LIWC Replication Manual

The source code for this Replication of LIWC 2007 is found at https://github.com/fabianboth/LIWC-Replication. Also a runable compilation of this program is downloadable.

The Java replication of LIWC 2007 uses a LIWC dictionary to classify a given input text into categories. These dictionaries need to be placed in the folder “LIWCDictionaries” in the working directory of the Java application. Hereby, all .dic filenames in that folder are scanned for the keywords “German” and “English” to determine the type of dictionary. The java implementation expects exactly one German and one English dictionary to be found in the given folder, else an error is thrown. If you wish to change the used dictionaries, you need to exchange the dictionaries in that folder. Also important to note is that the .dic files should be encoded in UTF8 format to circumvent unforeseen problems. Those LIWC dictionaries cannot be distributed in the frame of this work and need to be acquired separately.

Further, this implementation generates three times as many threads as available cpu cores for multi-threading and splits the dictionary into several pieces for work distribution. If a machine with more than 16 cpu cores is used, this routine becomes less efficient due to the limited amount of words in such a dictionary and one might wish to alter the multithreading scheme in the source code.

In the source code, a construction class with command line capabilities is available to act as interface to any other program. This constructor class can setup the whole framework needed for analysis and alter the analytic scheme. Using the compiled .jar program, the .jar file can be evoked with command line parameters. For easier access and handling, also a .bat file has been added, which can be run directly. For custom analysis and definition of input and output files, the command line parameters within the .bat file can be altered with a simple text editor. Following the function of the different command line parameters is explained. Parameters three to six are optional and don’t need to be defined. But if for example parameter four should be defined, parameter three needs to be defined, too. All input parameters are read sequentially and hence no parameter can be skipped.

Command Line Arguments:

[1] **full pathname** of .txt file to analyze or folder path of files to analyze. If the input parameter is a folder path, all .txt files within this folder and subfolders are analyzed

[2] **path to write** the "result.txt" file

[3] (optional, default "en") **language mode** determines the dictionary to be used; possible input: "auto", "en", "de" - note if auto is used, make sure both dictionaries (en, de) are compatible (year 2001)

[4] (optional, default "off") **replication mode**: If set to "on", an improved algorithmic version will be used - if set to "off" the closest replication of LIWC's behavior is used

[5] (optional, default "off") **line mode**: If set to "on", each line in a text file will be analyzed separately, indicated by “\n”. If a line is empty it will be skipped.

[6] (optional, default "off") **umlaut mode** (only use if the used dictionaries have been adapted adequately!): If set to "on", all umlauts in a text file are replaced by their alternative form - i.e. "ö" will be replaced with "oe".

Following, examples for possible command line arguments are provided to clarify the usage of those. These can either be defined in the .bat file or directly in the command console. Also the constructor class in the source code can be delivered with a String array containing those parameters directly.

Examples:

(1) ["C:\test.txt" "C:\" en off off off]
In this example the text file test.txt will be analyzed with the English dictionary and results will be stored to “C:\” in the file results.txt. This input setting is equal to [javaw -jar LIWCReplication.jar "C:\test.txt" "C:\"] utilizing only default values.

(2) ["C:\test.txt" "C:\" de off on off]
Here, the same file will be analyzed with the German LIWC dictionary, whereby all lines within the text file are investigated separately

(3) ["C:\test.txt" "C:\" auto on]
In this example, the text file is analyzed with automated language detection, which means language is determined beforehand analysis by the program. Also the improved algorithm is utilized in the fourth input parameter. Parameters five and six are not mentioned and will be initialized with their default values.

(4) ["C:\files" "C:\" en on]
With this command parameter setup, all .txt files that are found in “C:\files” and its subfolders will be analyzed. Therefore the English dictionary will be used and the enhanced analysis algorithm will be used