it does not require a transformation of the original annotations, but instead, leaves the original annotations untouched, and only complements them with a more interoperable interpretation.

For more than 100 languages, OLiA covers different aspects of linguistic annotation, including Part of Speech (PoS) annotation, syntax, and inflectional morphologies. Aspects of discourse semantics (discourse structure, discourse relations, information structure, anaphora, coreference, named entities) are subject to a separate discourse extension (Chiarcos, 2014). Despite its potential genefits in interoperability and interpretability, it can be complicated for the developer of a corpus or an NLP tool to produce a certain type of annotations to provide an OLiA Annotation Model, because this requires a set of technical skills that neither most linguists nor most web developers, nor most NLP specialists, possess.

This paper aims to address the challenge to create annotation models. For the integration of a language resource into the OLiA ecosystem, this normally represents the first step to take, but a relatively hard one for, say, a linguist working on an annotated corpus, or a developer not intrinsically familiar with RDF technology. Our proposed solution is to integrate ontology development into established documentation workflows, so that users are creating an ontology along with their regular

work without even noticing it.

2 The Ontologies of Linguistic Annotation

The OLiA ontologies define a set of reference categories for linguistic annotations. On the one hand, this pertains to linguistic concepts as used in tagsets, annotation schemes and lexical resources (OLiA Reference Model),[1] on the other hand, OLiA provides formalizations of entire annotation schemas or (families of) language resources (OLiA Annotation Models).[2]

Anntation Model concepts are linked to OLiA Reference Model concepts by means of `rdfs:subClassOf`/`rdfs:subPropertyOf` relationships, exploiting the full band-width of OWL2/DL semantics (i.e., class intersection $\sqcap$, union $\sqcup$ and complement $\neg$ operators). Every annotation model resides in a separate, stand-alone ontology, and for every annotation model, there is at least one linking model in which the mapping to OLiA Reference Model concepts is provided.[3] This declarative, machine-readable mapping helps to disentangle definition and interpretation, and, moreover, it facilitates debugging, future revisions and portability across different platforms. Also, it is a feature that sets OLiA apart from other, past and present, standardization efforts such as EAGLES (Calzolari and Monachini, 1996), ISOcat (Kemps-Snijders et al., 2008) or the Universal Dependencies (De Marneffe et al., 2021) – all of these employ(ed) opaque scripts to produce standard tags which can only be debugged and consulted *in code* – if publicly available at all.

In a similar way, the OLiA Reference Model is also linked with other, community-maintained reference terminologies such as ISOcat (Kemps-Snijders et al., 2008) or the General Ontology of Linguistic Description (Farrar and Langendoen, 2010), and the OLiA Reference Model partially builds on these, but further domain-, theory- or language-specific reference terminologies are likewise integrated with OLiA (Chiarcos et al., 2020a). This includes, for example, UniMorph (McCarthy et al., 2020, specific to inflection morphology), LexInfo (McCrae et al., 2017, specific to linguistic terminology for lexical resources in OntoLex-Lemon), or the BLL Thesaurus (Chiarcos et al., 2016, linguistic metadata for a linguistic bibliography).

In the context of LLOD, OLiA serves mostly as an additional layer of interoperable annotations over language resources such as corpora (Bosque-Gil et al., 2018), but also, it is a central component of the NLP Interchange Format, and thus, of web services that dynamically cater linguistic annotations (Hellmann et al., 2013). Yet, OLiA provides potential users and contributors with a certain entry bias, as it is based on RDF technologies as its technical backbone. This paper aims to address one of the aspects of the challenge, the creation of annotation models.

We provide three components designed for bootstrapping OLiA Annotation Models from conventional annotation documentation: (1) a configurable tool to convert MediaWiki source files into OWL ontologies, (2) a novel Annotation Model for morphological analyzers from Apertium, and (3) an Annotation Model for linguistic glosses from Wikipedia. Our converter is a relatively small, but generic piece of code. It can be configured for different constellations, and it requires the source data to provide Wiki tables with one row corresponding to one individual in the end. It is optimized for the extraction tasks at hand, but it is sufficiently that, for any data that comes in a similar form, it can be either directly employed or easily adapted.

3 An Annotation Model for Apertium

Apertium[4] is an open-source machine translation (MT) system, developed by a large community of volunteers and enthusiasts. Apertium focuses on symbolic, rule-based approaches on machine translation, which are particularly fruitful for closely related language pairs with insufficient resources to train a neural or statistical MT system on. Indeed, rule-based generation requires textbook expertise and bilingual word lists for its development, but not necessarily parallel corpora.

The Apertium ecosystem comprises

1. a machine translation engine,
2. tools to manage the necessary linguistic data for a given language pair, and

[1] Namespace prefix `olia:`, reference URL `http://purl.org/olia/olia.owl#`.

[2] As an example, the Penn Treebank schema, namespace prefix `penn:`, resides under `http://purl.org/olia/penn.owl#`.

[3] For the Penn Treebank tagset, the linking model resides under `http://purl.org/olia/penn-link.rdf`.

[4] `https://www.apertium.org`

3. language resources (morphological analyzers, dictionaries) for 51 languages and 53 language pairs considered stable (plus 135 languages and 249 language pairs with experimental support and at different degrees of maturity).[5]

3.1 Apertium Morphosyntactic Annotations

Apertium implements symbolic, transfer-based machine translation, where source language input is first morphologically and syntactically analyzed, then, the lemmas are word-wise translated into the target language, where restructuring rules and surface generation takes place. As such, it provides or wraps a large ensemble of morphological generators and analyzers, often based on finite state transducers (FST).

Apertium tags and morphosyntactic features are not standardized across languages, but they share some common conventions.[6] To some extent, these are in a continuous state of flux, as new language pairs are coming in (and bring in new terminology), while the community presses for more consistency across them. These update processes are relatively slow, as new languages are coming in at a moderate rate, so, any annotation model built from this documentation is likely to remain valid for the coming years, but still needs to be regularly updated. As there is no overall versioning applied across all Apertium language pairs, the documentation and any OLiA Annotation Model derived from it reflects the status at a particular state in time, and requires a timestamp as metadata to make this explicit.

