Human Language Technology Conference of the North American
Chapter of the Association for Computational Linguistics
Demonstrations, Boston, MA, May 2004

WordNet::Similarity - Measuring the Relatedness of Concepts

Ted Pedersen
Department of Computer Science
University of Minnesota
Duluth, MN 55812
tpederse@d.umn.edu

Siddharth Patwardhan
School of Computing
University of Utah
Salt Lake City, UT 84102
sidd@cs.utah.edu

Jason Michelizzi
Department of Computer Science
University of Minnesota
Duluth, MN 55812
mich0212@d.umn.edu

http://search.cpan.org/dist/WordNet-Similarity
http://wn-similarity.sourceforge.net

Abstract

WordNet::Similarity is a freely available software package that makes it possible to measure the semantic similarity and relatedness between a pair of concepts (or synsets). It provides six measures of similarity, and three measures of relatedness, all of which are based on the lexical database WordNet. These measures are implemented as Perl modules which take as input two concepts, and return a numeric value that represents the degree to which they are similar or related.

1 Introduction

WordNet::Similarity implements measures of similarity and relatedness that are all in some way based on the structure and content of WordNet.

Measures of similarity use information found in an *is–a* hierarchy of concepts (or synsets), and quantify how much concept A is like (or is similar to) concept B. For example, such a measure might show that an *automobile* is more like a *boat* than it is a *tree*, due to the fact that *automobile* and *boat* share *vehicle* as an ancestor in the WordNet noun hierarchy.

WordNet is particularly well suited for similarity measures, since it organizes nouns and verbs into hierarchies of *is–a* relations. In version 2.0, there are nine separate noun hierarchies that include 80,000 concepts, and 554 verb hierarchies that are made up of 13,500 concepts.

Is–a relations in WordNet do not cross part of speech boundaries, so similarity measures are limited to making judgments between noun pairs (e.g., *cat* and *dog*) and verb pairs (e.g., *run* and *walk*). While WordNet also includes adjectives and adverbs, these are not organized into *is–a* hierarchies so similarity measures can not be applied.

However, concepts can be related in many ways beyond being similar to each other. For example, a *wheel* is a part of a *car*, *night* is the opposite of *day*, *snow* is made up of *water*, a *knife* is used to cut *bread*, and so forth. As such WordNet provides relations beyond *is–a*, including *has–part*, *is–made–of*, and *is–an–attribute–of*. In addition, each concept is defined by a short gloss that may include an example usage. All of this information can be brought to bear in creating measures of relatedness. As a result these measures tend to be more flexible, and allow for relatedness values to be assigned across parts of speech (e.g., the verb *murder* and the noun *gun*).

This paper continues with an overview of the measures supported in WordNet::Similarity, and then provides a brief description of how the package can be used. We close with a summary of research that has employed WordNet::Similarity.

2 Similarity Measures

Three of the six measures of similarity are based on the *information content* of the least common subsumer (LCS) of concepts A and B. Information content is a measure of the specificity of a concept, and the LCS of concepts A and B is the most specific concept that is an ancestor of both A and B. These measures include res (Resnik, 1995), lin (Lin, 1998), and jcn (Jiang and Conrath, 1997).

The lin and jcn measures augment the information content of the LCS with the sum of the information content of concepts A and B themselves. The lin measure scales the information content of the LCS by this sum, while jcn takes the difference of this sum and the information content of the LCS.

