Algorithms for Natural Language Processing

Word Sense Disambiguation

WORD SENSE DISAMBIGUATION

Homonymy and Polysemy

- As we have seen, multiple words can be spelled the same way (**homonymy**; technically homography)
- The same word can also have different, related senses (**polysemy**)
- Various NLP tasks require resolving the ambiguities produced by homonymy and polysemy.
- **Word sense disambiguation** (WSD)

Two Versions of the WSD Task

- **Lexical sample**
  - Choose a sample of words
  - Choose a sample of senses for those words
  - Identify the right sense for each word in the sample
- **All-words**
  - Systems are given the entire text
  - Systems are given a lexicon with senses for every content word in the text
  - Identify the right sense for each content word in the text

Supervised WSD

- If we have hand-labelled data, we can do supervised WSD
- Lexical sample tasks
  - *Line-hard-serve* corpus
  - SENSEVAL *corpora*
- All-word tasks
  - Semantic concordance
    - SemCor—subset of Brown Corpus manually tagged with WordNet senses
  - SENSEVAL-3
- Can be viewed as a classification task

But What Features Should I Use?

- As Weaver (1955) noted,

  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 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?"

  What information is available in that window of length N that allows us to do WSD?

But What Features Should I Use?

- **Collocation features**
  - "Encode information about *specific* positions located to the left or right of the target word"
  - For *bass* (hypothetical, from J&M):
    - [w_{i-2}, POS_{i-2}, w_{i-1}, POS_{i-1}, w_{i+1}, POS_{i+1}, w_{i+2}, POS_{i+2}]
    - [guitar, NN, and, CC, player, NN, stand, VB]
- **Bag-of-words features**
  - Unordered set of words occurring in window
  - Relative sequence is ignored
  - Used to capture domain
  - For *bass* (hypothetical, adapted from J&M)
    - [*fishing*, *big*, *sound, player*, … *band*]
    - *[0, 0, 0, 1, … 0]*

Naïve Bayes for WSD

- The intuition behind the naïve Bayes approach to WSD is that choosing the best sense *s* among the possible senses *S*, given a feature vector *f* is about choosing the most probable sense given the vector.
- Starting there, we can derive the following:

$$\hat{s} = \underset{s \in S}{\operatorname{argmax}} P(s) \prod_{j=1}^{n} P(f_j|s)$$

- Of course, in practice, you map everything to log space and perform additions instead of multiplications

What's so Naïve about Naïve Bayes?

- **Reminder**: Naïve Bayes is **naïve** in that it "pretends" that the features in *f* are independent
- Often, this is not really true
- Nevertheless, Naïve Bayes Classifiers frequently (lol) perform very well in practice

Decision List Classifiers for WSD

- The decisions handed down by naïve Bayes classifiers (and other similar ML algorithms) are difficult to interpret.
  - It is not always clear why, for example, a particular classification was made
  - For reasons like this, some researchers have looked to decision list classifiers, a highly interpretable approach to WSD
- Decision List: list of statements
  - Each statement is essential a conditional
  - Item being classified falls through the cascade until a statement is true
  - The associated sense is then returned
  - Otherwise, a default sense is returned
- But where does the list come from?

Learning a Decision List Classifier

- Yarowsky (1994) proposed a way for learning such a classifier, for binary homonym discrimination, from labelled data
- Generate and order tests:
  - Each individual feature-value pair is a test
  - Contribution of the test is obtained by computing the probability of the sense given the feature
  - How discrimintative is a feature between two senses?

$$\left| log \left(\frac{P(Sense_1|f_1)}{P(Sense_2|f_2)} \right) \right|$$

  - Order tests according to log-likelihood ratio

How to Evaluate WSD Systems?

Extrinsic evaluation

- Also called **task-based**, **end-to-end**, and **in vivo** evaluation
- Measures the contribution of a WSD (or other) component to a larger pipeline
- Requires a large investment and hard to generalize to other tasks

Intrinsic evaluation

- Also called **in vitro** evaluation
- Measures the performance of a WSD (or other) component in isolation
- Do not necessarily tell you how well the component contributes to a real test (which is what you really want to know)

Baselines

- **Most frequent sense**
  - Senses in WordNet are typically ordered from most to least frequent
  - For each word, simply pick the most frequent
  - Surprisingly accurate
- **Lesk algorithm**
  - Really, a family of algorithms
  - Measures overlap in words between gloss/examples and context

Simplified Lesk Algorithm

```
function SIMPLIFIED LESK(word, sentence) returns best sense of word

  best-sense ← most frequent sense for word
  max-overlap ← 0
  context ← set of words in sentence
  for each sense in senses of word do
    signature ← set of words in the gloss and examples of sense
    overlap ← COMPUTEOVERLAP(signature, context)
    if overlap > max-overlap then
        max-overlap ← overlap
        best-sense ← sense
  end
  return(best-sense)
```

Figure 20.3 The Simplified Lesk Algorithm. The COMPUTEOVERLAP function returns the number of words in common between two sets, ignoring function words or other words on a stop list. The original Lesk algorithm defines the *context* in a more complex way. The *Corpus Lesk* algorithm weights each overlapping word w by its $-\log P(w)$, and includes labeled training corpus data in the *signature*.

What about Selectional Restrictions?

