WordNet Similarity Metrics

Charlie Greenbacker

based on material from Jurafsky and Martin,
Speech and Language Processing (2nd Ed.)
Ch. 20.6 Word Similarity: Thesaurus Methods,
plus NLTK documentation

Overview

- Defining "word similarity"
- Use in Assignment 3
- Thesaurus vs. Distributional methods
- Five thesaurus-based similarity measures
- Tools for implementation

"Word Similarity"

- Synonymy
- Antonymy
- Hyponymy
- Hypernymy
- Meronymy

"Word Similarity"

- **Synonymy**
- Antonymy
- Hyponymy
- Hypernymy
- Meronymy

"Word Similarity"

- **Synonymy**
  - Binary relation between words
  - Two words are either synonyms or not

"Word Similarity"

- **Synonymy**
  - Binary relation between words
  - Two words are either synonyms or not
- **Similarity**
  - Looser metric of semantic distance

"Word Similarity"

- **Similarity**
  - More similar if words share more features of meaning
    - "near-synonyms"
- Less similar if words have fewer common meaning elements
  - greater "semantic distance"

“Word Similarity”

- Relations between words

“Word Similarity”

- ~~Relations between words~~

"Word Similarity"

- ~~Relations between words~~
- Relations between senses

"Word Similarity"

- ~~Relations between words~~
- Relations between senses
  - "bank" (financial sense) is more similar to "fund"
  - "bank" (river sense) is more similar to "slope"

Use in Assignment 3

- **Spelling Correction with Semantics**
  - Rank candidate corrections by similarity to nearby words
  - Compare candidates to words in some context window
  - Combine with minimum distance edit (or similar) for ranking

Thesaurus vs. Distributional

- Thesaurus methods
  - Measure distance between two senses
    - Use on-line thesaurus (e.g., WordNet, MeSH)
- Distributional methods
  - Estimate word similarity
  - Find words that have similar distributions in a corpus

Thesaurus vs. Distributional

- **Thesaurus methods**
  - Measure distance between two senses
  - Use on-line thesaurus (e.g., **WordNet**, MeSH)
- Distributional methods
  - Estimate word similarity
  - Find words that have similar distributions in a corpus

Five similarity measures

- Path length
- Resnik similarity
- Lin similarity
- Jiang-Conrath distance
- Extended Lesk measure

1. Path length

- Intuition: *the shorter the* **path** *between two words/senses in a thesaurus hierarchy graph, the more similar they are*
  - Words are quite similar to parents & siblings
  - Less similar to words far away in the network