The default source for information content for concepts is the sense–tagged corpus SemCor. However, there are also utility programs available with WordNet::Similarity that allow a user to compute information content values from the Brown Corpus, the Penn Treebank, the British National Corpus, or any given corpus of raw text.

```
> similarity.pl --type WordNet::Similarity::lin car#n#2 bus#n#1
car#n#2  bus#n#1  0.530371390319309        # railway car versus motor coach

> similarity.pl --type WordNet::Similarity::lin car#n bus#n
car#n#1  bus#n#1  0.618486790769613        # automobile versus motor coach

> similarity.pl --type WordNet::Similarity::lin --allsenses car#n bus#n#1
car#n#1  bus#n#1  0.618486790769613        # automobile versus motor coach
car#n#2  bus#n#1  0.530371390319309        # railway car versus motor coach
car#n#3  bus#n#1  0.208796988315133        # cable car versus motor coach
```

Figure 1: Command Line Interface

Three similarity measures are based on path lengths between a pair of concepts: lch (Leacock and Chodorow, 1998), wup (Wu and Palmer, 1994), and path. lch finds the shortest path between two concepts, and scales that value by the maximum path length found in the *is–a* hierarchy in which they occur. wup finds the depth of the LCS of the concepts, and then scales that by the sum of the depths of the individual concepts. The depth of a concept is simply its distance to the root node. The measure path is a baseline that is equal to the inverse of the shortest path between two concepts.

WordNet::Similarity supports two hypothetical root nodes that can be turned on and off. When on, one root node subsumes all of the noun concepts, and another subsumes all of the verb concepts. This allows for similarity measures to be applied to any pair of nouns or verbs. If the hypothetical root nodes are off, then concepts must be in the same physical hierarchy for a measurement to be taken.

3 Measures of Relatedness

Measures of relatedness are more general in that they can be made across part of speech boundaries, and they are not limited to considering *is-a* relations. There are three such measures in the package: hso (Hirst and St-Onge, 1998), lesk (Banerjee and Pedersen, 2003), and vector (Patwardhan, 2003).

The hso measures classifies relations in WordNet as having direction, and then establishes the relatedness between two concepts A and B by finding a path that is neither too long nor that changes direction too often.

The lesk and vector measures incorporate information from WordNet glosses. The lesk measure finds overlaps between the glosses of concepts A and B, as well as concepts that are directly linked to A and B. The vector measure creates a co–occurrence matrix for each word used in the WordNet glosses from a given corpus, and then represents each gloss/concept with a vector that is the average of these co–occurrence vectors.

4 Using WordNet::Similarity

WordNet::Similarity can be utilized via a command line interface provided by the utility program *similarity.pl*. This allows a user to run the measures interactively. In addition, there is a web interface that is based on this utility. WordNet::Similarity can also be embedded within Perl programs by including it as a module and calling its methods.

4.1 Command Line

The utility *similarity.pl* allows a user to measure specific pairs of concepts when given in *word#pos#sense* form. For example, *car#n#3* refers to the third WordNet noun sense of *car*. It also allows for the specification of all the possible senses associated with a *word* or *word#pos* combination.

For example, in Figure 1, the first command requests the value of the lin measure of similarity for the second noun sense of *car* (railway car) and the first noun sense of *bus* (motor coach). The second command will return the score of the pair of concepts that have the highest similarity value for the nouns *car* and *bus*. In the third command, the *–allsenses* switch causes the similarity measurements of all the noun senses of *car* to be calculated relative to the first noun sense of *bus*.

4.2 Programming Interface

WordNet::Similarity is implemented with Perl's object oriented features. It uses the WordNet::QueryData package (Rennie, 2000) to create an object representing WordNet. There are a number of methods available that allow for the inclusion of existing measures in Perl source code, and also for the development of new measures.

When an existing measure is to be used, an object of that measure must be created via the *new()* method. Then the *getRelatedness()* method can be called for a pair of word senses, and this will return the relatedness value. For example, the program in Figure 2 creates an object of the lin measure, and then finds the similarity between the

```
#!/usr/bin/perl -w

use WordNet::QueryData;                          # use interface to WordNet
use WordNet::Similarity::lin;                    # use Lin measure

 $wnObj = new WordNet::QueryData;                  # create a WordNet object
$linObj = new WordNet::Similarity::lin($wnObj); # create a lin object

$value = $linObj -> getRelatedness ('car#n#1', 'bus#n#2');  # how similar?
```

Figure 2: Programming Interface

first sense of the noun *car* (automobile) and the second sense of the noun *bus* (network bus).

