Word Sense Disambiguation

WordNet (http://wordnet.princeton.edu/) is a database of lexical relations:
- Nouns (117,798); verbs (11,529); adjectives (21,479) & adverbs (4,481)
  - https://wordnet.princeton.edu/wordnet/man/wnstats.7WN.html

WordNet contains different senses of a word, defined by synsets (synonym sets):
- {chump¹, fool², gull¹, mark³, patsy¹, fall guy¹, sucker¹, soft touch¹, mug²}
  - Words are substitutable in some contexts
- gloss: a person who is gullible and easy to take advantage of

See http://babelnet.org for other languages

Context

A (word) sense represents one meaning of a word
- bank¹: financial institution
- bank²: sloped ground near water

Various relations:
- homonymy: 2 words/senses happen to sound the same (e.g., bank¹ & bank²)
- polysemy: 2 senses have some semantic relation between them
  - bank¹ & bank³ = repository for biological entities

WSD Evaluation

- Extrinsic (in vivo) evaluation: evaluate WSD in the context of another task, e.g., question answering
- Intrinsic (in vitro) evaluation: evaluate WSD as a stand-alone system
  - Exact-match sense accuracy
  - Precision/recall measures, if systems pass on some labelings

Baselines:
- Most frequent sense (MFS): for WordNet, take first sense
- Lesk algorithm (later)

Ceiling: inter-annotator agreement, generally 75-80%
Feature extraction

1. POS tag, lemmatize/stem, & perhaps parse the sentence in question
2. Extract context features within a certain window of a target word
   - Feature vector: numeric or nominal values encoding linguistic information

Feature extraction

Bag-of-words features

Bag-of-words features encode unordered sets of surrounding words, ignoring exact position
- Captures more semantic properties & general topic of discourse
- Vocabulary for surrounding words usually pre-defined e.g., 12 most frequent content words from bass sentences in the WSJ:

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

leading to this feature vector:

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

Naive Bayes assumption

- Treat the context (c) as a bag of words (v_j)
- Make the Naive Bayes assumption that every surrounding word \( v_j \) is independent of the other ones:

\[
\text{(3)} \quad P(c|s_k) = \prod_{v_j} P(v_j|s_k)
\]

\[
\text{(4)} \quad s = \arg\max_{s_k} \sum_{v_j} \log P(v_j|s_k) + \log P(s_k)
\]

We get maximum likelihood estimates from the corpus to obtain \( P(s_k) \) and \( P(v_j|s_k) \)

Bayesian WSD

- Look at a context of surrounding words, call it \( c \), within a window of a particular size
- Select the best sense \( s \) from among the different senses

\[
\text{(1)} \quad s = \arg\max_{s_k} P(s_k|c) = \arg\max_{s_k} \frac{P(c|s_k)P(s_k)}{P(c)} = \arg\max_{s_k} P(c|s_k)P(s_k)
\]

Computationally simpler to calculate logarithms, giving:

\[
\text{(2)} \quad s = \arg\max_{s_k} [\log P(c|s_k) + \log P(s_k)]
\]

Dictionary-based WSD

Lesk algorithm

Use general characterizations of the senses to aid in disambiguation

Intuition: words found in a particular sense definition can provide contextual cues, e.g., for ash:

<table>
<thead>
<tr>
<th>Sense</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s_1 ): tree</td>
<td>a tree of the olive family</td>
</tr>
<tr>
<td>( s_2 ): burned stuff</td>
<td>the solid residue left when combustible material is burned</td>
</tr>
</tbody>
</table>

If tree is in the context of ash, the sense is more likely \( s_1 \)
### Lesk algorithm

Look at words within the sense definition and the words within the definitions of context words, too (unionsing over different senses)

1. Take all senses $s_i$ of a word $w$ and gather the set of words for each definition
   - Treat it as a bag of words
2. Gather all the words in the definitions of the surrounding words, within some context window
3. Calculate the overlap
4. Choose the sense with the higher overlap

### Problems with dictionary-based WSD

- Not very accurate: 50%-70%
- Highly dependent upon the choice of dictionary
- Not always clear whether the dictionary definitions align with what we think of as different senses

### Heuristic-based WSD

Can use a heuristic to automatically select seeds

- **One sense per discourse**: the sense of a word is highly consistent within a given document
- **One sense per collocation**: collocations rarely have multiple senses associated with them

### One sense per collocation

Rank senses based on what collocations the word appears in,

- e.g., *show interest* might be strongly correlated with the 'attention, concern' usage of *interest*
- The collocational feature could be a surrounding POS tag, or a word in the object position
- For a given context, select which collocational feature will be used to disambiguate, based on which feature is strongest indicator
  - Avoid having to combine different pieces of information this way

Rankings are based on the following, where $f$ is a collocational feature:

$$ P(s_k | f) $$

### Calculating collocations

1. Initially, calculate the collocations for $s_k$
2. Calculate the contexts in which an ambiguous word is assigned to $s_i$, based on those collocations
3. Calculate the set of collocations that are most characteristic of the contexts for $s_k$, using the formula:

$$ P(s_i | f) $$

4. Repeat steps 2 & 3 until a threshold is reached.

Example

(5) This cigar **burns** slowly and creates a stiff **ash**.