Here, we focus on morphological analyzers within Apertium, and, normally, these represent the first component to be provided for any particular language – and, in fact, for some language pairs, machine translation is or can be implemented using only the FST technology that is also underlying the morphological analysis. This is somewhat different from earlier approaches on connecting Apertium with LLOD technology, as this was solely focusing on the dictionaries also contained in Apertium (Gracia et al., 2018; Chiarcos et al., 2020b; Gracia et al., 2020), and the most recent version of this data includes a manually verified mapping from abbreviations/tags to the LexInfo 3.0 ontology,[7] and thus, indirectly, to OLiA. However, this is necessarily incomplete, as the dictionaries account for open-class lexemes and selected parts of speech only, but not for morphological processes, function words and their morphosyntactic features – all of as these are handled via hand-crafted grammar rules in Apertium, but not by the Apertium dictionaries.

As opposed to this, we aim to provide a more exhaustive mapping that also allows the future development of RDF-based web services as wrappers around Apertium *analyzers*, the LLOD publication of Apertium-compliant corpora, or the linking of such corpora with Apertium-based and other OntoLex dictionaries. It is to be noted, however, that we rely exclusively on the available documentation and provide a fully automated conversion only. If there are omissions or errors in the documentation, or if any particular tool does not adhere to the overall recommendations, these aspects will not be covered by our annotation model.

3.2 Conversion to RDF

Apertium symbol definitions are provided in a wiki page[6] with tables for different kinds of annotations, separated by headlines (see Fig. 4 in the appendix). For converting Apertium data, we operate with wiki text (MediaWiki source code). This is because in established Apertium workflows, the list of symbols is designed to be scrapeable, it provides additional information in its comments, and explicit guidelines for systematicising tables, headline formatting and the marking of tags.

We aim for a generic tool, so we do not *depend* on these conventions (also cf. Fig. 4 as an illustration for the degree of variation observed on the page), but we *respect* them. Our conversion operates as follows:

1. We retrieve the original wikitext using the flag `?action=raw` (cf. Fig. 1).

2. We create the class `:Symbol` as a top-level class, using a user-provided base URI as namespace.

3. For every headline under which (directly or indirectly) at least one table is found, we create a class from the label enclosed in `<!- ... ->`, if this is not available, we operate with the section title, instead. The class

[5] https://wiki.apertium.org/wiki/List_of_language_pairs

[6] https://wiki.apertium.org/wiki/List_of_symbols

[7] lexinfo.net/

```
==Part-of-speech Categories== <!-- POS -->

{|class=wikitable
! Symbol             !! Gloss                    !! Notes                                  !! Universal POS
|-
| <code>n</code>     || Noun                     || ''see 'np' for proper noun''                    || NOUN
|-
| <code>vblex</code> || Standard ("lexical") verb || ''see also: vbser, vbhaver, vbmod, vaux, vbdo'' || VERB
|-
```

Figure 1: Apertium list of symbols (wikitext, excerpt).

name is normalized by enforcing CamelCase, removal of whitespaces, and URL encoding. Also, if the class name happens to have been previously created during the conversion, we produce a unique name by attaching a numerical suffix. The original section header is given as an `rdfs:label`.

4. Based on the hierarchy of headlines, every class is assigned a super-class generated from its header, resp., `:Symbol` for top-level section headers.

   Output generated so far from the snippet given above is:

   ```
   :POS rdfs:subClassOf :Symbol;
     rdfs:label
       "Part-of-speech Categories"@en .
   ```

5. For every wikitable, we determine the column labels from its header how, splitting at `!!`. Column labels are normalized by camelCase conversion, lower-casing of the first word and whitespace removal. These will become RDF properties when processing the following rows. If a wikitable does not provide a header row, we re-use the last established header row. If no header has been established before, the table is skipped with a warning.

   For the table in Fig. 1, the normalized column labels are `symbol`, `gloss`, `notes`, and `universalPOS`.

6. For every row within a table, we split its columns at `||` and align them with the column labels provided in the header.

   For the first row in the snippet above, this yields (shown here as a JSON dictionary):

   ```
   {"symbol" : "<code>n</code>",
    "gloss" : "Noun",
    "notes" : "''see 'np' ...",
    "universalPOS" : "NOUN" }
   ```

7. For every row, determine its identifier by following a sequence of user-provided column names (by default `symbol`, `symbols`, `tag`, `xMLTag`, `xMLAttributeValue`, as needed for the Apertium page): for the first of these column labels found in the current table, we retrieve the cell value as label. We remove XML markup from this label, normalize whitespaces and punctuation to _ and apply lowercasing and URI encoding to obtain (the local name for) the URI. If the resulting symbol is not unique, we attach a numerical suffix. The row URI is assigned the class derived from its section header as an `rdf:type`:

   ```
   :n a :POS .
   ```

8. For every column in the current row, we create a triple where the property (derived from the normalized column labels) provides the cell content (stripped of markup and white-space normalized) as a string value:

   ```
   :n :symbol "n" ;
      :gloss "Noun" ;
      :notes "''see 'np' ...'''";
      :universalPOS "NOUN" .
   ```

This conversion is applicable to any wikitext page that provides wiki tables with explicit headers. Section headers are optional. It is required, though, that a user provides the base URI and a (normalized) column label that determines how to identify the columns from which row URIs are to be created.

