Enriching ODIN

Fei Xia† William D. Lewis‡ Michael Wayne Goodman†

Joshua Crowgey† and Emily M. Bender†

†University of Washington, PO Box 352425, Seattle, WA 98195, USA
‡Microsoft Research, One Microsoft Way, Redmond, WA 98052, USA
{fxia, goodmami, jcrowgey, ebender}@uw.edu, wilewis@microsoft.com

Abstract

In this paper, we describe the expansion of the ODIN resource, a database containing many thousands of instances of Interlinear Glossed Text (IGT) for over a thousand languages. A database containing a large number of instances of IGT, which are effectively richly annotated and heuristically aligned bitexts, provides a unique resource for bootstrapping NLP tools for resource-poor languages. To make the data in ODIN more readily consumable by tool developers and NLP researchers, we propose a new XML format for IGT, called XigtẆe call the updated release ODIN-II.

Keywords: Resource-poor languages, multilingual resources, bootstrapping NLP tools, Interlinear Text

1. Introduction

In the past two decades, there has been substantial progress in natural language processing (NLP), largely due to the large number of annotated corpora that have been created, such as treebanks and PropBanks (Kingsbury et al., 2002) , which have been used to train and test statistical NLP systems. However, creating these resources is expensive and labor intensive; as a result, the vast majority of the world's languages lack such resources, and consequently, high-performance NLP tools.

To address this deficiency, recent studies have proposed taking advantage of resources containing data in both resource-rich and resource-poor languages. One such method involves processing one side of bitexts (the resource-rich language) and projecting the enriched information to the other side of the bitext (the resource-poor language) via word alignments (Yarowsky and Ngai, 2001; Hwa et al., 2005). A challenge to this approach is that there might not be a large enough supply of bitexts for both languages to allow for the training of a high-quality statistical word aligner. In our previous work (Georgi et al., 2013), we have shown that one can bootstrap NLP tools for resource-poor languages by taking advantage of Interlinear Glossed Text (IGT).

IGT is a common format that linguists use to present language data relevant to a particular analysis. It is most commonly presented in a three-line form, a sample of which is shown in (1). The first line, the *language line*, gives data for the language in question, and is either phonetically encoded or transcribed in the language's native orthography. The second line, the *gloss line*, contains a morpheme-by-morpheme or word-by-word gloss for the data on the language line. The third line, the *translation line*, contains a translation of the first line, often in a resource-rich language such as English. There could be additional lines showing other information such as a citation and a language name and/or code. In Ex (1), *(Bailyn, 2001)* is the author of the IGT instance, and *cym* is the language code for Welsh.

(1) Rhoddodd yr athro lyfr i'r bachgen ddoe
gave-3sg the teacher book to-the boy yesterday
"The teacher gave a book to the boy yesterday" (Bailyn, 2001) [cym]

In this paper, we describe a resource that will enable the community to more readily experiment with the IGT data type. To create this resource, which we call ODIN-II, we begin with the Online Database of Interlinear Text (ODIN, Lewis and Xia (2010)), a collection of IGT data from linguistic documents on the Web, described in Section 2. Next, we describe the process of enriching the IGT data (which includes cleaning and normalizing IGT instances, and projecting syntactic information from the translation line to the language line), and demonstrate that the enriched IGT data can be used to bootstrap NLP tools such as parsers. Third, we review Xigt, a new XML format for representing the enriched IGT data. The ODIN data (including the original and enriched IGT) is available to the public.[1]

2. Building ODIN

The mechanical processes for constructing a database of interlinear text such as ODIN were first described in (Lewis, 2003). Lewis (2003) observed that IGT is as a rich source of linguistic markup, and a collection of harvested IGT could be treated as a gateway to the construction of a resource representing the conceptual space of the field of linguistics, such as an ontology of linguistic concepts (Lewis et al., 2001; Farrar and Langendoen, 2003). It quickly became clear that the database of IGT itself was directly of use to the field of linguistics, in addition to secondary resources like the ontologies derived from it (Xia and Lewis, 2008; Lewis and Xia, 2008). The ODIN database is created in two stages: automatic construction, followed by manual correction.

