Journal of Contemporary Eastern Asia Vol. 16, No. 2: 110-144
DOI: 10.17477/jcea.2017.16.2.110

Segmenting Chinese Texts into Words for Semantic Network Analysis

James A. Danowski[1]

Unlike most languages, written Chinese has no spaces between words. Word segmentation must be performed before semantic network analysis can be conducted. This paper describes how to perform Chinese word segmentation using the Stanford Natural Language Processing group's Stanford Word Segmenter v. 3.8.0, released in June 2017.Keywords: Transnational Studies, Social capital, Online Sharing, Sender-effect, Young adults

Problem Statement

Semantic network analysis of texts has become widely practiced since its beginnings some 40 years ago (Danowski, 1982; 1993a; 1993b; Carley, 1993). A search of Google Scholar on November 28, 2017 using the term "semantic network" found 88,500 hits since 2013. One reason for this popularity is ease of processing. Most languages place a space between written words. This enables direct computation of semantic networks, based on proximate word cooccurrences using a window 7 words wide, centered on each word in the text, yielding words as vertices and edges between words as links, weighted by their frequency or other variables (Danowski, 1993b, 2009).

Nevertheless, Asian languages, such as Chinese, Japanese and Myanmar, do not separate words by spaces. This makes semantic analysis difficult because continuous strings of characters are separated only by commas and periods. As a result, one must segment the continuous texts of these languages into isolated words, a prerequisite for many natural language processing applications.

Accordingly, semantic network analysis of Chinese has lagged. One probable cause is that machine segmentation of words was a much more difficult problem to solve than creating algorithms and tools for semantic network analysis such as WORDij (Danowski, 1993a) and others (see Yang, S & González-Bailón, 2017; Lewis, Carley, & Diesner, 2016).

[1] Department of Communication, University of Illinois at Chicago

Earlier work combining word segmentation and semantic network analysis (Yuan, Feng, & Danowski, 2013) used a segmenter not readily available on the web. It required use of the python programming language, leaving out researchers not literate in it. This article seeks to widen accessibility of Chinese word segmentation tools for communication and other social scientists. It covers how to use the Stanford Word Segmenter, originally released in 2006, with version 3.8.0 released in June 2017, to prepare Chinese texts for semantic analysis (https://nlp.stanford.edu/software/segmenter.shtml).

Stanford Word Segmenter

The Stanford Word Segmenter is an ongoing product of the natural language processing group at the university. Using it, researchers can avoid reinventing tools that already exist or one-off alpha-level programs written specifically for the paper that describes its first use (see for example, Pei, Ge, & Chang, 2014; Wang, Utiyama, Finch, & Sumita, 2014).

The Stanford Word Segmenter is based on conditional random fields (CRF) models (Tseng, Chang, Andrew, Jurafsky and Manning, 2005). CRFs are a member of the sequence modeling family, and are a specific class of Hidden Markoff Modeling (HMM). A CRF is a probabilistic model, analyzing distributions of linear character chains to predict sequences of labels for sequences of input samples (Lafferty, McCallum, & Pereira, 2001). Peng, Feng, & McCallum (2004) first used this framework for Chinese word segmentation, where each character is labeled either as the beginning of a word or a continuation of one.

Features

Three categories of features used are: 1) character identity n-grams, 2) morphological features, and 3) character reduplication features. For each state, the character identity features (Ng & Low 2004, Xue & Shen 2003, Goh et al. 2003) are represented using feature functions that mark the identity of the character in the current, proceeding and subsequent positions.

Specifically, the Stanford Word Segmenter uses four types of unigram feature functions, labeled C0 (current character), C1 (next character), C-1 (previous character), C-2 (the character two characters back). Furthermore, four types of bi-gram features are used: C0C1, C-1C0, C-1C1, C2C-1, and C2C0. The result is highly valid segmentation of character strings into words.

Additional Pre-Processing

Once words are identified, researchers often perform additional preprocessing. Carley, 1997a) suggests a basic preprocessing decision is whether to either eliminate whole categories of words, as do Corman et al (2001) and whether to translate specific words and phrases into more basic concepts (Carley, 1997b). Translation, for example, may include building ontological categories

by combining words or word pairs to form them. Another way is to make an include list of words/nodes that are the only ones the researcher is interested in, and compute the semantic network using only these terms, For example, Danowski and Cepela (2010) analyzed the White House member networks of the Nixon through Obama administrations the by way of co-appearances in news stories of White House cabinet members, the president, vice president, and key staff, using an include list of names and aliases and using the WORDij option of the "include list", the opposite of a stop-word list. Such moves enable the analyst to capture textual features of interest, in this case, the measurement of networks among key members of the administration.

Other pre-processing techniques include stemming words to their roots, although we find this reduces validity of optimal messages (Danowski, 1993b). A common pre-processing is to use a *stop word list* or *drop list* to remove function words that have no intrinsic meaning, articles, transition terms, indirect references or proper nouns, and text markup language. Carley (1997b) points out that a thesaurus can be used to collapse words sharing the same general meaning into higher-order concepts. For example, in a thesaurus the words "macaroni" and "spaghetti" might both be matched to the higher-level concept "pasta."

Sometimes, an opposite approach is useful, to take a core list of concepts and expand them by adding additional words from a thesaurus. For example, we wanted to create a method for converting open-ended text comments describing a fixed choice set of items. The NESSE program of the U.S federal government seeks to measure student engagement using fixed-choice scales. In a work in progress, we took all the main words from each of the questionnaire items and wrote a term expansion program to extract synonyms and antonyms from the Wordnet thesaurus. This expanded list was then used to encode open-ended comments about engagement aspects, to capture relevant text from student responses. Finally, we performed regression classification analysis to calibrate the fixed-choice method with automatic coding of the openended responses to measure engagement. In further studies, engagement would be automatically indexed by the open-ended responses, enabling removal of the fixed-choice questions.

Another example of inserting a mission-specific include list for semantic network analysis is to use an include list generated from an expert crowd. In an online survey, we obtained data from 437 members of the Public Relations Society of America. One question asked them to list their key publics. We compiled an aggregate list of publics across all respondents, then edited the list to remove duplicates or alternative spellings. This became our master publics list that could be used to assess any organization's publics network. One may download corporate annual reports, then do a semantic network analysis with the include list of publics (Danowski, 2012b). If the same text sources are used as input, this provides a valid way to measure differences in the networks of publics each organization includes in their reports.

These variables can be analyzed with other variables to test hypotheses.

Post processing can improve the specificity of the semantic network analysis to match the specificity of the question. For example, Danowski & Park (2014) took the aggregate semantic

network from documents retrieved containing the term "jihad" in the 47 Muslim-majority countries' web documents and news posts before and after the Arab Spring events in Egypt, then focused in on the term "jihad" and all other words up to three link-steps away, and computed metrics on the semantic network structure to show that Arab Spring was immediately followed by increased centrality of radical jihadist ideological word pairs as Iran, already highly central in the inter-Muslim country network, become even more central (Danowski and Park, 2013).

Other post-processing techniques include aligning the semantic network data with other data. One example linked change over time in the semantic networks of news posts in relation to ego-centric network structures based on national mobile metadata (Danowski, 2017). Another is finding the strong correlations between sentiment word pairs and a dependent variable such as ego-centric network density, then inputting only these word pairs into further semantic network procedures to find optimal messages from weighted shortest paths between seed and target words using WORDij's OptiComm program (Danowski, 2017b).

Other post-processing moves are to analyze relationships between semantic network variables and characteristics of the social units that produced the content. For example, one can test hypotheses that include semantic network variables. Zywica and Danowski (2008) investigated differences in Facebook users' semantic networks for the term "popularity" in relation to psychological variables.

With time-series semantic network data, one can flip from the network of words to the network of time points, allowing the investigator to find the most central time periods and examine the word pairs it contains that are significantly more frequent that at other time periods. This turning of time outside-in reveals pivotal time points in a series and why they are such by the content they contain (Danowski, 2012a).

Data

Nexis provided the source "Xinhua_News_in_Chinese_for_Overseas_Service_2017." The search term was the Chinese characters for North Korean President Kim Jong-un (KJU): 金正恩. All available dates were selected. Documents returned numbered 114 after duplicate removal. The documents were downloaded as text files which we converted to UTF-8 format using Notepad in Windows. Next, we ran our DeDup program that removes duplicates that may not have been caught by Nexis, and more importantly, that strips the document of meta-data tag constants such as DATE, HEADLINE, BODY, etc. that if left in would damage the internal validity of the probabilistic semantic network analysis.

The next step is segmentation of the file. The Stanford Segmenter 3.8.0 software was run on this file. Figure 2 shows a news document in unsegmented and segmented form. One can observe the spaces that now separate words in the right-side column.

Figure 2. Unsegmented and Segmented News Document

October 21, 2017 Saturday 1:05 PM GMT 眾院明大選 安倍誓不讓人民受北韓威脅

（法新社東京 21 日電）

日本眾議院大選明天登場，在競選活動最後一日，首相安倍晉三矢言保護民眾遠離北韓威脅。北韓威脅議題是這次大選的主軸。

颱風來襲之際，安倍在東京市中心告訴冒雨參加活動的群眾，這場大選非常艱困。

他說：「執政聯盟...是唯一可以保護人們生命和捍衛我們幸福生活（的選擇）。」安倍此言顯然指涉兩度試射飛彈飛越日本領空，並揚言擊沈日本的北韓政權。

民調顯示，安倍領導的自由民主黨可望贏得明天大選，讓他對北韓的強硬立場與「安倍經濟學」成長策略獲得選民新授權。

安倍面對北韓威脅採鷹派立場，強調應對金正恩政權施加最大壓力，並表示支持美國立場，「所有選項」都可以討論。

在短短 12 天競選活動期間，安倍持續譴責平壤當局，最近還對群眾說：「在像這樣的時刻...我們不能動搖。各位，我們絕對不能屈服於北韓的威脅之下！」（譯者：中央社陳妍君）

October 21 , 2017 Saturday 1:05 PM GMT 眾院明大選安倍誓不讓人民受北韓威脅（法新社東京 21 日電）日本眾議院大選明天登場，在競選活動最後一日，首相安倍晉三矢言保護民眾遠離北韓威脅。北韓威脅議題是這次大選的主軸。

颱風來襲之際，安倍在東京市中心告訴冒雨參加活動的群眾，這場大選非常艱困。他說：「執政聯盟...是唯一可以保護人們生命和捍衛我們幸福生活（的選擇）。」安倍此言顯然指涉兩度試射飛彈飛越日本領空，並揚言擊沈日本的北韓政權。

民調顯示，安倍領導的自由民主黨可望贏得明天大選，讓他對北韓的強硬立場與「安倍經濟學」成長策略獲得選民新授權。

安倍面對北韓威脅採鷹派立場，強調應對金正恩政權施加最大壓力，並表示支持美國立場，「所有選項」都可以討論。

在短短 12 天競選活動期間，安倍持續譴責平壤當局，最近還對群眾說：「在像這樣的時刻...我們不能動搖。各位，我們絕對不能屈服於北韓的威脅之下！」（譯者：中央社陳妍君）

Semantic Network Analysis

Now the segmented file is ready for semantic network analysis using a tool such as WORDij. Here is a short WORDij description to give readers, who are unfamiliar with it, an idea of its modules and capabilities: WORDij is a text analysis program that treats words as nodes and word pairs as links for network analysis and other statistical analysis. The software, available at:
(https://wordij.net) runs on Windows PC, Mac and Linux systems and is free for academic use.