WordNet::Similarity enables detailed tracing that shows a variety of diagnostic information specific to each of the different kinds of measures. For example, for the measures that rely on path lengths (lch, wup, path) the tracing shows all the paths found between the concepts. Tracing for the information content measures (res, lin, jcn) includes both the paths between concepts as well as the least common subsumer. Tracing for the hso measure shows the actual paths found through WordNet, while the tracing for lesk shows the gloss overlaps in WordNet found for the two concepts and their nearby relatives. The vector tracing shows the word vectors that are used to create the gloss vector of a concept.

5 Software Architecture

Similarity.pm is the super class of all modules, and provides general services used by all of the measures such as validation of synset identifier input, tracing, and caching of results. There are four modules that provide all of the functionality required by any of the supported measures: *PathFinder.pm*, *ICFinder.pm*, *DepthFinder.pm*, and *LCSFinder.pm*.

PathFinder.pm provides *getAllPaths()*, which finds all of the paths and their lengths between two input synsets, and *getShortestPath()* which determines the length of the shortest path between two concepts.

ICFinder.pm includes the method *IC()*, which gets the information content value of a synset. *probability()* and *getFrequency()* find the probability and frequency count of a synset based on whatever corpus has been used to compute information content. Note that these values are pre–computed, so these methods are simply reading from an information content file.

DepthFinder.pm provides methods that read values that have been pre–computed by the *wnDepths.pl* utility. This program finds the depth of every synset in WordNet, and also shows the *is–a* hierarchy in which a synset occurs. If a synset has multiple parents, then each possible depth and home hierarchy is returned. The depth of a synset is returned by *getDepthOfSynset()* and *getTaxonomyDepth()* provides the maximum depth for a given *is–a* hierarchy.

LCSFinder.pm provides methods that find the least common subsumer of two concepts using three different criteria. These are necessary since there is multiple inheritance of concepts in WordNet, and different LCS can be selected for a pair of concepts if one or both of them have multiple parents in an *is–a* hiearchy. *getLCSbyIC()* chooses the LCS for a pair of concepts that has the highest information content, *getLCSbyDepth()* selects the LCS with the greatest depth, and *getLCSbyPath()* selects the LCS that results in the shortest path.

6 Related Work

Our work with measures of semantic similarity and relatedness began while adapting the Lesk Algorithm for word sense disambiguation to WordNet (Banerjee and Pedersen, 2002). That evolved in a generalized approach to disambiguation based on semantic relatedness (Patwardhan et al., 2003) that is implemented in the SenseRelate package (*http://senserelate.sourceforge.net*), which utilizes WordNet::Similarity. The premise behind this algorithm is that the sense of a word can be determined by finding which of its senses is most related to the possible senses of its neighbors.

WordNet::Similarity has been used by a number of other researchers in an interesting array of domains. (Zhang et al., 2003) use it as a source of semantic features for identifying cross–document structural relationships between pairs of sentences found in related documents. (McCarthy et al., 2004) use it in conjunction with a thesaurus derived from raw text in order to automatically identify the predominent sense of a word. (Jarmasz and Szpakowicz, 2003) compares measures of similarity derived from WordNet and Roget's Thesaurus. The comparisons are based on correlation with human relatedness values, as well as the TOEFL synonym identification tasks. (Baldwin et al., 2003) use WordNet::Similarity to provide an evaluation tool for multiword expressions that are identified via Latent Semantic Analysis. (Diab, 2003)

combines a number of similarity measures that are then used as a feature in the disambiguation of verb senses.

7 Availability

WordNet::Similarity is written in Perl and is freely distributed under the Gnu Public License. It is available from the Comprehensive Perl Archive Network (*http://search.cpan.org/dist/WordNet-Similarity*) and via SourceForge, an Open Source development platform (*http://wn-similarity.sourceforge.net*).