[1] http://faculty.washington.edu/fxia/odin/

2.1. Automatic construction

The automatic construction stage has three steps. First, we crawl the Web for linguistic documents and collect those documents that most likely contain IGT. It is done by throwing queries against an existing search engine, extracting the relevant URLs from the results of the queries, crawling the pages returned (i.e., search returned pages for relevant URLs), and downloading the pages and documents that contain IGT. We found that one of the most successful queries was one that used strings contained within IGT itself. Since the markup vocabulary for IGT often contains "grams" (e.g., NOM, ACC, ERG, etc.), the most successful strategy involves using the highest frequency grams as search terms.[2]

Second, IGT within those documents is detected and extracted. We treat IGT detection as a sequence labeling problem, and apply machine learning methods to the task: first, we train a learner and use it to tag each line in a document with a BIO tag, and then we convert the best tag sequence into a span sequence. The feature set includes word ngrams, shapes of a line (e.g., whether the line starts with an example number), and other cues for the presence of IGT. When trained on 41 documents which contained 1573 IGT instances and tested on 10 documents, the f-scores for exact and partial span match on the test data are 81.65% and 96.75% respectively (Xia and Lewis, 2008).

Third, each extracted IGT instance is assigned a language name and a language code. While existing methods for language ID perform very well in a typical language ID setting (e.g., only a dozen languages with a large amount of training data), they all require training data in these languages in order to build a language model or a character ngram list. They do not work well in this setting because the number of languages represented by IGT on the Web numbers in the thousands and for many of those languages we do not have any training data. To address this challenge, we proposed to treat language identification as a coreference resolution task, where an IGT instance is linked to a language name appearing in the same document. When trained on 1372 IGT instances from 125 languages and tested on 1516 instances (only 55.45% of them belong to a language that appears in the training set), the accuracy is 83.08%, much higher than 55.45% (the upper bound of any language ID algorithm that relies on having training data for the languages that the test data belong to) (Xia et al., 2009).

We ran the IGT detection and language ID systems on three thousand IGT-bearing documents crawled from the Web and the extracted IGTs were stored in the ODIN database. Table 1 shows shows the language distribution of the IGT instances in the database according to the output of the language ID system. For instance, the third row says that 122 languages each have 100 to 999 IGT instances, and the 40,260 instances in this bin account for 21.27% of all IGT in the ODIN database.

[2]The first attested use of the word *gram* in the context of IGT that we are aware of is in **??**.

Range of IGT instances	# of languages	# of IGT instances	% of IGT instances
$>$ 10000	3	36,691	19.39
1000-9999	37	97,158	51.34
100-999	122	40,260	21.27
10-99	326	12,822	6.78
1-9	838	2,313	1.22
total	1326	189,244	100

Table 1: The language distribution of IGT instances in ODIN after stage 1 (automatic construction). The IGT instances are extracted from 2868 documents, and the language IDs are those assigned by the language ID system.

2.2. Manual correction

To ensure the high quality of the ODIN database, we manually correct the output of Steps 2 and 3 in the automatic construction stage. This is done in three steps.

First, the annotators correct the boundary of IGT instances found by the IGT detection module. In addition, they label each line in IGT with a *xx-yy* tag. The *xx* part is the main tag, indicating whether the line is a language line (*L*), a gloss line (*G*), a translation line (*T*), a blank line (*B*), or a line with other information such as the citation or linguistic construction name (*M*). The *yy* part is called the secondary tag; it provides additional properties of the line; for instance, *CR* means that the current line has been corrupted when the document retrieved by the crawler is converted from a *pdf* file to a text file by an off-the-shelf *pdf-to-text* converter. The tags are used for automatic enrichment of IGT instances, as discussed in the next section.

