Developing ODIN: A Multilingual Repository of Annotated Language Data for Hundreds of the World's Languages

William D. Lewis
Microsoft Research, USA

Fei Xia
Department of Linguistics, University of Washington, USA

Abstract

In this article, we review the process of building ODIN, the Online Database of Interlinear Text (http://odin.linguistlist.org) a multilingual repository of linguistically analyzed language data. ODIN is built from interlinear text that has been harvested from scholarly linguistic documents posted on the web. At the time of this writing, ODIN holds nearly 190,000 instances of interlinear text representing annotated language data for more than 1,000 languages (representing data from >10% of the world's languages). ODIN's charter has been to make these data available to linguists and other language researchers via search, providing the facility to find instances of language data and related resources (i.e. the documents from which data were extracted) by language name, language family, and even annotations used to markup the data (e.g. NOM, ACC, ERG, PST, 3SG). Further, we have sought to enrich the data we have collected and extract 'knowledge' from the enriched content. To enrich the data, we use a variety of statistical tagging and parsing methods applied in the English translations. An enhanced search facility allows users to find data across languages for a variety of syntactic constructions and constituent orders, facilitating unprecedented automated and online discovery of language data.

Correspondence:
William D. Lewis,
Microsoft Research,
One Microsoft Way,
99\1637, Redmond,
WA 98052, USA
E-mail:
wilewis@microsoft.com.

1 The Value of Large Multilingual Resources

As vast amounts of language data has become available electronically, linguistics is gradually transforming itself into a discipline where science is often conducted using corpora. The development of tools to annotate and enrich language data semi-automatically have enhanced the research experience by enabling richer interactions with data. The fact that a linguist can ask a question using tools such as tgrep[1] against treebank data for a dozen or so of the world's languages, looking for the tell-tale structural signatures of relative clauses, topicalization, or double-objects, has opened doors to large-scale inquiry that until a couple of decades ago was unimaginable. Unfortunately, data for the vast majority of the world's languages are not readily available electronically, let alone in a richly annotated form (such as treebank). Although resources such as the Open Language Archives Community (OLAC) have allowed linguists to locate resources for thousands of the world's languages, once a relevant resource is discovered; however, actually interacting with the data are often idiosyncratic, haphazard, or nonexistent, and queries across multiple resources or data points in dozens or more languages are out of the question.

The next stage in the evolution of scientific discovery in the field of linguistics depends on the evolution of large multilingual data stores, which will allow linguists to locate relevant data or structures across dozens, hundreds, or even thousands of languages simultaneously.[2] Richly annotated resources for dozens of the world's languages have already proven themselves to be valuable. Having the same available for hundreds to thousands of the world's languages would be truly transformative.

A fortunate consequence of over a hundred years of annotating language data is that there is already a significant amount of annotated language data for many of the world's languages. This data, often included in scholarly discourse posted to the web, acts as a kind of 'back door' for larger scale multilingual query. A common, semi-structured data type used throughout the discipline is Interlinear Glossed Text (IGT).

In this article, we discuss the harvesting of IGT commonly found in scholarly linguistic papers. The resulting database, The Online Database of Interlinear Text (ODIN), is a resource regularly used by the linguistic community and represents a first step in the development of similar resources that can benefit linguists and other language experts.

This article is organized as follows: in Section 2, we give some background on IGT and why IGT might be a reasonable data source for building a large multilingual resource. In Section 3, we discuss how we construct web crawlers to find documents containing IGT and how we identify and extract instances of the data, comparing both heuristic-based approaches for IGT detection versus adaptive, machine learning approaches. In Section 4, we review language identification over IGT, and why it presents challenges to existing language identification algorithms (e.g. due to small sample sizes, hundreds to thousands of languages, etc.). In Section 5, we review the search facility in ODIN, focusing in particular on the utility of finding data and resources for a large number of the world's languages. ODIN's search facility allows users to find data by language name, code, or family, and by annotations and linguistic constructions. In Section 6, we show how we can use existing data enrichment algorithms (e.g. taggers, parsers) to enrich language data over IGT instances. Data enrichment opens the door to an enhanced search facility and to automated tools that can be applied to resource poor languages (the bulk of languages documented with IGT are languages for which very few resources exist). Finally, we conclude the article in Section 7.

2 Background

2.1 The ODIN vision

ODIN is a database of interlinear text 'snippets', harvested mostly from scholarly documents posted to the web. ODIN was developed as part of the greater effort within the GOLD Community of Practice (Farrar and Lewis, 2006)[3] and the Electronic Metastructure for Endangered Languages Data efforts (EMELD)[4], whose goals are to promote best practice standards and software, specifically those that facilitate interoperation over disparate sets of linguistic data. ODIN's genesis came from the realization that despite the fact that significant amounts of language data are being posted and maintained on the web, there is no uniform search strategy for discovering these data, and most that can be discovered cannot be easily manipulated or used. The Linguistics cyberinfrastructure may be expansive and rich, yet 'discovering' language data on the web often depends on haphazard, low-precision, low-recall string-based search strategies (using tools such as Google[5], Yahoo[6], or Bing[7]), or even on decidedly low-tech discoveries made by word-of-mouth.[8]

In our pursuit of a better way to locate and use language data within the existing infrastructure, we came to realize certain norms in the presentation of data that could be tapped for automated discovery and manipulation. One of the more typical semi-structured formats that linguists use is IGT. We conceived of ODIN as a means to locate instances of IGT on the web by language name and code, such that the linguist doing a search could be reasonably confident that the resources discovered do in fact contain data for the language of interest.

As we built ODIN, we realized that IGT lent itself to other types of search and knowledge discovery. Beyond the obvious search facility for language data by language name, code or family, it is also possible to search for language data by annotations used within it. We also realized that because IGT natively contains language data for two languages—a source language and a translation—the application of automated taggers and parsers to the translation could lead to other types of discovery, moving beyond merely finding data, but actively manipulating and enriching it.

2.2 IGT

IGT is a common method for encoding and displaying linguistic data, and is often used in scholarly articles 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 gives data for the language in question, either phonetically encoded or transcribed in the language's native orthography. The line is broken down into words and morphemes, where words are usually delimited by spaces, and morphemes by dashes ('-'), although other characters can be used to delimit morphemes, such as '+' and '=' (the latter two are most often used for delimiting clitics). The second line contains a morpheme-by-morpheme or word-by-word gloss for the data in the first line, or a mixture of the two. The second-line delimiters are generally the same as those in the first, and morphemes and words usually align between the two lines. Where a given word or morpheme can be glossed by more than one term, periods ('.') or colons (':'), and sometimes dashes or spaces, separate the additional glosses (for instance, in (1) 'Rhoddodd' is glossed as 'gave' and '3sg'). Finally, the third line contains a free-translation of the first line.[9]

(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)

Glosses in the second line take two forms: those representing grammatical information (usually formal or semantic features), which we label with the term gram (in the spirit of Bybee and Dahl, 1989), and described in more detail in Lewis (2003), and those that contain unconstrained translations of words and morphemes, which we will label by the generic term gloss. Grams are often put in upper case by the linguist to differentiate them from glosses.

