Lingloss

Welcome to the Lingloss project page!

In 1967, I designed what was meant to be an international auxiliary language called Lingloss. Like many other such projects, it was never really ready enough to release upon the public. In 2012 Lingloss still remains a work in progress; however, I believe I have recently made some progress on one aspect of the overall problem. The reasons for this belief are more fully detailed at
http://www.richardsandesforsyth.net/docs/bunnies.pdf
.

Through this webpage I share some software which, when more fully developed, may help designers of the coming international auxiliary language (yes, there will have to be one eventually: the human race must grow up some day!) to overcome an obstacle that prior efforts have never convincingly overcome, namely the problem of establishing a suitable core vocabulary.

What you will find when you unzip
[glossoft.zip]
is a pair of programs written in Python3 (along with various ancillary files) which address the following aspects of the vocabulary-building problem:

1. How to choose a core collection of lexical items, i.e. what Hogben (1963) calls a "list of essential semantic units" (LESU), which is concise enough to be learnt in a matter of weeks and at the same time extensive enough to support the great majority of essential communicative functions;
2. How to choose a suitable international word for each of the items in the LESU.

Towards a Core Vocabulary

The program corevox1.py takes in several lists of essential semantic units (formatted one item per line) and produces a consensus list consisting of all the items that occur in at least minfreq of the input lists, where minfreq is an integer from 1 (in which case the output is all the items that occur in any of the input lists) to N, the number of input lists (in which case the output is only those items common to all the input lists).

Where do the input lists come from? Well, to test the program, four files containing previous attempts to come up with a LESU are provided (baslist, hoglist, longlist and maclist). These are, respectively: the Basic English wordlist (Ogden,1937); the LESU of "Essential World English" (Hogben, 1963); the defining vocabulary of the Longman English Dictionary (Longman, 2003); the defining vocabulary of the MacMillan English dictionary for advanced learners (MacMillan, 2002). [subfolder: lexicons]

Ogden and Hogben were trying to establish minimal subsets of words needed for the majority of communicative purposes in simplified versions of English. Compilers of

the Longman and MacMillan dictionaries were trying to establish basic word lists in terms of which all the other entries in their dictionaries could be defined. Thus all four lists represent principled attempts to create concise but effective vocabularies. They didn't all settle on the same words, but any term that appears in more than one of these sets is likely have a strong claim for inclusion in anyone's core vocabulary.

Note that, although most of the entries in these lists are relatively common, they are not mere frequency lists. They result from attempts to cover the most commonly used concepts without redundancy. Therefore some high-frequency terms will be excluded if they are redundant.

I should perhaps apologize for anglocentric bias here; although in mitigation it should be noted that there is nothing in this software that limits it to the English language. I am most at home with English examples, but I would hope that others could apply the same methods to other languages: the comparisons would be instructive.

Towards an International Vocabulary

The second program, avwords3.py, is more innovative, as far as the field of interlinguistics is concerned. It finds the 'verbal average' of a number of different words. As far as I know, nobody has ever defined what a verbal average might be; so, to be a little more specific, the heart of this program is a function that takes in a number of strings (usually words, though they could be short phrases) and produces a string which is, in a certain sense, the most typical representative of those input strings. As currently implemented, it works in 2 stages. Firstly, using a string-similarity scoring function, the string in the group which is most similar to all the others of that group is chosen. Secondly, certain manipulations, such as dropping a character or swapping 2 adjacent characters, are tried to see if they increase the similarity score of that string in relation to the rest and, if so, the modified string is accepted.

For example, given the following inputs

    ['cheval', 'caballo', 'cavallo', 'cavalo', 'cal', 'equus', 'cavall']

which are the French, Spanish, Italian, Portuguese, Romanian, Latin and Catalan words for 'horse', the program computes that

    'cal'

is the most central or typical item. In this case, no deletions or letter-exchanges make it more typical, so it is retained.

The program works by reading in several (utf8) files in the format exemplified below.

```
young      giovane
you    voi
yes    sì
yellow      giallo
year        anno
would       sarebbe
```

```
work        lavoro
word        parola
wool        lana
wood        legno
woman       donna
with        con
wire        filo
wing        ala
wine        vino
window      finestra
```

This is an extract from a simple English-Italian lexicon: each line consists of a source-language term followed by a target-language equivalent, with tab character separating them.

Each of these input lexicons uses the same source language (English in the examples provided) with a different target language (various Romance languages in the examples provided). These sample bilingual lexicons can be found in the **lexicons** folder after you have unzipped the software.

Incidentally, the part that hasn't been automated is going from the LESU produced as output by corevox1.py to the several lexicons needed as input by avwords3.py. There are lots of public-domain bilingual lexicons, so it would be possible to write software that took a LESU and an existing lexicon (English-to-target-language in the present case) and produced suitable input for avwords3.py, but to do it properly would, I suspect, require human scrutiny anyway, so that task is left as "an exercise for the reader".

The output of avwords3.py is a lexicon in the same format as the inputs, where each source-language item is associated with the 'verbal average' of the terms in the various target languages -- intended as a first approximation to an English-Lingloss dictionary. Example output produced from the seven small example inputs in the lexicons folder follows below.

```
Mon Dec 24 16:28:24 2012
window      fenestra
wine  vin
wing  ala
wire  fil
with  con
woman       mulier
wood  lea
woods       bos
wool  lana
word  parala
work  trabaar
would       voudrais
year  ano
yellow      gallo
yes   si
you   voi
young       jove
```

On the basis of the example data provided here, Lingloss, if it ever gets into circulation, would look very much like a Romance language, a kind of simplified, modernized Latin. However, that decision is by no means set in stone. The main point of computerizing parts of the process is to permit exploration of alternative design decisions.