Our automatic language ID module labels each IGT instance with a language name and an ISO 639-3 language code. The annotators correct the language name in Step 2 and the language code in Step 3. The reason that we separate the two steps is that the mapping between language names and language codes is many-to-many, and Step 3 must be done by a linguist who can choose the correct language code for an ambiguous language name. See (Xia et al., 2010) for more information about these two steps.

We have finished manual correction of more than 83% of the IGT instances in ODIN, and the language distribution of this subset of data is in Table 2. Notice that the number of languages in this subset is higher than the number of languages in Table 1. That is because our automatic language ID module maps an ambiguous language name to the most common language code associated with the name. This process is error-prone; manual correction reveals that the ODIN data actually covers more languages than indicated by the automatic construction stage.

The canonical form of an IGT instance includes a language line (*L*), a gloss line (*G*), and a translation line (*T*), but linguists often do not follow this canonical form, especially if they use multiple IGT instances in a group. For instance, an IGT instance might include only the *L* line, because the line has slightly different word order as the language line in a previous instance, and the readers could infer what the gloss or translation lines should be from the previous IGT. Table 3 gives a breakdown of the number of IGTs by the

Range of IGT instances	# of languages	# of IGT instances	% of IGT instances
> 10000	1	10,814	6.88
1000-9999	31	81,218	51.69
100-999	139	46,420	29.55
10-99	460	15,650	9.96
1-9	862	3,012	1.92
Total	1,493	157,114	100

Table 2: The language distribution of IGT instances in ODIN after stage 2 (manual correction). The IGT instances are extracted from 2025 documents, and the language ids are according to human correction of the language ID output.

presence of *L*, *G*, *T* lines. Some lines are marked as *L-G* as *L* and *G* are displayed side-by-side. IGT like those fall into the category *Other types*. The table shows that only 74.92% of IGT are in the canonical form. For the rest, additional work is required to recover the *"missing"* lines from the context.

Lines in an IGT	# of IGT instances	% of IGT instances
L only	749	0.48
G only	611	0.39
T only	155	0.10
L and G	19,750	12.57
L and T	7,912	5.04
G and T	469	0.30
L, G, and T	117,717	74.92
Other types	9,751	6.21
Total	157,114	100

Table 3: The IGT type distribution in ODIN after stage 2. The IGT data are the same as in Table 2.

3. Enriching IGT data

The unique structure of IGT makes it an extremely rich source of information for resource-poor languages: Implicit in an IGT instance is not only a short bitext between that language and a language of wider communication (almost universally English, but instances of Spanish and German have been discovered as well), but also information encoded in the so-called gloss line about the grammatical morphemes in the source language and word by word translations to lemmas of the translation language. Thus even small quantities of IGT could be used to bootstrap tools for resource-poor languages through structural projection (Yarowsky and Ngai, 2001; Xia and Lewis, 2007). However, bootstrapping tools often require the original IGT to be enriched, as explained in this section.

3.1. Cleaning and normalizing IGT instances

The first step of enrichment is to clean the original IGT and separate out different fields in an IGT. Cleaning steps are applied both to remove extraneous data and to try to recover from PDF extraction or OCR errors. These cleaning steps remove things like example numbers, leading and trailing quotation marks, parentheses, or whitespaces, and descriptors like "intended:" or "lit:". Cleaning also merges lines wrongly split by the *pdf-to-text* converter. We also want to normalize various formatting deviations, such as IGTs that were wrapped at a column border, those with language lines and translations on the same line, those with multiple translations, and so on.

