A QUICK INTRODUCTION TO THE ***SEMILAR*** APPLICATION

Mihai Lintean, Rajendra Banjade, and Vasile Rus
vrus@memphis.edu
linteam@gmail.com
rbanjade@memphis.edu

The Goal of this Document

This document introduce the user to the SEMILAR application which offers easy access to semantic similarity methods available in the SEMILAR library (Rus et al., 2013) from a graphical user interface (GUI). The SEMILAR application is a standalone Java application which allows users to pre-process texts and compute semantic similarity between pairs of texts using menus and mouse clicks. This document offers a quick introduction to the SEMILAR application and not an comprehensive description of it. Furthermore, the current GUI-based SEMILAR application is a simplified version of the full-fledged SEMILAR application which has not been yet publicly released for it is too rich and we cannot offer, for the time being, training to our large user base due to lack of resources.

Important Note: The SEMILAR application comes with its own version of the SEMILAR library which is different from the public SEMILAR library available for download. This is due to the fact that we have kept updating the SEMILAR library while the GUI-based SEMILAR application has yet to be integrated with the latest version of the SEMILAR library. Therefore, when you download the SEMILAR application it is very important to download all the files that are described in the download page, including the SEMILAR library.

Citation information:
Rus, V., Lintean, M. C., Banjade, R., Niraula, N. B., & Stefanescu, D. (2013, August). SEMILAR: The Semantic Similarity Toolkit. In *ACL (Conference System Demonstrations)* (pp. 163-168)

Where to Start?

The first step is to download the SEMILAR application and configure it. Please visit "SEMILAR APP" page in SEMILAR website (www.semanticsimilarity.org) for download instructions. To start the application, go to the SEMILAR application home directory (i.e., Semilar-1.0.19) and double click Semilar.jar or run *java -jar semilar.jar* on the command line.

Failed to start the application? Please go through the download instructions available in SEMILAR Application download page and make sure everything is as specified. If you really feel that the issue cannot be easily resolved, please let us know with as much details as possible.

Once you start the application, you can run sample data which is available in test-data folder or start with your own dataset.

How to use the interface?

We will describe important features of the application as we go through the step-by-step example given below. We would suggest you work with the tool as it is virtually impossible to create a full fledged user guide.

A Step-by-Step Example

We will now look at a step-by-step example of how to use the SEMILAR application to compute the semantic similarity between pairs of texts provided in an input text file.

We will use a subset of instances from the Microsoft Research Paraphrase corpus (Dolan, B., Quirk, C., & Brockett, 2004). This subset can be found in test-data folder which is in the same folder as Semilar.jar.

Notes:

1. The descriptions are based on the assumption that you have downloaded all the required files from SEMILAR website and configured properly (please see “Where to start?” section), and you are on Semilar-1.0.19 folder.
2. Please check messages displayed on the application console whenever you run into trouble. The error message, if any, displayed on the console might help you figure out the problem.

Format of the input file

The texts in the input must be in the following format:

- (optional) the first line defines the structure of the next lines; it contains a maximum of 5 tab-separated strings which can be:
    - id = defines the id of one of the two texts
    - text = defines one of the two texts
    - class = defines the relation between the two texts

      Example:

      ```
      class id    text id   text
      1           456       This is the first text.      837     Here is the second text.
      ```
- each of the following lines represent one instance (i.e. a pair of sentences is an instance)
    - lines should have at least one 'text' and at most two 'id's, two 'text's and one 'class'. The length of texts can vary from just pair of words (if you are measuring word level similarity) to pair of sentences or paragraphs. Be aware though that some of the metrics will take much longer to process on

instances of longer texts. The class is expected to be a binary value, indicating whether the instance pair has a semantic similarity or not (i.e. paraphrase or not).

- if format line is missing, the defaults based on the determined number of columns (tab-separated) are:
  - 5 columns: class id1 id2 text1 text2
  - 3 columns: class text1 text2
  - 2 columns: text1 text2
  - 1 columns: text1

Details about the input format are also available in the SEMILAR application. Once you start the application, click the What's The Format? button in the Project Select pane, which is the default pane when you start SEMILAR. Also look at the sample input file in test-data folder for a reference.

Step 1. Start SEMILAR

To start the SEMILAR application you must run the Semilar.jar executable file.

