Unit 4- Continue

Word Senses and relations

- **Word Senses** refer to the different meanings a word can have in various contexts.
- Each distinct meaning of a word is called a "sense."
- The study of word senses is critical in natural language processing (NLP) for understanding and generating human language accurately.
- **Relationships Between Senses** are the semantic connections that exist between different senses of words.
- WordNet, a lexical database of English, extensively documents these relationships.

Types of Word Senses

- **Synonymy**:
  - Words with the **same** or nearly the same meaning.
  - Example: {car, automobile}
- **Antonymy**:
  - Words with **opposite** meanings.
  - Example: {hot, cold}
- **Hyponymy**:
  - A specific word whose meaning is included in a more **general** word.
  - Example: {dog} (hyponym of {animal})
- **Hypernymy**:
  - A general word that includes the meanings of more **specific** words.
  - Example: {animal} (hypernym of {dog})

- **Meronymy**:
  - A word that denotes a **part** of a larger whole.
  - Example: {wheel} (meronym of {car})
- **Holonymy**:
  - A word that denotes a **whole** of which a part is mentioned.
  - Example: {car} (holonym of {wheel})
- **Troponymy**:
  - A verb that denotes a specific **manner** of performing another verb.
  - Example: {to jog} (troponym of {to run})
- **Entailment**:
  - A verb that **implies** the action of another verb.
  - Example: {to snore} (entails {to sleep})
- **Coordinate Terms**:
  - Words that share a common hypernym and are at the same level of specificity.
  - Example: {car, bus, bicycle} (coordinate terms under {vehicle})

Word Sense Disambiguation (WSD)

- Word Sense Disambiguation (WSD) is the process of determining which sense of a word is used in a given context. This is essential for accurate language understanding in applications such as machine translation, information retrieval, and more.

Approaches to WSD:

1. Supervised Methods:

1. Use labeled data to train machine learning models to predict the correct word sense based on context.
2. Techniques: Support Vector Machines (SVM), Neural Networks.

2. Unsupervised Methods:

1. Cluster contexts of word usage to identify different senses without labeled data.
2. Techniques: Clustering algorithms, distributional similarity.

3. Knowledge-Based Methods:

1. Leverage lexical resources like WordNet to determine the correct sense.
2. Techniques: Lesk algorithm, similarity measures based on definitions.

4. Contextualized Embeddings:

1. Use deep learning models (e.g., BERT) to generate context-aware word embeddings that capture different senses.
2. Example: BERT can distinguish between "bank" in "river bank" and "financial bank" based on context.

Disambiguation Process:

1. Identify the context words surrounding the target word.
2. Compare the context with the definitions (glosses) of each sense in WordNet.
3. Choose the sense with the highest overlap or most relevant meaning based on the context.

The Simplified Lesk algorithm

- Let's disambiguate "**bank**" in this sentence:

  The **bank** can guarantee deposits will eventually cover future tuition costs because it invests in adjustable-rate mortgage securities.

- given the following two WordNet senses:

$bank^1$	Gloss:	a financial institution that accepts deposits and channels the money into lending activities
	Examples:	"he cashed a check at the bank", "that bank holds the mortgage on my home"
$bank^2$	Gloss:	sloping land (especially the slope beside a body of water)
	Examples:	"they pulled the canoe up on the bank", "he sat on the bank of the river and watched the currents"

The Simplified Lesk algorithm

Choose sense with most word overlap between gloss and context
(not counting function words)

The **bank** can guarantee deposits will eventually cover future tuition costs because it invests in adjustable-rate mortgage securities.

$bank^1$	Gloss:	a financial institution that accepts deposits and channels the money into lending activities
	Examples:	"he cashed a check at the bank", "that bank holds the mortgage on my home"
$bank^2$	Gloss:	sloping land (especially the slope beside a body of water)
	Examples:	"they pulled the canoe up on the bank", "he sat on the bank of the river and watched the currents"

Drawback

- Glosses and examples migh be too short and may not provide enough chance to overlap with the context of the word to be disambiguated.

The Corpus(--based) Lesk algorithm

- Assumes we have some sense--labeled data (like SemCor)
- Take all the sentences with the relevant word sense:

  *These short, "streamlined" meetings usually are sponsored by local **banks**[1], Chambers of Commerce, trade associations, or other civic organizations.*