2.3 Why IGT?

Although other forms of interlinear text exist, ODIN is built primarily around IGT, i.e. interlinear text that is commonly found in scholarly linguistic discourse. IGT is a semi-structured data type, and as such, is amenable to some degree of automation. There are consistencies in its use across the discipline—the standard three- or four-line format, phonetically or orthographically encoded language data, a morpheme-by-morpheme or word-by-word gloss, a free translation, and some markup terminology regularities—which provide a rich underlying structure that make the data format more readily accessible to automated manipulation than other less-structured data types. Of even greater importance is its ubiquity: it is used throughout the discipline to present language data, and can be found in thousands of web-accessible linguistic documents.

That being said, IGT is a presentation format, and as such, is geared for human consumption, not for automation. Two problems are presented for automation: first, structural associations between elements are not clear, in that the alignment between lines is not explicit, nor is the relation between elements on the same line necessarily evident. Second, the semantics used for describing grammatical concepts, the grams, is not consistent across instances, and can vary by language studied, theoretical tradition, and even by researcher.

On the other hand, its relatively consistent structure and format make it conducive to mining and enrichment. Finding IGT involves locating salient linguistic documents and then searching these documents for text that looks like IGT. Once discovered, instances of IGT can be 'unpacked', that is, the relations between elements across and within the lines of IGT can be made explicit and stored in such a way to facilitate search. In addition, term ambiguities can be resolved through manual and automated means. Finally, the content of IGT can be enriched through the use of statistical taggers and parsers applied to the near universal English translation found within IGT instances.[10] These methods for discovery and enrichment are covered in detail in the next sections.

3 Crawling for and Detecting IGT

Locating IGT on the web is done in two steps. First, documents that might contain IGT are located. This is done by crawling the web for linguistic documents and collecting those documents that most likely contain IGT. Second, IGT within these documents is detected and extracted. IGT detection and extraction involves scanning the text of a document for patterns that resemble IGT, and then extracting and storing the detected instances in the ODIN database.

3.1 Crawling for IGT

The major difficulty with locating documents that contain IGT is reducing the size of the search space. We decided very early in the development of ODIN that unconstrained web crawling was too time and resource intensive a process to be feasible due to the web's massive size. Focused crawls, *á la* Mercator (Najork and Heydon, 2002), which optimize crawling performance by quickly eliminating irrelevant pages, presented a good method for reducing the search space, but still required large, resource-intensive crawls. We discovered that highly focused meta-crawls were far more fruitful. Meta-crawling essentially involves 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. In addition, we found that precision increases when we include two or more grams per query. Thus, for example, although ERG alone returns a large number of linguistic documents, ERG combined with ABS (or any other high frequency gram) returns a far less noisy and far more relevant set of documents.

Other successful queries we use include: language names and codes [drawn from the Ethnologue database (Gordon, 2005)], linguists' names and the languages they work on (drawn from the LinguistList linguist database), linguistically relevant terms (drawn from the SIL linguistic glossary), and queries that look for particular language data, such as words or morphemes found in IGT.

Given a set of documents returned from a query, we then search through them for signs of IGT. Integrated with the crawlers is an IGT detector, whose mandate is a limited one: if a targeted document appears to contain 'at least one' instance of IGT, that document is harvested for inclusion in a subsequent more thorough off-line detection process.

Table 1 shows the statistics for the most successful crawls and their related search term 'types'. Calculated from the top 100 queries for each type, the table presents the most successful query types, the average number of documents returned for each query type (of those in the top 100), and the average

Table 1 The most successful query types

Query Type	Avg no. docs	Avg no. docs w/IGT
Gram(s)	1,184	239
Language name(s)	1,314	259
Both grams and names	1,536	289
Language words	1,159	193

number of documents in which IGT was actually found.

At the time of this writing, nearly 1.5 million documents have been crawled. Of these, approximately 150,000 documents have been identified as potentially containing IGT. We anticipate a substantial increase in the number of IGT instances in ODIN in the upcoming months and years as we process these documents, and as we continue to crawl and add data to the database.

3.2 Detecting and extracting IGT

We have explored two methods for detecting and extracting IGT in source documents. The first method uses regular expression (regex) 'templates'—grep-like regexs that look for multi-line instances—to look for the tell-tale signatures of common IGT types. The second method uses machine learning, treating the regex templates and a host of other heuristics as features for a learner.

3.2.1 *Detection using regexes*

An example of regex templates is shown in (2), which looks for a three-line IGT instance which indented any number of spaces, the first line of which begins with a number in parentheses, a second line that contains anything, and a third line that begins with a quote. This regex would 'discover' an instance of IGT like that shown in (1), and the two numbered examples in Table 1.

(2)

```
\t*(\()\d*\).*\n
\t*.*\n
\t*\'.*\n
```

The IGT detector uses a chart (Earley, 1970) to track the templates currently being processed, temporarily storing the active hypotheses in the chart. When the longest matching currently active rule is completed successfully (cf. Abney, 1995), the relevant instance of IGT is extracted and added to the database as a potential instance of IGT. The chart is then cleared and processing continues.

Using regexs in an IGT detector suffers from fairly low precision (61%) and recall (52%). Since we felt that precision was far more important than recall—the assumption being that linguists would be far less forgiving of incorrectly identified IGT than of missing instances—we had concentrated most of our efforts on improving the precision at the cost of recall.

Our original regex detector was enhanced with a few heuristics designed to improve precision. The first involves a rather 'unforgiving' alignment algorithm that filtered out IGT instances where the first and second lines fail to align (in other words, counting common delimiters such as spaces, dashes, etc., and throwing out instances where the counts differ). Precision rose to 88% using this method, but recall dropped rather precipitously to 18%.

Language identification (Language ID) was also used as a filter (Language ID is covered in more detail in Section 4). We developed two language ID methods that we used in this manner. The first looked for language names in the surrounding text. Since the database of language names contained in the Ethnologue is used (Gorden, 2005), each language name that is found can be mapped to a unique three-letter language code. The second involved building statistical language models (Gorden and Trenkle, 1994) over the first line of IGT, the line encoding the language data of interest, which, when compared against an inventory of language models built over already collected IGT, returns one or more ranked language codes. The language codes that were returned were then compared against the language code returned by the first heuristic, and if a match is found, the IGT instance was automatically added to the database.

Ultimately, precision for IGT detection rose to 98% using these heuristics, although recall dropped even further, to 13%.

3.2.2 *Problems with the regex detection method*

The substantial increase in precision is clearly desirable, however, with recall in the low teens, many instances of IGT are not detected by the algorithm. The hand-written templates are rigid and not forgiving of even the slightest differences in format. Further, since there are many formats for IGT, it is not possible to write a comprehensive 'grammar' to catch all types. The canonical form of IGT, as presented in Section 2.2, may consist of three distinct parts and each generally residing on a different line (i.e. the prototypical source language, gloss, and translation lines), but it is not uncommon for IGT found in documents to deviate from this norm, sometimes substantially. There are several reasons for this. First, when IGT examples appear in clusters within a article—for example, a set of examples from the same language that differ very little from one another—authors will often abbreviate instances into two parts versus three. In other cases, authors will enrich a particular instance of IGT beyond the norm by adding layers of annotation, such as additional phonological or morphological analyses, or in rare cases, even additional translations or orthographic transcriptions.[11]

Second, dictated by formatting constraints, long IGT examples may need to be wrapped one or more times, and there are no conventions on how wrapping should be done, nor how many times it can be done. In contrast, short IGT examples might have the translation placed to the right of the target line rather than below it. Thus, for any given IGT example, data may appear on multiple lines or it may be clumped together onto a single line.