WORDij has seven modules:

1. Wordlink: this base module counts words and proximate word pairs and the results are used by other modules.
2. QAPNet: calculates an overall measure of the similarity of two whole networks using a correlation coefficient from +1 to -1.
3. Z-Utilities: compares two text files and finds significant differences in frequencies for either the words or the word pairs. A Z-score and a Chi-Square are computed for the word and word pair comparisons.
4. VISij: graphs visualization of networks of words and links. If multiple files are included an animation will play a network sequence change from one file to another.
5. OptiComm: produces messages that could be used to either promote change to move two words closer, move them further apart, or to reinforce aspects of the semantic networks.
6. Utilities: extracts Proper nouns, and TimeSegs, a program to create time-series files of documents from Lexis/Nexis or NewsBank into sequential files by user selection of intervals of daily, weekly, monthly or yearly.
7. Conversions: converts WORDij files for use with MultiNet/Negopy UCINET, NetDraw, Pajek, and NodeXL. And, it includes a Node(cen)Tric utility to extract all links of a focal node up to 5 steps away and outputs .net file according to a Pajek convention.

Tutorials

Included here are two detailed step-by-step tutorials, one for MACs and one for Windows PCs. First, we present the steps in text form, followed by the insertion of screenshots into the steps.

Installing the Stanford Segmenter and Segmenting Chinese on a PC with Windows 10 Update 1079 Without Screenshots As of October 31, 2017.

1. Check that you have the latest Java version installed.
2. In the Windows search box, type: Configure Java And, press Enter. The Java Control Panel will open.

   Note: As of October 31, 2017, the Java version was 9.01 (build 9.0.1 +11)

3. Click on the Update tab.
4. Click Check Now.
5. If Java needs updating, it will download the latest version. Follow the prompts to install the update.
6. If Java is up-to-date, then a message will appear stating you have the latest Java.
7. If your Java is up-to-date, Skip to step 18 to download and install the Stanford Segmenter for Chinese and Arabic.
8. If Java Control panel does not open, then you need to install the Java Development Kit
   (JDK), which includes the Java Runtime Environment (JRE)

   Note: To run WORDij 64 bit on a PC you need the JDK. Continue to Step 9.

9. Click on the link below to install the Java SE Java Development Kit (JDK), which includes the Java Runtime Environment (JRE)
   http://www.oracle.com/technetwork/java/javase/downloads/index-jsp-138363.html#javasejdk

10. Click on the Download JDK button. This link will take you to the download page:
http://www.oracle.com/technetwork/java/javase/downloads/index-jsp-138363.html#javasejdk

11. Click “Accept License Agreement” and,
12. Download the Windows jdk-9.0.1_windows-x64_bin.exe file.
13. Go to your Downloads folder and click on the file”: jdk-9.0.1_windows-x64_bin.exe. The Java setup begins. Click Next.
14. A Status bar will display the installation progress
15. Click Next.
16. Another Status bar will be displayed.
17. Click Close, after the successful installation.
18. **Download the Stanford Segmenter for Chinese and Arabic.**
    https://nlp.stanford.edu/software/segmenter.html

19. Click the download link
20. Select the 3.8.0 version (or whatever is the most recent version)
21. Unzip the file: stanford-segmenter-2017-06-09.zip (or whatever the most recent version is)..
    Select the file:

22. And, Under Compressed Folder Tools, Click Extract.
23. Click Extract ALL.
24. Select your destination folder, such as, Documents.
25. Click Extract.
26. Review the extracted files.
27. In the Windows search box, type: CMD and, press Enter.

    Note: CMD stands for "Command Prompt." This will start a command Prompt window.

28. Important: Change the Directory to where the folder "stanford-segmenter-2017-06-09" is located

    For example:

    C:\Users\riope>cd C:\Users\riope\Documents\stanford-segmenter-2017-06-09

    Note:

    The generic PC segmenter command syntax is as follows:

    segment.bat ctb|pku] path/to/input.file <encoding> <size> > path/to/segmented.file

Examples: segment.bat ctb test.simp.utf8 UTF-8 0 > textsegment_ctb_out.txt

Or

segment.bat pku test.simp.utf8 UTF-8 0 > textsegment_pku_out.txt

If you place the input file in the “stanford-segmenter-2017-06-09” folder then you do not have to specify the complete file path.

The parameters are: ctb : Chinese Treebank pku : Beijing University path/to/: is the file path input.file: The file you want to segment. Each line is a sentence. encoding: UTF-8, GB18030, etc.

(This must be a character encoding name known by Java) size: size of the n-best list (just put '0' to print the best hypothesis without probabilities). segmented.file: The output file segmented. If no segmented file is given the output is displayed in the terminal screen.

Two segmentation models are provided. The "ctb" model was trained with the Chinese treebank(CTB) segmentation, and the "pku" model was trained with Beijing University's (PKU) segmentation. PKU models provide smaller vocabulary sizes and OOV rates on test data than CTB models.

See README-Chinese.txt more details.

The following are three examples.

29. Example 1:
    The below PC command will display the results in the terminal window.

    segment.bat ctb test.simp.utf8 UTF-8 0

30. Example 2:
    The below command will save the results to the file: textsegment_ctb.txt

    segment.bat ctb test.simp.utf8 UTF-8 0 > textsegment_ctb.txt

31. Example 3:

    The example below uses the full path for each file:

C:\Users\riope\Documents\stanford-segmenter-2017-06-09/segment.bat ctb test.simp.utf8 UTF-8 0 > C:\Users\riope\Documents\stanford-segmenter-2017-06-09/textsegment_ctb.txt

32. Download and unzip the Chinese Stop Words file at:
    https://gist.github.com/dreampuf/5548203

    Unzip the file and add the file extension ".txt"

    This will change the file type from Terminal to Text.

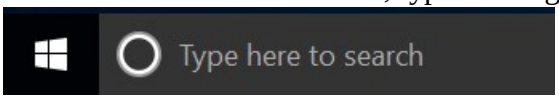
    Use this file as the Droplist with WORDij's WordLink module.

Note: Do NOT check the "Chinese Filter" option. That should be used only if you wish to input a raw Chinese text file, unsegmented, and want to produce all the adjacent pairs of characters.

33. Download WORDij at: https://wordij.net
34. Complete the WORDij download form and click Submit at the bottom of the form.
35. Choose the Windows 64-bit option. You should see the WORDij.zip file as a folder in your Downloads folder. We suggest you move the WORDij folder to your Documents folder as a more permanent file location. Note: a MAC will automatically unzip the folder, which is not the case on a Windows PC.
36. Open the WORDij folder. Locate the WORDij.jar and right click on it, or Control Click.
37. Click Open.
38. WORDij will start and the main menu screen will appear.
39. For technical support, or any other reason, please contact: Email : jdanowski@gmail.com

Installing the Stanford Segmenter and Segmenting Chinese on a PC with Windows 10 Update 1079 As of October 31, 2017 with Screenshots.

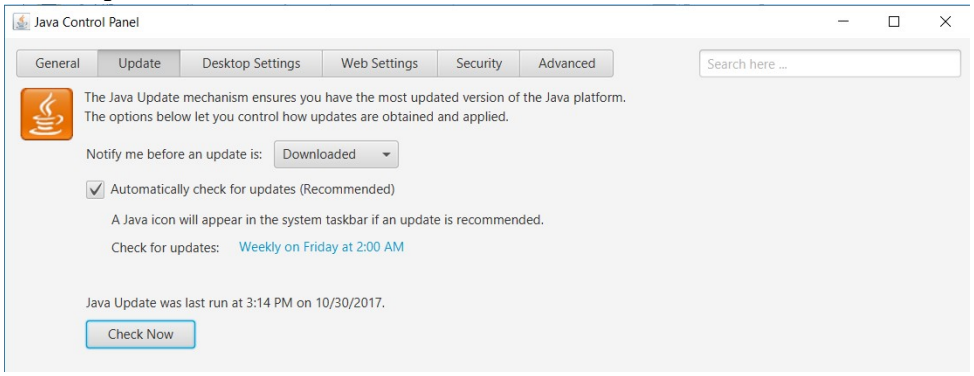
1. Check that you have the latest Java version installed.
2. In the Windows search box, type: Configure Java And, press Enter.



3. The Java Control Panel will open.

   Note: As of October 31, 2017, the Java version was 9.01 (build 9.0.1 +11).

4. Click on the Update tab.



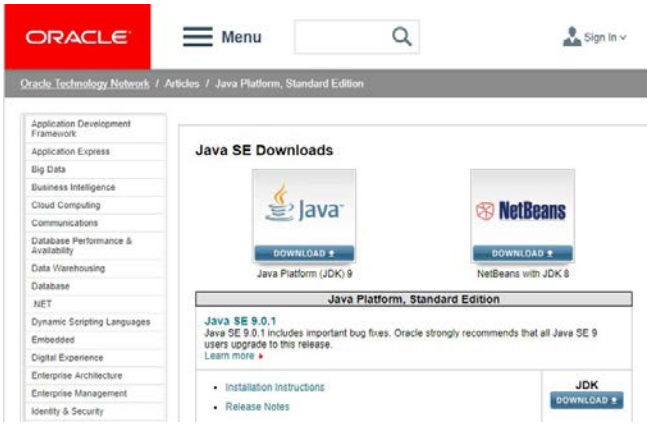
5. Click Check Now.
6. If Java needs updating, it will download the latest version. Follow the prompts to install the update.
7. If Java is up-to-date, then a message will appear stating you have the latest Java.