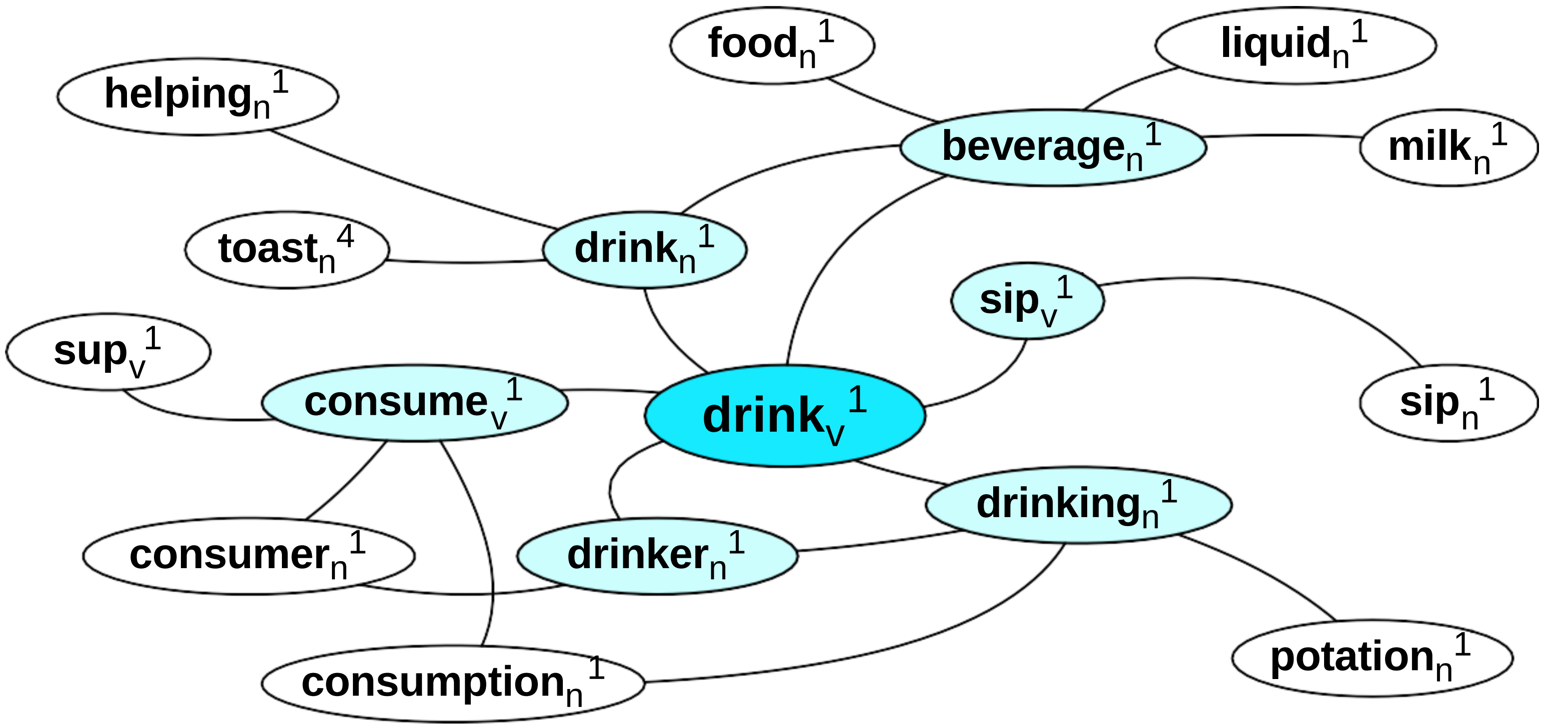
- Now add these to the gloss + examples for each sense, call it the "signature" of a sense. Basically, it is an expansion of the dictionary entry.
- Choose sense with most word overlap between context and signature (ie. the context words provided by the resources).

Corpus Lesk: IDF weighting

- Instead of just removing function words
  - Weigh each word by its `promiscuity' across documents
  - Down--weights words that occur in every `document' (gloss, example, etc)
  - These are generally function words, but is a more fine--grained measure
- Weigh each overlapping word by **inverse document frequency (IDF)**.

Graph based methods

- First, WordNet can be viewed as a graph
  - senses are nodes
  - relations (hypernymy, meronymy) are edges
  - Also add edge between word and unambiguous gloss words



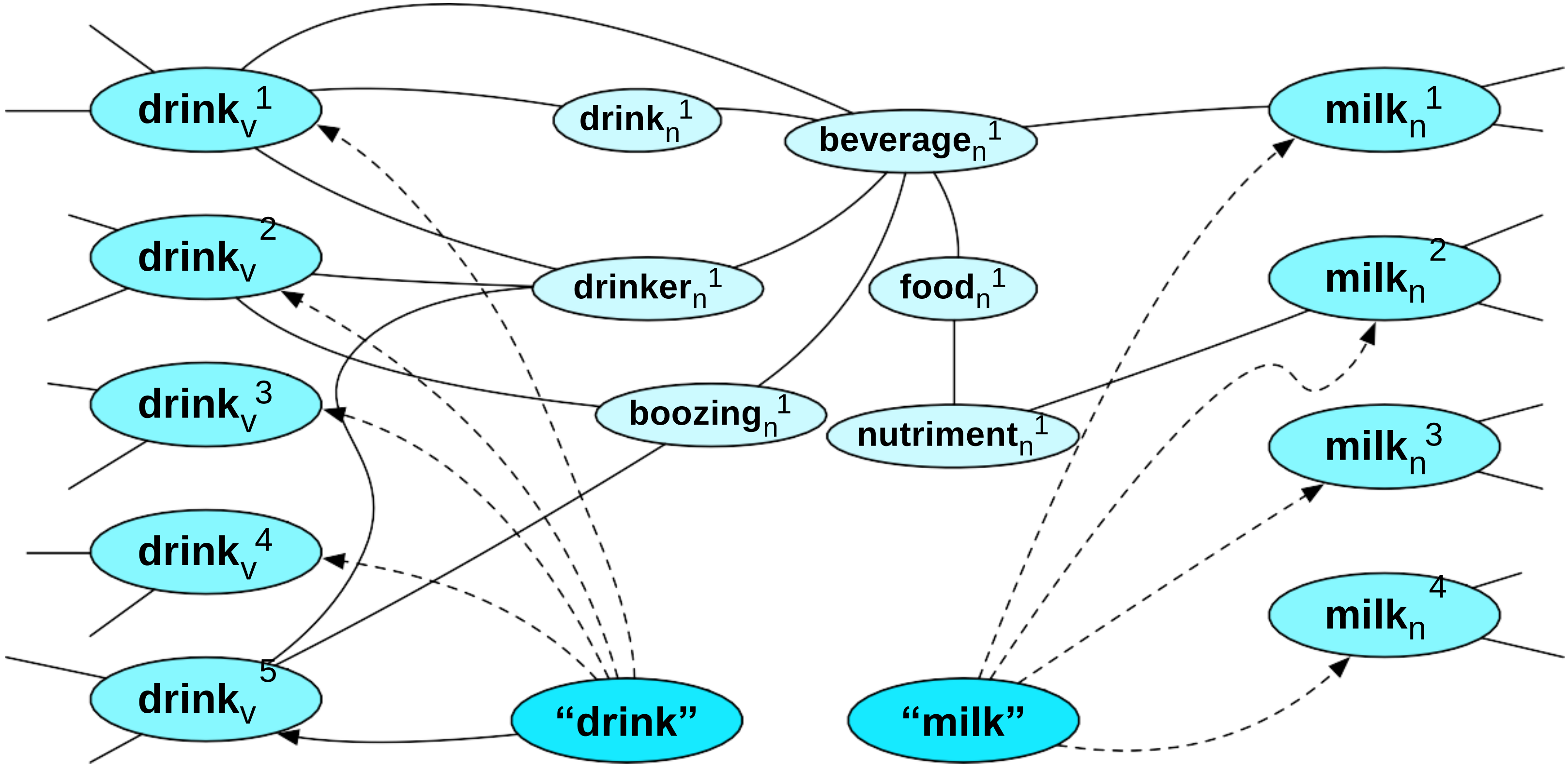
An undirected graph is set of nodes tha are connected together by bidirectional edges (lines).