Third, most IGT-bearing documents on the web are in PDF, and the off-the-shelf PDF-to-text conversion tool we ran in a preprocessing step will sometimes corrupt instances, spreading text across multiple lines, or corrupting the encoding if nonstandard fonts were used (the latter is particularly true of the source line). In some instances, some words or morphemes on the source or gloss line are inadvertently dropped in the conversion.

The reader will note example (3) (from Kim, 1997) as problematic for many of the reasons described above: the first line of the IGT instance is displaced up by one line; this is caused by the PDF-to-text process. Also note that the annotations (e.g. AGRP, Adj, NP, etc.) are integrated with the language line, so the example is missing a separate gloss line.

```
(3)
[DP [D0 Ku] [AGRP [Adj ketaran] AGR0 [NP namwu]]]
a.
        the            big               tree
```

All of this complicates the IGT detection task, making the highly specific regex approach 'brittle' and less accurate, dropping its recall. A statistical approach can outperform a rule-based one.

3.2.3 *using machine learning in the detection process*

In our revamped detection process, 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 tag in a predefined tag set; then we convert the best tag sequence into a span sequence. A span is a (start, end) pair, that indicates the beginning and ending line numbers of an IGT instance.

Among all the tagging schemes we experimented with, the following five-tag scheme works the best on the development set: BL (any blank line), O (a non-BL line outside IGT), B (the first line in an IGT), E (the last line in an IGT), I (a non-BL line inside an IGT that is not a B or an E).

A learner is then trained to tag a document with these five tags. We trained our learner on the following four types of features:

F_1: the words that appear on the current line. These are features typically used in a text classification task.

F_2: sixteen features that look at various cues for the presence of IGT. For example, whether the line starts with a quotation, whether the line starts with an example number (e.g. (1)), and whether the line contains a large portion of hyphenated or non-English tokens.

F_3: in order to find good tag sequences, we include features for the tags of the previous two lines (e.g. if the learner tagged the preceding line with BL, O, etc.).

F_4: the same features as in F_2, but checked against the neighboring lines. For instance, if a feature f_5 in F_2 checks whether the current line contains a citation, f_5^{+1} checks whether the next line contains a citation.

After the lines in a document are tagged by the learner, we identify IGT instances by finding all the spans in the document that match the '$B\ [I|BL]^*\ E$' pattern; that is, a span that starts with a B, ends with an E, and has zero or more I or BL in between.[12]

3.2.4 *Experimental results*

To evaluate the two IGT detectors, we randomly selected sixty-one ODIN documents and manually marked the location of IGT in the documents. The files were then split into training, development, and test sets; the size of each set is shown in Table 2. The annotation speed was about 4000 lines per hour. Each file in the development and test sets was annotated independently by two annotators, and the inter-annotator agreement (*f*-score) on IGT boundary was 93.74% when using 'exact match' (i.e. two spans match iff they are identical). When 'partial match' (i.e. two spans match iff they overlap) was used, the *f*-score increased to 98.66%.

We used four machine learning algorithms implemented in Mallet (Mccallum, 2002): decision tree, naïve Bayes, maximum entropy (MaxEnt), and conditional random field (CRF). We tried all possible combinations of the feature types with each of the learners, and found that MaxEnt and CRF outperformed decision tree and naïve Bayes. Table 3 shows the MaxEnt model's performance on the development set with different combinations of features: the highest *f*-score for exact match in each group is marked in boldface.[13]

Table 4 shows the results on the test data. The performance of MaxEnt on this data set is slightly worse than on the development set mainly because the test set contains much more corrupted data (due to pdf-to-text conversion) than both the training and development sets. Both tables show that the machine learning approach outperforms the regex approach, reducing the error rate by >50%, and simultaneously increasing 'both' precision and recall (identified in bold in the f-score measure). In particular, corrupted examples such as (3.2.2) are more likely to be discovered by the machine learning approach than the regex-based approach, as the latter approach would require adding a specific rule for this kind of corruption. In addition, the partial match results for both approaches are much better than exact match results, indicating that many span errors could be potentially fixed by postprocessing.

Table 2 Data sets for the IGT detection experiments

	No. of files	No. of lines	No. of IGTs
Training data	41	39,127	1,573
Dev data	10	8,932	447
Test data	10	14,592	843

4 Language ID

Automated language identification (language ID) is an essential task for harvesting language data: since the principal use of ODIN is to locate data and resources for specific languages, ODIN's users would be quite unforgiving if examples in the

Table 3 Performance on the development set

Features	System span num	Classification accuracy	Exact match			Partial match		
			Prec	Recall	*f*-score	Prec	Recall	*f*-score
Regex templates	269	N/A	68.40	41.16	**51.40**	99.26	59.73	74.58
F_1	130	81.50	68.46	19.91	30.85	97.69	28.41	44.02
F_2	405	93.28	58.27	52.80	**55.40**	95.56	86.58	90.85
$F_1 + F_3$	180	80.26	61.67	24.8	35.40	81.11	32.66	46.57
$F_1 + F_2$	420	94.42	63.09	59.28	61.13	93.81	88.14	90.88
$F_2 + F_3$	339	92.68	75.81	57.49	65.39	93.21	70.69	80.40
$F_2 + F_4$	456	96.91	80.92	82.55	**81.73**	93.64	95.53	94.57
$F_1 + F_2 + F_3$	370	93.39	75.14	62.20	68.05	93.51	77.40	84.70
$F_1 + F_2 + F_4$	444	97.00	84.68	84.11	84.40	95.95	95.30	95.62
$F_2 + F_3 + F_4$	431	97.79	86.77	83.67	**85.19**	97.68	94.18	95.90
$F_1 + F_2 + F_3 + F_4$	431	98.00	90.02	86.80	**88.38**	97.22	93.74	95.44

'System span number' is the number of IGT identified by the system (cf. the number of IGT in the gold standard is 447); 'Classification accuracy' is the accuracy of tagging all the lines in the documents; 'Exact match' and 'partial match' check whether the spans identified by the system match the ones in the gold standard; N/A, not applicable.

Table 4 Performance on the test set (the span number in the gold standard is 843)

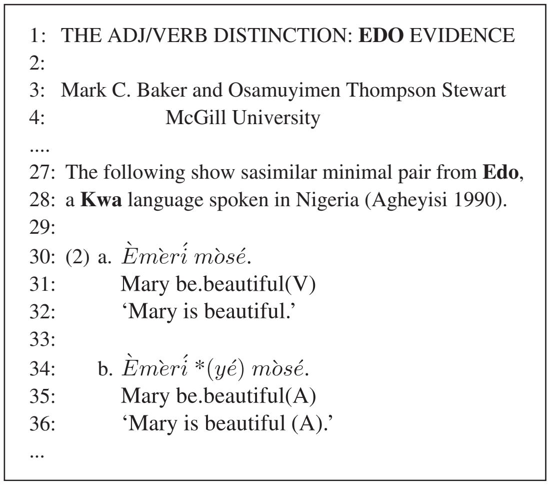
Features	System span num	Classification accuracy	Exact match			Partial match		
			Prec	Recall	*f*-score	Prec	Recall	*f*-score
Regex templates	587	N/A	74.95	52.19	**61.54**	98.64	68.68	80.98
F_2	719	92.45	57.02	48.64	52.50	94.02	80.19	86.56
$F_2 + F_4$	849	95.66	75.50	76.04	75.77	93.76	94.42	94.09
$F_2 + F_3 + F_4$	831	95.95	77.14	76.04	76.58	95.19	93.8	94.50
$F_1 + F_2 + F_3 + F_4$	830	96.80	82.29	81.02	**81.65**	96.51	95.02	95.76

database had the language misidentified. Although 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), language data on the web, specifically IGT, present significant problems for standard language ID methods in a number of ways.