1. Path length

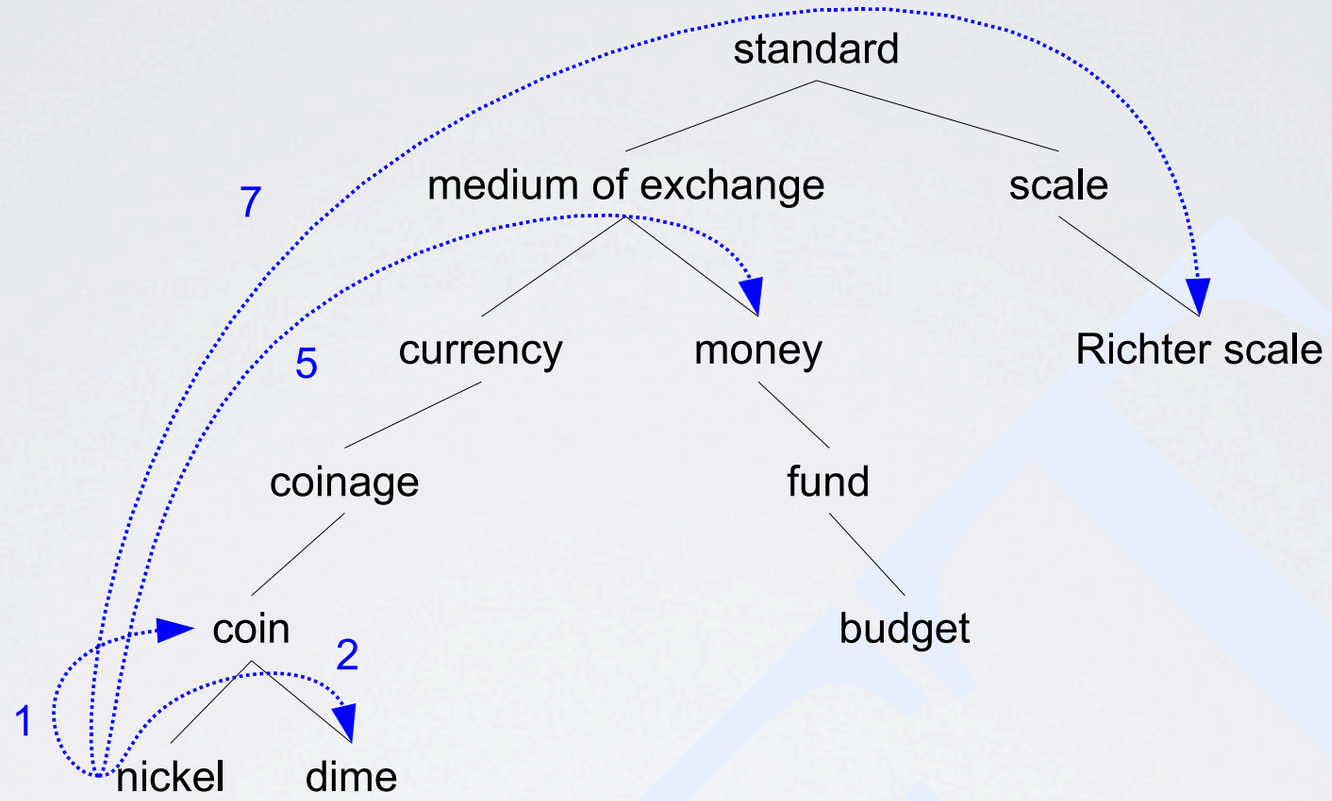

Figure ~~19.6~~ 20.6 A fragment of the WordNet hypernym hierarchy, showing path lengths from *nickel* to *coin* (1), *dime* (2), *money* (5), and *Richter scale* (7).

$pathlen(c_1, c_2)$ = number of edges in shortest path

1. Path length

$pathlen(c_1, c_2)$ = number of edges in shortest path

- Path-based similarity often involves a log transform
  - **path-length based similarity**:

$$sim_{path}(c_1, c_2) = -\log pathlen(c_1, c_2)$$

- Weaknesses: requires sense-tagged data, assumes uniform cost

2. Resnik similarity

- **information-content word-similarity**:
  - Still relies on structure of thesaurus
  - Refines path-based approach using normalizations based on hierarchy depth
  - Represents distance associated with each edge
  - Adds probabilistic information derived from a corpus

2. Resnik similarity

- Probability of random word being an instance of concept c:

$$P(c) = \frac{\sum_{w \in words(c)} count(w)}{N}$$

where $words(c)$ is set of words subsumed by concept c,
N is the number of words in corpus and also in thesaurus