How to use the graph for WSD

“She drank some milk”

- choose the *most central* sense

(several algorithms have been proposed recently)



Word Meaning and Similarity

Word Similarity: Thesaurus Methods

Word Similarity

- **Synonymy**: a binary relation
  - Two words are either synonymous or not
- **Similarity** (or **distance**): a looser metric
  - Two words are more similar if they share more features of meaning
- Similarity is properly a relation between **senses**
  - We do not say "The word "`bank`" is not similar to the word "`slope`" ", bu w say.
    - Bank^1 is similar to fund^3
    - Bank^2 is similar to slope^5
- But we'll compute similarity over both words and senses

Why word similarity

- Information retrieval
- Question answering
- Machine translation
- Natural language generation
- Language modeling
- Automatic essay grading
- Plagiarism detection
- Document clustering

Word similarity and word relatedness

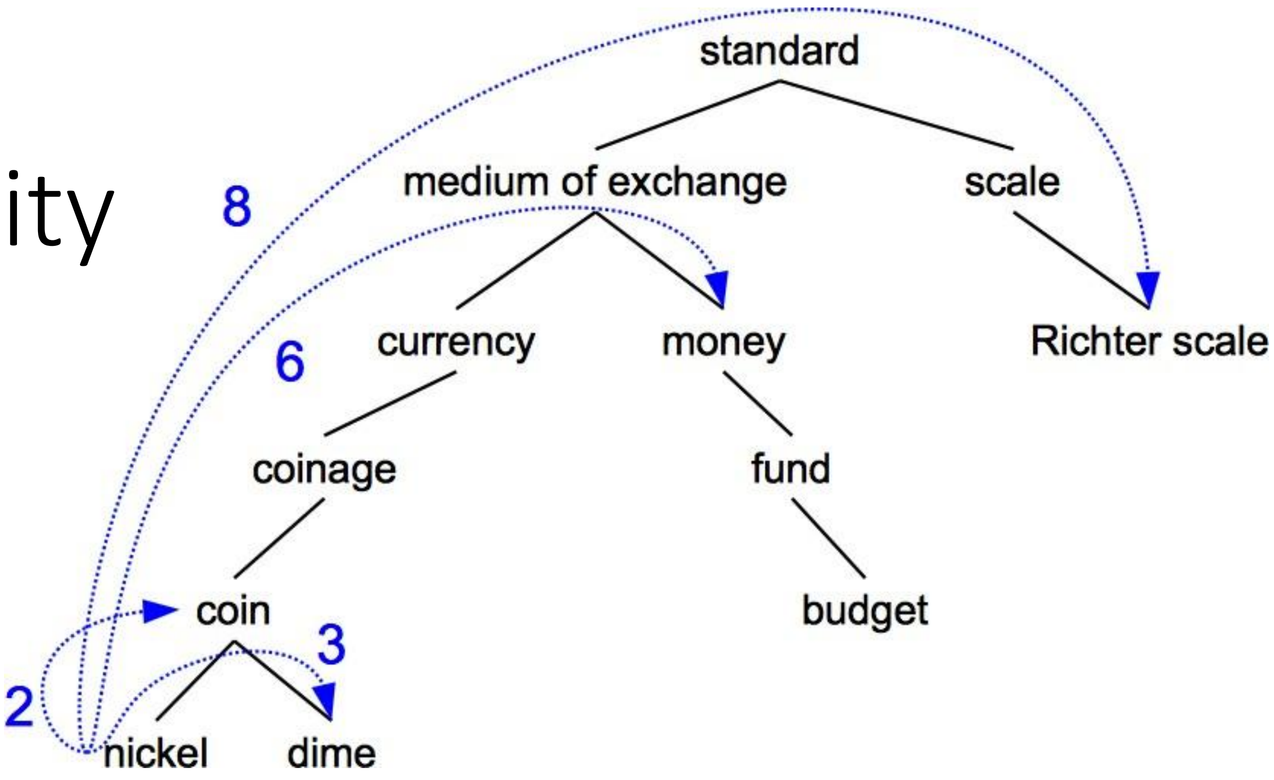
- We often distinguish **word similarity** from **word relatedness**
  - **Similar words**: near--synonyms
    - `car, bicycle:` **similar**
  - **Related words**: can be related any way
    - `car, gasoline:` **related**, not similar

Cf. Synonyms: car & automobile

Two classes of similarity algorithms

- Thesaurus--based algorithms
  - Are words “nearby” in hypernym hierarchy?
  - Do words have similar glosses (definitions)?
- Distributional algorithms:

Path-based similarity



- Two concepts (senses/synsets) are similar if they are near each other in the thesaurus hierarchy
  - =have a short path between them
  - concepts have path 1 to themselves

Refinements to path--based similarity

- $\text{pathlen}(c_1,c_2)$ = **(distance metric)** = 1 + number of edges in the shortest path in the hypernym graph between *sense nodes* c_1 and c_2

- $$\text{simpath}(c_1,c_2) = \frac{1}{\text{pathlen}(c_1, c_2)}$$

Sense similarity metric: 1 over the distance!