First, the number of languages represented by IGT on the web numbers in the thousands, which is significantly more than most language ID work has had to contend with. This is further compounded by the fact that the amount of data available for most languages represented on the web is highly limited. For example, more than half of languages in ODIN have less than ten IGT instances. It is generally the case that language ID requires relatively large samples of training data (1,000+ characters at a minimum) to build accurate models.

Second, many IGT instances in the test data belong to some languages that have never appeared in the training data. We call this the unseen language problem. Existing language ID algorithms cannot handle this problem since they cannot build models for the languages that they have never seen before. Third, the target sentences in IGT are very short (e.g. a few words), making the task more challenging. Fourth, for languages that do not use a latin-based writing system, the target sentences are often transliterated, making the character encoding scheme less informative (and subject to greater variation across samples).

Given these properties, applying existing language ID algorithms to the task here produces poor results. For instance, although Cavnar and Trenkle's n-gram-based algorithm yields an accuracy as high as 99.8% when tested on newsgroup articles in eight languages (Cavnar and Trenkle 1994),[14] the same algorithm run over IGT data gives an accuracy as low as 1.66% (specifically when there is very little language data for training and the number of languages being evaluated reaches a few hundred).



1: THE ADJ/VERB DISTINCTION: **EDO** EVIDENCE
2:
3: Mark C. Baker and Osamuyimen Thompson Stewart
4: McGill University
....
27: The following show sasimilar minimal pair from **Edo**,
28: a **Kwa** language spoken in Nigeria (Agheyisi 1990).
29:
30: (2) a. *Èmèrí mòsé.*
31: Mary be.beautiful(V)
32: 'Mary is beautiful.'
33:
34: b. *Èmèrí *(yé) mòsé.*
35: Mary be.beautiful(A)
36: 'Mary is beautiful (A).'
...

Fig. 1. A linguistic document that contains IGT: words in boldface are language names

4.1 Language identification as coreference resolution

It might be tempting to simply link each IGT instance to the language names that occur just before them in the document, but that approach has its limits due to two factors. First, linguists are not consistent with labeling IGT instances in their articles: language names can appear before, after, or not at all. For instance, in Fig. 1, both *Edo* and *Kwa* are valid language names; however, the two

IGT instances are *Edo*, not *Kwa* (the language name that appears right before the IGT instances). Second, language names can be ambiguous; that is, it is possible for one name to represent multiple languages. For instance, Tiwa is used both for a Sino-Tibetan language spoken in India and genetically unrelated Kiowa-Tanoan language spoken in the USAs. Since language names are ambiguous, we use ISO 639-3 in our database, however, choosing the correct language code for an ambiguous language name is not an easy task, even for humans.

Since the language name associated with an IGT instance almost always appears 'somewhere' in the document, we treat the language ID task as a coreference resolution problem, where IGT instances and language names appearing in the document are the 'mentions' and the language codes corresponding to these language names are the 'entities'. A language identifier simply needs to link the mentions to the entities, allowing us to apply any good pre-existing resolution algorithms and to provide an elegant solution to the unseen language problem.

Following previous work such as Soon *et al.* (2001), Ng (2005), and Luo (2007), our coreference resolution system processes the mentions sequentially and determines for each mention whether it should start a new entity or be linked to an existing mention; that is, the approach makes a series of 'link' versus 'not-to-link' decisions, one decision per (mention, entity) pair. The decision is made by a classifier that uses the following features:

F_1: the nearest language that precedes the IGT instance.

F_2: the language names appearing in the neighborhood of the target IGT instance.

F_3: character and word n-grams for the language line of each IGT instance. These are compared against n-gram models built over each language for which training data exists.

F_4: Similar to (F_3), but the comparison is between the n-gram lists built over the current IGT instance with the ones built over other IGT instances 'in the same document'. If some of the IGT instances in the document share tokens, they are likely to belong to the same language.

To identify language names in a document, we implemented a simple language name detector that scans the document from left to right and finds the longest string that is a language name according to a language table. The language table we constructed is the result of merging three existing language tables: (1) ISO 639-3 maintained by SIL International;[15] (2) the fifteenth edition of the Ethnologue;[16] and (3) the list of ancient and dead languages maintained by LinguistList.[17] The merged table contains 7,816 language codes and 47,728 (language code, language name) pairs.

4.2 Experimental results

In the experiment, the training set contained 1,372 IGT instances from 125 languages and the test set contained 1,516 instances from 133 languages. In Table 5, the third column shows the percentage of instances in the test data that belong to a 'known' language (i.e. a language that has at least one instance in the training data), and this is the upper bound of any existing language ID algorithms that use training data for building language-specific models (such as the n-gram models in Cavnar and Trenkle's algorithm). The last column lists the accuracy of the new algorithm. The first two rows of the table show that when there is very little training data the existing algorithms fail because they cannot handle unseen languages; their upper bound is unacceptably low. In contrast, the new algorithm performs decently, because it can correctly label an IGT instance with a language code even if it has not seen any data for that language in the training data.

We ran the new language ID algorithm on all currently discovered IGT data, and Table 6 shows the language distribution of the IGT instances in ODIN according to the output of the algorithm.

Table 5 Language ID accuracy on the test data

Percentage of training data used for training	**No. of IGTs**	**Upper bound** (%)	**Accuracy of the new algorithm** (%)
1	14	1.66	70.15
10	137	21.55	80.24
100	1372	54.45	83.08

5 Exposing the Data to Search

A principal focus of ODIN is and has always been to search: how can linguists find the data that they are interested in and how the data can be stored in such a way as to accommodate the variety of queries that a linguist might ask. We initially limited search to queries by language name and code, which maintained compatibility with OLAC, the Open Language Archives Community (Bird and Simons 2003). More recently we have extended the search facility by adding what we have labeled 'Advanced Search', which provides two additional search query types: search by language family and search by annotation (the annotation contained within the gloss of IGT instances). An additional search feature, search by linguistic construction, will be discussed in Section 6.1.