3.3 Introducing Standard Vocabularies

An additional parameter that a user can provide is a mapping from normalized column labels to RDF vocabularies, provided as a JSON dictionary. The defaults account for converting the Apertium page:

```
{
    ":symbol" : "olias:hasTag",
    ":symbols": "olias:hasTag",
    ":tag" : "olias:hasTag",
```

```
    ":xMLAttributeValue": "olias:hasTag",
    ":xMLTag": "olias:hasTag",
    ":gloss" : "rdfs:label",
    ":notes" : "rdfs:comment",
    ":means" : "rdfs:comment",
    ":description" : "rdfs:comment"
}
```

Properties not listed here are preserved. In the Apertium data, this applies to `:appearsInAttributeNotes`, `:appearsInXMLTagsNotesExamples`, `:universalFeature`, `:universalFeatures`, and `:universalPOS`. With these replacements, we arrive at the following representations of the first row in our data set:

```
:POS rdfs:subClassOf :Symbol;
  rdfs:label
  "Part-of-speech Categories"@en .

:n a :POS .
:n olias:hasTag "n" ;
   rdfs:label "Noun" ;
   rdfs:description
     "''see 'np' for proper noun''";
   :universalPOS "NOUN" .
```

What remains to do to qualify this as an OLiA annotation model is to declare this file an ontology and to provide elementary metadata:

```
<.../apertium.owl> a owl:Ontology ;
  rdfs:comment
    "OLiA Annotation Model for
    Apertium ..." ;
  rdfs:isDefinedBy
    <https://wiki.apertium.org/wiki/
                      List_of_symbols> ;
  owl:versionInfo "2023-03-07 12:06:48" .
```

The object URI of `rdfs:isDefinedBy` is extrapolated from the base URI – unless explicitly specified by the user. As OLiA Annotation Models are traditionally provided as RDF/XML, the resulting Turtle file is converted with off-the-shelf tools. The resulting OWL file can be loaded and processed with off-the-shelf Semantic Web tools, e.g., with the ontology browser Protégé, cf. Fig. 2.

4 Wikipedia Glossing Abbreviations

Wikipedia[8] is the prime example for a collaboratively constructed, community-maintained resource, and it is acknowledged as that since more than two decades. Unsurprisingly, it also found some popularity among people interested in or professionally working with language, and as such, it serves as a knowledge hub for linguistically relevant topics, and often the first place to look for orientation.

[8] https://www.wikipedia.org/

One such application is that Wikipedia seems to be used by students and linguistic practicioners as a central point to collect and to document glosses used as abbreviations in linguistic literature, in particular in the context of interlinear glossed text (IGT, cf. Appendix Fig. 5).[9] IGT is a format consisting of multiple lines where the first line usually represents a source language string, the following lines provide linguistic analyses, e.g., a transliteration, linguistic glosses, morphological segmentation, morpheme glosses, etc. Typically, the last line comprises a translation into the description language.

This formalism is widely used for educational purposes, for language documentation and in linguistic typology, and it has also been converted to a Linked Data representation and produced a native RDF vocabulary specifically for this purpose, Ligt (Chiarcos and Ionov, 2019; Nordhoff, 2020; Ionov, 2021). Ligt, however, only captures the *structure* of IGT formats, for the semantics of the tags used in that context, it relies on OLiA – which provides a small number of IGT-relevant annotation models, e.g., the UniMorph schema (Chiarcos et al., 2020a) and the glossing guidelines of Dipper et al. (2007), which incorporated the Leipzig Glossing Rules (Committee of Editors of Linguistics Journals, 2008/2015) and extended them to syntax and information structure.

A second usage in the context of Wikipedia itself is that it provides templates for producing interlinear glossed text as part of Wikipedia pages, and these abbreviations are recommended for use. At a future point in time, they may actually be automatically linked to the current website if mentioned in the template, but at the moment, the automated linking operates on a shorter, and older excerpt of these abbreviations. Both the Wikipedia templates and their surface rendering are illustrated in the appendix (Fig. 6). As of March 1, 2023, the English Wikipedia contains 7,639 instances of the `interlinear` template on 651 pages,[10] plus an unknown number of applications of derived templates (e.g., `fs_interlinear` or language- or script-specific templates).

The Wikipedia gloss labels are not directly tied to any particular data, but their usage in combina-

[9] https://en.wikipedia.org/wiki/List_of_glossing_abbreviations

[10] https://bambots.brucemyers.com/TemplateParam.php?wiki=enwiki&template=Interlinear

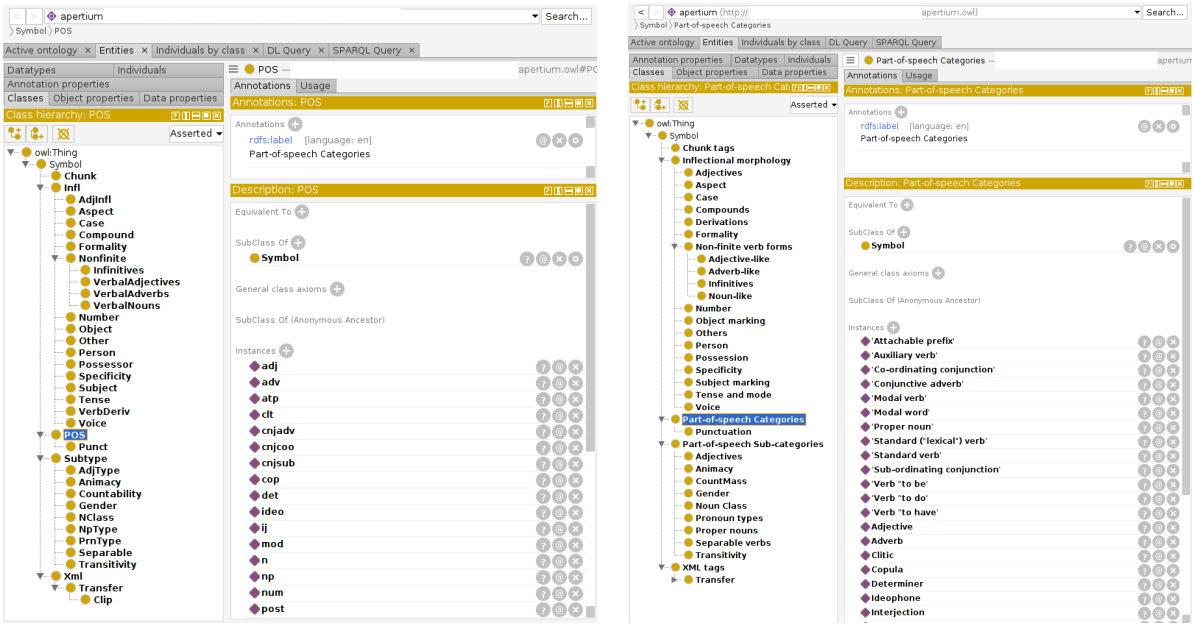

Figure 2: Apertium Annotation Model, visualized with Protégé, configured to display of URIs (left), resp. labels (right).

tion with Wikipedia templates for interlinear glossing is recommended. Furthermore, they are frequently consulted (and extended) by practicioners in the field, in particular by students, so that they attain a certain near-normative function. In the context of efforts to mine scientific papers for machine-readable versions of interlinear glossed text comprised in them (Lewis and Xia, 2010; Nordhoff and Krämer, 2022), it becomes increasingly relevant also to provide machine-readable semantics for the abbreviations, especially if such data is to become the basis for further linguistic research or language technological solutions for low-resource languages as repeatedly proposed over the years (Bender et al., 2014; Zhou et al., 2019).

