

Converting On-Line Bilingual Dictionaries from Human-Readable to Machine-Readable Form

James Mayfield and Paul McNamee

The Johns Hopkins University Applied Physics Laboratory
11100 Johns Hopkins Road, Laurel MD 20723-6099 USA
+1 240-228-6944

{mayfield,mcnamee}@jhuapl.edu

ABSTRACT

We describe a language called ABET that allows rapid conversion of on-line human-readable bilingual dictionaries to machine-readable form.

1. INTRODUCTION

A key resource for many approaches to cross-language information retrieval (CLIR) is a bilingual dictionary (bidict). Unfortunately, like other cross-language resources, machine-readable on-line bidicts that are suitable for use in CLIR are scarce. However, the availability of on-line cross-language dictionaries in human readable format (such as PDF versions of printed dictionaries, or HTML wordlists) is broader. It makes sense then to examine how human-readable dictionaries might be converted to machine-readable form, such as might be useful for CLIR or machine translation (MT).

Such dictionary conversion is tedious work. Each dictionary has its own format. Often, because machine readability is not a primary goal of dictionary authors, the format is inconsistent throughout the document. Extraction of the text form of the human-readable dictionary from its native document type (e.g., PDF) prior to conversion can introduce further noise that makes identification of the source language words or their target language translations difficult. Yet there are certain key processes that are the same across most extraction tasks. We have developed a language called the *APL Bidict Extraction Tool* (ABET) that automates these common processes, while providing for simple descriptions of document-specific formatting. Using ABET, it is possible to quickly convert a human-readable on-line dictionary to machine-readable form, without getting bogged down in the mechanistic details of the conversion. ABET is easy to learn, and lends the full power of Perl pattern matching to the scriptwriter.

More heavily automated approaches include pattern and wrapper induction [3], maximum entropy [1] and HMMs [2]. Such approaches are most useful when many documents use the same format, and when more general document models are needed.

2. The ABET Language

Because bidicts are essentially tabular data, the ABET document model holds that every bidict or bidict component of interest comprises an (optional) header, followed by a list of one or more entries. For example, a typical bidict might contain some introductory text (which can likely be ignored), followed by a sequence of dictionary entries. The model is recursive, so that the individual components can also be expressed as header plus list. For example, each dictionary entry might comprise a source

agyrmak to hurt	akylyly smart, clever
agyrtmak to hurt, cause pain	akylsyz stupid
agyry 1. ill 2. troubles	akylsyzlyk stupidity
agyzy beklemek to fast	akym stream, flow
agyzy mouth	akytmak to pour
agyzyryk horse's bit	al pink
	ala multi-colored (2 colors)

Figure 1. Sample Turkmen/English human-readable (PDF) dictionary

```
{ PRE_KILL => [\&strip_ascii_nulls],
  CROP => "\nA\n(.*)\nA\n",
  FIND => "\n([\n]{3,})",
  POST_KILL => [\&kill_parens,\&kill_brackets],
  NO_TRIM => "TRUE",
  PARSE => {
    HEADER_FIND => "(.*)\s{2,}([\^:]*)",
    SPLIT => "\s*[,;]\s*|\s*\[dIV]\.\s*",
    POST_KILL => [\&kill_leading_abbrevs],
    FN => sub {
      my ($text,$dict,$context,$offsets) = @_;
      add($dict,'en',$text,'tk',$context->[0]);
    }
  }
}
```

Figure 2. Sample ABET script for extracting Turkmen/English dictionary

language word, followed by one or more comma-separated translations. Figure 1 shows a sample of such a dictionary,¹ mapping Turkmen to English, produced by the U. S. Peace Corps. This document model significantly simplifies extraction patterns (scripts) for data that conform to the model, and makes ABET easier to use than tools designed for more general models.

The ABET language structures the extraction process around this simple document model. An ABET script is a brace-delimited block containing a set of key/value pairs whose keys are drawn from a fixed set of action keywords. Figure 2 depicts a sample ABET script, which successfully extracts a machine-readable bidict from the Turkmen/English dictionary of Figure 1. (Readers familiar with Perl will realize that an ABET script is simply a Perl hash; however, very little knowledge of Perl beyond pattern matching is required to use ABET.)

At each level, extraction processing comprises four steps: text preprocessing; header and body identification; text postprocessing; and recursive header and body parsing, or entry

¹ <<http://www.chaihana.com/lang.html>>

into output dictionary. First, preprocessing of the entire text under consideration is performed. Preprocessing commands are **CROP** to delete surrounding extraneous material, **PRE_KILL** to perform an arbitrary transformation on the entire text before parsing (called **KILL** because the most common operation is to delete text, *e.g.*, parenthesized expressions, extraneous characters, *etc.*), and **PRE_MACROS** to expand macro lists, such as replacing the HTML entity **<** with **<**. The top level of our example in Figure 2 uses **PRE_KILL** to delete ASCII null characters from the input, and **CROP** to delete extraneous surrounding text.