In addition to the language data (*L*), gloss (*G*), and translation (*T*) parts of IGT, an IGT often contains other information such as a language name (*-LN*), citation (*-AC*), construction names (*-CN*), and so on. An example is shown in (2), in which the first line contains the language name and citation,[3] the third line includes coindexes *i* and *i/j*, and the last two lines show two possible translations of the sentence. Here, the language line is displayed as two lines due to errors made by the off-the-shelf converter that converted the crawled pdf documents into text.

```
(2) Haitian CF (Lefebvre 1998:165)
                    ak
    Jani   pale          lii/j
    John   speak  with    he
    (a) 'John speaks with him' (b) 'John
        speaks with himself'
```

The goal of this step is to separate out different fields in an IGT, fix display errors caused by the pdf-to-text converter, and store the results in a uniform data structure such as the one shown in Ex (3), itself a correction of Ex (2). The task is not trivial; for instance, the coindex *i* in *Jani* and *lii/j* on the third line of Ex (2) could easily be mistaken as being part of the word. The *xx-yy* tags added by the manual correction step provide some help; for instance, the second and third lines in (2) will be labeled as *L-CR*, and the *-CR* tag signals the corruption in the language lines.

```
(3) Language: Haitian CF
    Citation: (Lefebvre 1998:165)
    L:        Jan   pale   ak     li
    Coindx:   (Jan, i), (li, i/j)
    G:        John  speak  with   he
    T1:       John speaks with   him
    T2:       John speaks with himself
```

3.2. Adding word alignment and syntactic structure

After we have cleaned IGT, the next step is to add word alignment and syntactic structure. In our previous work (Xia and Lewis, 2007), we proposed an algorithm to leverage the structure of IGT to enrich it further. We do so in three steps: (1) parse the English translation with an English parser, (2) align the language line and the English translation via the gloss line, and (3) project syntactic structures from English onto the language line. Given the IGT in Ex (1), the algorithm will produce the word alignment in Fig 1, the dependency structures in Fig 2, and the phrase structures in Fig 3.

[3] *CF* here stands for French-lexified creole.

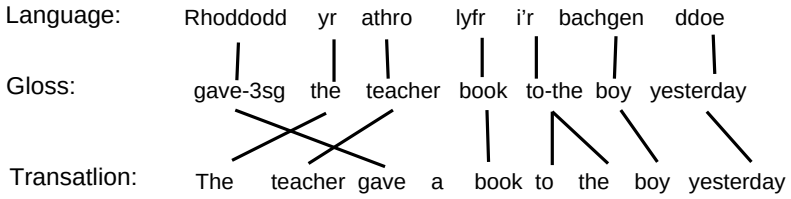

Figure 1: Aligning the language line and the English translation with the help of the gloss line

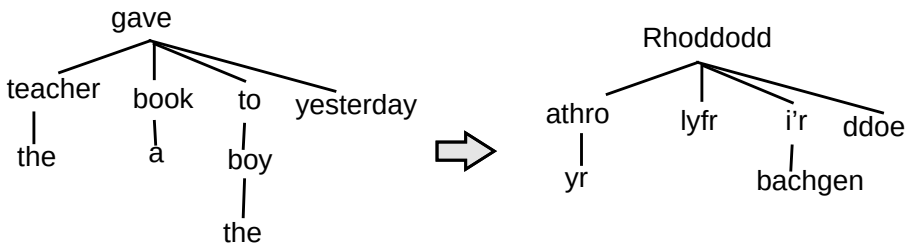

Figure 2: Projecting dependency structure from the translation line to the language line

3.3. Using enriched IGT

The syntactic structures produced by syntactic projection often are not perfect because the structures in the translation line and the language line may differ significantly.[4] Nevertheless, the enriched IGT can help linguistic study and NLP in many ways. For instance, it allows linguists to search ODIN for linguistic constructions (e.g., passives, conditionals, double object structures) contained in the IGT instances. Enriched IGT also allows discovery of computationally relevant typological features, such as word or constituent order, or the presence or absence of particular grammatical features, and does so with high accuracy (Lewis and Xia, 2008; Bender et al., 2013). Enriched IGT can also be used to bootstrap NLP tools; for instance, adding features extracted from projected syntactic structures to a statistical parser provided a significant boost to parser performance (Georgi et al., 2013).

