Emdros HAL example (Hyperspace Analogue to Language)

Ulrik Petersen

January 26, 2008

Contents

1 Introduction

Welcome to the Emdros HAL example. Here you will find information on:

- What is a HAL space?
- What does the example do?
- Running the example

2 Part I: Preliminaries

- What is a HAL space?
- What does the example do?

2.1 What is a HAL space?

2.1.1 Background

HAL stands for:

Hyperspace **A**nalogue to **L**anguage

It was invented by a research group at University of California at Riverside. It has a homepage here.

HAL is a numeric method for analysing text. It does so by running a *sliding window* of fixed length across a text, calculating a matrix of word cooccurrence values along the way.

2.1.2 Informal definition

A HAL space is an QxQ matrix of integers, where Q is the number of distinct word-forms in the text. For example, if the text contains 50,000 running words, of which 6,000 are unique forms, then the corresponding HAL space will be a 6,000x6,000 matrix.

Each entry in the matrix is a sum of all values arising from running the sliding window over the text.

The sliding window is n words wide, e.g., 8.

Then, for a given word, say, the one at index t, a value is added to the matrix for each of the word pairs which arise by pairing the word at index

t

with each of the words at indexes

t-n to t-1 .

This basically means the n words before the one at index t are paired with the word at index t.

For each of these word-pairs, a value is calculated as follows: If the word before t is at index j (e.g., t-3), then the value is

n - $|t$-$j|$ + 1

For example (t=10, j=10-3=7):

8 - |10 - 7| + 1 = 6

This basically means that words close to t get a higher score than words farther away.

If the word-form corresponding to t has number p in the HAL-matrix, and the word-form corresponding to j has number q in the HAL-matrix, then this value is added to the p'th row and the q'th column.

2.1.3 Mathematical definition

Assume or define the following:

- Assume that the words are numbered 1..M.
- The words themselves will be called T(1), T(2), ..., T(M) for easy reference.
- Assume that there are Q unique forms. Then the forms will be called F(0), F(1), F(2), ..., F(Q-1).
- Assume that the function Form(n) maps word-indexes in 1..M to form-indexes in 0..Q-1. For example, if T(2) has the form F(10), then Form(2) = 10.
- Assume that the window-width is n.
- Define the delta function D(a,b) as follows: D(a,b) = n - |a-b| + 1
- Assume that Matrix[0..Q-1][0..Q-1] is the HAL matrix, and that all values are initially 0.

The HAL space is calculated as follows:

1. Initialize all matrix-cells to be 0.

2. For each word-index i in the range 2..M:

   (a) For each word-index *i2* in the range *i*-n..*i*-1 (the lower bound of this range must be adjusted if there is no such word, e.g., if n=4 and i is 2, the first value will be 1, not -2):

      i. Let d = D(i2,i)

      ii. Add d to the HAL matrix's Form(*i*)'th row and Form(*i2*)'th column. I.e., add d to the existing value of Matrix[Form(i)][Form(i2)].

2.2 What does the example do?

2.2.1 Overview

The example can do these things:

1. It can load an Emdros database with a text, initializing the database for use with the example.
2. It can build HAL-spaces of the in the database
3. It can load a previously generated HAL space from the database
4. It can emit two kinds of output:

   (a) A complete HAL-space as a Comma-Separated-Value (CSV) text file suitable for loading into a spreadsheet.

   (b) Certain parts of a HAL-space, based on certain word-forms that are of interest.

2.2.2 What output does it give?

- For building the database:
  - A loaded Emdros database
  - A list of word-forms found in the text (the Q word forms spoken of on the HAL space page).
- For querying the database:
  - Optionally, a Comma-Separated Value (CSV) file containing the entire HAL-space, suitable for loading into a spreadsheet.
  - An output file with data for words which you are specially interested in.

    For each word, a list is given of the words with which it cooccurs most frequently and closely.

    If the word form you are interested in is w1 and the word form with which it cooccurs is w2, then the score is calculated as Matrix[w1][w2] + Matrix[w2][w1].

    This score is printed twice: First, it is printed in a scaled form. The score is multiplied by a user-specified factor and divided by the text

length. And second, it is printed in its raw form, as it came from the matrix.
The list is sorted, so that the "heaviest" words come first. The user can specify how many words to put in the list.

2.2.3 What input does it need?

1. For loading the database: Only a text file.
2. For querying the database: A configuration file with a special format.

2.2.4 What is a configuration?

A configuration file is a plain text file which looks like a Unix configuration file, and holds information necessary for running the HAL example. See the later page for its format.