It is to be noted, however, that glosses and concepts used in the literature reside in an n:m relationship, so that the same abbreviation is used for one purpose by a particular researcher, but for another by another person. As an example, the abbreviation `AC` is defined as “motion across (as opposed to up/down-hill, -river)”, as “animacy classifier”, or as “accusative case”. This is why "conventional glosses" have been singled out, and except for a small number of exceptions, these provide a 1:1 mapping. For the specific case of `AC`, this is not considered a conventional gloss at all (because of its ambiguity), and for the functions mentioned before, only accusative case (with the tag `ACC`) receives that status.

The application of our converter to Wikipedia was straight-forward. The extraction was performed via the Wikipedia API, but the resulting wikitext followed the same conventions (albeit much less constrained than in Apertium). Beyond that, user parameters (base URI, source URI, column labels and their mapping to properties) were adjusted: The row URI is taken from the column with the (normalized) labels `conventionalGloss` (for grammatical abbreviations, punctuations and numbers), and `2LetterGloss` (for kinship terms). As not every row provides a conventional gloss, we also added the column `variants` to the list of URI-defining columns: By ordering preferences, this is used for URI generation only of neither `conventionalGloss` nor `2LetterGloss` are found.

Our conversion of abbreviation variants is lossless in the sense that these are preserved, but only as attribute values, we do not create a distinct tag with its specific `olias:hasTag` for each of them. This was done in order to properly distinguish preferred (readings of) glosses from dispreferred (glosses or readings). The original objective of distinguishing conventional and variant glosses in Wikipedia seems to be that the same gloss was used for different, unrelated meanings (1:m mappings), while at the same time, the same meaning could be expressed by a variety of tags. The current distinc-

tion has been introduced to enable a 1:1 mapping (even though this is not fully achieved).

The resulting ontology is analoguous in structure and vocabulary to the Apertium ontology. A difference is that the concept hierarchy of Wikipedia is much shallower, grouping all morphosyntactic features and categories together under the umbrella of `:GrammaticalAbbreviations`.

5 Automatically Supported Linking

To facilitate the creation of OLiA Linking Models, we provide a command-line tool that takes three main parameters, one reference model (that provides concepts that represent superclasses in the linking), one annotation model (that provides concepts and individuals that are assigned superclasses in the linking) and one linking model (specifying the file into which the resulting mapping is to be written).[11] By default, the linking procedure only creates `rdfs:subClassOf` links between concepts and `rdfs:subPropertyOf` links between properties, but with the flag `-indiv`, it also creates `rdf:type` links between annotation model instances and reference model classes.

The comparison is performed in several steps. If one step produces no linking candidates, it resorts to the next. For a given annotation model concept (or individual), check all reference model concepts in the following way:

1. Convert local names of the URIs from camel case to lower-cased whitespace segmentation. If both strings match, the reference model URI is a linking candidate.

2. Convert local names and RDF/SKOS labels to lower-cased whitespace segmentation. If two strings match, the reference model URI is a linking candidate.

3. Convert local names and RDF/SKOS labels to lower-cased whitespace segmentation and retrieve the set of words used for describing for both URIs. If there is an overlap between both sets of words, the reference model URI is a linking candidate.

The linking tool is interactive, and for every annotation model word for which at least two candidates are found, it presents these to the user as an ordered list. The user can manually select one of the candidates by entering its number, optionally add a comment or state that no linking candidate is applicable. If there is one linking candidate, it is automatically linked (and marked by an `rdfs:comment` in the Linking Model), if there are none, this is marked by an `rdfs:comment`.

This way of linking is restricted, as it is incomplete and heuristic, but it is also *very fast*. In most cases, processing an Annotation Model concept requires 2-3 key strokes: the number of the selected reference model concept (or 0 for no match) and <ENTER>. Yet, manual refinement is highly recommended, and automated comments are generated to guide the way.

We can bootstrap a baseline linking with the OLiA Reference Model from the existing LexInfo linking for Apertium dictionary – but this accounts only for parts of speech, not for grammatical features. In total, 197 Apertium Wiki tags can be linked in this way. Overall, the Apertium ontology comprises 37 classes (headlines) and 301 instances (tags). In addition to this, the automated procedure produced 26 `rdfs:subClassOf` and 22 `rdf:type` links against the OLiA Reference Model, and 15 `rdfs:subClassOf` links against the OLiA Top Model. The limited coverage of linking for instances is partially due to the degree of underspecification they are presented in the table. In parts, however, it is also due to gaps in OLiA. As such, OLiA does currently not support Bantu nominal classes (that alone accounts for 3% of the gaps) and other features specific to certain languages or language families. While language-specific features are generally beyond scope for OLiA, we strongly suggest to extend it with features relevant to entire language families.

6 Manual Linking for Wikipedia Glossing Abbreviations

For Wikipedia glosses, we found that only 82 (16%) were previously covered by the Linking Models for UniMorph (68 in total) or the Dipper et al. (2007) model (42 in total, 28 in both). This linking exploits that the same set of conventional tags were inherited from the literature into these models, but with the automatically supported linking, this number could only be increased by 9 `rdf:type` links. On the one hand, this indicates a certain level of underspecification and idiosyncrasy in both resources, as clearly evident from the brevity of definitions

[11] This tool is not specific to OLiA, so we use lower case spelling. Indeed, any pair of ontologies can be linked in that manner.

in Wikipedia, for example; in parts, this is due to gaps in OLiA (for example, it doesn't currently account for kinship terms as there do not seem to exist any corpora that contain or tools that produce such annotations, kinship terms alone represent 7.5% of conventional Wikipedia glosses). On the other hand, this discrepancy may also indicate a fundamental difference between Wikipedia glossing abbreviations (resp., the scholarly tradition from which these emerge) and OLiA (developed with a focus on linguistically annotated corpora, not text book examples).