4. Representing enriched IGT in Xigt

A preliminary version of the ODIN database has been released to the public;[5] the subset contains 130,351 instances of IGT across 1,274 languages. The release includes only the original IGT data in the plain text format, reflecting the information as extracted from the source documents and the language that the IGT belongs to. An example is given in Figure 4. The first line shows the document id, the position of the IGT in the document (in lines 959-961), and the type of each line in the IGT. The second line gives the language name and language code. The next three lines are the original text from the document.

While the plain text format may be sufficient for representing original IGT for reading by humans, it does not make explicit the relationships among various tiers and tokens, nor does it scale well for representing the enriched IGT data we would like to encode—namely, cleaned IGT, bilingual word alignments, syntactic structures for translation and/or language lines, and any additional information that could be relevant. For that purpose, we propose the use of a new data model and XML format, Xigt (Goodman et al., forthcoming), which is designed specifically for the bulk processing of IGT. Xigt makes explicit the function of each line of each IGT instance and can encode alignments between them through ID-reference annotations. Making these alignments, implicit in the formatting of IGT in linguistics papers, explicit is the first step in enriching IGT for further processing.

Xigt was designed to be extensible so that further enrichment, such as that described above, can be encoded without having to modify the structure or content of the original data. For ODIN data, this includes the following (Figure 5 shows an enriched instance of IGT for the original example shown in Figure 4):

Provenance: Metadata elements at the corpus, IGT, and tier levels allow for the tracking of provenance of the documents from which the ODIN data was initially extracted. Attributes on these elements can be used for finer specificity, such as for page numbers. Xigt has some basic metadata types available (see *metadata* at the beginning of Figure 5), but we also plan to provide extensions for other metadata formats, such as OLAC.[6]

Stand-off annotation: We wish to preserve the original ODIN data as extracted from the source PDF documents and encode all further information as stand-off annotation against that original. Thus, the first Xigt extension for ODIN is a new tier type, `odin-raw`, which encapsulates text lines of ODIN into an XML element and gives them identifiers for later reference. We encode the data in Unicode by default, so nearly any Unicode character is acceptable.[7]

[4] Dorr (1994) provides a detailed analysis of divergence in languages.

[5] http://odin.linguistlist.org

[6] http://www.language-archives.org/OLAC/metadata.html

[7] Unacceptable characters are those illegal in XML documents, such as the form feed character (`0x000C`) and other Unicode control characters. If the original IGT data contain any unacceptable

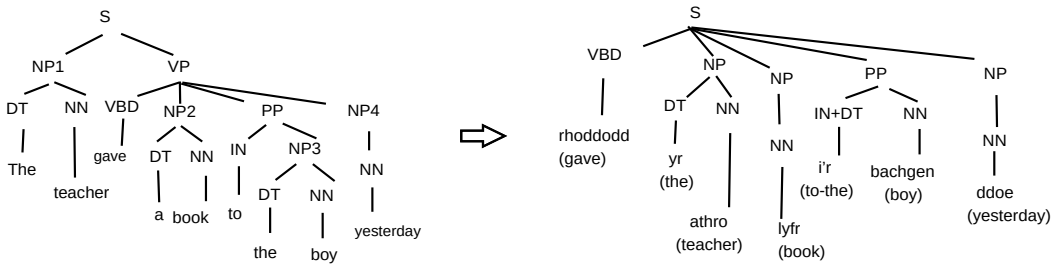

Figure 3: Projecting phrase structure from the translation line to the language line

```
doc_id=397 959 961 L G T
language: korean (kor)
line=959 tag=L: (1) Nay-ka ai-eykey pap-ul mek-i-ess-ta
line=960 tag=G:     I-Nom child-Dat rice-Acc eat-Caus-Pst-Dec
line=961 tag=T:     'I made the child eat rice.'
```

Figure 4: An IGT example in plain text format

Storing cleaned and normalized IGT: Because IGT instances are extracted automatically from Web documents and linguists do not follow the a consistent protocol for creating IGT, [8] the original IGT instances can be noisy. After completing the cleaning step discussed in Section 3.1., the cleaned version of the IGT is stored as the second tier, `odin-clean`.