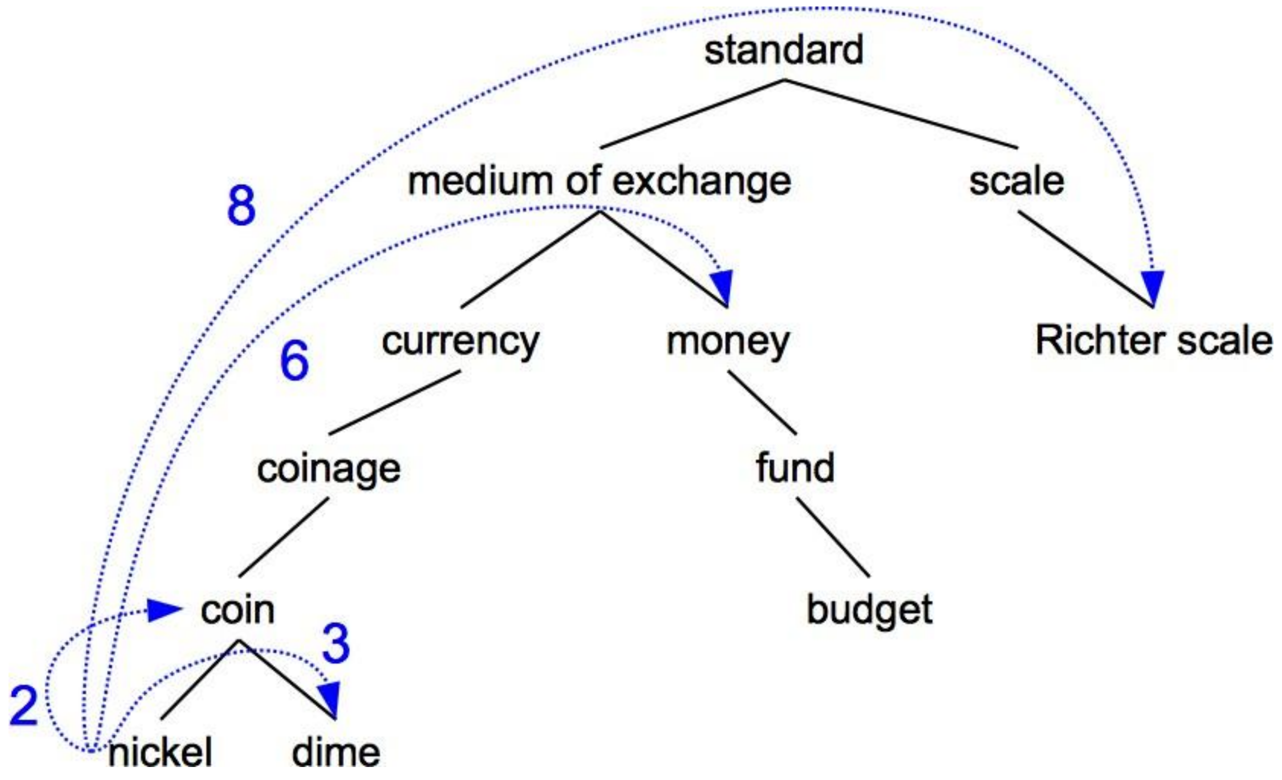
- $$\text{wordsim}(w_1,w_2) = \max_{c_1 \in \text{senses}(w_1), c_2 \in \text{senses}(w_2)} \text{sim}(c_1,c_2)$$

Word similarity metric: max similarity among pairs of senses.

For all senses of w1 and all senses of w2, take the similarity between each of the senses of w1 and each of the senses of w2 and then take the maximum similarity between those pairs.

Example: path--based similarity

$$\text{simpath}(c_1,c_2) = 1/\text{pathlen}(c_1,c_2)$$



simpath(*nickel,coin*) = 1/2 = .5

simpath(*fund,budget*) = 1/2 = .5

simpath(*nickel,currency*) = 1/4 = .25

simpath(*nickel,money*) = 1/6 = .17

simpath(*coinage,Richter scale*) = 1/6 = .17

Problem with basic path--based similarity

- Assumes each link represents a uniform distance
  - But *nickel* to *money* seems to us to be closer than *nickel* to *standard*
  - Nodes high in the hierarchy are very abstract
- We instead want a metric that
  - Represents the cost of each edge independently
  - Words connected only through abstract nodes
    - are less similar

Information content similarity metrics

Resnik 1995. Using information content to evaluate semantic similarity in a taxonomy. IJCAI