8. If your Java is up-to-date, Skip to step 21 to download and install the Stanford Segmenter for Chinese and Arabic.

9. If Java Control panel does not open, then you need to install the Java Development Kit
   (JDK), which includes the Java Runtime Environment (JRE)
   Note: To run WORDij 64 bit on a PC you need the JDK. Continue to Step 10.

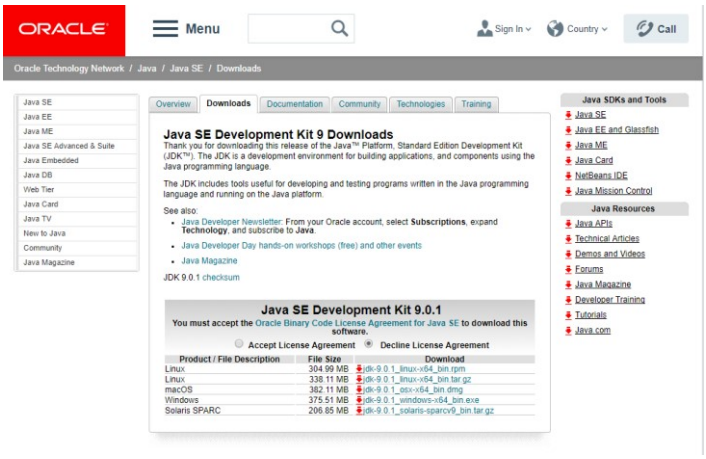
10. Click on the link below to install the Java SE Java Development Kit (JDK), which includes the Java Runtime Environment (JRE)
    http://www.oracle.com/technetwork/java/javase/downloads/index-jsp-138363.html#javasejdk

11. Click on the Download JDK button.



This link will take you to the download page:

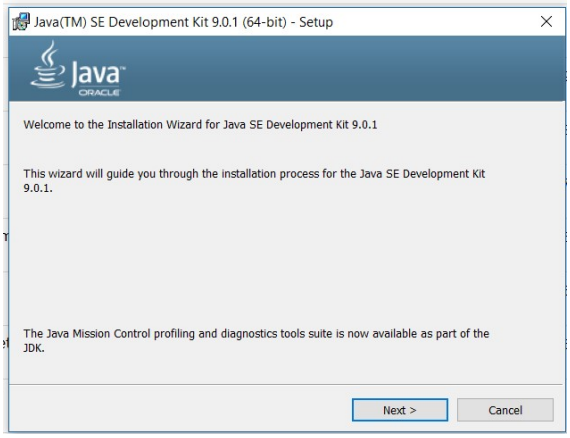
http://www.oracle.com/technetwork/java/javase/downloads/index-jsp138363.html#javasejdk



12. Click “Accept License Agreement” and,
13. Download the Windows jdk-9.0.1_windows-x64_bin.exe file.

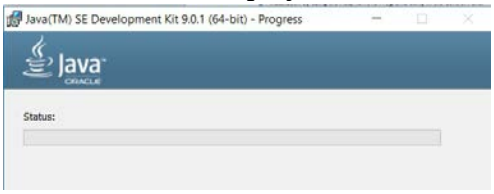
14. Goto your Downloads folder and click on the file”:
    jdk-9.0.1_windows-x64_bin.exe.

15. The Java setup begins. Click Next.

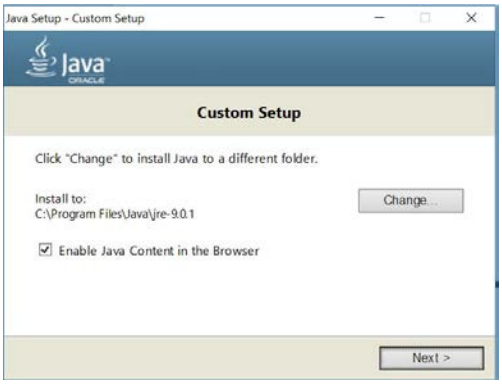


16. Click Next

17. A Status bar will display the installation progress



18. Click Next.

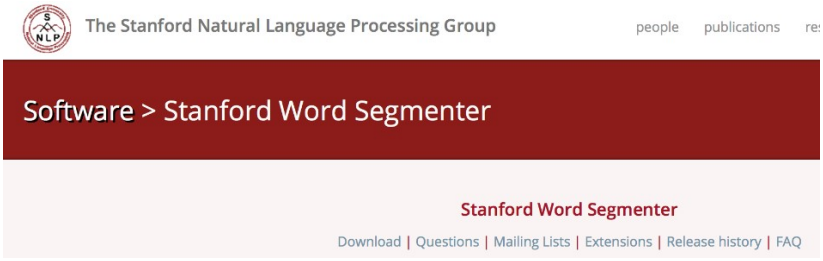


19. Another Status bar will be displayed.

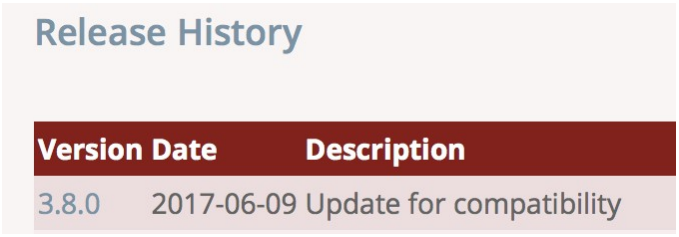
20. Click Close, after the successful installation.

21. **Download the Stanford Segmenter for Chinese and Arabic.**
    https://nlp.stanford.edu/software/segmenter.html

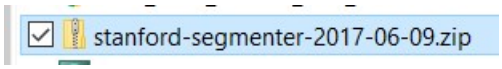
22. Click the download link



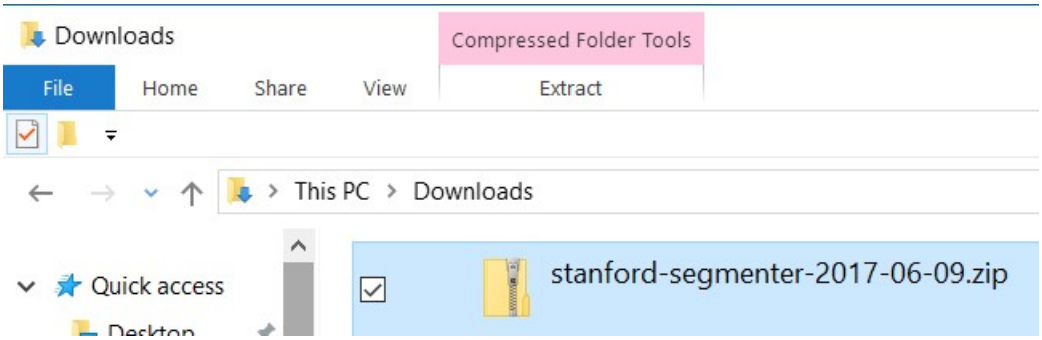
23. Select the 3.8.0 version or whatever is the most recent version)



24. Unzip the file: stanford-segmenter-2017-06-09.zip (or whatever the most recent version is).. Select the file:



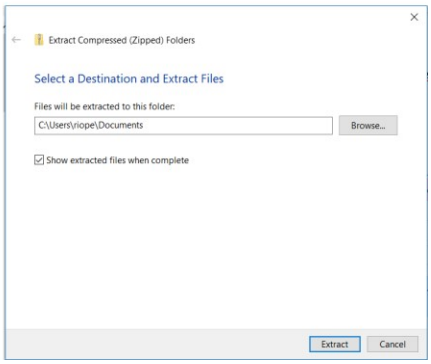
25. And , Under Compressed Folder Tools, Click Extract.



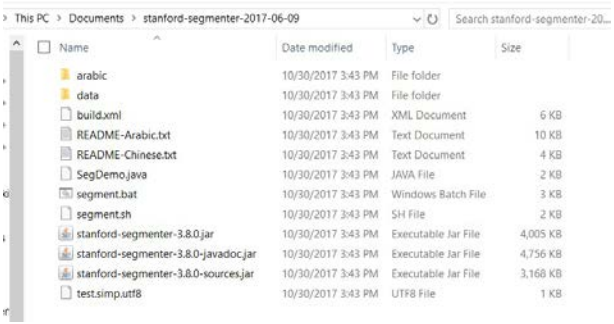
26. Click Extract ALL.

27. Select your destination folder, such as, Documents.

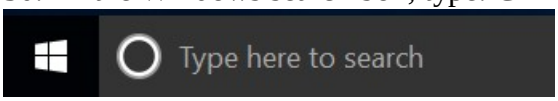
28. Click Extract.



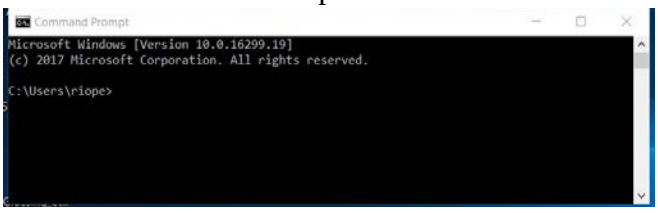
29. Here is a screenshot of the extracted files.



30. In the Windows search box, type: CMD and, press Enter.



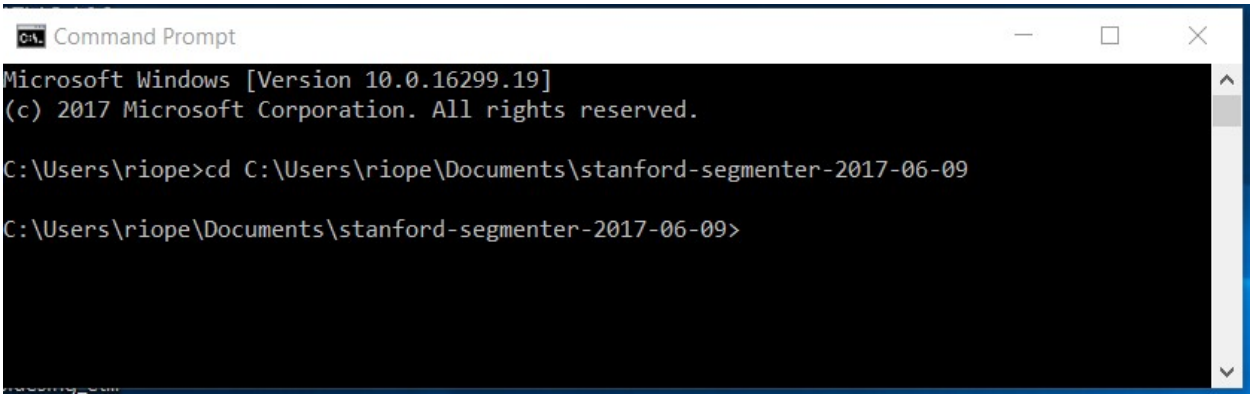
Note: CMD stands for “Command Prompt.”
This will start a command Prompt window.