Segmentation and alignments: Given a (cleaned) IGT instance, the next step of enriching the data is to segment each line into tokens and align tokens between adjacent lines. The tiers `words`, `glosses` and `translations` store the results of segmentation and word alignments. The segmentation part uses character offsets to reference the data in the `odin-raw` tier (or the `odin-clean` tier if it exists). Notice that the alignment between words and glosses is one-to-one and monotonic by definition;[9] in contrast, the alignment between glosses and translations can be one-to-many (e.g., last word in the gloss line aligns to *made* and *eat* in Figure 4) or one-to-zero (e.g., *the* in the translation line in Figure 4 aligns to nothing).

Syntactic structures: An important source of information in IGT is the implicit structure of the string in the translation line—this is equally true for human readers of IGT who use their knowledge as speakers of the "target" language (the language translated into) to gain an understanding of the source line, just as it is to automatic processors of IGT which can take advantage of the greater resources available for English (the most common language for IGT translation lines) to gain information about the structure of the source line. We extended Xigt to allow for the encoding of syntactic structures (dependency structures or parse trees). The last tier in Figure 5 shows the dependency structure of the translation line. One can easily add another tier to show the dependency structure of the language line or phrase structures of the translation/language lines. Adding these additional tiers (e.g., tiers containing parses) to existing data in the ODIN database is work that is in progress.

Partial representations: Finally, note that the Xigt encoding of enriched IGT allows for partial representations; that is, because of noise in the original IGT, we cannot always produce all levels of enrichment. A minimal Xigt ODIN entry will have the `odin-raw` tiers encoding the information as extracted directly from the PDF, a citation for the original source, and a language ID. The users of ODIN data can add their own tiers of annotations and make that available to the public. Thus we see another benefit of encoding ODIN data in Xigt: Xigt allows easy encoding of information in an enriched IGT and helps facilitate the exchange of IGT data with various levels of annotation.

5. Conclusion

The ODIN resource has already proved valuable in a number of research projects. Our goal in the present work is to make it more accessible to the community on several levels. First, the ODIN data will be available for download, both in bulk and on a per-language basis. Second, users will have the option of accessing the original ODIN data (as extracted from the source PDFs) or the Xigt-encoded version. Xigt is designed to support automated processing of IGT data, and as such should allow ODIN users to avoid retracing the same preprocessing steps referenced in this paper.

Third, and most importantly, much previous work on ODIN has built on the enrichment steps described in Section 3., but the enriched data have not previously been available for use by the broader research community. The current release of the Xigt-encoded version of the resource includes the fundamental enrichment which makes explicit the alignments encoded in the visual layout of IGT. In future releases we will include additional enrichments, including projected syntactic structures. In both cases, the new release, which we are calling ODIN-II, allows other users to benefit from the enriched and cleaned aspects of the data without having to reimplement the relevant steps.

characters, we replace them with the Unicode replacement character (`0xFFFD`).

[8] The problem persists despite efforts to do impose consistency, such as the Leipzig Glossing Rules (Bickel et al., 2004).