- Some of the earliest approaches to WSD relied heavily on selection restrictions
  - Catch a bass
  - Play a bass
  - You know which sense to pick by selectional restrictions from the verb
    - A fish is “catchable”
    - A musical instrument is “playable”
- This is a useful, but imperfect, source of information for sense disambiguation

Limits to Selectional Restrictions

- Consider the following sentences (from J&M):
  - But it fell apart in 1931, perhaps because people realized you can't **eat** gold for lunch if you're hungry.
  - In his two championship trials, Mr. Kulkarni **ate** glass on an empty stomach, accompanied only by water and tea.
- **Upshot**: we cannot say that, just because a sense does not satisfy the selectional restrictions of another word in the sentence (e.g. a verb), it is the wrong sense
- We need to be more clever…

Selectional Preference Strength

- "The general amount of information that a predicate tells us about the semantic class of its arguments."
  - *Eat* tells us a lot about its object, but not everything
  - *Be* tells us very little
- From J&M:
  The **selectional preference strength** can be defined by the difference in information between two distributions: the distribution of expected semantic classes *P(c)* (how likely it is that a direct object will fall into a class *c*) and the distribution of expected semantic classes for the particular verb *P(c|v)* (how likely it is that the direct object of the specific verb *v* will fall into semantic class *c*). The greater the difference between these distributions, the more information the verb is giving us about possible objects.
- **Relative entropy** or the **Kullback-Leibler divergence**

Help! I Can't Label All This Data!

- There are bootstrapping techniques that can be used to obtain reasonable WSD results will minimal amounts of labelled data
- One of these is Yarowsky's algorithm (Yarowsky 1995)
- Starts with a heuristic—**one sense per collocation**
  - **Insight:** *plant life* means *plant* is a life form; *manufacturing plant* means *plant* is a factory; there are similar collocations for other word senses
  - Don't label a bunch of data by hand
  - Build seed collocations that are going to give the right senses by hand
  - Then use the technique we discussed for decision list classifiers to "build out" from the seeds

Yarowsky in Action

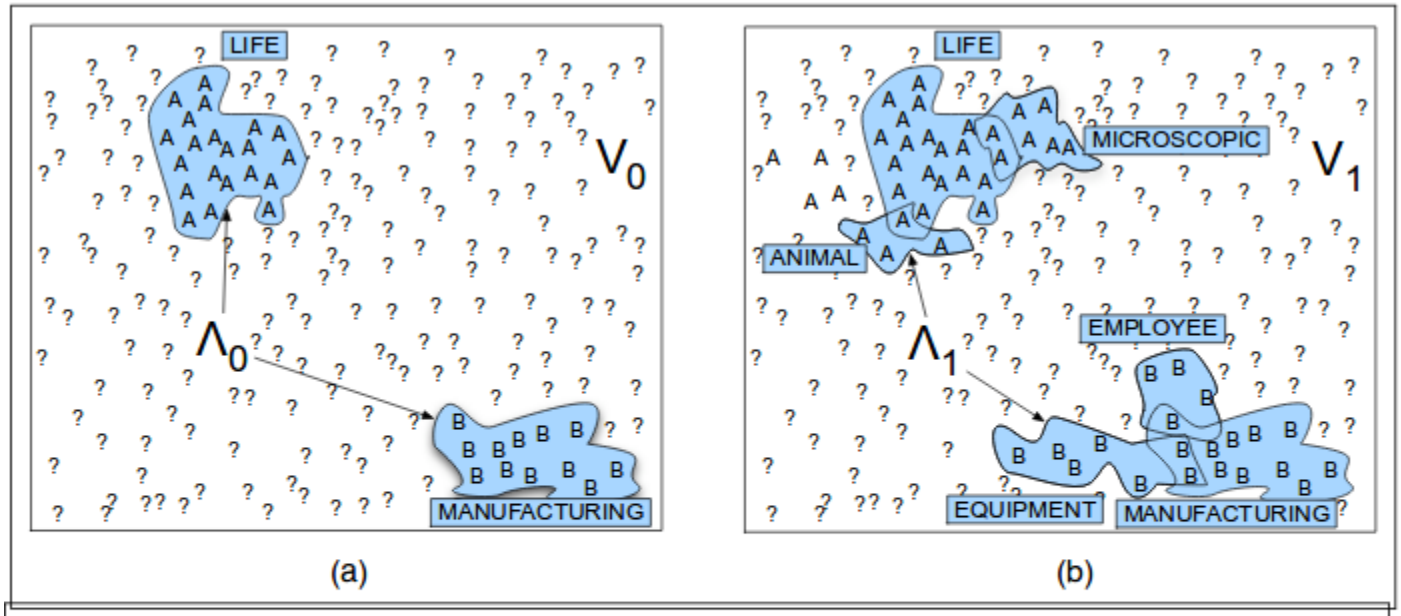

Figure 20.4 The Yarowsky algorithm disambiguating 'plant' at two stages; '?' indicates an unlabeled observation, A and B are observations labeled as SENSE-A or SENSE-B. 'LIFE' indicates observations occur with collocate "life". The initial stage (a) shows only seed sentences Λ_0 labeled by collocates ('life' and 'manufacturing'). An intermediate stage is shown in (b) where more collocates have been discovered ('equipment', 'microscopic', etc) and more instances in V_0 have been moved into Λ_1, leaving a smaller unlabeled set V_1. Figure adapted from Yarowsky (1995).