31. Important:

    Change the Directory to where the folder “stanford-segmenter-2017-06-09” is located

For example:
C:\Users\riope>cd C:\Users\riope\Documents\stanford-segmenter-2017-06-09



Note:
The generic PC segmenter command syntax is as follows:

segment.bat ctb|pku] path/to/input.file <encoding> <size> > path/to/segmented.file

Examples: segment.bat ctb test.simp.utf8 UTF-8 0 > textsegment_ctb_out.txt

Or

segment.bat pku test.simp.utf8 UTF-8 0 > textsegment_pku_out.txt

If you place the input file in the "stanford-segmenter-2017-06-09" folder then you do not have to specify the complete file path.

The parameters are: ctb : Chinese Treebank
pku : Beijing University path/to/: is the file
path input.file: The file you want to
segment. Each line is a sentence.
encoding: UTF-8, GB18030, etc.
(This must be a character encoding name known by Java) size: size of the n-best list (just put '0' to print the best hypothesis without

probabilities). segmented.file: The output file segmented. If no segmented file is given the output is displayed in the terminal screen.

Two segmentation models are provided. The "ctb" model was trained with the Chinese treebank(CTB) segmentation, and the "pku" model was trained with Beijing University's (PKU) segmentation. PKU models provide smaller vocabulary sizes and OOV rates on test data than CTB models.

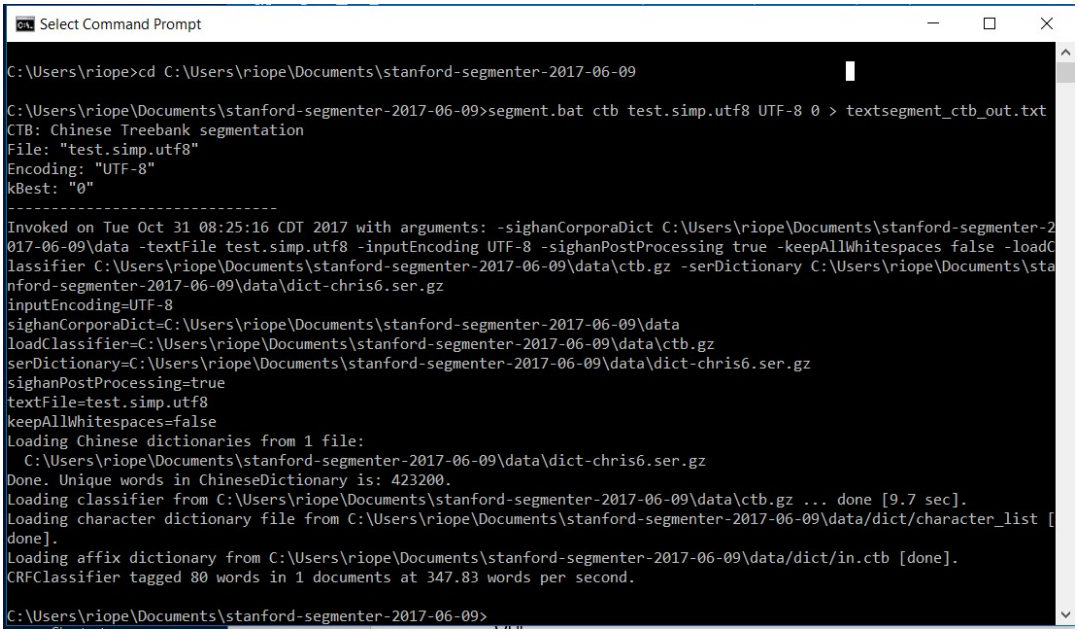
See README-Chinese.txt more details.

The following are three examples.

32. Example 1:

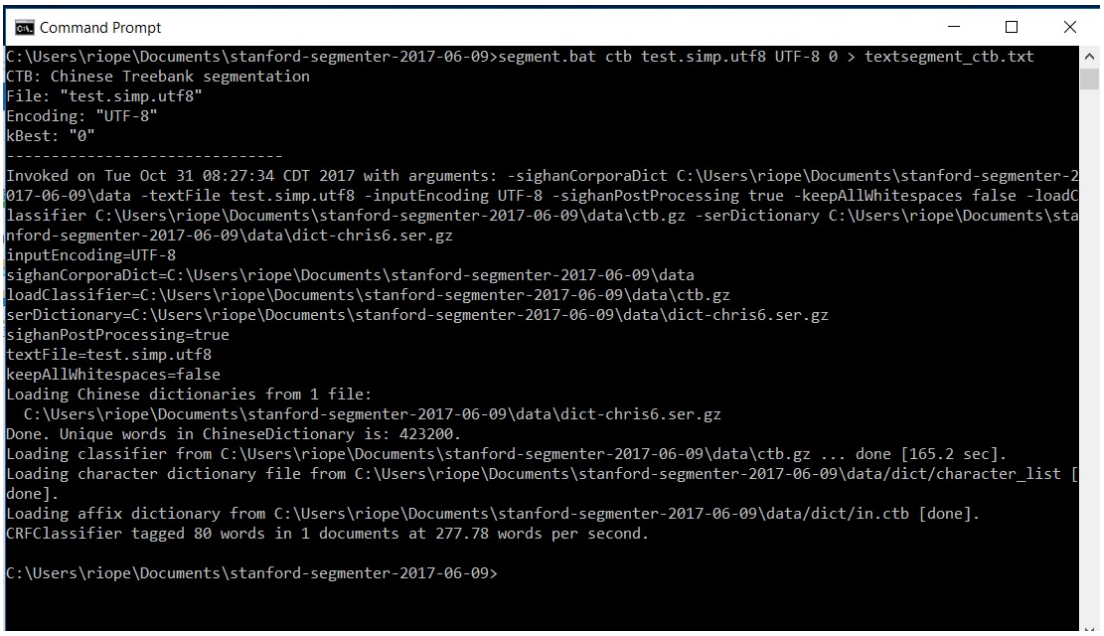
The below PC command will display the results in the terminal window.

segment.bat ctb test.simp.utf8 UTF-8 0



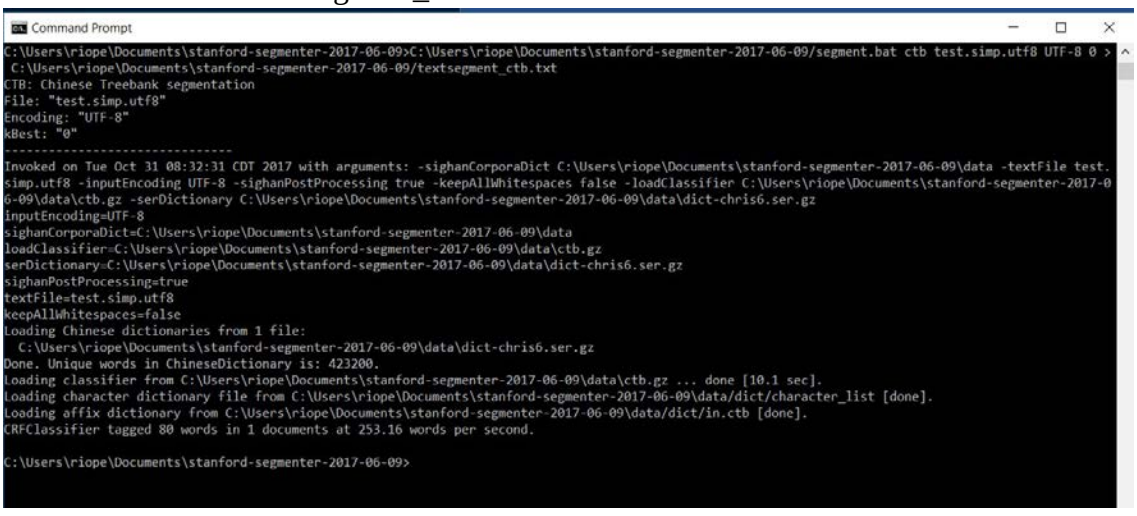
33. Example 2:

The below command will save the results to the file: textsegment_ctb.txt

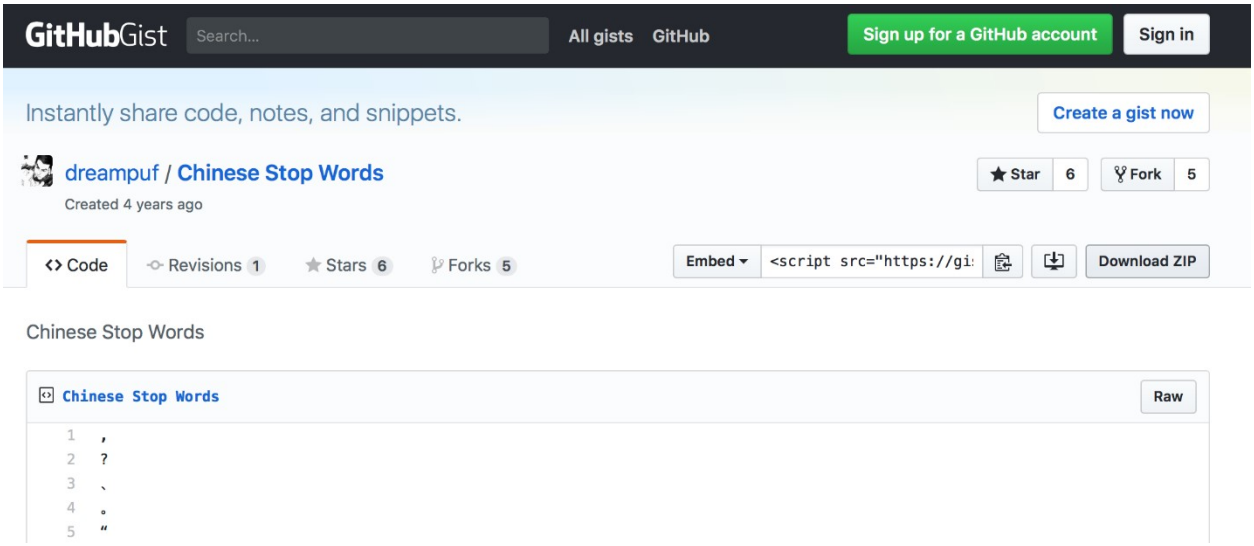


34. Example 3:

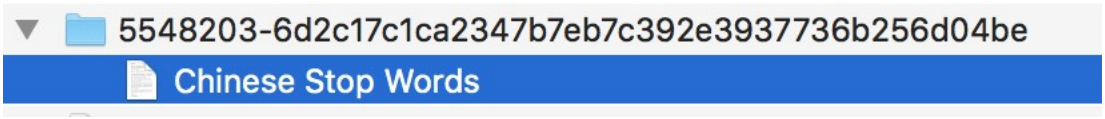
The example below uses the full path for each file: C:\Users\riope\Documents\stanford-segmenter-2017-06-09/segment.bat ctb test.simp.utf8 UTF-8 0 > C:\Users\riope\Documents\stanford-segmenter-2017-06-09/textsegment_ctb.txt

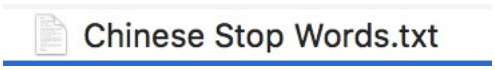


35. Download and unzip the Chinese Stop Words file at: https://gist.github.com/dreampuf/5548203



Unzip the file and add the file extension ".txt"
This will change the file type from Terminal to Text.