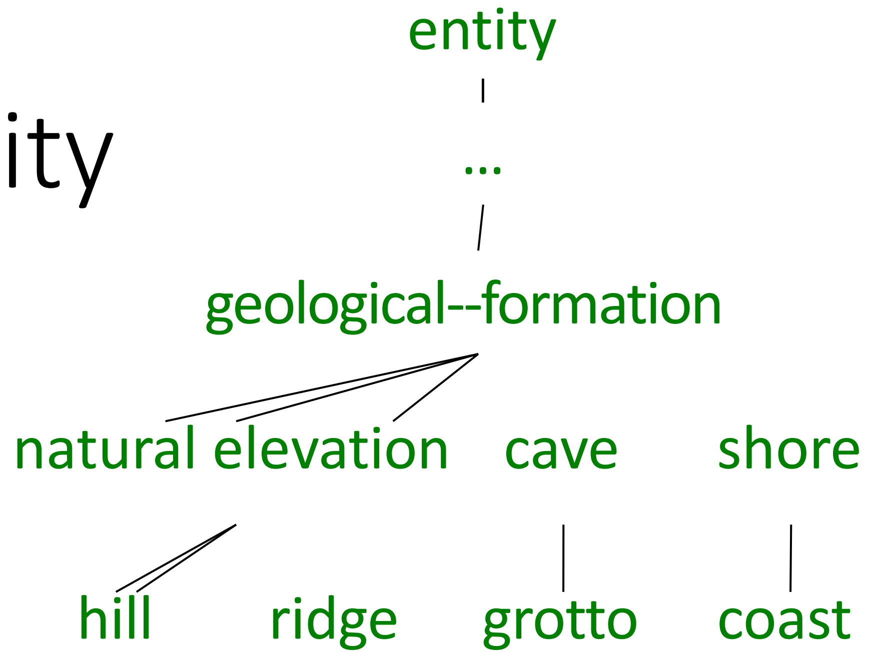
- In simple words:
  - We define the probability of a concept C as the probability that a randomly selected word in a corpus is an instance of that concept.
  - Basically, for each random word in a corpus we compute how probable it is that it belongs to a certain concepts.

Formally: Information content similarity metrics

Resnik 1995. Using information content to evaluate semantic similarity in a taxonomy. IJCAI

- Let's define $P(c)$ as:
  - The probability that a randomly selected word in a corpus is an instance of concept c
  - Formally: there is a distinct random variable, ranging over words, associated with each concept in the hierarchy
    - for a given concept, each observed noun is either
      - a member of that concept with probability $P(c)$
      - not a member of that concept with probability $1\text{-}P(c)$
  - All words are members of the root node (Entity)
    - $P(\text{root})=1$
  - The lower a node in hierarchy, the lower its probability

Information content similarity



- For every word (ex “natural elevation”), we count all the words in that concepts, and then we normalize by the total number of words in the corpus.
- we get a probability value that tells us how probable it is that a random word is a an instance of that concept

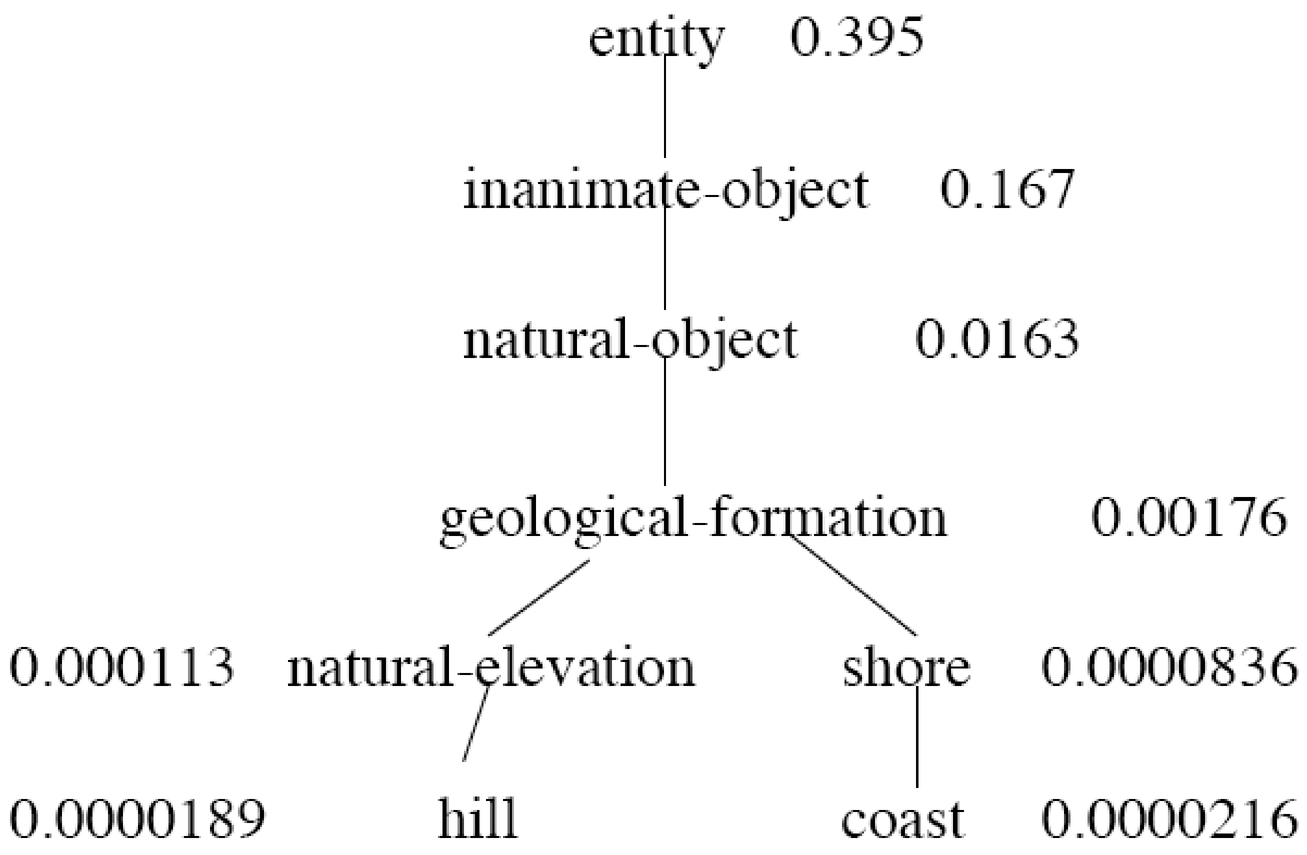
$$P(c) = \frac{\sum_{w \in words(c)} count(w)}{N}$$