Table 6 The language distribution of IGT instances in ODIN (according to the output of the language ID system)

Range of IGT instances	No. of languages	No. of IGT instances	Percentage of IGT instances
>10,000	3	36,691	19.39
1,000–9,999	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	1,326	189,244	100

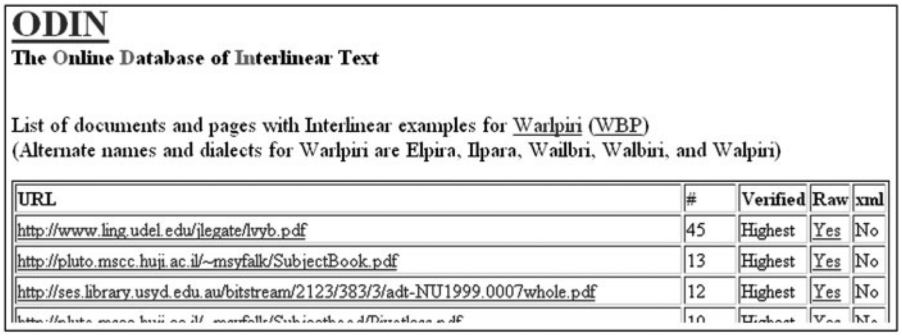
5.1 Language Name/Code search

As noted, the initial search facility provided by ODIN was designed to maintain compatibility with OLAC. OLAC provides for interoperation over language repositories through well-defined metadata based on the Dublin Core metadata set.[18] Registered repositories are harvested regularly by OLAC, and made available to search through the OLAC portals at LinguistList[19] and the LDC[20]. Using these portals, users can search for language resources using language names and codes, where results are presented as a list of relevant repositories and URLs. ODIN generates OLAC metadata as new data are discovered. Results to OLAC queries are supplied via a PHP script that lists all relevant documents for a particular language (the topic of a user's query). A sample output for one such query, for the language Warlpiri (a Pama-Nyungan language spoken in Australia), is shown in Fig. 2.

The user can further 'drill-down' and locate not only the articles that contain data for the language of interest, but also the actual data itself. In Fig. 3, data for Warlpiri are shown (from Legate, 2003).

5.2 Language family search

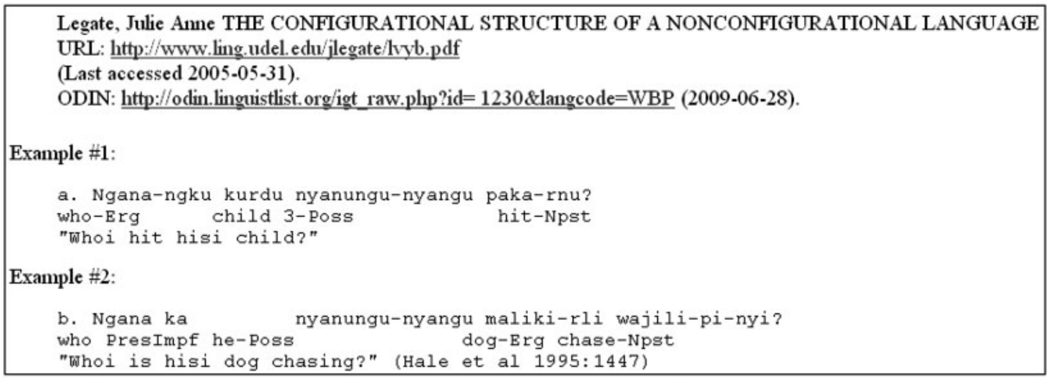
An extension to language name search is search by language family. We use the language families as defined in Ethnologue, and allow users to select family names from a pull-down list. Once selected, results are returned as a list of languages within the selected family, specifically for languages for which data exists in ODIN. An example output is shown in Fig. 4. The user has the option of displaying


ODIN
The Online Database of Interlinear Text

List of documents and pages with Interlinear examples for Warlpiri (WBP)
(Alternate names and dialects for Warlpiri are Elpira, Ilpara, Wailbri, Walbiri, and Walpiri)

URL	#	Verified	Raw	xml
http://www.ling.udel.edu/jlegate/lvyb.pdf	45	Highest	Yes	No
http://pluto.mscc.huji.ac.il/~msyfalk/SubjectBook.pdf	13	Highest	Yes	No
http://ses.library.usyd.edu.au/bitstream/2123/383/3/adt-NU1999.0007whole.pdf	12	Highest	Yes	No
http://pluto.mscc.huji.ac.il/~msyfalk/Subjecthood/Pivotless.pdf	10	Highest	Yes	No

Fig. 2 A list of documents that contain Warlpiri IGT



Legate, Julie Anne THE CONFIGURATIONAL STRUCTURE OF A NONCONFIGURATIONAL LANGUAGE
URL: http://www.ling.udel.edu/jlegate/lvyb.pdf
(Last accessed 2005-05-31).
ODIN: http://odin.linguistlist.org/igt_raw.php?id= 1230&langcode=WBP (2009-06-28).

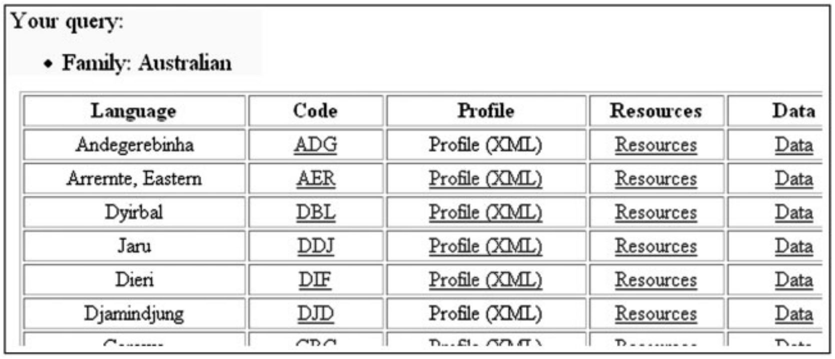
Example #1:

```
a. Ngana-ngku kurdu nyanungu-nyangu paka-rnu?
who-Erg      child 3-Poss           hit-Npst
"Whoi hit hisi child?"
```

Example #2:

```
b. Ngana ka         nyanungu-nyangu maliki-rli wajili-pi-nyi?
who PresImpf he-Poss               dog-Erg chase-Npst
"Whoi is hisi dog chasing?" (Hale et al 1995:1447)
```

Fig. 3 Warlpiri IGT in one of the retrieved documents



Your query:

- Family: Australian

Language	Code	Profile	Resources	Data
Andegerebinha	ADG	Profile (XML)	Resources	Data
Arrernte, Eastern	AER	Profile (XML)	Resources	Data
Dyirbal	DBL	Profile (XML)	Resources	Data
Jaru	DDJ	Profile (XML)	Resources	Data
Dieri	DIF	Profile (XML)	Resources	Data
Djamindjung	DJD	Profile (XML)	Resources	Data

Fig. 4 Query results for Australian languages

'Resources' for each language, which are effectively the documents the IGT instances were extracted from, or he or she can view the instances of IGT themselves (the *Data* link). The user can also display an XML-encoded Language Profile, which contains a 'grammar fragment', *á la* (Farrar and Lewis, 2006) for the language and is automatically compiled from harvested IGT (i.e. from the grams). Profiles contain information about grammatical categories discovered in IGT, such as grammatical case, aspect, tense, etc. An example Profile extracted from ODIN for the language Yoruba (a Niger-Congo language spoken in Nigeria) can be seen in (4). The profile says that in Yoruba the canonical word order is Subject-Verb-Object (SVO), determiners appear after nouns, and the language has Accusative case, Genitive case, Nominative case, and so on. The concepts such as AccusativeCase come from the GOLD Ontology (Farrar, 2003; Farrar and Langendon, 2003).[21]

(4)

```
<Profile>
  <language code="WBP">Yoruba</language>
  <ontologyNamespace prefix="gold">
    http://linguistic-ontology.org/gold.owl#
  </ontologyNamespace>
  <feature="word_order"><value>SVO</value>
  </feature>
  <feature="det_order"><value>NN-DT</value>
  </feature>
  <feature="case">
    <value>gold:AccusativeCase</value>
    <value>gold:GenitiveCase</value>
    <value>gold:NominativeCase</value>
                ...
</Profile>
```

The Language Family search can also be combined with the Concept/Gram and Construction Search discussed in Sections 5.3 and 6, respectively. The combination allows users to constrain the results from these searches to specific language families, thus reducing the number of documents and IGT instances returned for any particular query.