- $P(root) = 1$ since all words are subsumed by root concept

- The lower a concept in the hierarchy, the lower the probability

2. Resnik similarity

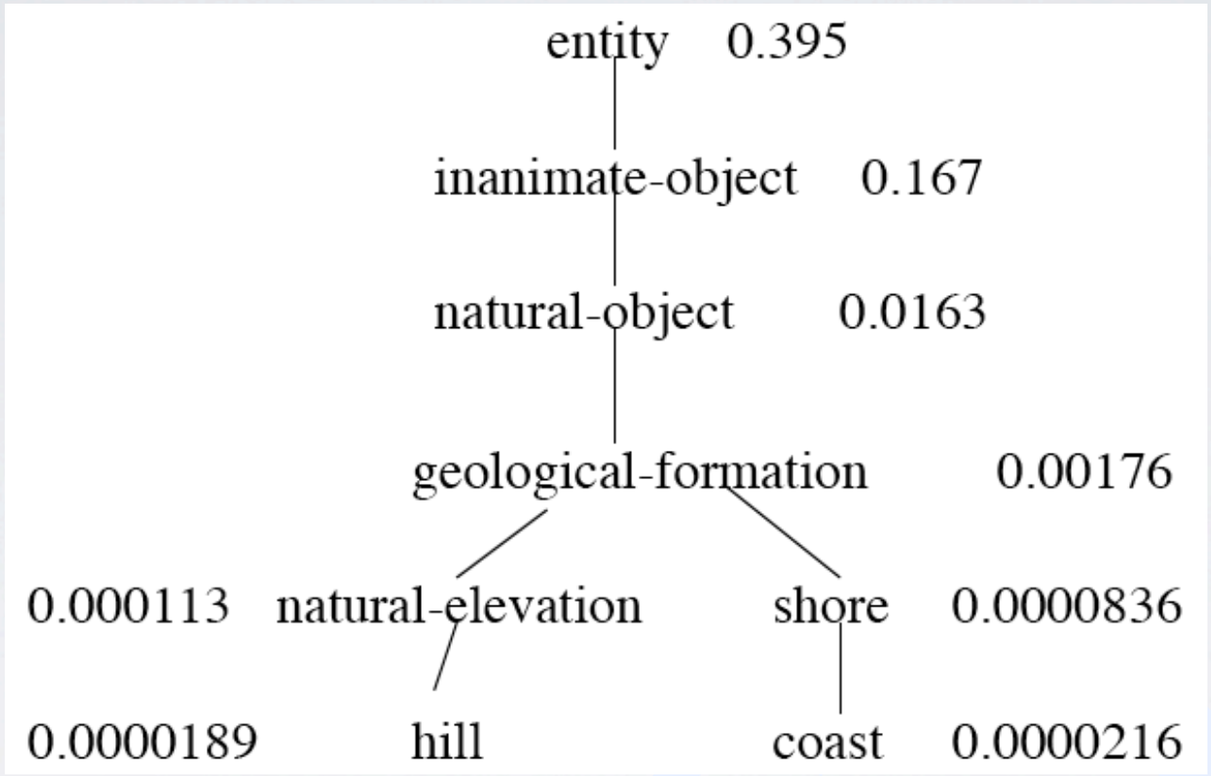

Figure ~~19.7~~20.7PLACEHOLDER FIGURE: A fragment of the WordNet hierarchy, showing the probability $p(c)$ attached to each content, from Lin (1998b)

- Train probabilities by counting in a corpus: each word counts as an occurrence of all concepts "containing" it

2. Resnik similarity

- Two more definitions are needed...
  - **Information content** of a concept c: $IC(c) = -\log P(c)$
    - basic information theory
- **Lowest common subsumer**: $LCS(c_1, c_2)$

  = lowest node in hierarchy that is a hypernym of c_1 & c_2

2. Resnik similarity

- Finally... the Resnik similarity measure:

$$sim_{Resnik}(c_1, c_2) = -\log P(LCS(c_1, c_2))$$

- estimates common amount of information between words by **information content** of **lowest common subsumer**

3. Lin similarity

- Similarity is about more than just common information... the more **differences** between A & B, the less similar they are

  - Commonality: $IC(common(A, B))$
  - Difference: $IC(description(A, B)) - IC(common(A, B))$

    where $description(A, B)$ "describes" A and B

3. Lin similarity

- **Similarity Theorem**: The Similarity between A and B is measured by the ratio between the amount of information needed to state the commonality of A and B and the information needed to fully describe what A and B are.

$$sim_{Lin}(A, B) = \frac{common(A, B)}{description(A, B)}$$

- the information in common between two concepts is twice the information in the lowest common subsumer

3. Lin similarity

- Final **Lin similarity function** for concepts in a thesaurus:

$$sim_{Lin}(c_1, c_2) = \frac{2 \times \log P(LCS(c_1, c_2))}{\log P(c_1) + \log P(c_2)}$$