In order o compute the probability of the term "natural elevation", we take ridge, hill + natural elevation itself

Information content similarity

- WordNet hierarchy augmented with probabilities P(c)

D. Lin. 1998. An Information--Theoretic Definition of Similarity. ICML 1998

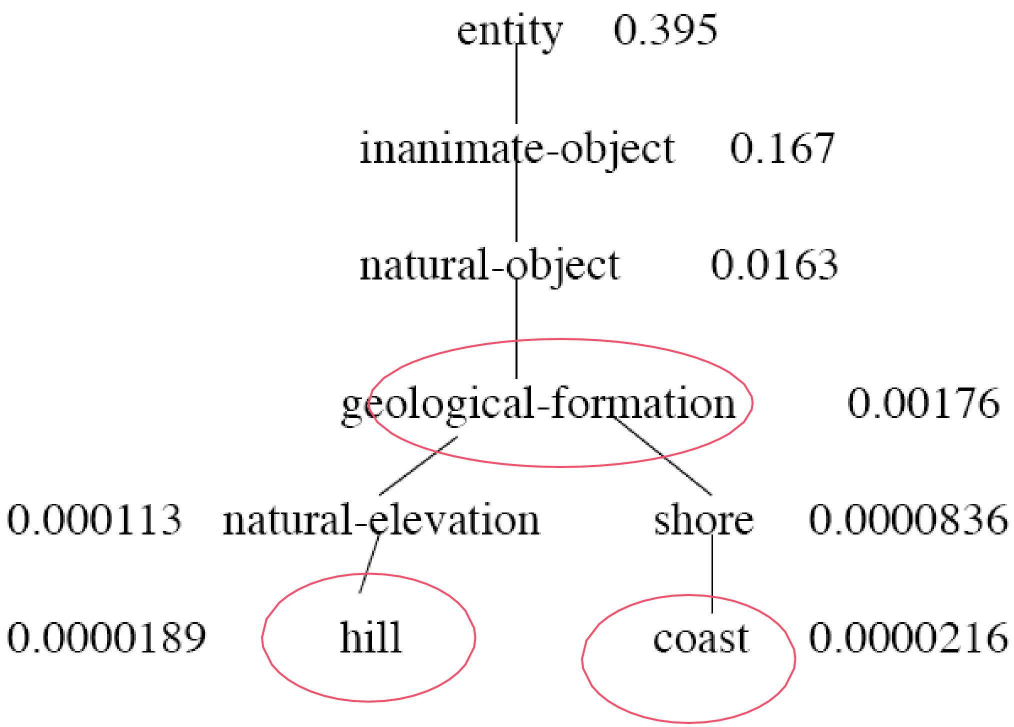


Information content: definitions

1. Information content:
    1. $IC(c) = -\log P(c)$
2. Most informative subsumer (Lowest common subsumer)

   $LCS(c_1,c_2) =$

   The most informative (lowest) node in the hierarchy subsuming both c_1 and c_2



- A lot of people prefer the term **surprisal** to information or to information content.

$$-\log p(x)$$

It measures the amount of surprise generated by the event x.

The smaller the probability of x, the bigger the surprisal is.

It's helpful to think about it this way, particularly for linguistics examples.

Using information content for similarity: the Resnik method

Philip Resnik. 1995. Using Information Content to Evaluate Semantic Similarity in a Taxonomy. IJCAI 1995.
Philip Resnik. 1999. Semantic Similarity in a Taxonomy: An Information--Based Measure and its Application to Problems of Ambiguity in Natural Language. JAIR 11, 95–130.

- The similarity between two words is related to their common information
- The more two words have in common, the more similar they are
- Resnik: measure common information as:
  - The information content of the most informative (lowest) subsumer (MIS/LCS) of the two nodes
  - $sim_{resnik}(c_1,c_2) = -\log P(\ LCS(c_1,c_2)\ )$

Dekang Lin method

Dekang Lin. 1998. An Information--Theoretic Definition of Similarity. ICML

- Intuition: Similarity between A and B is not just what they have in common
- The more **differences** between A and B, the less similar they are:
  - Commonality: the more A and B have in common, the more similar they are
  - Difference: the more differences between A and B, the less similar
- Commonality: IC(common(A,B))
- Difference: IC(description(A,B)--IC(common(A,B))

Dekang Lin similarity theorem

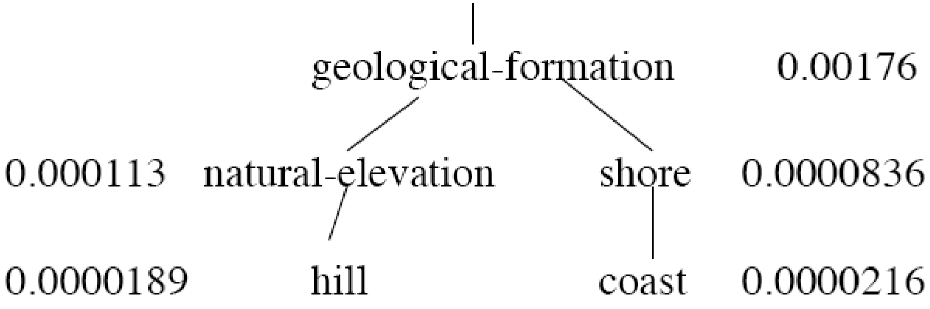
- The similarity between A and B is measured by the ratio between the amount of information needed to state the commonality of A and B and the information needed to fully describe what A and B are