[9] *Monotonic* means that the word alignment arcs do not cross; that is, if the word w in the language line aligns to a gloss g in the gloss line, any words after w have to align to glosses after g.

```
<?xml version="1.0" encoding="utf-8"?>
<xigt-corpus alignment-method="auto">
  <metadata type="xigt-meta">
    <meta type="language" name="korean" iso-639-3="kor" tiers="words"/>
    <meta type="source" id="src1"
          title="ARGUMENT COMPOSITION AND THE LEXICON:
            LEXICAL AND PERIPHRASTIC CAUSATIVES IN KOREAN"
          author="Bratt, Elizabeth Owen"
          year="1996"/>
  </metadata>
  <igt id="i1">
    <tier type="odin-raw" id="o">
      <item id="o1">doc_id=397 959 961 L G T</item>
      <item id="o2">language: korean (kor)</item>
      <item id="o3">line=959 tag=L: (1) Nay-ka ai-eykey pap-ul mek-i-ess-ta</item>
      <item id="o4">line=960 tag=G:     I-Nom child-Dat rice-Acc eat-Caus-Pst-Dec</item>
      <item id="o5">line=961 tag=T:     'I made the child eat rice.'</item>
    </tier>
    <tier type="odin-clean" id="c" alignment="o">
      <item id="c1" alignment="o3"
            line="959" tag="L">Nay-ka ai-eykey pap-ul mek-i-ess-ta</item>
      <item id="c2" alignment="o4"
            line="960" tag="G">I-Nom child-Dat rice-Acc eat-Caus-Pst-Dec</item>
      <item id="c3" alignment="o5" line="961" tag="T">I made the child eat rice.</item>
    </tier>
    <tier type="phrases" id="p" content="c">
      <item id="p1" content="c1"/>
    </tier>
    <tier type="words" id="w" segmentation="p">
      <item id="w1" segmentation="p1[0:6]"/>
      <item id="w2" segmentation="p1[7:15]"/>
      <item id="w3" segmentation="p1[16:22]"/>
      <item id="w4" segmentation="p1[23:35]"/>
    </tier>
    <tier type="glosses" id="g" alignment="w" content="c">
      <item id="g1" alignment="w1" content="c2[0:5]"/>
      <item id="g2" alignment="w2" content="c2[6:15]"/>
      <item id="g3" alignment="w3" content="c2[16:24]"/>
      <item id="g4" alignment="w4" content="c2[25:41]"/>
    </tier>
    <tier type="translations" id="t" alignment="p" content="c">
      <item id="t1" alignment="p1" content="c3"/>
    </tier>
    <tier type="words" id="tw" segmentation="t">
      <item id="tw1" segmentation="t1[0:1]"/>
      <item id="tw2" segmentation="t1[2:6]"/>
      <item id="tw3" segmentation="t1[7:10]"/>
      <item id="tw4" segmentation="t1[11:16]"/>
      <item id="tw5" segmentation="t1[17:20]"/>
      <item id="tw6" segmentation="t1[21:25]"/>
    </tier>
    <tier type="alignments" id="a" source="tw" target="g">
      <item id="a1" source="tw1" target="g1/>
      <item id="a2" source="tw2" target="g4"/>
      <item id="a3" source="tw4" target="g2"/>
      <item id="a4" source="tw5" target="g4"/>
      <item id="a5" source="tw6" target="g3"/>
    </tier>
    <tier type="dependencies" id="dt" dep="tw" head="tw">
      <item id="dt1" dep="tw1" head="tw2"/>
      <item id="dt2" dep="tw3" head="tw4"/>
      <item id="dt3" dep="tw4" head="tw2"/>
      <item id="dt4" dep="tw5" head="tw2"/>
      <item id="dt5" dep="tw6" head="tw5"/>
    </tier>
  </igt>
</xigt-corpus>
```

Figure 5: The Xigt representation of the IGT in Figure 4

Finally, IGT encoded in the Xigt format can be easily extended to add more tiers or alternative annotations. Xigt thus serves as a vehicle for users of ODIN to improve annotations on the ODIN data, which in turn can provide for iterative improvements to the resource. The successive versions of ODIN can therefore have broader utility across the community.

6. Acknowledgments

We would like to thank Ryan Georgi and Glenn Slayden for discussion on the Xigt format and the issues with the original IGT data. We are also grateful to the three anonymous reviewers for helpful comments.