8 Acknowledgements

WordNet::Similarity was preceeded by the distance.pl program, which was released in June 2002. This was converted into the object oriented WordNet::Similarity package, which was first released in April 2003 as version 0.03. The most current version as of this writing is 0.07, which was released in March 2004.

The distance.pl program and all versions of WordNet::Similarity up to and including 0.06 were designed and implemented by Siddharth Patwardhan as a part of his Master's thesis at the University of Minnesota, Duluth. Version 0.07 was designed and implemented by Jason Michelizzi as a part of his Master's thesis.

The lesk measure in WordNet::Similarity was originally designed and implemented by Satanjeev Banerjee, who developed this measure as a part of his Master's thesis at the University of Minnesota, Duluth. Thereafter Siddharth Patwardhan ported this measure to WordNet::Similarity.

This work has been partially supported by a National Science Foundation Faculty Early CAREER Development award (#0092784), and by a Grant-in-Aid of Research, Artistry and Scholarship from the Office of the Vice President for Research and the Dean of the Graduate School of the University of Minnesota.

References

T. Baldwin, C. Bannard, T. Tanaka, and D. Widdows. 2003. An empirical model of multiword expression decomposability. In *Proceedings of the of the ACL-2003 Workshop on Multiword Expressions: Analysis, Acquisition and Treatment*, pages 89–96, Sapporo, Japan.

S. Banerjee and T. Pedersen. 2002. An adapted Lesk algorithm for word sense disambiguation using WordNet. In *Proceedings of the Third International Conference on Intelligent Text Processing and Computational Linguistics*, pages 136–145, Mexico City, February.

S. Banerjee and T. Pedersen. 2003. Extended gloss overlaps as a measure of semantic relatedness. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*, pages 805–810, Acapulco, August.

M. Diab. 2003. *Word Sense Disambiguation within a Multilingual Framework*. Ph.D. thesis, The University of Maryland.

G. Hirst and D. St-Onge. 1998. Lexical chains as representations of context for the detection and correction of malapropisms. In C. Fellbaum, editor, *WordNet: An electronic lexical database*, pages 305–332. MIT Press.

M. Jarmasz and S. Szpakowicz. 2003. Roget's thesaurus and semantic similarity. In *Proceedings of the Conference on Recent Advances in Natural Language Processing*, pages 212–219, Borovets, Bulgaria.

J. Jiang and D. Conrath. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings on International Conference on Research in Computational Linguistics*, pages 19–33, Taiwan.

C. Leacock and M. Chodorow. 1998. Combining local context and WordNet similarity for word sense identification. In C. Fellbaum, editor, *WordNet: An electronic lexical database*, pages 265–283. MIT Press.

D. Lin. 1998. An information-theoretic definition of similarity. In *Proceedings of the International Conference on Machine Learning*, Madison, August.

D. McCarthy, R. Koeling, and J. Weeds. 2004. Ranking WordNet senses automatically. Technical Report CSRP 569, University of Sussex, January.

S. Patwardhan, S. Banerjee, and T. Pedersen. 2003. Using measures of semantic relatedness for word sense disambiguation. In *Proceedings of the Fourth International Conference on Intelligent Text Processing and Computational Linguistics*, pages 241–257, Mexico City, February.

S. Patwardhan. 2003. Incorporating dictionary and corpus information into a context vector measure of semantic relatedness. Master's thesis, University of Minnesota, Duluth, August.

J. Rennie. 2000. WordNet::QueryData: a Perl module for accessing the WordNet database. http://www.ai.mit.edu/people/jrennie/WordNet.

P. Resnik. 1995. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pages 448–453, Montreal, August.

Z. Wu and M. Palmer. 1994. Verb semantics and lexical selection. In *32nd Annual Meeting of the Association for Computational Linguistics*, pages 133–138, Las Cruces, New Mexico.

Z. Zhang, J. Otterbacher, and D. Radev. 2003. Learning cross-document structural relationships using boosting. In *Proceedings of the 12th International Conference on Information and Knowledge Management*, pages 124–130.