Once you started the SEMILAR application, you will see the main screen which is divided into three parts:

1) The menu bar at the top which should indicate the five tabs or panes:
   a. Project Select
   b. Data View
   c. Preprocessing
   d. Similarity Methods, and
   e. SandBox
2) The current pane area in the middle where features related to the selected tab are shown; and
3) The Console area at the bottom where messages are displayed about the processing steps performed by the user. More details about each of the tabs and related features will be revealed as needed in the following steps. The reader of this manual should keep in mind that this document is just a quick intro to using the SEMILAR app and not a fully detailed manual of all the options. We will only describe here the options necessary to illustrate our step-by-step example.

Step 2. Load the text

In order to load the text to be processed in the SEMILAR application, select the Project Select tab.

From the Import New Data box, click the Browse button next to the Training Data File to browse for the desired input file that contains the texts to be processed.

In our step-by-step example, we use a sample input file containing the subset of MSR paraphrase corpus. The sample input file named msr_paraphrase_subset.txt is available in test-data folder.

Once you located the file, please click the Import button. The data should now be loaded in the SEMILAR application unless an error has happened during the opening of the file. If an error occurred it is most likely to an incorrect input format of the data – please check the input file format in the corresponding section above. Also, check the console message, if any.

Step 3: View the input text in Data View

If the loading of the input file was successful, you can explore the loaded texts in the Data View pane which can be activated by clicking on the Data View tab in the menu bar. The Data View pane is quite rich in features. The goal of this pane is to allow the user to check the data, select parts of it for further processing, see the output of processing steps, etc.

For now, we will simply draw the attention to the two most important columns in the table-like view of the data: Text A and Text B. These two columns should show the texts in each pair of texts provided in the input file. All subsequent semantic similarity computations will be performed on such paired texts.

Right below the table-like view of the input data, there is a number of tabs which offer the users various useful views of the data. Some of these tabs only display something meaningful after preprocessing steps or semantic similarity computation were performed. Continue to read and learn more on how to preprocess the input texts.

Step 4: Preprocessing The Texts

The role of the preprocessing is to obtain various linguistics information such as lemmas, parts of speech and syntactic dependencies that are very useful for computing semantic similarities between two texts. To preprocess the input texts, select the Preprocessing tab from the menu bar.

There are four majors preprocessing steps that can be applied:

A. tokenization,
B. extraction of word's base forms,
C. part-of-speech tagging, and
D. syntactic parsing.

Tokenization is the separation of punctuation from words. The base forms can be obtained either as lemmas or as stems (the result of stemming). Part-of-speech tagging and syntactic parsing can be performed using either the OpenNLP package or the Stanford package. OpenNLP only offers phrase-based syntactic parsing which means no dependency (as in dependency parsing) will be available.
Once you selected your options for each of the steps above, you can then click the big Preprocess Data button to start the preprocessing process. In our case, we will choose Stanford Standard as tokenizer, use Stanford as part-of-speech tagger, Porter Stemmer for base form extraction, and Stanford for parsing.

While preprocessing the input texts, please watch the Console area at the bottom where the status of the preprocessing will be continuously displayed.

When the preprocessing step is done, the SEMILAR application will have available important linguistic information about your texts which can be used further for computing semantic similarity between pairs of texts, which we describe next. You may also see the linguistic information obtained during preprocessing from the Data View pane. Use the Lexical Data and Syntactic Data tabs in the Data View pane to access the corresponding lexical and syntactic information. You must select a particular instance for which you may want to see the lexical and syntactic information. In order to select an instance, you may just click on it in the table-like view. Once selected, the instance will be highlighted in blue.

Step 5: Computing Semantic Similarity

The main purpose of the SEMILAR application is to offer users an easy way to compute how semantically similar two texts are (in the range of 0 to 1, 1 being equivalent in meaning and 0 being not similar at all). So far, we have only prepared the input texts for this step. The Similarity Methods tab offers users access to a number of methods to compute semantic similarity. The category of methods expand word-to-word semantic similarity methods to text level (Rus & Lintean, 2012; Lintean & Rus, 2015). Appendix A provides a brief description of all these methods along with the corresponding references which present them in more detail. In our example, we will illustrate how the user can compute the semantic similarity using a lexical overlap method.