In order to explore this further, we resort to manual linking of Wikipedia glossing abbreviations, and we expect that this process may lead to a number of suggestions regarding extensions or restructuring of the OLiA Reference Model as a side-product of the process: The annotation model developed so far represents a solid basis from which a concept hierarchy can be manually crafted in an ontology editor. Unfortunately, the current data is represented in a relatively shallow way, as a limitation for Wikipedia glosses is that (except for the basic distinction between punctuation and numbers, grammatical abbreviations and kinship terms), they are relatively unstructured: Abbreviations are provided as an alphabetically organized list, without being grounded in an overarching taxonomy. The task is thus to pick instances (representing conventional or variant glosses) from an unstructured list and to put them into the OLiA categories they below, ideally using drag-and-drop mechanisms.

Protégé is a seminal OWL editor and it allows both to manually create a concept hierarchy and provides an interface for quickly reclassifying individuals by means of drag and drop.[12] To this end, we created a novel ontology and imported both the generated Wikipedia ontology and the OLiA top-level ontology and manually classified the Wikipedia glosses according to their type. The top-level ontology defines the root concepts of OLiA, i.e., types of units (e.g., `oliat:Word`) and features (e.g., `oliat:MorphosyntacticFeature`, `oliat:GenderFeature`, etc.). Although this coarse-grained classification does not yet establish a proper linking between Wikipedia glosses and the OLiA Reference Model, it allows for a rough classification that can be the basis for subsequent refinements, or serve to evaluate future linking methods. Figure 3 illustrates the manual reclassification procedure.

At the moment, this process of re-classification is still ongoing. Preliminary findings indicate that many Wiktionary glosses are ambiguous or underspecified in that they really act like *abbreviations* for terms, not like *tags* for linguistic annotation. And the same term may occur in different contexts. As such, the conventional tag `REP` stands for 'repetitive', but the meaning is further explained as either 'repetitive aspect' (otherwise referred to as iterative aspect), 'repeated word in repetition' (echo word) or 'repetitive numeral' (numeral formed by reduplication of a basic numeral).[13] A linking to existing OLiA Reference Model concepts is possible, and using OWL2/DL semantics, the ambiguity can be expressed in OLiA:

```
wiki:screp ∈
  olia:IterativeAspect⊔
  olia:EchoWord⊔
  (olia:Reduplication⊓olia:Numeral)
```

Such a complicated linking cannot be established with the automated linking procedure described below, nor with manual the drag-and-drop method, both of which only support direct type assignments. The necessary anonymous classes representing intersections or unions have to be constructed manually, and this is also supported by Protégé. Moreover, this example also illustrates to some extent *why* the linking is failing at times: The OLiA terms 'iterative aspect', 'echo word', and 'reduplication' have no counterpart in the Wikipedia description.

7 Summary and Discussion

This paper described the automated creation of OLiA Annotation Models for different community projects, based on the conversion of wikitext and its layout conventions for section headings and tables. The converter and the associated linking tool are published under open source as part of the OLiA GitHub repository.[14]. Both tools are relatively

[12] This functionality is available from the "Individuals by type" view, not enabled by default, but available via Window|Views|Individual views (Protégé 5.5.0, Desktop).

[13] According to Turner (1967, p.285), the Chontal phrase *núli núli* 'completely' is a repetitive numeral based on *núli* 'one'.

[14] https://github.com/acoli-repo/olia/tree/master/tools. Also, the ontologies are provided there, currently under https://github.com/acoli-repo/olia/tree/master/owl/experimental/meta. Later on, they are expected to migrate to the stable release (https://github.com/acoli-repo/olia/tree/master/owl/stable

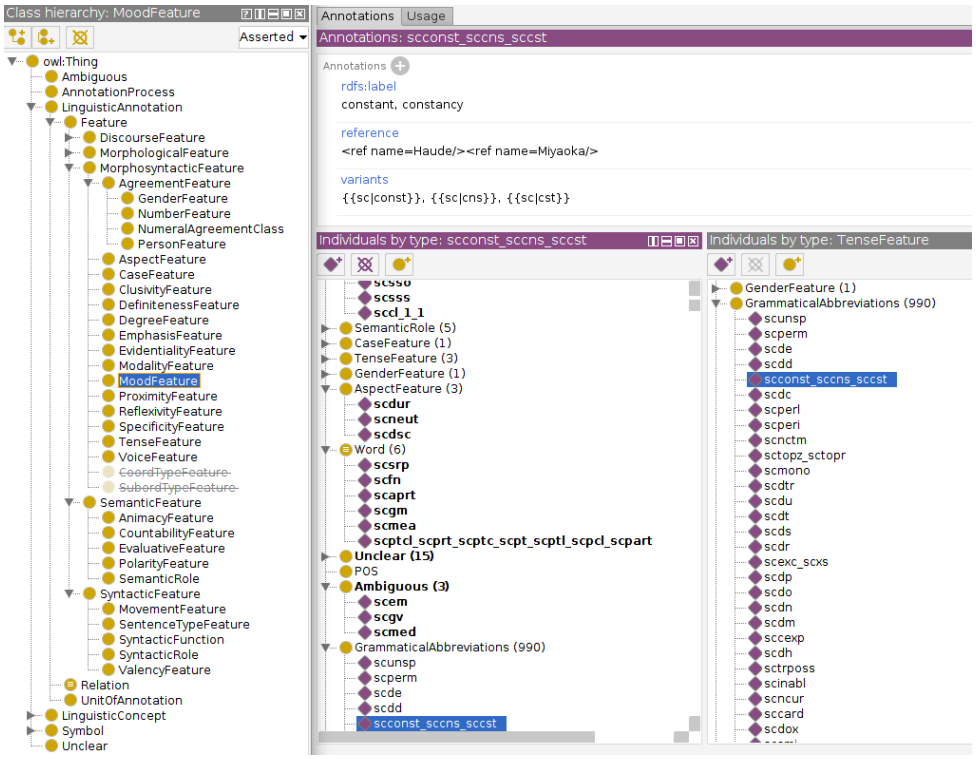

Figure 3: Drag-and-drop classification of Wikipedia abbreviations against OLiA top-level concepts: Between both "Individuals per type" tabs, RDF individuals can be moved by drag and drop. The Annotation view tab above shows the annotations of the current individual. On the left, you see (and can edit) the concept hierarchy.

simple command-line tools with a high level of genericity. The converter is applicable to any MediaWiki content with tables, the linker is applicable to any pair of ontologies.