Second, an optional header, and a list of entries are identified. Three main ABET commands are available: **HEADER_FIND** to identify a pattern that selects the header entry; **FIND** to identify a pattern that selects the list items; and **SPLIT** to identify a pattern that detects the text *between* list items (which is sometimes easier than matching the items themselves). In addition, **HEADER_POS**, **START_POS** and **END_POS** may be used to select the indices of the header, and of the first and last legitimate list items, within the list of items created by **FIND** or **SPLIT**. In this case, matching the entries themselves is easier than matching the text between entries, so the **FIND** command is used.

The third phase, text postprocessing, includes the commands **POST_KILL** and **POST_MACROS**, which are used like **PRE_KILL** and **PRE_MACROS**, but are applied to individual list items after they are identified; **NO_TRIM**, which suppresses removal of whitespace from the ends of identified list items; and **KEEP_EMPTYIES**, which retains list items that contain no text (and which may be useful *e.g.* to keep two lists in synch with one another). In the example, parenthesized and bracketed expressions are deleted using the **POST_KILL** command, and whitespace removal is suppressed with **NO_TRIM**.

The final processing phase is either to recursively parse each item, or to invoke a function on each item (typically one that adds an entry to the output dictionary). Commands include **HEADER_PARSE** and **PARSE**, to recursively parse the header entry and list items, respectively, using a different ABET script; **HEADER_FN** and **FN**, to apply an arbitrary function to the header or list items respectively; and **LIST_FN**, to apply a function to the list as a whole, rather than item-by-item. The most commonly used function simply adds an entry to the output dictionary. One of the arguments to various functions is the (now parsed) surrounding context. This can, for example, provide the function with the text of a column header indicating the language of the current entry. In our example, **PARSE** is used to recursively parse each line of the dictionary. **PARSE** takes as its argument another ABET script. This latter script recognizes the Turkmen word or phrase using **HEADER_FIND**, identifies the English words and

phrases (or more accurately, the characters that separate them) using **SPLIT**, and adds each resulting pairing to the output dictionary with **FN**. In addition to the extraction commands described, various debugging commands and switches are available to assist the user write effective extraction scripts. ABET also comes with a library of commonly used transformations, *e.g.*, **kill_parens** to remove parenthesized expressions.

3. DISCUSSION

Over the past year, we have collected over 50MB of on-line, human-readable bidicts amenable to harvesting by ABET. These bidicts appear as HTML, PDF, and a host of nonstandard, undocumented flat file formats. The longest ABET extraction script to date is a mere twenty-four lines, which we attribute to the fact that the ABET document model matches the data so well. The results of our extraction are shown in Table 1, which indicates for each language the number of English words for which at least one translation has been extracted; many are languages that have no commercial MT available, and no E.U. or U.N. coverage.

ABET is difficult to evaluate empirically, because it is a programming language rather than a utility. For example, extraction accuracy is a property of a particular ABET program, not of the ABET language. Our subjective evaluation is that ABET allows the scriptwriter to quickly capture the structure of a source, and to identify and correct common errors in the data; this rapid turnaround induces us to use ABET whenever we locate a new on-line bidict.

ABET is important for four reasons:

1. **Lexical coverage is critical.** There is a growing consensus in the CLIR community that lexical coverage is the most important factor in CLIR performance. Thus, tools that increase lexical coverage are desirable.
2. **Resources are scarce.** Machine-readable bidicts, parallel corpora, and MT systems are all rare. ABET broadens this set of resources to include human-readable on-line bidicts.
3. **Extraction is tedious.** ABET automates a significant portion of the extraction process, and allows the user to decompose extraction into simple pattern-matching steps.
4. **Data are restricted.** ABET allows resource sharing without risking copyright infringement—the tool itself is shared, but the data are not. Individual sites may then apply the tool to those resources to which they have access.

Future work includes updating the system to handle Unicode (once Unicode support is integrated into Perl) and integrating text extraction from formats such as PDF. ABET is publicly available at <http://www.csee.umbc.edu/~mayfield/ABET/>.

4. References

- [1] Borthwick, A., Sterling, J., Agichtein, E. and Grishman, R. ‘Exploiting diverse knowledge sources via maximum entropy in named entity recognition.’ *Sixth COLING-ACL Workshop on Very Large Corpora*. 1998.
- [2] Freitag, D. and McCallum, A. ‘Information extraction with HMMs and shrinkage.’ *AAAI-99 Workshop on Machine Learning for Information Extraction*. 1999.
- [3] Muslea, Ion. ‘Extraction patterns for information extraction tasks: A survey.’ *AAAI-99 Workshop on Machine Learning for Information Extraction*. 1999.

Table 1. Dictionaries built using ABET

Lang	#Words	Lang	#Words	Lang	#Words
Afrikaans	6499	Gaelic	1218	Portuguese	31575
Albanian	1319	German	94901	Romanian	2305
Bulgarian	2156	Greek	8647	Russian	31983
Catalan	1142	Hungarian	63940	Slovak	1221
Chinese	1488	Icelandic	2134	Spanish	25028
Danish	5371	Indonesian	23376	Swahili	1031
Dutch	15591	Italian	18461	Swedish	25372
Finnish	3063	Japanese	50404	Tagalog	1438
French	23332	Latin	10033	Turkish	2495
Frisian	4727	Norwegian	6655	Turkmen	8928