5.3 Concept/Gram search

The Concept/Gram search allows users to search through IGT for instances that contain morphemes or words that encode particular grammatical concepts, such NominativeCase, PerfectiveAspect, SubjunctiveModality, etc. The user specifies the desired concepts in terms of the General Ontology of Linguistic Description (GOLD; Farrar and Langendoen 2003), and the instances of data that map to these concepts are searched for. Grams such as PST and PAST are normalized to a common semantic, namely PastTense, so the query for PastTense will return both of these forms, as well as 1SGPast, HodiernalPast, and others. The user can also request morphemes encoded as affixes (prefixes, suffixes) or clitics (proclitics, enclitics), which, although not explicitly encoded in IGT, can be deduced from their relationships to root morphemes. Thus a query might look something like 'PastTense encoded as a Suffix, Singular encoded as an Enclitic', which will return all instances of IGT where both conditions hold.

The Concept search in ODIN is not as sophisticated as that discussed in Simons, *et al.* (2004), where the full power of GOLD was brought to bear on an RDF-encoded database. Due to the speed concerns, ODIN has limited the search facility to a two-tiered, GOLD compatible hierarchy: users can search for a specific concept, such as ErgativeCase, or a parent concept, such as Case. In Simons *et al.* (2004), the full GOLD hierarchy could be queried, allowing users to abstract away from specific categories and relationships, using the full inference capability of RDF. Although users may wish such power in a linguistic query, implementing such queries are not tractable in real time given current technology.

6 Data Enrichment and Construction Search

6.1 Facilitating search by enriching the English translation

As noted, one motivation behind ODIN is to locate instances of IGT and expose these instances to search. Although the initial search strategy behind ODIN was to comply with the OLAC model, the rich annotated content of IGT, both explicit (e.g. grams) and implicit (e.g. the use of delimiters), opens the door to additional query over IGT, namely search over concepts/grams, as noted in the previous section. Further, since the translation lines in IGT are almost universally English, it is possible to enrich IGT instances even further—namely through the application of robust POS taggers and parsers against the English content—and apply search over the enriched English translations. Queries across the enriched English translations can give us clues as to the structures that 'might' exist in the source language data.

An experimental search interface called 'Construction Search' has been created in ODIN to allow linguists to search for linguistic constructions contained in the IGT that have been 'discovered' by enriching the English translation. The interface is considered experimental since the enrichment process could introduce errors. For instance, the POS tagger and parser are not 100% accurate, and the source language line could have a structure different from that of the English translation line. Despite this deficiency, we believe that activating this search feature adds more value to the linguistic public, giving another way to find data of interest. Currently active construction queries include: Passives, Conditionals, Counterfactuals, Imperatives, Questions, Sententially Negated structures, etc. The English translations for all examples in ODIN have been tagged using the Ratnaparkhi MaxEnt tagger (Ratnaparkhi, 1998). A subset of the examples have also been parsed using the Collins Parser (Collins, 1999).

A search for passive constructions, for instance, will return examples where the tell-tale sign of passives in English can be found (in tagged English text, some form of the copula followed by the past participle of the verb).[22] The source language data might contain passive-like structures, which will need to be reviewed by the linguist to be sure. A search for passive in the ODIN database returns a list of sixty languages where the English translation contains a passive. Selecting one of the languages returns a list of possible passives in that language. As an example, Fig. 5 shows one of the twelve passive examples returned for the langauge Manggarai, an Austronesian language spoken in Indonesia (all examples are from Arka and Kosmas, 2002).

Note that there is an obvious passive construction in the English translation. The linguist would need to decide if this example does in fact represent a passive in the source language data. This appears to be the case in this example.

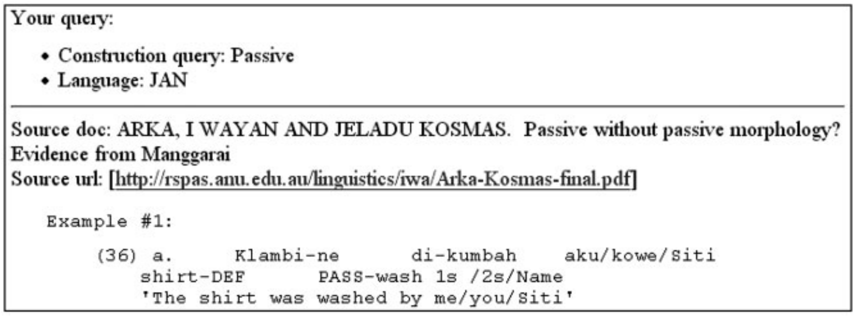
Figure 6 gives the results of another query, one that looks for relative clauses, using a treebank structure matching the following:

```
(NP (NP...)
      (SBAR (WHNP (WP))
             (S...)))
```

This query looks for relative clauses (not reduced) in the parsed English translation, assuming that a relative or relative-like structure exists in the target language data. Data for over 172 languages are found when this query is run, with the example from Breton shown in Fig. 6 (from Phillips, 1996). It is not clear that Breton has relative clauses, but the PCL morpheme (a pronominal clitic) marks the boundaries of the subordinate clause that corresponds to the relative in English.

6.2 Possible paths for source language enrichment

In this section, thus far, we have demonstrated the potential for Construction Search over an enriched (i.e. parsed or tagged) English translation. Determining if the construction also exists in the source language data is left to the linguist. In Xia and Lewis (2007), we proposed an algorithm to enrich IGT in three steps: (1) parse the English translation with an English parser, (2) align the language line and the English translation using the


Your query:

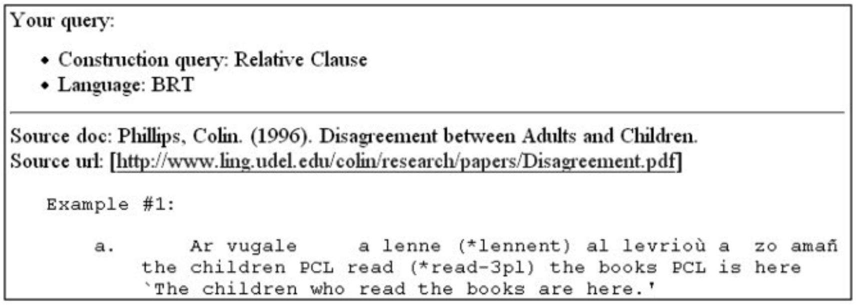
- Construction query: Passive
- Language: JAN

Source doc: ARKA, I WAYAN AND JELADU KOSMAS. Passive without passive morphology? Evidence from Manggarai
Source url: [http://rspas.anu.edu.au/linguistics/iwa/Arka-Kosmas-final.pdf]

```
Example #1:

    (36) a.     Klambi-ne      di-kumbah    aku/kowe/Siti
        shirt-DEF      PASS-wash 1s /2s/Name
        'The shirt was washed by me/you/Siti'
```