2.2.5 What information does an input file contain?

An input file contains the following:

- The **database name** which holds the text to analyze.
- The **sliding window width**, *n*.
- The name of the **CSV-file** containing the HAL-space in a CSV-format, suitable for reading into a spreadsheet. ("none" if you don't want a CSV file).
- The name of the **output file** containing the output for the words you are interested in.
- The **words you are interested in**.
- The **maximum number of values** for each word you are interested in.
- The **factor by which to multiply** each value for a given word. This is first divided by the number of words in the text. This can come in handy if you wish to compare texts of different lengths.

3 Part II: Running

The example can be run as follows:

1. Building the database
2. Querying the database which has two sub-parts
   (a) Writing the input file
   (b) Running the example

3.1 Building the database

3.1.1 How

The database needs only be built once and for all. To do so, run the

```
halblddb
```

command-line program with the right options.

3.1.2 halblddb usage

To see the halblddb usage, run the following command:

```
halblddb --help
```

This displays the following output:

```
halblddb version <version-number> on <backend-name>
Usage: halblddb [options]
OPTIONS:
   -d , --dbname db     Use this database
   -f textfilename      Use this text input file
   -o wordlistfilename  Use this wordlist output file
   --help               Show this help
   -V , --version       Show version
   -h , --host host     Set hostname to connect to
   -u , --user user     Set database user to connect as (default: 'emdf')
   -p , --password pwd  Set password to use for database user
```

3.1.3 Example

Assume that you want the following:

- Build a database called **hal_test**
- using an input-file called **mytext.txt**
- writing a word-list to a file called **wordlist.txt**

And assume the following (not necessary if using SQLite):

- You have set up your backend database with the **user** called **emdf**
- having a **password** called **changeme**.

The the following command-line would do it:

```
halblddb -d hal\_text -f mytext.txt -o wordlist.txt -p changeme -u emdf
```

That's it.

3.2 Querying the database

Having built the database, your next step is to query it in interesting ways.

There are two steps to this:

1. Writing the input file
2. Running the example

3.2.1 Writing the input file

Overview The format of the file is much like a Windows .ini file or a Unix configuration file:

- The file contains a number of *key=value* pairs.
- Lines beginning "#" are ignored (i.e., are comments).

Self-documenting example The following is a self-documenting example.

Please refer back to the "What does the example do?" page for a description of each of the following settings.

```
\# The db name
database = hca

\# The factor by which to multiply each value in the output.
\# This is divided by the text length, so for example if your text
\# has 70000 words, 100000 would be a good value
factor = 100000

\# The window width
n = 5

\# The csv output filename
csv = haltest.csv.txt

\# The value-vector output filename
output = haltest.out.txt

\# The maximum number of values in each value-vector
max\_values = 20

\# Place the words here, with one 'word' key per line, e.g.:
word = mor
word = barn
word = blomst
```

3.2.2 Running the example

How First, open a command-line prompt.
Then follow this schema:

```
mqlhal -c myconfigfile.cfg
```

If on MySQL or PostgreSQL, you may need to use the "-u username", "-p password", and/or the "-h hostname" options.

Example This example is for Windows users.

1. Open a command-line prompt.
2. cd to the right directory, e.g., "C:\Emdros\"

```
C:\>C:
c:\>cd C:\Emdros\
C:\Emdros>
```

3. Run the program. Assume that your configuration file is called "C:\temp\myconfig.cfg"

```
C:\Emdros>bin\mqlhal.exe -c "C:\temp\myconfig.cfg"
```

That's it. Now sit back and watch as the program spits out various messages and runs to completion.

What happens next? Afterwards, it's time to analyze the output. Perhaps you need to adjust the configuration file and rerun the program to get new output. This will likely be an iterative process until you find what you are looking for, or have tested your hypothesis about the text.

4 Part III: References

Here, we have two kinds of references:

1. Bibliography
2. Links

4.1 Bibliography

4.1.1 Articles

- Burgess, C., K. Livesay, and K. Lund (1998). *Explorations in Context Space: Words, Sentences, Discourse*, Discourse Processes, Volume 25, pp. 211 - 257.
  See http://locutus.ucr.edu/abstracts/97-bll-expl.html from where you can also download the paper.

- Burgess, C. and K. Lund. (1997). *Modelling parsing constraints with high-dimensional context space.* Language and Cognitive Processes, Volume 12, pp. 1-34.
  See http://citeseer.nj.nec.com/context/398051/0.

- Lund, Kevin and Curt Burgess. (1996) *Producing high-dimensional semantic spaces from lexical co-occurrence*, Behavior Research Methods, Instruments and Computers, Volume 28, number 2, pp. 203–208.
  See http://locutus.ucr.edu/abstracts/96-lb-prod.html from where you can also download the paper.

4.2 Links

- Official HAL Homepage
- Emdros homepage