Conversion from HTML and other web formats is a standard task and has been conducted countless times. For example, DBpedia,[15] DBnary[16] and UniMorph[17] are all based on extraction templates applied over Wikipedia, resp. Wiktionary – although for different types of data. DBpedia and DBnary are also routinely updated in this manner, whereas UniMorph data is frozen and conversion scripts do not seem to be publicly available. Our approach differs in that we do not extract a dataset (ABox), but an ontology (TBox), and that it operates on a much more fine-grained scale. This allows, for example, to expose the result of the build process directly to the user, again.

In particular, the build process can be extended to produce either a graphical representation of the resulting ontology or to apply an interactive browser to the result, so that users can dynamically explore, browse and search their annotation models with off-the-shelf tooling. The integration of existing documentation with such visualizations remains, however, a subject of future efforts, as different possibilities exist for this purpose, and the preferences within the communities need to be taken into consideration. The classical approach to ontology visualization is to convert RDF to the Dot language and to generate a static image with GraphViz.[18] Similarly, SVG and SVG renderers can be used for the same end.[19] The downside of this approach is that the image is to be manually uploaded to or updated in the respective wiki. Alternatively, it is possible to directly link interactive visualization tools such as WebVOWL, along with the URL that contains the ontology to be visual-

[15] https://www.dbpedia.org/
[16] http://kaiko.getalp.org/about-dbnary/
[17] https://unimorph.github.io/

[18] This has been the basis for a number of classical ontology/RDF visualizers integrated in the Protégé ontology viewer. At the present day, the conversion to Dot can also be performed by a web service, e.g., https://www.easyrdf.org/converter?out=dot&raw=1&uri=, followed by the ontology URL. For generating an actual image, different layout schemes can be employed, and we recommend using a local installation of GraphViz, because this is more easily scriptable than online services such as WebGraphViz (http://www.webgraphviz.com/).

[19] As provided, for example, by yWorks: https://www.yworks.com/use-case/visualizing-an-ontology.

ized.[20]

What is interesting about the approach is that it allows to fully automatically create formal ontologies (OLiA Annotation Models) on the basis of established community workflows. We could *build* on established Apertium conventions for their list of symbols, and we could *build* on the current practices in the maintenance and development of the Wikipedia glossing abbreviations. (And, as both as community-maintained, if these conventions would ever be broken by another contributor, and this is noted by our tools, we can fix those issues directly.) At no point did we have to *enforce* new requirements to enable the creation of an OLiA Annotation Model, and neither did we ask Apertium or Wikipedia contributors to operate with a cumbersome tool for handling RDF and linked data. In other words, the entry barrier for OLiA and LLOD technology has been almost eliminated for these groups of users. This also sets it apart from solutions such as VocBench (Stellato et al., 2020) or OpenRefine (Miller and Vielfaure, 2022), which already require their users to have an innate interest in Linked Data or Semantic Web technologies, so that they are actively operating towards this goal with the intent to create a mapping into a machine-readable format. This is not required here, as, instead, the converter is already provided. Moreover, we are concerned with crowd-sourced, community-maintained data, which has a certain quality of being in a continuous update and revision process. So, extraction needs to be repeated relatively frequently – but OpenRefine and VocBench are not designed for repeated conversion, as these are highly interactive tools.

The creation of Linking Models, then, requires a higher level of technical expertise, of course, but this does not have to be provided by an Apertium or Wikipedia contributor, instead, it can come from the LLOD community. And if more technically oriented community members see scientific or technological value in that kind of data *for their own purposes*, this is likely to happen.

It should be noted that the approach to create ontologies as a side-product of established community conventions for maintaining and creating their documentation, is not the first of its kind either. We conducted an earlier, unpublished experiment that infused RDFa attributes into Jekyll templates, so that HTML pages generated from Markdown (as used by the Universal Dependency community to document their annotation schemas) would already contain a machine-readable representation of these schemas. The technology worked very well, and a prototype over an older version of UD guidelines with RDFa markup is still online,[21] and using an RDFa reader on the published HTML pages, a full-fledged ontology could be derived on the fly and queried with SPARQL. From the perspective of a UD contributor, nothing changed, and the process was taking advantage of established conventions originally intended to streamline the layout, especially the usage of explicit variables for certain aspects, and the section structure of the Markdown document. A downside here was that the build process was relatively unstable, and it turned out to take too long for efficiently debugging and maintaining this setup (several minutes, but sometimes more), so that eventually, this experimental prototype was discontinued, and with a change of layout and Markdown conventions with the transition from version 1.0 to 2.0 of the Universal Dependencies, they have not been updated.

With our converter, we do not rely on such a complicated setup. Instead, we provide a simple script for building Annotation Models, and using a cron job, they can be repeatedly called to provide up-to-date RDF data for Annotation Models and visualizations. If deployed on a web server, these can be produced by a third party, independently from the infrastructure of the particular community involved.

Our tools and annotations have been integrated into the OLiA GitHub repository,[22] so they will remain accessible to the community as long as OLiA remains a relevant resource. Moreover, they will be subject to any long-term sustainability solution developed for OLiA in the future.

[20] At the time of writing, the recommended URL for that purpose would be `http://vowl.visualdataweb.org/webvowl-old/webvowl-old.html#iri=`, followed by the ontology URL. However, as the `-old` link indicates, the system is currently in transition to a novel backend, so that link might change.

[21] See `http://fginter.github.io/docs/`. Note the small RDF logos that trigger the RDFa parsing process. However, these URLs contain a GET request at a public web service for RDFa parsing, after more than a decade of successful operation, was shut down mid-last year, so that these links yield a status page, not RDF data in Turtle, anymore. Alternative web services are available, but the links in this prototype have not been updated, yet.