- For example, from Figure 20.7:

$$sim_{Lin}(hill, coast) = \frac{2 \times \log P(geological\text{-}formation)}{\log P(hill) + \log P(coast)} = 0.59$$

4. Jiang-Conrath distance

- Related to sim_{Lin} - expressed as distance instead of similarity:

$$dist_{JC}(c_1, c_2) = 2 \times \log P(LCS(c_1, c_2)) - (\log P(c_1) + \log P(c_2))$$

- Transform into a similarity measure by taking the reciprocal:

$$sim_{JC}(c_1, c_2) = \frac{1}{2 \times \log P(LCS(c_1, c_2)) - (\log P(c_1) + \log P(c_2))}$$

5. Extended Lesk measure

- Extends Lesk algorithm for word sense disambiguation
- **Dictionary-based** method
- makes use of glosses, a property of dictionaries

5. Extended Lesk measure

- **Extended gloss overlap**: two concepts/senses are similar if their glosses contain overlapping words
  - *drawing paper*: paper that is specially prepared for use in drafting
  - *decal*: the art of transferring designs from specially prepared paper to a wood or glass or metal surface
- For each n-word phrase seen in both glosses, eLesk adds n^2; longer overlaps are rare, and should be weighted more heavily

5. Extended Lesk measure

- **Extended gloss overlap**: two concepts/senses are similar if their glosses contain overlapping words
  - *drawing paper*: paper that is specially prepared for use in drafting
  - *decal*: the art of transferring designs from specially prepared paper to a wood or glass or metal surface

$$1^2 + 2^2 = 5$$

- For each n-word phrase seen in both glosses, eLesk adds n^2; longer overlaps are rare, and should be weighted more heavily

5. Extended Lesk measure

- Extends Lesk looks for overlap in hypernyms, hyponyms, meronyms, and other relations of the two concepts... not just in the glosses of the two synsets
  - if considering hyponyms only, and $gloss(hypo(A))$ as the concatenation of all glosses of all hyponym senses of A, then the total relationship between concepts A and B is:

$$\begin{aligned} similarity(A,B) = & \; overlap(gloss(A), gloss(B)) \\ & + overlap(gloss(hypo(A)), gloss(hypo(B))) \\ & + overlap(gloss(A), gloss(hypo(B))) \\ & + overlap(gloss)hypo(A)), gloss(B)) \end{aligned}$$

5. Extended Lesk measure

- Let $RELS$ be the set of possible WordNet relations with glosses we compare, then define **Extended Lesk** measure as:

$$sim_{eLesk}(c_1, c_2) = \sum_{r,q \in RELS} overlap(gloss(r(c_1)), gloss(q(c_2)))$$

Implementation Tools

- No need to implement these similarity metrics on your own
  - NLTK WordNet interface for Python
  - `Wordnet::Similarity` package for Perl
    - ported to Java: http://www.cogs.susx.ac.uk/users/drh21/

NLTK WordNet Interface

```
>>> import nltk
>>> from nltk.corpus import wordnet as wn
>>> dog = wn.synset('dog.n.01')
>>> cat = wn.synset('cat.n.01')
>>> from nltk.corpus import wordnet_ic
>>> brown_ic = wordnet_ic.ic('ic-brown.dat')
>>> dog.path_similarity(cat)
0.2
>>> dog.res_similarity(cat, brown_ic)
7.911666509036577
>>> dog.lin_similarity(cat, brown_ic)
0.8768009843733973
>>> dog.jcn_similarity(cat, brown_ic)
0.449775285516739
>>> # NLTK does not implement eLesk(?)
>>> # but offers Leacock-Chodorow & Wu-Palmer measures
```

Wordnet::Similarity

- Includes all five metrics discussed (even eLesk), and others
  - Sourceforge: http://wn-similarity.sourceforge.net/
  - CPAN: http://search.cpan.org/dist/WordNet-Similarity/
  - Web interface demos (servers are often busy):
    - http://marimba.d.umn.edu/cgi-bin/similarity/similarity.cgi
    - http://talisker.d.umn.edu/cgi-bin/similarity/similarity.cgi

So which metric is best?

- That's for **you** to figure out!