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

- Lin (altering Resnik) defines IC(common(A,B)) as 2 x information of the LCS

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

Lin similarity function



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

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

$$= \frac{2\ln 0.00176}{\ln 0.0000189 + \ln 0.0000216}$$

$$= .59$$

The (extended) Lesk Algorithm

- A thesaurus--based measure that looks at **glosses**
- Two concepts are similar if their glosses contain similar 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 that's in both glosses
  - Add a score of n^2
  - Paper and specially prepared for $1 + 2^2 = 5$
  - Compute overlap also for other relations
    - glosses of hypernyms and hyponyms

Summary: thesaurus--based similarity

$$\text{sim}_{\text{path}}(c_1,c_2) = \frac{1}{pathlen(c_1,c_2)}$$

$$\text{sim}_{\text{resnik}}(c_1,c_2) = -\log P(LCS(c_1,c_2))$$

$$\text{sim}_{\text{lin}}(c_1,c_2) = \frac{2\log P(LCS(c_1,c_2))}{\log P(c_1)+\log P(c_2)}$$

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

Libraries for computing thesaurus--based similarity

- NLTK
  - http://nltk.github.com/api/nltk.corpus.reader.html?highlight=similarity - nltk.corpus.reader.WordNetCorpusReader.res_similarity
- WordNet::Similarity
  - http://wn--similarity.sourceforge.net/
  - Web-based interface:
    - http://marimba.d.umn.edu/cgi--bin/similarity/similarity.cgi

Machine Learning based approach

Basic idea

- If we have data that has been hand--labelled with correct word senses, we can used a supervised learning approach and learn from it!
  - We need to extract features and train a classifier
  - The output of training is an automatic system capable of assigning **sense labels** TO **unlabelled words** in a context.

Two variants of WSD task

- Lexical Sample task
  - (we need labelled corpora for individual senses)
  - Small pre--selected set of target words (*ex difficulty*)
  - And inventory of senses for each word
  - **Supervised machine learning: train a classifier for each word**
- All-words task
  - (each word in each sentence is labelled with a sense)
  - Every word in an entire text
  - A lexicon with senses for each word

SENSEVAL 1–2–3

Supervised Machine Learning Approaches

- Summary of what we need:
  - the **tag set** (“sense inventory”)
  - the **training corpus**
  - A set of **features** extracted from the training corpus
  - A **classifier**

Supervised WSD 1: WSD Tags

- What’s a tag?
  A dictionary sense?
- For example, for WordNet an instance of “bass” in a text has 8 possible tags or labels (bass1 through bass8).

8 senses of “bass” in WordNet

1. bass –(the lowest part of the musical range)
2. bass, bass part –(the lowest part in polyphonic music)
3. bass, basso –(an adult male singer with the lowest voice)
4. sea bass, bass –(flesh of lean-fleshed saltwater fish of the family Serranidae)
5. freshwater bass, bass –(any of various North American lean--fleshed freshwater fishes especially of the genus Micropterus)
6. bass, bass voice, basso –(the lowest adult male singing voice)
7. bass –(the member with the lowest range of a family of musical instruments)
8. bass –(nontechnical name for any of numerous edible marine and freshwater spiny--finned fishes)

SemCor

SemCor: 234,000 words from Brown Corpus, manually tagged with WordNet senses

<wf pos=PRP>**He**</wf>

<wf pos=VB lemma=recognize wnsn=4 lexsn=2:31:00::>**recognized**</wf>

<wf pos=DT>**the**</wf>

<wf pos=NN lemma=gesture wnsn=1 lexsn=1:04:00::>**gesture**</wf>

<punc>.</punc>

Supervised WSD: Extract feature vectors Intuition from Warren Weaver (1955):

"If one examines the words in a book, one at a time as through an opaque mask with a hole in it one word wide, then it is obviously impossible to determine, one at a time, the meaning of the words…

But if one lengthens the slit in the opaque mask, until one can see not only the central word in question but also say N words on either side, then if N is large enough one can unambiguously decide the meaning of the central word… the window

The practical question is : ``What minimum value of N will, at least in a tolerable fraction of cases, lead to the correct choice of meaning for the central word?"

Feature vectors

- **Vectors** of sets of feature/value pairs

Two kinds of features in the vectors

Generally speaking, a **collocation** is a sequence of words or terms that co-occur **more often than would be expected by chance.** But here the meaning is not exactly this…

- **Collocational** features and **bag-of-words** features
  - **Collocational/Paradigmatic**
    - Features about words at **specific** positions near target word
      - Often limited to just word identity and POS
  - **Bag-of-words**
    - Features about words that occur anywhere in the window (regardless of position)
      - Typically limited to frequency counts

Examples

- Example text (WSJ):

  An electric guitar and **bass** player stand off to one side not really part of the scene

- Assume a window of +/–2 from the target

Examples

- Example text (WSJ)