[22] `https://github.com/acoli-repo/olia/`

Acknowledgements

We would like to thank three anonymous reviewers for comments and feedback, which have been integrated into this paper, the authors of the Wikipedia glossing page and the Apertium documentation, upon whose work we build here, and the developers of the earlier Apertium-Lexinfo mapping, most notably Julia Bosque-Gil and Max Ionov.

References

Emily M. Bender, Joshua Crowgey, Michael Wayne Goodman, and Fei Xia. 2014. Learning grammar specifications from IGT: A case study of chintang. In *Proceedings of the 2014 Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pages 43–53, Baltimore, Maryland, USA. Association for Computational Linguistics.

Julia Bosque-Gil, Jorge Gracia, Elena Montiel-Ponsoda, and Asunción Gómez-Pérez. 2018. Models to represent linguistic linked data. *Natural Language Engineering*, 24(6):811–859.

Nicoletta Calzolari and Monica Monachini. 1996. EAGLES Proposal for Morphosyntactic Standards: in view of a ready-to-use package. In G. Perissinotto, editor, *Research in Humanities Computing*, volume 5, pages 48–64. Oxford University Press, Oxford, UK.

Christian Chiarcos. 2008. An ontology of linguistic annotations. *LDV Forum*, 23(1):1–16.

Christian Chiarcos. 2010. Towards robust multi-tool tagging. An OWL/DL-based approach. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 659–670.

Christian Chiarcos. 2014. Towards interoperable discourse annotation. discourse features in the ontologies of linguistic annotation. In *Proceedings of the Eigth International Conference on Language Resources and Evaluation (LREC-2014)*, pages 4569–4577.

Christian Chiarcos, Christian Fäth, and Frank Abromeit. 2020a. Annotation interoperability for the post-ISOCat era. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5668–5677.

Christian Chiarcos, Christian Fäth, and Maxim Ionov. 2020b. The acoli dictionary graph. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 3281–3290.

Christian Chiarcos, Christian Fäth, and Maria Sukhareva. 2016. Developing and using the ontologies of linguistic annotation (2006-2016). In *Proceedings of the 5th Workshop on Linked Data in Linguistics: Managing, Building and Using Linked Language Resources (LDL-2016)*, pages 63–72, Portorož, Slovenia.

Christian Chiarcos and Michael GÃ-tze. 2007. A linguistic database with ontology-sensitive corpus querying. In *Datenstrukturen für linguistische Ressourcen und ihre Anwendungen. Frühjahrstagung der Gesellschaft für Linguistische Datenverarbeitung (GLDV 2007)*, Tübingen, Germany.

Christian Chiarcos and Maxim Ionov. 2019. Ligt: An llod-native vocabulary for representing interlinear glossed text as rdf. In *2nd Conference on Language, Data and Knowledge (LDK 2019)*. Schloss Dagstuhl-Leibniz-Zentrum Für Informatik.

Christian Chiarcos and Maxim Ionov. 2021. Linking discourse marker inventories. In *3rd Conference on Language, Data and Knowledge (LDK 2021)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik.

Christian Chiarcos and Maria Sukhareva. 2015. OLiA - Ontologies of Linguistic Annotation. *Semantic Web Journal*, 518:379–386.

Committee of Editors of Linguistics Journals. 2008/2015. Leipzig glossing rules, conventions for interlinear morpheme-by-morpheme glosses. Technical report, University of Leipzig, Germany.

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

Stefanie Dipper, Michael Götze, and Stavros Skopeteas. 2007. Information structure in cross-linguistic corpora: Annotation guidelines for phonology, morphology, syntax, semantics, and information structure. *Interdisciplinary Studies on Information Structure (ISIS), Working papers of the SFB 632*, 7.

Scott Farrar and D Terence Langendoen. 2010. An owl-dl implementation of gold. *Linguistic Modeling of Information and Markup Languages*, pages 45–66.

Jorge Gracia, Christian Fäth, Matthias Hartung, Max Ionov, Julia Bosque-Gil, Susana Veríssimo, Christian Chiarcos, and Matthias Orlikowski. 2020. Leveraging linguistic linked data for cross-lingual model transfer in the pharmaceutical domain. In *The Semantic Web–ISWC 2020: 19th International Semantic Web Conference, Athens, Greece, November 2–6, 2020, Proceedings, Part II 19*, pages 499–514. Springer.

Jorge Gracia, Marta Villegas, Asuncion Gomez-Perez, and Nuria Bel. 2018. The Apertium bilingual dictionaries on the web of data. *Semantic Web*, 9(2):231–240.

Sebastian Hellmann, Jens Lehmann, Sören Auer, and Martin Brümmer. 2013. Integrating nlp using linked data. In *12th International Semantic Web Conference, 21-25 October 2013, Sydney, Australia*.

Maxim Ionov. 2021. Apics-ligt: Towards semantic enrichment of interlinear glossed text. In *3rd Conference on Language, Data and Knowledge (LDK 2021)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik.

Marc Kemps-Snijders, Menzo Windhouwer, Peter Wittenburg, and Sue Ellen Wright. 2008. ISOcat: Corralling data categories in the wild. In *Proceedings of the 2008 International Conference on Language Resource and Evaluation (LREC)*.

William D Lewis and Fei Xia. 2010. Developing odin: A multilingual repository of annotated language data for hundreds of the world's languages. *Literary and Linguistic Computing*, 25(3):303–319.

Arya D McCarthy, Christo Kirov, Matteo Grella, Amrit Nidhi, Patrick Xia, Kyle Gorman, Ekaterina Vylomova, Sabrina J Mielke, Garrett Nicolai, Miikka Silfverberg, et al. 2020. Unimorph 3.0: Universal morphology. In *Proceedings of The 12th language resources and evaluation conference*, pages 3922–3931. European Language Resources Association.

John P McCrae, Julia Bosque-Gil, Jorge Gracia, Paul Buitelaar, and Philipp Cimiano. 2017. The ontolex-lemon model: development and applications. In *Proceedings of eLex 2017 conference*, pages 19–21.