From the Class of Methods list please select Lexical (the default selection). In order to customize the lexical overlap method of choice, the user must select specific values for each of the five options available for this category of methods. First, the user must indicate whether to keep all tokens in the texts or filter out some. In our case, we will select Stop Words which means that the stop words in the input text will be discarded when computing the semantic similarity. Second, the user must specify whether or not to use the raw words or their base form (if Use Base Form is selected). Furthermore, the user may specify whether words should be all mapped to Lowercase (if Lowercase selected) before computing the semantic similarity. When the Lowercase options is selected, common nouns that appear at the beginning of a sentence and are typically spelled with a first capital letter, will be treated the same with occurrences in the middle of the sentence. For our example - do not check any of the options. Third, the user must specify how words in one text would be paired with words in the other text. There are three options available for pairing: greed, optimal (Rus & Lintean, 2012), and quadratic assignment (QAP). Please select QAP (Lintean & Rus, 2015). For Token Similarity, which indicates which word-to-word similarity to be used to compute the similarity between paired words, please choose Lin (scroll all the way down).

The Lower Match Limit option indicates a threshold below which the similarity of a pair of words is discarded. In other words, if two words are somehow paired and their word-to-word similarity is below the threshold then the words and their similarity are not considered further for computing the overall similarity of the larger texts. For our example, we will keep the default value of 0.5.

The fourth options of the lexical overlap methods indicates whether tokens should be weighted or not. Please choose IDF (Inverted Document Frequency).
The Normalize Score option indicates how the overlap score between two texts will be normalized. In our case, we will select MAX which indicates that the length of the text which has the most (MAX) number of tokens will be used to normalize the overall text-to-text similarity score.

As the user selects options, it can observed that the text field next to the Method Name is being automatically generated by SEMILAR. In other words, SEMILAR will auto-generate a name for the specific overlap method the user just created by selecting various values for the various options. If you think about the space of possibilities, there are thousands of lexical overlap methods (for more details, see Rus, Banjade, & Lintean, 2014). The user can overwrite the auto-generate name. Once the user is happy with the method name, s/he can click Add To List to add the method to the Defined Methods list. The user has the opportunity to go back and define another lexical overlap method and add it to the list. This is useful when trying to compare different ways to compute lexical overlap. For instance, the user can define two overlap methods: one which uses stop-words and one which discards stop words in order to understand the impact of stop words to compute semantic similarity using lexical overlap.

In our example, we will do just that. We will define another semantic similarity methods in which we discard the stop words (you must check the Stop Words button) and then add it to the Defined Methods list. Once the two methods are in the Defined Methods list, we can go ahead and run these methods on the input data by clicking the Compute All Similarities button. Please observe the Console to see the progress.

Once the similarities were computed, the user may save the results in a CSV file by pressing the Export All Similarities button. By checking “export with class”, the label of each instances provided in the input file will also be saved. Please observe the Console to see where the CSV file was saved and the file name.

Final Comments

The SandBox pane is just that: a place where the user can play with various semantic similarity methods without the need to provide an input file or apply the methods on a big set of data. In the SandBox the user can just type a pair of words or a pair of texts and then select a semantic similarity method on the pair and see the results immediately.
As we indicated earlier, this document is meant to be a quick introduction to the SEMILAR application and by no means a detailed, comprehensive description of it. We plan to release a more detailed manual in the future as our time and resources allow.

For comments and questions please email Rajendra Banjade (rbanjade@memphis.edu) or Dr. Vasile Rus (vrus@memphis.edu).

Thank you for using SEMILAR!

Mihai Lintean, PhD
Rajendra Banjade, PhD student
Vasile Rus, PhD
The University of Memphis, Tennessee, USA
http://semanticsimilarity.org

REFERENCES

Dolan, B., Quirk, C., & Brockett, C. (2004, August). Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In*Proceedings of the 20th international conference on Computational Linguistics* (p. 350). Association for Computational Linguistics.

Lintean, M., & Rus, V. (2015, May). An Optimal Quadratic Approach to Monolingual Paraphrase Alignment. In *Nordic Conference of Computational Linguistics NODALIDA 2015* (p. 127).