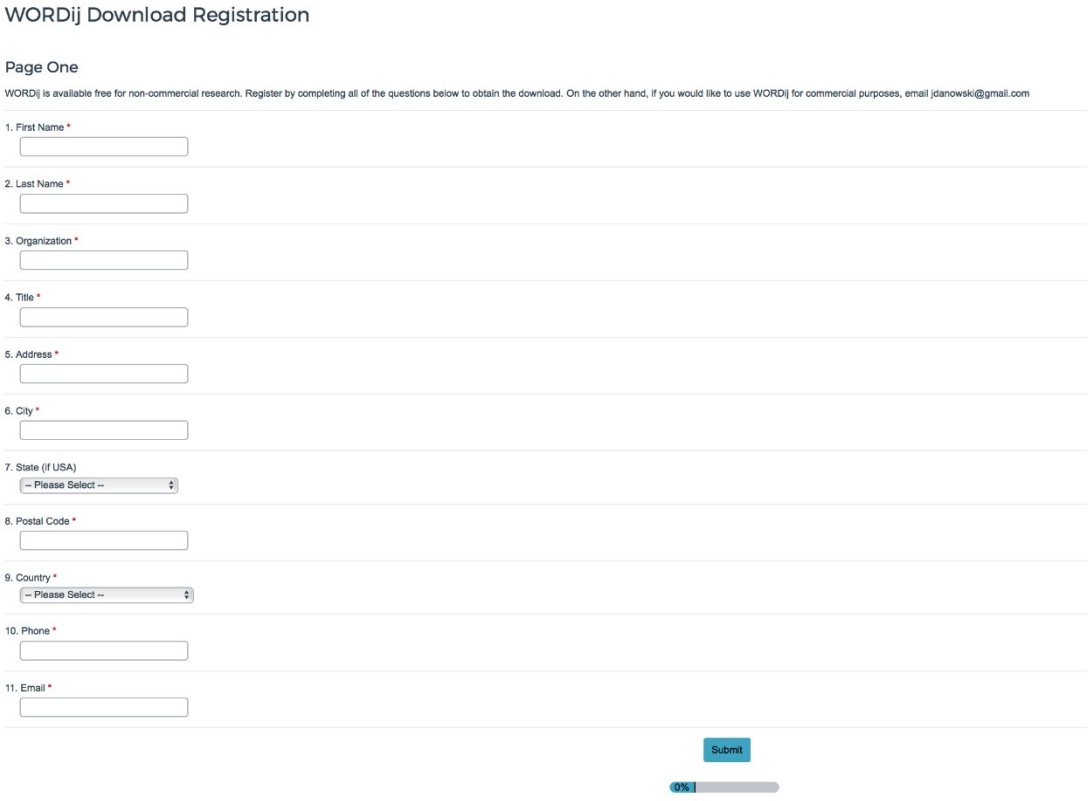


Use this file as the Droplist with WORDij's WordLink module.

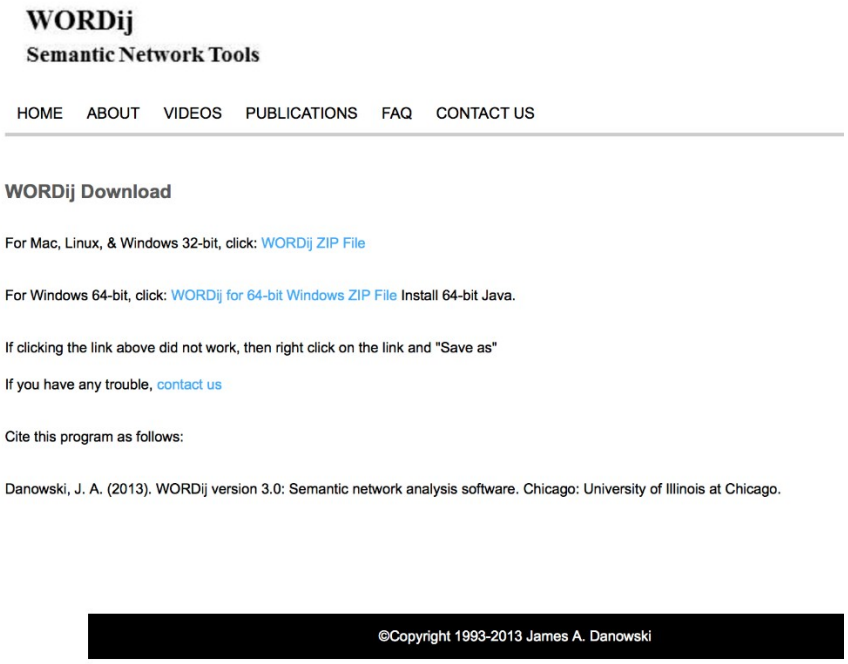
Note: Do NOT check the "Chinese Filter" option. That should be used only if you wish to input a raw Chinese text file, unsegmented, and want to produce all the adjacent pairs of characters.

36. Download WORDij at: http://wordij.net

37. Complete the WORDij download form and click Submit at the bottom of the form.

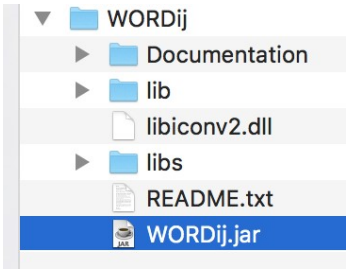


38. Choose the Windows 64-bit option



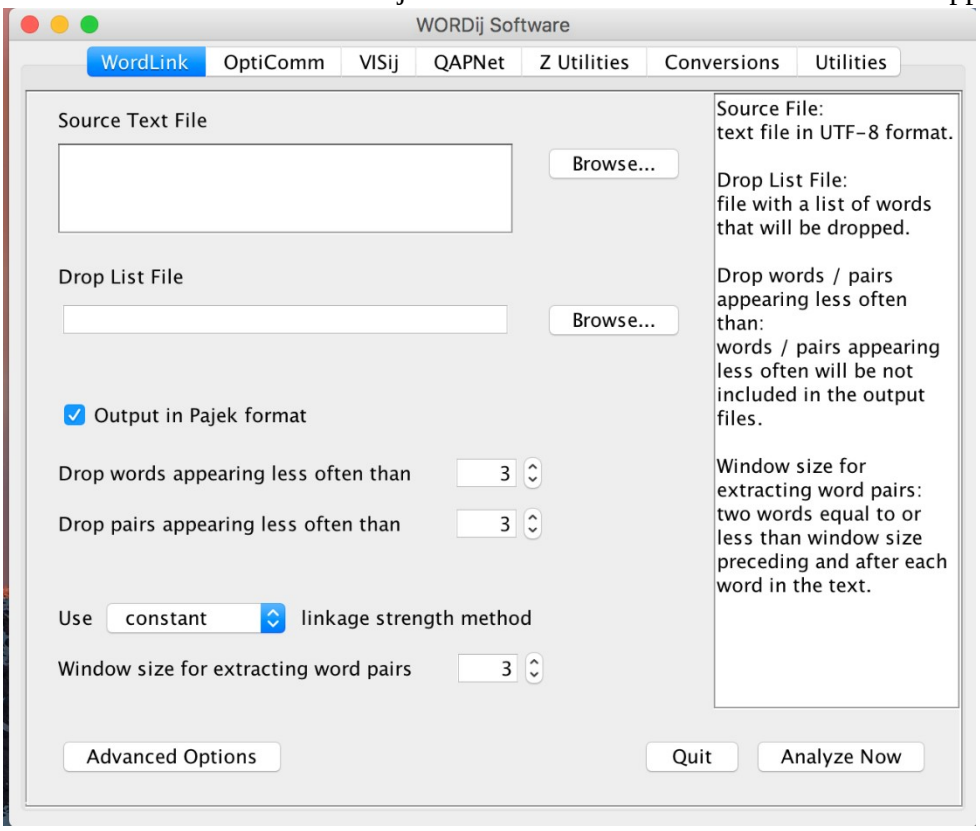
39. You should see the WORDij.zip file as a folder in your Downloads folder.
We suggest you move the WORDij folder to your Documents folder as a more permanent file location. Note: a MAC will automatically unzip the folder, which is not the case on a Windows PC.

40. Open the WORDij folder. Locate the WORDij.jar and right click on it, or Control Click.



41. Click Open.

42. WORDij will start and the main menu screen will appear.



Installing the Stanford Segmenter and Segmenting Chinese on a Mac with macOS Sierra Version 10.12.6, Text Only Instructions, No screenshots

1. Begin by checking that you have the latest Java version installed.
2. Click the Black Apple in the upper left corner.
3. Select System Preferences and look for the Java Icon
4. If you do not see the Java icon, then you need to install Java
5. Note: if you see the Java icon, click on it and check for updates.
6. Otherwise, go to the next step and download the Java JDK for macOS.
7. Note: As of November 9, 2017, the current Java is 9 (build 9.0.1 +11)
8. Download the Java Development Kit or JDK at this link:
9. http://www.oracle.com/technetwork/java/javase/downloads/index-jsp-138363.html#javasejdk (Note: The Java Development Kit or JDK also includes the Java Runtime Environment or (JRE). To run WORDij on a Mac you need the JDK.
10. Click the JDK Download option.