Meg Miller and Natalie Vielfaure. 2022. Openrefine: An approachable open tool to clean research data. *Bulletin-Association of Canadian Map Libraries and Archives (ACMLA)*, 170.

Sebastian Nordhoff. 2020. Modelling and annotating interlinear glossed text from 280 different endangered languages as linked data with ligt. In *Proceedings of the 14th Linguistic Annotation Workshop*, pages 93–104.

Sebastian Nordhoff and Thomas Krämer. 2022. Imtvault: Extracting and enriching low-resource language interlinear glossed text from grammatical descriptions and typological survey articles. In *Proceedings of the 8th Workshop on Linked Data in Linguistics within the 13th Language Resources and Evaluation Conference*, pages 17–25.

Armando Stellato, Manuel Fiorelli, Andrea Turbati, Tiziano Lorenzetti, Willem Van Gemert, Denis Dechandon, Christine Laaboudi-Spoiden, Anikó Gerencsér, Anne Waniart, Eugeniu Costetchi, et al. 2020. Vocbench 3: A collaborative semantic web editor for ontologies, thesauri and lexicons. *Semantic Web*, 11(5):855–881.

Maria Sukhareva and Christian Chiarcos. 2016. Combining ontologies and neural networks for analyzing historical language varieties. A case study in Middle Low German. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC-2016)*, pages 1471–1480.

Paul R Turner. 1967. Highland chontal phrase syntagmemes. *International Journal of American Linguistics*, 33(4):282–286.

Zhong Zhou, Lori S. Levin, David R. Mortensen, and Alexander H. Waibel. 2019. Using interlinear glosses as pivot in low-resource multilingual machine translation. *arXiv: Computation and Language*.

Appendix: Illustrative Sample Data

Others [edit]

Symbol	Gloss	Notes
abbr	Abbreviation (e.g. *etc.*, *Mr.*)	Acronyms are also included (see acr)
date	Dates, years...	
email	Electronic Mail	Shorten form of Electronic Mail
file	Filenames	
mon	Money	
percent	Percentage	e.g. 25%, 0.9%
time	Time	
url	Web address	
web	Links and Emails	
year	Years	
maj	Large script in which every letter is the same height	
min	small script in which every letter is the same height	

Compounds [edit]

Symbol	Gloss	Notes	Universal feature
cmp	Compound Noun		

Chunk tags [edit]

Tag	Description
<SN>	Noun phrase / noun group (*sintagma nominal*)
<SA>	Adjective phrase / adjective group
<SV>	Verb phrase / verb group (*sintagma verbal*)

XML tags [edit]

Note: All XML tags are explained in depth in the PDF documentation, see also the dix.dtd and dix.rng files in the GitHub repository.

XML tag	Means	Appears in XML tags / notes / examples
<dictionary>	Mono- or bilingual dictionary	Toplevel tag for all dictionaries
<alphabet>	Set of characters in the language	In <dictionary>
<sdefs>	Symbol definitions	In <dictionary>

Figure 4: Apertium list of symbols (excerpt) .

VOL		volitive mood; volitional (cf. AVOL avolitional)	[112][117]
	VP	verbal particle	[19]
V_r	VR, v.r.	verb, reflexive (e.g. as a covert category)	[129]
	VSM	verb-stem marker	[67][23]
V_t	VT, v.t.	verb, transitive (e.g. as a covert category)	[129][15]
	WH.EX	exclamatory *wh-* clause ('what a ...!')	[*citation needed*]
	WH	interrogative pronoun (*wh*-word), *wh-* agreement	[56][16]
WHQ	WH.Q	*wh-* question	[16][131][20]
WIT		witnessed evidential (cf. EXP)	[38][16]
	WP, WPST	witnessed past	[80][99]
X	?	(unidentified morpheme)	[32][31]
	YNQ, PQ, P.INT, PI	yes–no question, polar question/interrogative (e.g. PC vs CQ)	[131][16][19][1]
	-Z	-(al)izer (*e.g.* ADJZ adjectivizer, NZ nominalizer, TRZ transitivizer, VBZ verbalizer)	
ZO		zoic gender (animals)	[132]

Kinship [edit]

It is common to abbreviate grammatical morphemes but to translate lexical morphemes. However, kin relations commonly have no precise translation, and in such cases they are often glossed with anthropological abbreviations. Most of these are transparently derived from English; an exception is 'Z' for 'sister'. (In anthropological texts written in other languages, abbreviations from that language will typically be used, though sometimes the single-letter abbreviations of the basic terms listed below are seen.) A set of basic abbreviations is provided for nuclear kin terms (father, mother, brother, sister, husband, wife, son, daughter); additional terms may be used by some authors, but because the concept of e.g. 'aunt' or 'cousin' may be overly general or may differ between communities, sequences of basic terms are often used for greater precision. There are two competing sets of conventions, of one-letter and two-letter abbreviations:[133][134][47][24]

1-Letter Gloss	2-Letter Gloss	Meaning	Equivalent sequence of nuclear relations
A	**Au**	aunt	= MZ or FZ / MoSi or FaSi
B	**Br**	brother	[basic term]
C	**Ch**	child	= S or D / So or Da
	Cu	cousin	= MZD, MZS, MBD, MBS, FZD, FZS, FBD, FBS = MoSiDa, MoSiSo, MoBrDa, MoBrSo, FaSiDa, FaSiSo, FaBrDa, FaBrSo
D	**Da**	daughter	[basic term]
e, E	**o, el**	elder/older	(e.g. eB, eZ)[54]

Figure 5: Wikipedia list of glossing abbreviations (excerpt) .

```
{{interlinear|lang=jig|spacing = 3| box = yes
|Nyama-baji imimikin-bili-rni-rni ardalakbi-wurru-ju
|DEM-PL old.woman-ANIM.DU-F-ERG hot-3PL-do
|'The two old women feel hot.'}}
```

Nyama-baji	*imimikin-bili-rni-rni*	*ardalakbi-wurru-ju*
DEM-PL	old.woman-ANIM.DU-F-ERG	hot-3PL-do

'The two old women feel hot.'

Figure 6: Wikipedia template `Interlinear` and its rendering, example from https://en.wikipedia.org/wiki/Template:Interlinear.