So, sense $s_2$ goes with the first sentence and $s_1$ with the second

- Note that, depending on the dictionary, **leaf** might also be a contextual cue for sense $s_1$ of **ash**
The Flip-Flop Algorithm (roughly)

1. Randomly partition $P$ (possible senses/translations) into $P_1$ and $P_2$
2. While improving mutual information scores,
   2.1 Find the partition $Q$ (possible indicators) into $Q_1$ and $Q_2$ which maximizes $I(P, Q)$
   2.2 Find the partition $P$ into $P_1$ and $P_2$ which maximizes $I(P_1, Q_1)$

**Word similarity**

**Idea:** expect synonyms to behave similarly

Define this in two ways:
- Knowledge-based: thesaurus-based WSD
- Knowledge-free: distributional methods

Word similarity computations are useful for IR, QA, summarization, language modeling, etc.

**Thesaurus-based WSD**

Use essentially the same set-up as dictionary-based WSD, but now:
- instead of requiring context words to have overlapping dictionary definitions
- we require surrounding context words to list the focus word $w$ (or the subject code of $w$) as one of their topics
e.g., If an animal or insect appears in the context of bass, we choose the fish sense instead of the musical one

Alternative: use path lengths in an ontology like WordNet to calculate word similarity

**Translation-based WSD**

**Idea:** when disambiguating a word $w$, look for a combination of $w$ and some contextual word which translates to a particular pair, indicating a particular sense
- interest can be ‘legal share’ (Beteiligung in German) or ‘concern’ (Interesse)
- In the phrase show concern, we are more likely to translate to Interesse zeigen than Beteiligung zeigen

So, in this English context, the German context tells us to go with the sense that corresponds to Interesse

**Information-theoretic WSD**

Instead of using all contextual features—which we assume are independent—an information-theoretic approach tries to find one disambiguating feature
- Take a set of possible indicators and determine which is the best, i.e., which gives the highest mutual information in the training data

Possible indicators:
- object of the verb
- the verb tense
- word to the left
- word to the right
- etc.

When sense tagging, find value of that indicator to tag

**Partitioning**

More specifically, determine what the values ($x_i$) of the indicator indicate, i.e. what sense ($s_i$) they point to.
- Assume two senses ($P_1$ and $P_2$), which can be captured in subsets $Q_1 = \{x_i | x_i$ indicates sense 1 $\}$ and $Q_2 = \{x_i | x_i$ indicates sense 2 $\}$
- We will have a set of indicator values $Q$; our goal is to partition $Q$ into these two sets

The partition we choose is the one which maximizes the mutual information scores $I(P_1, Q_1)$ and $I(P_2, Q_2)$
- The Flip-Flop algorithm is used when you have to automatically determine your senses (e.g., if using parallel text)
Disambiguation

After determining the best indicator and partitioning the values, disambiguating is easy:

1. Determine the value \( x_i \) of the indicator for the ambiguous word.
2. If \( x_i \) is in \( Q_1 \), assign it sense 1; otherwise, sense 2.

This method is also applicable for determining which indicators are best for a set of translation words.

Unsupervised WSD

Perform sense discrimination, or clustering

- In other words, group comparable senses together—even if you cannot give a correct label.

We will look briefly at the EM (Expectation-Maximization) algorithm for this task, based on a Bayesian model.

EM algorithm: Bayesian review

Bayesian WSD for supervised learning:

- Look at a context of surrounding words, call it \( v_j \) (word in context), within a window of a particular size.
- Select the best sense \( s \) from among the different senses.

\[
\hat{P}(c_i|v_j) = \arg \max_s \hat{P}(c_i|s_k) \hat{P}(s_k)
\]

We need some other way to get estimates of \( \hat{P}(s_k) \) and \( \hat{P}(c_i|s_k) \).

EM algorithm (cont.)

2. Maximization: Use the expected probabilities to re-estimate the parameters:

\[
\hat{P}(v_j|s_k) = \frac{\sum_{i \in v_j} \hat{P}(c_i|s_k)}{\sum_{k} \sum_{i \in v_j} \hat{P}(c_i|s_k)}
\]

\( \rightarrow \) Of all the times that \( v_j \) occurs in a context of any of this word's senses, how often does \( v_j \) indicate \( s_k \)?

\[
\hat{P}(s_k) = \frac{\sum_{k} \hat{P}(c_i|s_k)}{\sum_{k} \sum_{j} \hat{P}(c_i|s_k)}
\]

\( \rightarrow \) Of all the times that any sense generates \( c_j \), how often does \( s_k \) generate it?

Surveys on WSD Systems

Surveys:

  - Covers: decision lists, decision trees, Naive Bayes, neural networks, instance-based learning, SVMs, ensemble methods, clustering, multilinguality, Senseval/Senseval+, etc.