Click the "Accept License Agreement" and then

11. Download the maxOS 371.64 MB jdk-9_osx-x64_bin.dmg, or latest version. 12. Goto your Downloads folder and click on the file: jdk-9_osx-x64_bin.dmg
13. A window opens and double click on the JDK 9. package icon.
14. Click Continue
15. Click Install
16. Enter your computer's password or use your Touch ID
17. Click Close.
18. Keep or move the install file to trash
19. Download the Stanford Segmenter for Chinese and Arabic at this link below.

https://nlp.stanford.edu/software/segmenter.html

20. Click the download link
21. Select the 3.8.0 version or whatever is the most recent version)
22. Unzip the file: stanford-segmenter-2017-06-09.zip (or whatever the most recent version is). 23. Start a Terminal Session *24. Important:*
25. Change the Directory to where the folder "stanford-segmenter-2017-06-09" is located
26. For example:

27. cd /Users/kenriopelle/Downloads/stanford-segmenter-2017-06-09 28. The generic segmenter command syntax is as follows:

segment.sh [-k] [ctb|pku] path/to/input.file <encoding> <size> > path/to/segmented.file

Example: ./segment.sh ctb ./test.simp.utf8 UTF-8 0

> ./textsegmentCTBout.txt The period forward slash "./"

before the "segment.sh" means to use the current directory

You can use the "./" before the input.file and the output segmented.file.

The parameters are:
-k: keep all

white spaces in

the input ctb:

Chinese

Treebank pku:

Beijing Univ.

path/to/: is the file path input.file: The file

you want to segment. Each line is a

sentence.

encoding: UTF-8, GB18030, etc.

(This must be a character encoding name known by Java) size:

size of the n-best list (just put '0' to print the best hypothesis

without probabilities).

segmented.file: The output file segmented. If no segmented file is given the

output is displayed in the terminal screen.

Two segmentation models are provided. The "ctb" model was trained with Chinese treebank (CTB) segmentation, and the "pku" model was trained with Beijing University's (PKU) segmentation. PKU models provide smaller vocabulary sizes and OOV rates on test data than CTB models. See README-Chinese.txt for more details.

The following are three examples.

29. Example 1:
30. The below command will display the results in the terminal window. ./segment.sh ctb ./test.simp.utf8 UTF-8 0

31. Example 2:
The below command will save the results to the file: textsegmentCTBout2.txt

./segment.sh ctb ./test.simp.utf8 UTF-8 0 > ./textsegmentCTBout2.txt

32. Example 3:
The example below uses the full path for each file:
/Users/kenriopelle/Downloads/stanford-segmenter-2017-06-09/segment.sh ctb

/Users/kenriopelle/Downloads/stanford-segmenter-2017-06-09/test.simp.utf8 UTF-8 0 >

/Users/kenriopelle/Downloads/stanford-segmenter-2017-06-09/textsegmentCTBout.txt

33. Download and unzip the Chinese Stop Words file at:
https://gist.github.com/dreampuf/5548203

Unzip the file and add the file extension ".txt"
This will change the file type from Terminal to Text.

Use this file as the Droplist with WORDij's WordLink module.

Note: Do NOT check the "Chinese Filter" option. That should be used only if you wish to input a raw Chinese text file, unsegmented, and want to produce all the adjacent pairs of characters.

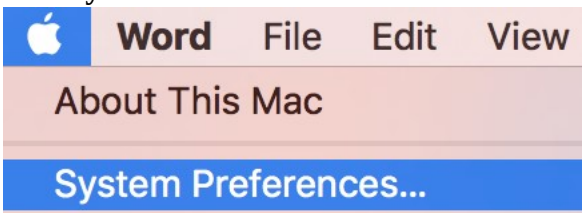
34. Download WORDij at: https://wordij.net

35. Complete the WORDij download form and click Submit at the bottom of the form.

36. Choose to download the Mac option

Installing the Stanford Segmenter and Segmenting Chinese on a Mac with macOS Sierra Version 10.12.6 with screenhsots

1. **Check that you have the latest Java version installed.**
2. **Click the Black Apple in the upper left corner.**
3. **Select System Preferences…**

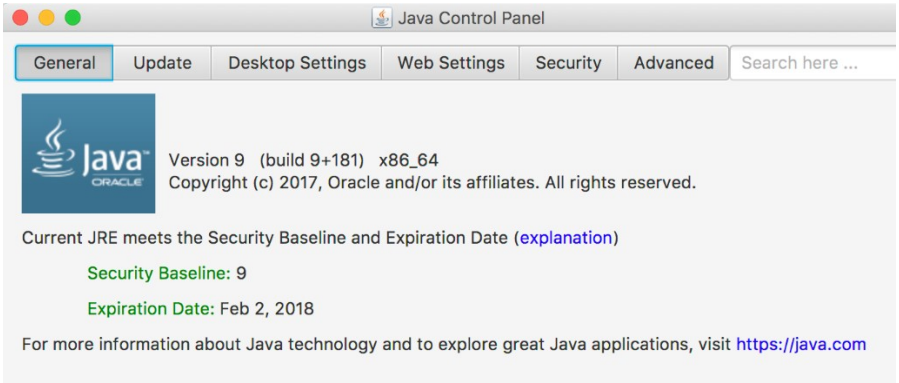


4. **If you do not see the Java icon, then you need to install Java**

a. **Note: if you see the Java icon, click on it and check for updates.**

b. **Otherwise, goto the next step and download the Java JDK for macOS.**

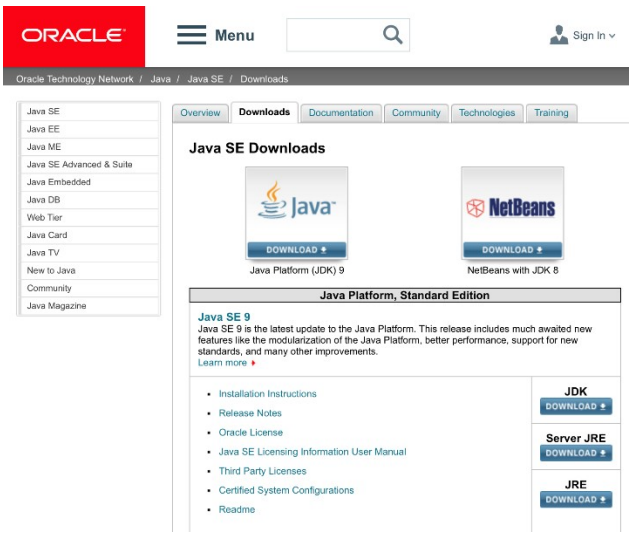
c. **Note: As of September 25th 2017, the current Java is 9 (build 9-181)**



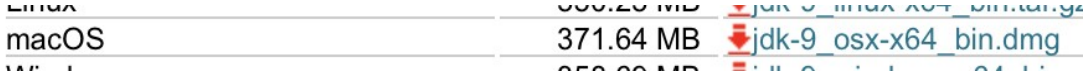
5. **http://www.oracle.com/technetwork/java/javase/downloads/index-jsp138363.html#javasejdk**

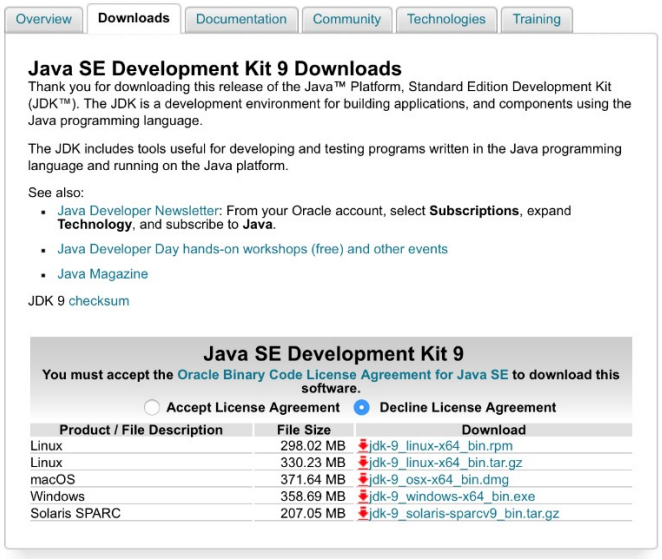
(This link is for the Java Development Kit or JDK which includes the JRE) Note: To run WORDij on a Mac you need the JDK.

6. **Click the JDK Download option.**

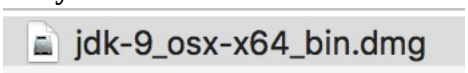


7. **Click the "Accept License Agreement" and then**
8. **Download the maxOS 371.64 MB jdk-9_osx-x64_bin.dmg**





9. **Goto your Downloads folder and click on the file**

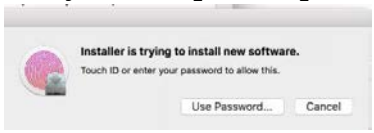


10. **A window opens and double click on the JDK 9. package icon.**

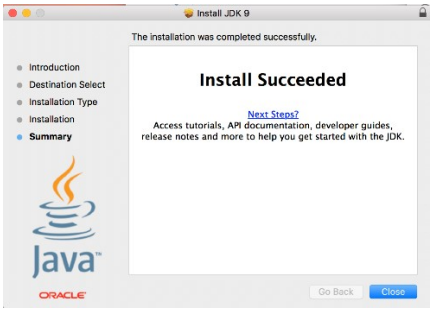
11. **Click Continue**

12. **Click Install**

13. **Enter your computer's password or use your Touch ID**



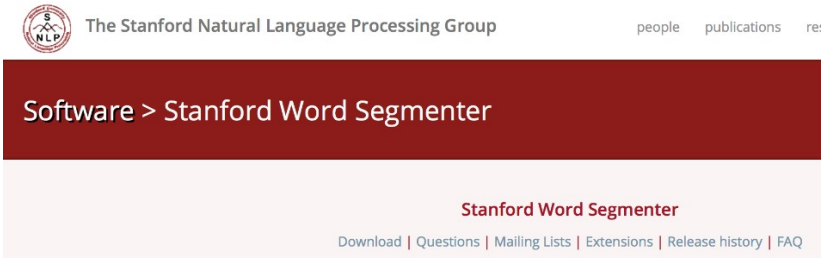
14. **Click Close.**



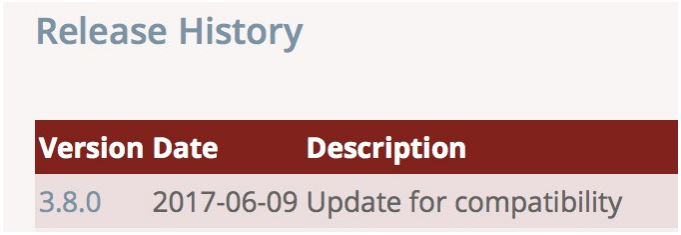
15. **Keep or move install file to trash**

16. **Download the Stanford Segmenter for Chinese and Arabic.**
https://nlp.stanford.edu/software/segmenter.html