The English word 'would' isn't expressed by a single word in these languages, thus illustrates the need for human pre-processing or post-processing. In fact, avword3.py also produces a listing file in which the quality of the 'verbal averages' is shown. This is meant to provide serious users with information to enable them to decide which of the proposed term equivalents need further attention.

These programs are prototypes, intended to illustrate a particular methodology, which I believe is novel. Much work remains to be done. For example, comparison of alternative string-similarity scoring functions would be a good idea; as would a test of whether each target word should be rendered into a common phonetic representation or just taken as spelled; and so on. The main point is to stimulate such work.

Running the programs

To execute the programs you will have to obtain Python (version 3 not 2) if you don't already have it. This can be found at

www.python.org

I have tested these programs under Windows7, but I believe they should run without alteration under Linux as well.

Then you will have to unzip the file

glossoft.zip

preferably at your top-level directory. This will have subfolders

lexicons	sample LESUs and small-scale bilingual lexicons
libs	common routines and variables for the programs in p3
op	default directory to receive output
p3	Python3 programs
parapath	directory to hold parameter files

Each program requires certain input parameters, which are put into a text file that can be edited by Notepad, Notepad++ or other text editors. Example parameter files for using the example data provided will be found on the parapath folder once the zipped file has been unpacked. Each line of a parameter file starts with a parameter name then one or more spaces then the value for that parameter. Unknown parameters are ignored. Parameters not given a value in the parameter file receive a default value.

A table of parameters used by the programs follows.

parameter name	type	default	description
casefold	0 .. 1	1	whether to fold uppercase to lower case on input; 1 implies yes, 0 implies no.
jobname	alphanumeric string	same name as program	name to link output files
minfreq	integer	2	minimum number of input LESU files in which a term must appear in to be kept for output
outgloss	Windows or Linux file-spec	avwords_glos	output file for consensus lexicon
vocfile	Windows or Linux file-spec	corevox_vocs	output file for consensus LESU
voclists	Windows or Linux file-spec	lesu.dat / lexicons.txt	input text file containing list of input file-specs, 1 per line
withkey	0 .. 1	0	whether to include the source-language term along with the target-language equivalents in avwords (1), or not (0)

The content of coretest.txt, a simple initial parameter file for corevox1.py, is copied below.

```
voclists  c:\glossoft\parapath\lesu.txt
vocfile  c:\glossoft\op\corelist.txt
minfreq  2
```

The content of wordavs.txt, a starter parameter file for avwords3.py, is copied below.

```
voclists  c:\glossoft\parapath\glossies.txt
outgloss  c:\glossoft\op\glossout.txt
withkey  0
```

Pretty simple, eh?

References

Hogben, L. (1943). *Interglossa*. Harmondsworth: Penguin Books.

Hogben, L. (1963). *Essential World English*. London: Michael Joseph Ltd.

Longman (2003). *Dictionary of Contemporary English*. Harlow: Pearson Educational Ltd.

Macmillan (2002). *MacMillan English Dictionary for Advanced Learners*. Oxford: MacMillan Education.

Ogden, C.K. (1937). *The ABC of Basic English*. London: Kegan, Paul, Trench, Trubner & Co. Ltd.

Appendix

Constructed Auxiliary Languages :

Year	Language	Surname	Forename(s)
1661	Universal Character	Dalgarno	George
1668	Real Character	Wilkins	Bishop
1699	Characteristica Universalis	Leibniz	Gottfried
1765	Nouvelle Langue	de Villeneuve	Faiguet
1866	Solresol	Sudre	Francois
1868	Universalglot	Pirro	Jean
1880	Volapuk	Schleyer	Martin
1886	Pasilingua	Steiner	Paul
1887	Bopal	de Max	Saint
1887	Esperanto	Zamenhof	Lazarus
1888	Lingua	Henderson	George
1888	Spelin	Bauer	Georg
1890	Mundolingue	Lott	Julius
1892	Latinesce	Henderson	George
1893	Balta	Dormoy	Emile
1893	Dil	Fieweger	Julius
1893	Orba	Guardiola	Jose
1896	Veltparl	von Arnim	Wilhelm
1899	Langue Bleu	Bollack	Leon
1902	Idiom Neutral	Rosenberger	Waldemar
1903	Latino sine Flexione	Peano	Giuseppe
1906	Ro	Foster	Edward
1907	Ido	de Beaufront	Louis
1913	Esperantido	de Saussure	Rene
1922	Occidental	de Wahl	Edgar
1928	Novial	Jespersen	Otto
1943	Interglossa	Hogben	Lancelot
1944	Mondial	Heimer	Helge
1951	Interlingua	Gode	Alexander
1957	Frater	Thai	Pham Xuan
1961	Loglan	Brown	James
1967	Lingloss	Forsyth	Richard
1983	Uropi	Landais	Joel
1996	Unish	Jung	Young Hee
1998	Lingua Franca Nova	Boeree	George
2002	Mondlango	Yafu	He
2011	Angos	Wood	Benjamin