Fig. 5 Query results for 'Passives in Manggarai'


Your query:

- Construction query: Relative Clause
- Language: BRT

Source doc: Phillips, Colin. (1996). Disagreement between Adults and Children.
Source url: [http://www.ling.udel.edu/colin/research/papers/Disagreement.pdf]

```
Example #1:

    a.      Ar vugale     a lenne (*lennent) al levrioù a  zo amañ
        the children PCL read (*read-3pl) the books PCL is here
        `The children who read the books are here.'
```

Fig. 6 Query results relative clauses for data in Breton

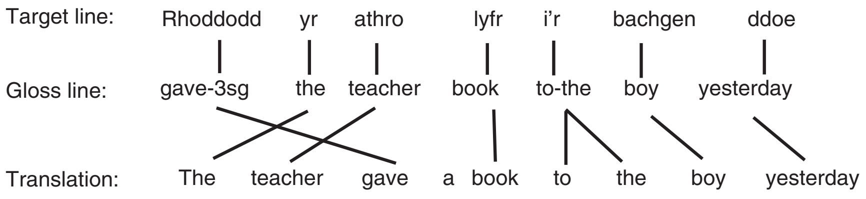

Target line:
Rhoddodd yr athro lyfr i'r bachgen ddoe
Gloss line:
gave-3sg the teacher book to-the boy yesterday
Translation:
The teacher gave a book to the boy yesterday

Fig. 7 Aligning the language line and the English translation with the help of the gloss line

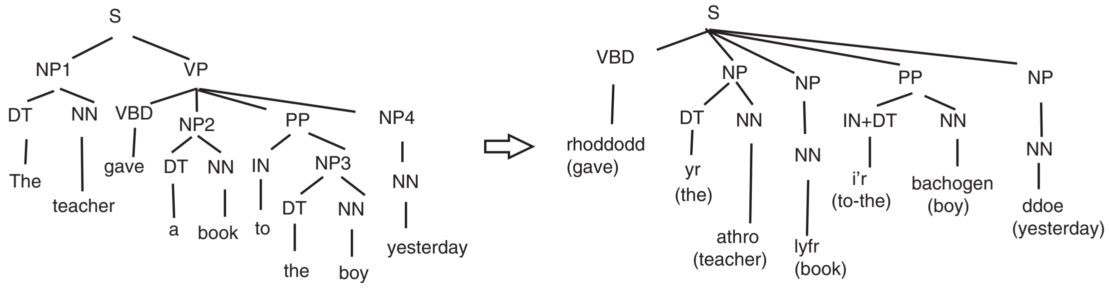


Fig. 8 Projecting phrase structure from the translation line to the language line

gloss line, and (3) project the syntactic structures from English onto the target line. Given the IGT in (1), the algorithm will produce the word alignment in Fig. 7 and the syntactic structures in Fig. 8. We evaluated the algorithm on a small set of 538 IGT instances for several languages. On average, the accuracy of the English syntactic structures is 93.48%; the *f*-score of the word alignment links between the translation and target lines is 94.03%, and the accuracy of the target syntactic structure produced by the projection algorithm is 81.45%. The details on the algorithms and the experiments can be found in Xia and Lewis (2007).

Although it might seem that projecting annotations and structures onto source language data may introduce more noise, and thus limit further the utility of query across such structures, projections do improve the value of construction-based query: the resulting structures follow the linear order of constituents in the source language data rather than that of English. Queries that are dependent on such linear orders (such as looking for examples where a relative clause appears before the noun phrase it modifies) will generate more accurate results. We plan to extend Construction Search to include query over projected structures at a later date.

7 Conclusion

The ODIN database is presented here as a model for leveraging existing infrastructure to facilitate sophisticated search, and as a model for interoperation over legacy data and legacy formats. It stands as an example of how we might proceed in developing the tools and methodologies to grow a richer Linguistics Cyberinfrastructure, especially by transforming and enriching existing web-accessible resources. ODIN itself is a resource that is searchable at the level of data, going beyond other systems both in depth, the granularity of search that is provided, and breadth, the number of languages served. In the future, we look apply to sophisticated statistical NLP technology to harvested language data, which will result in even more searchable content.

8 Fair Use

Crucial to the ODIN effort has been the fair use of the data that are collected. We fully recognize that any instance of data that is collected by ODIN is proprietary and the property either of the linguist who crafted the example or the native community where the example might have originated. In line with linguistic custom and Section 107 of Copyright Law,[23] all instances of IGT that are returned as results of a query are fully cited as to their source, and the source document is provided via link (no copies of source documents are maintained on the ODIN site). The citation information for the 3,000 documents currently referenced by ODIN has been added manually and stored in the database. Later, when more documents are automatically processed, it will not be possible to manually enter the citation information for these documents. In this case, our query engine will still search across the instances from the newer documents, but will only display links to the documents, not to the IGT instances themselves.

Acknowledgements

A special thanks goes to Terry Langendoen and Scott Farrar for their support of ODIN from the beginning. We gratefully acknowledge the support of the EMELD PIs and associates, especially Gary Simons, Helen Aristar-Dry, and Anthony Aristar. We would also like to extend special thanks to Emily Bender, Barbara Citko, Heidi Harley, and Simin Karimi for helping us with finding useful queries for the Construction Query feature in ODIN. The research described in the article is supported in part by National Science Foundation (NSF) under grant no BCS-0411348, BCS-0720670, and BCS-0748919, and by an RRF grant from the University of Washington. Any opinions, findings, and conclusions expressed are those of the authors and do not necessarily reflect the views or official policies of the funding institutions.

References

Abney, S. (1995). Chunks and Dependencies: Bringing Processing Evidence to Bear on Syntax. In Cole, J., Green, G. and Morgan, J. (eds), In *Computational Linguistics and the Foundations of Linguistic Theory*. Stanford, California: CSLI.

Arka, I. W. and Kosmas, J. (2002). Passive without passive morphology? Evidence from Manggarai. In *Paper presented at the 9th International Congress on Austronesian Linguistics*. Canberra, Australia.

Bailyn, J. F. (2001). Inversion, dislocation and optionality in Russian. In Zybatow, G., Junghanns, U., Mehlhorn, G., and Szucsich, L. (eds), *Current Issues in Formal Slavic Linguistics*, Frankfurt: Peter Lang AG. http://www.sinc.sunysb.edu/Clubs/nels/jbailyn/fdslbailyn.pdf (accessed September 2009).

Bird, S. and Simons, G. F. (2003). Extending Dublin Core metadata to support the description and discovery of language resources. *Computers and the Humanities*, **17**(4): 375–388.

Bow, C., Hughes, B., and Bird, S. (2003). Towards a general model of interlinear text. In *Proceedings of the E-MELD Language Digitization Project: Workshop on Digitizing and Annotating Texts and Field Recordings*. University of Michigan. http://emeld.org/workshop/2003/bowbadenbird-paper.pdf (accessed May 2010).

Bybee, J. L. and Dahl, O. (1989). The creation of tense and aspect systems in the languages of the world. *Studies in Language*, **13**(1): 51–103.