Rus, V., Lintean, M. C., Banjade, R., Niraula, N. B., & Stefanescu, D. (2013, August). SEMILAR: The Semantic Similarity Toolkit. In *ACL (Conference System Demonstrations)* (pp. 163-168).

Rus, V., & Lintean, M. (2012, June). A comparison of greedy and optimal assessment of natural language student input using word-to-word similarity metrics. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP* (pp. 157-162). Association for Computational Linguistics.

Rus, V., Banjade, R., & Lintean, M. (2014). On paraphrase identification corpora. In *Proceeding on the International Conference on Language Resources and Evaluation (LREC 2014)*.

APPENDIX A: Summary of Available Metrics in SEMILAR

The majority of the similarity methods that SEMILAR provides are presented with much detail in Dr. Lintean's dissertation on semantic similarity:

Lintean, M.C. (2011). Measuring Semantic Similarity: Representations and Methods (Doctoral dissertation). The University of Memphis, Memphis, TN

For easier reference, we will refer to this work as the "Dissertation". There are 5 main classes of methods, each with its own set of configurable properties.

I. Lexical Methods:

Compute a normalized similarity score from aligned token pairs that are deemed to be semantically similar.

- Step 1 allows to exclude specific types of tokens from pairing and scoring.
- Before looking for any possible matches, the token forms can be further reduced to a simpler form (step 2)
- The way tokens are matched is configured at step 3. Tokens are compared based on a selected "Token Similarity Metric" ("None" means the similarity is binary, based only on the form of the tokens: 1 for same exact forms, 0 otherwise). In addition, tokens will only be matched if their similarity score is above the "Lower match limit" threshold. Here are the available match options:

a. Greedy Pair (exclusive): each token A from one sentence is greedily matched with the most similar token B from the other sentence. If token B is already part of a previously found pair, the next most similar available token is chosen.
b. Greedy Pair (inclusive): each token A from the first sentence is greedily matched with the most similar token B from the second sentence. Tokens can belong to multiple pairs.
c. Optimal Pairing: Tokens are optimally paired (in an exclusive manner) such that the overall sum of all paired tokens values is maximum. The value of a pair is the product of both token weights and their similarity score. See the third referenced source for more details.
d. QAP Alignment: Tokens are optimally paired (in an exclusive manner) such that the overall score of both the set of the matched tokens and the set of their matched syntactic dependencies is maximized. See the third reference source for more details.

- Step 4 allows weighting the tokens based on their specificity
- The final similarity score is normalized through one of the options at the final step

References:
● *Dissertation: Chapters 3 and 4 (excluding section 4.8)*
● *Lintean, M. & Rus, V. (2012). Measuring Semantic Similarity in Short Texts through Greedy Pairing and Word Semantics. Proceedings of the 25th International Florida Artificial Intelligence Research Society Conference. Marco Island, FL.*
● *Lintean, M., & Rus, V. (2015, May). An Optimal Quadratic Approach to Monolingual Paraphrase Alignment. In Nordic Conference of Computational Linguistics NODALIDA 2015 (p. 127).*

II. Dependency Methods:

A similarity score is computed based on matched syntactic dependencies. Dependency relations must be available for this method to work.
Note: The matching is based on the "Token Similarity Metric" and the "Lower match limit" defined in the lexical methods option tab.

References:
● *Dissertation: Chapter 5*
● *Lintean, M., & Rus, V. (2010). Paraphrase Identification Using Weighted Dependencies and Word Semantics. Informatica, An International Journal of Computing and Informatics, 34(2010):19-28.*

III. LSA:

The texts are transformed into vectorial representations and a semantic similarity score is computed from the cosine of these vectors.

References:
● *Dissertation: Chapter 4, section 4.8*
● *Lintean, M., Moldovan, C., Rus, V., & McNamara D. (2010). The Role of Local and Global Weighting in Assessing the Semantic Similarity of Texts Using Latent Semantic Analysis. Proceedings of the 23rd International Florida Artificial Intelligence Research Society Conference. Daytona Beach, FL.*

IV: Corley & Mihalcea

This method is implemented as described in:

> *Corley, Courtney and Rada Mihalcea. 2005. Measuring the semantic similarity of texts. In Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment. Ann Arbor, MI.*

Options are to choose various word similarity metrics, decide the direction of the match and to include or not IDF weighting.