17. **Click the download link**

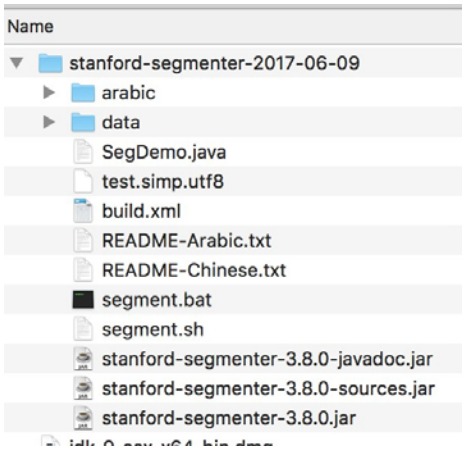


18. **Select the 3.8.0 version or whatever is the most recent version)**

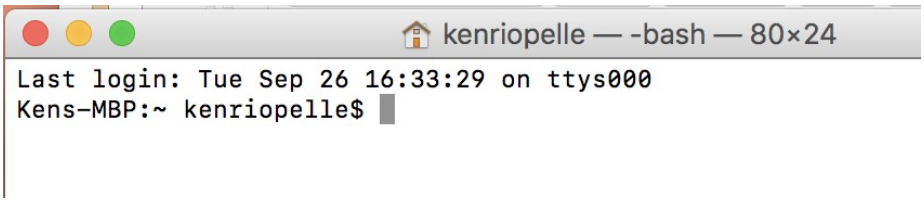


19. **Unzip the file: stanford-segmenter-2017-06-09.zip (or whatever the most recent version is).**

Here is a screenshot of the unzip file listing.



20. **Start a Terminal Session**



21. ***Important:***

Change the Directory to where the folder "stanford-segmenter-2017-06-09" is located

For example:
cd /Users/kenriopelle/Downloads/stanford-segmenter-2017-06-09

The generic segmenter command syntax is as follows:

segment.sh [-k] [ctb|pku] path/to/input.file <encoding> <size> > path/to/segmented.file

Example: ./segment.sh ctb ./test.simp.utf8 UTF-8 0 > ./textsegmentCTBout.txt

The period forward slash "./" before the "segment.sh" means to use the current directory You can use the "./" before the input.file and the output segmented.file.

The parameters are:
-k: keep all white spaces in the input
ctb :
Chin
ese
Treeb
ank
pku :
Beiji
ng
Univ.
path/to/: is the file path input.file: The file you want to segment. Each line is a sentence. encoding: UTF-8, GB18030, etc.
(This must be a character encoding name known by Java)
size: size of the n-best list (just put '0' to print the best hypothesis without probabilities).

segmented.file: The output file segmented. If no segmented file is given the output is displayed in the terminal screen.

Two segmentation models are provided. The "ctb" model was trained with Chinese treebank (CTB) segmentation, and the "pku" model was trained with Beijing University's (PKU) segmentation. PKU models provide smaller vocabulary sizes and OOV rates on test data than CTB models. CTB and PKU models representing slightly different feature sets. See README-Chinese.txt more details.

a. The following are three examples.

22. **Example 1:**

The below command will display the results in the terminal window.

./segment.sh ctb ./test.simp.utf8 UTF-8 0

```
Kens-MBP:stanford-segmenter-2017-06-09 kenriopelle$ ./segment.sh ctb ./test.simp.utf8 UTF-8 0
(CTB):
-n File:
./test.simp.utf8
-n Encoding:
UTF-8
---------------------------------
Invoked on Tue Sep 26 23:47:29 CEST 2017 with arguments: -sighanCorporaDict ./data -textFile ./test.simp.utf8 -inputEncoding UTF-8 -sighanPostProcessing true -
keepAllWhitespaces false -loadClassifier ./data/ctb.gz -serDictionary ./data/dict-chris6.ser.gz
inputEncoding=UTF-8
sighanCorporaDict=./data
loadClassifier=./data/ctb.gz
serDictionary=./data/dict-chris6.ser.gz
sighanPostProcessing=true
textFile=./test.simp.utf8
keepAllWhitespaces=false
Loading Chinese dictionaries from 1 file:
  ./data/dict-chris6.ser.gz
Done. Unique words in ChineseDictionary is: 423200.
Loading classifier from ./data/ctb.gz ... done [6.1 sec].
Loading character dictionary file from ./data/dict/character_list [done].
Loading affix dictionary from ./data/dict/in.ctb [done].
面 对 新 世 纪 ， 世 界 各 国 人 民 的 共 同 愿 望 是 ： 继 续 发 展 人 类 以 往 创 造 的 一 切 文 明 成 果 ， 克 服 20 世 纪 困 扰 着 人 类 的 战 争 和 贫 困 问 题 ， 推 进 和 平 与
发 展 的 崇 高 事 业 ， 创 造 一 个 美 好 的 世 界 。
CRFClassifier tagged 80 words in 1 documents at 544.22 words per second.
Kens-MBP:stanford-segmenter-2017-06-09 kenriopelle$
```

23. **Example 2:**

The below command will save the results to the file: textsegmentCTBout2.txt

./segment.sh ctb ./test.simp.utf8 UTF-8 0 > ./textsegmentCTBout2.txt

```
Kens-MBP:stanford-segmenter-2017-06-09 kenriopelle$ ./segment.sh ctb ./test.simp.utf8 UTF-8 0 > ./textsegmentCTBout2.txt
(CTB):
-n File:
./test.simp.utf8
-n Encoding:
UTF-8
---------------------------------
Invoked on Tue Sep 26 23:51:25 CEST 2017 with arguments: -sighanCorporaDict ./data -textFile ./test.simp.utf8 -inputEncoding UTF-8 -sighanPostProcessing true -
keepAllWhitespaces false -loadClassifier ./data/ctb.gz -serDictionary ./data/dict-chris6.ser.gz
inputEncoding=UTF-8
sighanCorporaDict=./data
loadClassifier=./data/ctb.gz
serDictionary=./data/dict-chris6.ser.gz
sighanPostProcessing=true
textFile=./test.simp.utf8
keepAllWhitespaces=false
Loading Chinese dictionaries from 1 file:
  ./data/dict-chris6.ser.gz
Done. Unique words in ChineseDictionary is: 423200.
Loading classifier from ./data/ctb.gz ... done [5.8 sec].
Loading character dictionary file from ./data/dict/character_list [done].
Loading affix dictionary from ./data/dict/in.ctb [done].
CRFClassifier tagged 80 words in 1 documents at 516.13 words per second.
Kens-MBP:stanford-segmenter-2017-06-09 kenriopelle$
```

24. **Example 3:**

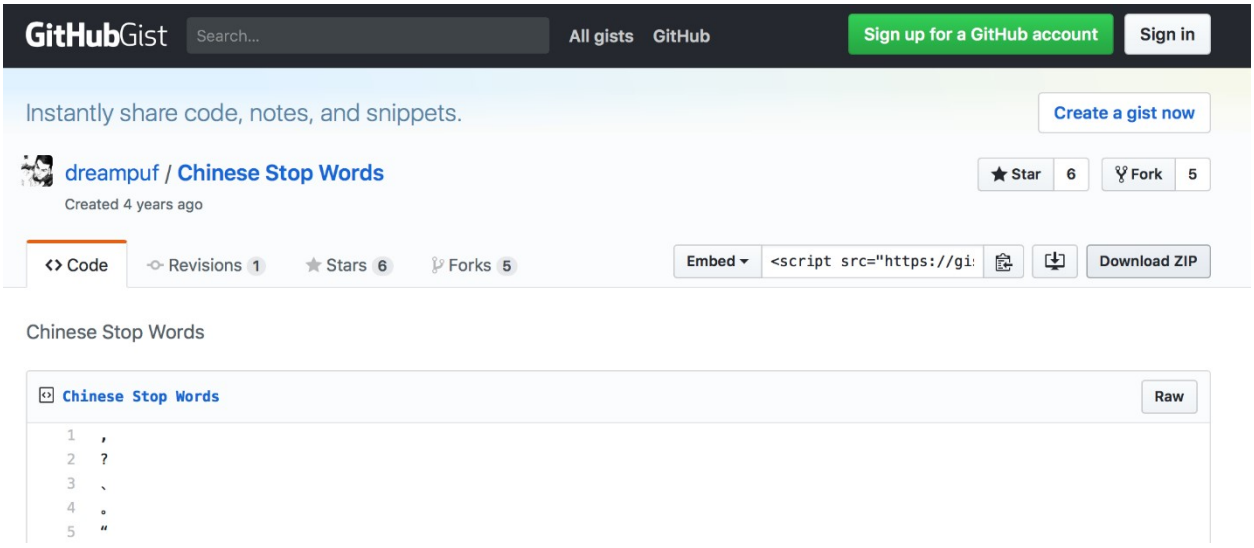
The example below uses the full path for each file:

/Users/kenriopelle/Downloads/stanford-segmenter-2017-06-09/segment.sh ctb

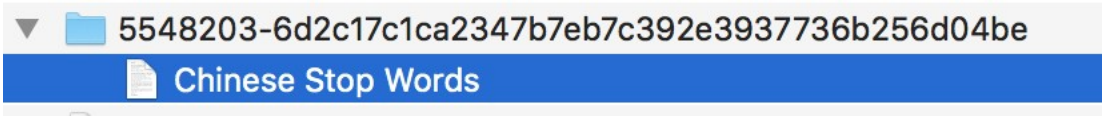
/Users/kenriopelle/Downloads/stanford-segmenter-2017-06-09/test.simp.utf8 UTF-8 0 >

/Users/kenriopelle/Downloads/stanford-segmenter-2017-06-09/textsegmentCTBout.txt

25. **Download and unzip the Chinese Stop Words file at: https://gist.github.com/dreampuf/5548203**



26. **Unzip the file and add the file extension ".txt" This will change the file type from Terminal to Text.**