Cavnar, W. B. and Trenkle, J. M. (1994). N-gram-based text categorization, In *Proceedings of Third Annual Symposium on Document Analysis and Information Retrieval*. Las Vegas: UNLV Publications/Reprographics, pp. 161–175.

Collins, M. (1999). *Head-Driven Statistical Models for Natural Language Parsing*. PhD. thesis, University of Pennsylvania.

Earley, J. (1970). An efficient context-free parsing algorithm. *Communications of the ACM*, **13**(2): 94–102.

Farrar, S. (2003). *An Ontology for Linguistics on the Semantic Web*. PhD. thesis. University of Arizona. www.u.arizona.edu/~farrar/papers/Far03b.pdf (accessed September 2009).

Farrar, S. and Langendoen, D. T. (2003). A linguistic ontology for the Semantic Web. *GLOT International*, **7**(3): 97–100. www.u.arizona.edu/~farrar/papers/FarLang03b.pdf (accessed October 2007).

Farrar, S. and Lewis, W. D. (2006). The GOLD Community of Practice: An infrastructure for linguistic data on the Web. *Language Resources and Evaluation.* http://faculty.washington.edu/wlewis2/papers/FarLew-06.pdf (accessed September 2009).

Gordon, R. G. (ed.) (2005). *Ethnologue: Languages of the World*, 15th edn. Dallas, TX: SIL International. http://www.ethnologue.com/ (accessed April 2010).

Kim, Y.-K. (1997). Agreement phrases in dp. In *UCL Working Papers in Linguistics 9*. London, University College. http://www.phon.ucl.ac.uk/publications/WPL/97papers/kim.pdf (accessed September 2009).

Legate, J. A. (2003). The configurational structure of a nonconfigurational language. In Pica, P. (ed.), *Linguistic Variation Yearbook*. Amsterdam: John Benjamins.

Lewis, W. D. (2003). *Mining and Migrating Interlinear Glossed Text*. Technical report, Workshop on Digitizing and Annotating Texts and Field Recordings, LSA Institute. http://emeld.org/workshop/2003/papers03.html (accessed September 2009).

Lewis, W. D., Farrar, S., and Langendoen, D. T. (2006). Linguistics in the internet age: Tools and fair use. In *Proceedings of EMELD 2006 Workshop on Digital Language Documentation: Tools and Standards: The State of the Art*. Lansing, MI. http://emeld.org/workshop/2006/papers/lewis.pdf (accessed September 2009).

Liberman, M. (2000). Legal, ethical, and policy issues concerning the recording and publication of primary language materials. In Bird, S. and Simons, G. (eds), *Proceedings of the Workshop on Web-based Language Documentation and description*. Philadelphia: University of Pennsylvania. http://www.ldc.upenn.edu/exploration/expl2000/ (accessed 17 May 2006).

Luo, X. (2007). Coreference or not: a twin model for coreference resolution. In *Proceeding. of the Conference on Human Language Technologies (HLT/NAACL 2007)*. New York: Rochester, pp. 73–80.

McCallum, A. K. (2002). Mallet: a machine learning for language toolkit. http://mallet.cs.umass.edu (accessed September 2009).

Najork, M. and Heydon, A. (2002). High-performance web crawling. In Abello, J. M., Pardalos, P. M., and Resende, M. G. C. (eds), *Handbook of Massive Data Sets*. Dordrecht/Boston/London: Kluwer Academic Press, pp. 25–45.

Ng, V. (2005). Machine learning for coreference resolution: From local classification to global ranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*. Ann Arbor, MI: University of Michigan, pp. 157–164.

Phillips, C. (1996). Disagreement between adults and children. In Mendikoetxea, A. and Uribe-Etxebarria, M. (eds), *Theoretical Issues on the Morphology-Syntax Interface*. San Sebastian: ASJU.

Ratnaparkhi, A. (1998). *Maximum Entry Models for Natural Language Ambiguity Resolution*. PhD. thesis. University of Pennsylvania.

Simons, G. F., Lewis, W. D., Farrar, S. O., Langendoen, D. T., Fitzsimons, B., and Gonzalez, H. (2004). The semantics of markup: mapping legacy markup schemas to a common semantics. In *Proceedings of the 4th workshop on NLP and XML (NLPXML-2004)*. Spain: Barcelona, pp. 25–32. held in cooperation with ACL-04. http://faculty.washington.edu/wlewis2/papers/Sim-etal04b.pdf (accessed April 2010).

Soon, W. M., Ng, H. T., and Lim, D. C. Y. (2001). A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, **27**(4): 521–544.

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

Notes

1 More information about ‘tgrep’ and ‘tgrep2’ (an extension of ‘tgrep’) can be found at http://tedlab.mit.edu/ dr/TGrep2/.

2 The scale of such queries is reminiscent of what biologists and biochemists can do currently, such as throwing queries for particular structural or chemical signatures across the tens of thousands of proteins and nucleic acids contained in the Protein Data Bank (http://www.pdb.org/pdb/home/home.do).

3 http://www.linguistics-ontology.org

4 http://emeld.org

5 http://www.google.com

6 http://www.yahoo.com

7 http://www.bing.com/

8 OLAC, is a major step in the direction of a unified means for language resource discovery. Despite having only forty-three archives, resources can be discovered for thousands of the world’s languages.

9 The canonical form of IGT is the three-line presentation, but the format of IGT does vary significantly across the discipline (e.g. additional lines of annotation, differing ways of presenting annotations, differing delimiters, etc.). See Bow *et al.* (2003) for a comprehensive study of IGT as it is used in the discipline. Also, see the Leipzig Glossing Rules (http://www.eva.mpg.de/lingua/files/morpheme.html) for specific examples of ‘best-practice’ with respect to

IGT. The EMELD site (http://www.emeld.org) also contains specific instructions on encoding IGT and other linguistic data types, and discusses best practice with respect to archiving, interoperation, etc.

10 Over 98% of the IGT instances that have discovered thus far are translated into English. Less common are instances translated into Spanish, German, and other languages.

11 Again, it is suggested the reader consult (Bow *et al.* 2003) for a thorough list of IGT types and examples of each.

12 Other heuristics for converting tag sequences to span sequences produce similar results.

13 F_4 is an extension of F_2, so every combination with F_4 should include F_2 as well. Also, F_3 should not be used alone. Therefore, Table 3 in fact lists all the possible feature combinations.

14 The accuracy ranges from 92.9% to 99.8% depending on the article length and a model parameter called 'profile length'.

15 http://www.sil.org/iso639-3/download.asp

16 http://www.ethnologue.com/codes/default.asp#using

17 http://linguistlist.org/forms/langs/GetListOfAncientLgs.html

18 http://dublincore.org/

19 http://linguistlist.org/forms/langs/find-a-language-or-family.html

20 http://www.language-archives.org/tools/search/

21 It is not our intention to discuss language profiles in detail here. The reader is referred to Farrar and Lewis (2006) for more thorough coverage of terminology sets and language profiles.

22 Linguists can also find passives in the source language data using a gram on the gloss line such as PASS. This is another way that passives can be found, and in ODIN's construction search interface, we use gloss line tags as another means to find constructions of interest (where they exist and are relevant).

23 See Liberman (2000) and Lew *et al.* (2006) for a thorough review of copyright law as it pertains to linguistics data.