This material is based upon work supported by the National Science Foundation under Grant No. BCS-1160274 and BCS-0748919. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

7. References

Bender, E. M., Goodman, M. W., Crowgey, J., and Xia,

F. (2013). Towards creating precision grammars from interlinear glossed text: Inferring large-scale typological properties. In *Proceedings of the 7th Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, pages 74–83, Sofia, Bulgaria, August.

Bickel, B., Comrie, B., and Haspelmath, M. (2004). The Leipzig Glossing Rules: Conventions for interlinear morpheme-by-morpheme glosses (revised version). Technical report, Department of Linguistics of the Max Planck Institute for Evolutionary Anthropology and the Department of Linguistics of the University of Leipzig, sept. http://www.eva.mpg.de/lingua/files/morpheme.html (2006-May-17).

Dorr, B. J. (1994). Machine translation divergences: a formal description and proposed solution. *Computational Linguistics*, 20(4):597–635.

Farrar, S. and Langendoen, D. T. (2003). A linguistic ontology for the semantic web. *GLOT International*, 7(3):97–100.

Georgi, R., Xia, F., and Lewis, W. D. (2013). Enhanced and portable dependency projection algorithms using interlinear glossed text. In *Proceedings of ACL 2013 (Volume 2: Short Papers)*, pages 306–311, Sofia, Bulgaria, August.

Goodman, M. W., Crowgey, J., Xia, F., and Bender, E. M. (forthcoming). Xigt: Extensible interlinear glossed text. *Language Resources and Evaluation*.

Hwa, R., Resnik, P., Weinberg, A., Cabezas, C., and Kolak, O. (2005). Bootstrapping Parsers via Syntactic Projection across Parallel Texts. *Special Issue of the Journal of Natural Language Engineering on Parallel Texts*, pages 311–325.

Kingsbury, P., Palmer, M., and Marcus, M. (2002). Adding semantic annotation to the Penn TreeBank. In *Proceedings of the Human Language Technology Conference (HLT-2002)*, San Diego, CA.

Lewis, W. D. and Xia, F. (2008). Automatically Identifying Computationally Relevant Typological Features. In *Proc. of the Third International Joint Conference on Natural Language Processing (IJCNLP-2008)*, Hyderabad, India.

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

Lewis, W. D., Farrar, S., and Langendoen, D. T. (2001). Building a knowledge base of morphosyntactic terminology. In *Proceedings of the IRCS Workshop on Linguistic Databases*, pages 150–156, University of Pennsylvania.

Lewis, W. D. (2003). Mining and migrating interlinear glossed text. Technical report, Workshop on Digitizing and Annotating Texts and Field Recordings, LSA Institute, July. http://emeld.org/workshop/2003/papers03.html.

Xia, F. and Lewis, W. D. (2007). Multilingual structural projection across interlinear text. In *Proc. of the Conference on Human Language Technologies (HLT/NAACL 2007)*, pages 452–459, Rochester, New York.

Xia, F. and Lewis, W. D. (2008). Repurposing Theoretical Linguistic Data for Tool Development and Search. In *Proc. of the Third International Joint Conference on Natural Language Processing (IJCNLP-2008)*, Hyderabad, India.

Xia, F., Lewis, W. D., and Poon, H. (2009). Language ID in the Context of Harvesting Language Data off the Web. In *Proceedings of The 12th Conference of the European Chapter of the Association of Computational Linguistics (EACL 2009)*, Athens, Greece, March 30 – April 3.

Xia, F., Lewis, C., and Lewis, W. D. (2010). The problems of language identification within hugely multilingual data sets. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC 2010)*, pages 2790–2797, Valletta, Malta.

Yarowsky, D. and Ngai, G. (2001). Inducing Multilingual POS Taggers and NP Bracketers via Robust Projection across Aligned Corpora. In *Proc. of the 2001 Meeting of the North American chapter of the Association for Computational Linguistics (NAACL-2001)*, pages 200–207.