An electric $\boxed{\text{guitar}}\boxed{\text{and}}$ **bass** $\boxed{\text{player}}\boxed{\text{stand}}$ off to

one side not really part of the scene,

- Assume a window of +/–2 from the target

Collocational features

- Position--specific information about the words and collocations in window
- $\boxed{\text{guitar}}\boxed{\text{and}}$ bass $\boxed{\text{player}}\boxed{\text{stand}}$

$$[w_{i-2}, \mathrm{POS}_{i-2}, w_{i-1}, \mathrm{POS}_{i-1}, w_{i+1}, \mathrm{POS}_{i+1}, w_{i+2}, \mathrm{POS}_{i+2}, w_{i-2}^{i-1}, w_{i}^{i+1}]$$

[guitar, NN, and, CC, player, NN, stand, VB, and guitar, player stand]

- word 1,2,3 grams in window of ±3 is common

Bag-of-words features

- "an unordered set of words" – position ignored
- Choose a vocabulary: a useful subset of words in a training corpus
- Either: the count of how often each of those terms occurs in a given window OR just a binary "indicator" 1 or 0

Co–Occurrence Example

- Assume we've settled on a possible vocabulary of 12 words in "bass" sentences:

[*fishing, big, sound, player, fly, rod, pound, double, runs, playing, guitar, band*]

- The vector for:

  guitar and bass player stand

  [0,0,0,1,0,0,0,0,0,0,1,0]

Word Sense Disambiguation

Classification

Classification

- *Input*:
  - a word w and some features *f*
  - a fixed set of classes $C = \{c_1, c_2,\ldots, c_J\}$

Any kind of classifier

- Naive Bayes
- Logistic regression
- Neural Networks
- Support-‐vector machines
- k-‐Nearest Neighbors
- etc.

- *Output*: a predicted class $c \in C$

Coherence

- Coherence in discourse refers to the way in which the sentences of a text relate to each other to form a unified whole. An important aspect of coherence is how references (like pronouns, noun phrases, etc.) are used to maintain the flow and connectivity of ideas. Coherence reference phenomena are mechanisms that help achieve this connection.

Coherence Reference Phenomena

1. Anaphora: Anaphora is a reference to something previously mentioned in the discourse.

   1. Example: "John went to the store. He bought some milk." ("He" refers to "John")

2. Cataphora: Cataphora is a reference to something that is mentioned later in the discourse.

   1. Example: "Before he could leave, John had to finish his work." ("he" refers to "John")

3. Exophora: Exophora is a reference to something outside the text, often in the physical or situational context.

   1. Example: "Look at that!" (where "that" refers to something in the physical environment)

4. Endophora: Endophora is a general term for both anaphora and cataphora, i.e., references within the text.

   1. Anaphoric endophora: "He was hungry. John ate an apple."
   2. Cataphoric endophora: "When he arrived, John was tired."

5. Coreference: Coreference occurs when two or more expressions in a text refer to the same entity.

   1. Example: "Alice lost her book. She can't find it anywhere." ("her" and "she" refer to "Alice", and "it" refers to "her book")

Penn Tree Bank

Penn Treebank is a large annotated corpus of English that is widely used in computational linguistics and natural language processing (NLP) for training and evaluating algorithms. It was created by the University of Pennsylvania's Linguistic Data Consortium and has been a foundational resource in the field.

Key Features of the Penn Treebank

- Annotated Corpus: The Penn Treebank includes syntactic and semantic annotations of texts, making it a valuable resource for developing and testing NLP models.
- Part-of-Speech (POS) Tagging: It provides POS tags for each word, which are essential for various NLP tasks such as lemmatization, parsing, and machine translation.
- Syntactic Trees: The corpus contains syntactic trees that represent the grammatical structure of sentences. These trees are crucial for tasks such as syntactic parsing and grammar induction.
- Wide Coverage: It includes texts from various genres, such as Wall Street Journal articles, telephone conversations, and more, offering a broad spectrum of English language use.

Penn Treebank POS Tags

- The Penn Treebank uses a set of POS tags to annotate words. Here is a list of some common tags:
- CC: Coordinating conjunction
- CD: Cardinal number
- DT: Determiner
- EX: Existential there
- FW: Foreign word
- IN: Preposition or subordinating conjunction
- JJ: Adjective
- JJR: Adjective, comparative
- JJS: Adjective, superlative
- LS: List item marker

- MD: Modal
- NN: Noun, singular or mass
- NNS: Noun, plural
- NNP: Proper noun, singular
- NNPS: Proper noun, plural
- PDT: Predeterminer
- POS: Possessive ending
- PRP: Personal pronoun
- PRP$: Possessive pronoun
- RB: Adverb
- RBR: Adverb, comparative
- RBS: Adverb, superlative

- RP: Particle
- SYM: Symbol
- TO: to
- UH: Interjection
- VB: Verb, base form
- VBD: Verb, past tense
- VBG: Verb, gerund or present participle
- VBN: Verb, past participle
- VBP: Verb, non-3rd person singular present
- VBZ: Verb, 3rd person singular present
- WDT: Wh-determiner
- WP: Wh-pronoun
- WP$: Possessive wh-pronoun
- WRB: Wh-adverb

- **Example Sentence with Penn Treebank POS Tags**
- Consider the sentence: "The quick brown fox jumps over the lazy dog."
- Here is how it might be tagged using Penn Treebank POS tags:
- The/DT quick/JJ brown/JJ fox/NN jumps/VBZ over/IN the/DT lazy/JJ dog/NN

- **Applications of the Penn Treebank**

1. **POS Tagging**: The POS-tagged data is used to train and evaluate POS taggers.
2. **Syntactic Parsing**: The syntactic trees are used to train parsers that can predict the syntactic structure of sentences.
3. **Machine Learning**: The annotated data serves as a benchmark for various machine learning models in NLP.
4. **Linguistic Research**: Researchers use the Penn Treebank to study syntactic and semantic phenomena in English.