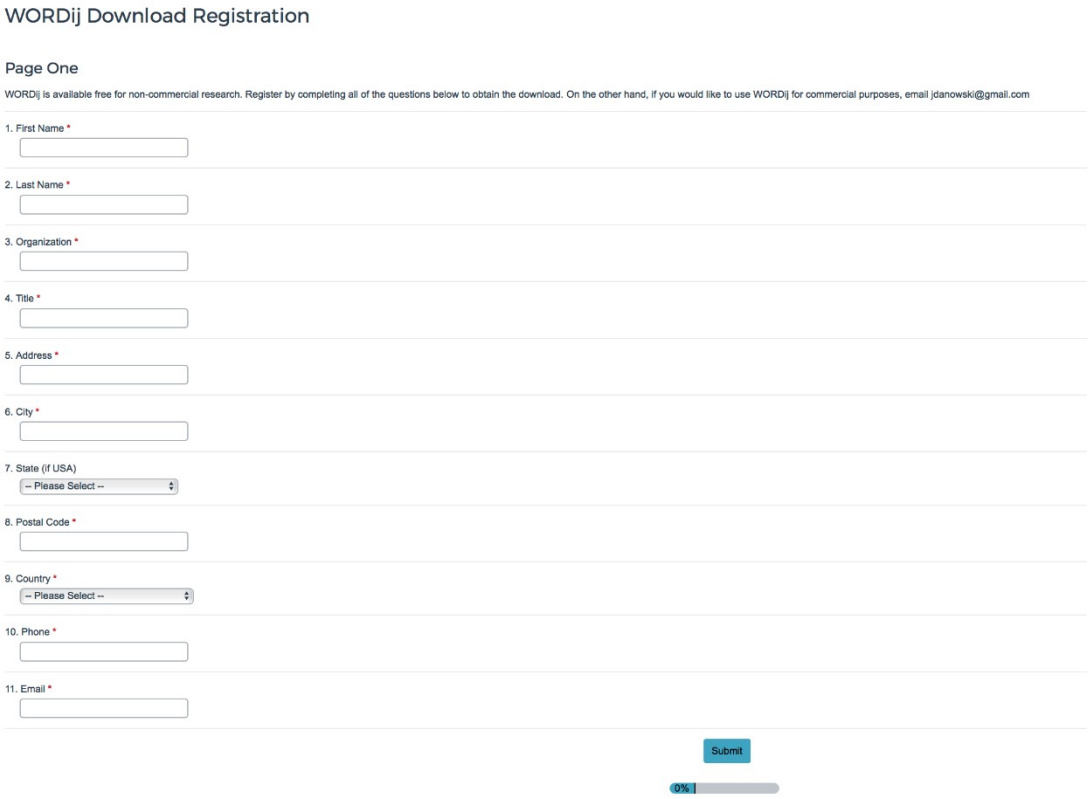


Use this file as the Droplist with WORDij's WordLink module.

Note: Do NOT check the "Chinese Filter" option. That should be used only if you wish to input a raw Chinese text file, unsegmented, and want to produce all the adjacent pairs of characters.

27. **Download WORDij at: https://wordij.net**

28. **Complete the WORDij download form and click Submit at the bottom of the form.**



29. **Choose the Mac option**

WORDij

Semantic Network Tools

HOME ABOUT VIDEOS PUBLICATIONS FAQ CONTACT US

WORDij Download

For Mac, Linux, & Windows 32-bit, click: WORDij ZIP File

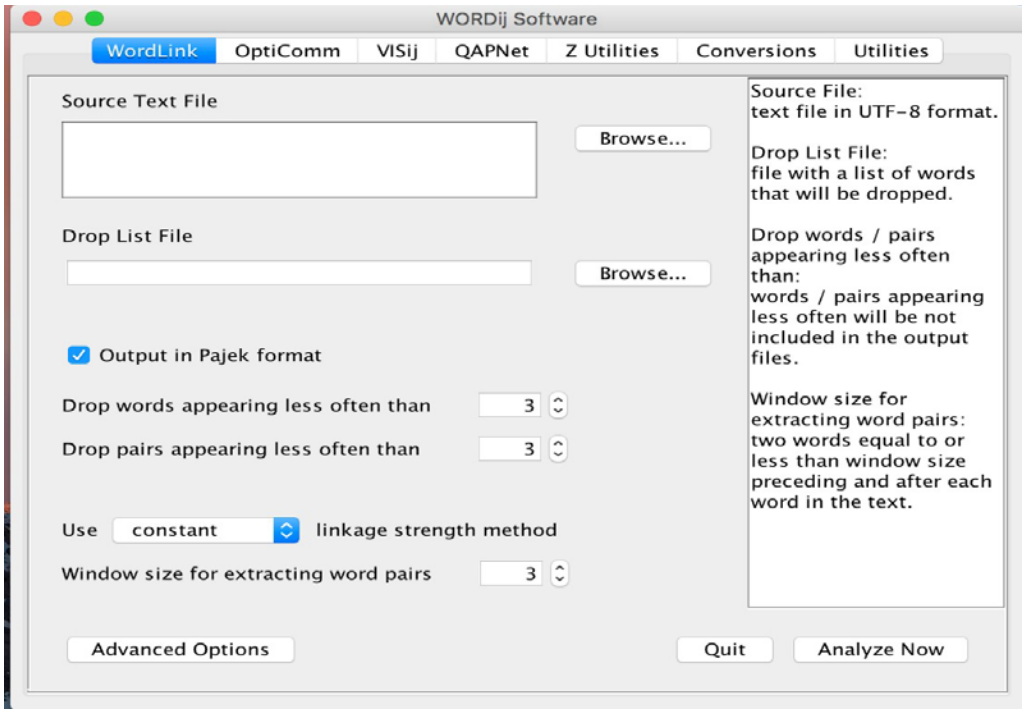
For Windows 64-bit, click: WORDij for 64-bit Windows ZIP File Install 64-bit Java.

If clicking the link above did not work, then right click on the link and "Save as"

If you have any trouble, contact us

Cite this program as follows:

Danowski, J. A. (2013). WORDij version 3.0: Semantic network analysis software. Chicago: University of Illinois at Chicago.



CONCLUSION

This paper, having shown how to segment Chinese for semantic neetwork analysis, opens the way for more communication and other social scientists to do more semantic network analysis on Chinese texts. The literature is likely to grow in two directions, horizontally as crosscultural comparisons are made, as well as a vertical widening and deepening of intracultural Chinese semantic network research. Research questions and hypotheses can now be addressed with more qualitative, quantitative, and mixed methods approaches including network analysis of message content.

REFERENCES

Carley, K. (1993). Coding choices for textual analysis: a comparison of content analysis and map analysis. *Sociological Methodology 23:* 75-126.

Carley, K. (1997a). Extracting team mental models through textual analysis. *Journal of Organizational Behavior 18*: 533-558.

Carley, K. (1997b.) Network text analysis: The network position of concepts. in text analysis for the social sciences, Carl W. Roberts (ed). Mahwah, NJ: Lawrence Erlbaum Associates, Inc. Pp. 79-102.

Corman, S., Kuhn, T., McPhee, R. and Dooley, K. (2001). Studying complex discursive systems: Centering resonance analysis of communication. *Human Communication Research 28*(2): 157-206.

Danowski, J. A. (1982). A network-based content analysis methodology for computer-mediated communication: An illustration with a computer bulletin board, in M. Burgoon (Ed.), *Communication Yearbook 5* (pp. 904-925). New Brunswick, NJ: Transaction Books.

Danowski, J. A. (1993a). WORDIJ: A word pair approach to information retrieval. Proceedings of the DARPA/NIST TREC Conference (pp. 131-136.) Washington, DC: National Institute of Standards and Technology.

Danowski, J. A. (1993b). Network analysis of message content. G. Barnett, & W. Richards (eds.). *Progress in communication sciences XII* (pp. 197-222). Norwood, NJ: Ablex.

Danowski, J.A. (2009). Network analysis of message content. In K. Krippendorff & M. Bock (Eds.) *The content analysis reader (*pp. 421-430). Sage Publications.

Danowski, J. A. (2012a, August). Semantic network analysis of islamist sources using time slices as nodes and semantic similarity as link strengths: Some implications for propaganda analysis about jihad. In *Intelligence and Security Informatics Conference (EISIC), 2012 European* (pp. 164-171). IEEE.

Danowski, J. A. (2012b). Analyzing change over time in organizations' publics with a semantic network include list: An illustration with Facebook. In *Advances in Social Networks Analysis and Mining (ASONAM), 2012 IEEE/ACM International Conference on* (pp. 954-959). IEEE.

Danowski, J. A. (2017a). WORDij version 4.0: 64-bit. [computer program]. Madison, WI: Communication and Technology Sciences. (https://wordij.net)

Danowski, J. A. (2017b). Creating Network-Based Communication Interventions to Increase

Community Resilience: A Demonstration for an African Nation Recovering from MuslimChristian Civil War. Presented at the 1st North American Social Networks Conference NASN2017, Washington, DC, 26-30 July 2017.

Danowski, J. A., & Cepela, N. (2010). Automatic mapping of social networks of actors from text corpora: Time series analysis. In *Data Mining for Social Network Data* (pp. 31-46). Springer US.

Danowski, J. A., & Park, H. W. (2014). Arab Spring effects on meanings for islamist web terms and on web hyperlink networks among muslim-majority nations: A naturalistic field experiment. *Journal of Contemporary Eastern Asia*, *13*(2).

Lafferty, J.; McCallum, A.; and Pereira, F. C. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Lewis, E. T., Carley, K. M., & Diesner, J. (2016). Displaying responsiveness or asserting identity in organizational language: how concept networks capture rhetorical strategies. *Center for the Computational Analysis of Social and Organizational Systems*.

Pei, W., Ge, T., & Chang, B. (2014, June). Max-margin tensor neural network for chinese word segmentation. In *ACL (1)*(pp. 293-303).

Peng, F., Feng, F., & McCallum, A. (2004, August). Chinese segmentation and new word detection using conditional random fields. In *Proceedings of the 20th international conference on Computational Linguistics* (p. 562). Association for Computational Linguistics.

Tseng, H., Chang, P., Andrew, G., Jurafsky, D., & Manning, D. (2005). A conditional random field word segmenter for SIGHAN Bakeoff 2005. In Proceedings of the 4th SIGHAN Workshop on Chinese Language Processing, volume 171. Jeju Island, Korea

Wang, X., Utiyama, M., Finch, A. M., & Sumita, E. (2014, June). Empirical Study of Unsupervised Chinese Word Segmentation Methods for SMT on Large-scale Corpora. In *ACL (2)* (pp. 752-758).

Yang, S., & González-Bailón, S. (2017). Semantic networks and applications in public opinion research. *The Oxford Handbook of Political Networks*, 327.

Yuan, E, J. Feng, M. & Danowski, J, A. (2013). Privacy in semantic networks on chinese social media: The case of Sina Weibo. *Journal of Communication 63* (2013) 1011–1031.

Zywica, J., & Danowski, J. (2008). The faces of Facebookers: Investigating social enhancement and social compensation hypotheses; predicting Facebook™ and offline popularity from sociability and self-esteem, and mapping the meanings of popularity with semantic networks. *Journal of Computer-Mediated Communication*, *14*(1), 1-34.