Sémantique lexicale computationnelle

Traitement sémantique automatique des langues naturelles

Partie 2: Désambiguïsation lexicale

(Slides basés sur Jurafsky & Martin (chap 20. 2004),
Perdersen et Mihalcea (2005), Mihalcea (2006 ESSLI) et Boleda & Evert (ESSLI 2009)

Pascal Denis
INRIA
`pascal.denis@inria.fr`

Master en Linguistique Informatique
Université Paris 7

Recap

- Introduction
- Lexical semantics basics
- Online resources: WordNet
- Computational lexical semantics
  - **Word Sense Disambiguation (WSD)**
  - Semantic Role Labeling (SRL)
  - **Word similarity**
  - Lexical Acquisition

Outline of this class

- Introduction
- Task description
- Evaluation
- Methods
  - supervised WSD
  - thesaurus-based WSD
  - semi-supervised WSD
- Loose ends

Introduction

Introduction

- Many words take on different senses depending on the context in which they are used
- Homonymous words
  - *bank*: slope vs. financial institution
  - *plant*: living vs. factory
  - *crane*: bird vs. machine
- Polysemous words
  - *bank*: financial institution vs. building
  - *chicken*: animal vs. meat

Motivations

- WSD = the task of selecting the correct sense of a word in a given context
- Potentially helpful in many applications
  - Information Extraction
  - Question answering
  - Text classification
  - Machine Translation
  - ...
- WSD distinct from word sense *discrimination*: problem of dividing the usages of a word into different senses, without existing inventory

Brief history

- First noted as problem for MT (Weaver, 1949)
  - A word can often only be translated if you know the specific sense intended (English *bill* could be *billet/addition* in French)
- Bar-Hillel (1960) declared the problem unsolvable:
  - *Little John was looking for his toy box. Finally, he found it. The box was in the pen. John was very happy.*

"Assume, for simplicity's sake, that pen in English has only the following two meanings: (1) a certain writing utensil, (2) an enclosure where small children can play. I now claim that no existing or imaginable program will enable an electronic computer to determine that the word pen in the given sentence within the given context has the second of the above meanings, whereas every reader with a sufficient knowledge of English will do this 'automatically'." (1960, p. 159)

Brief history (continued)

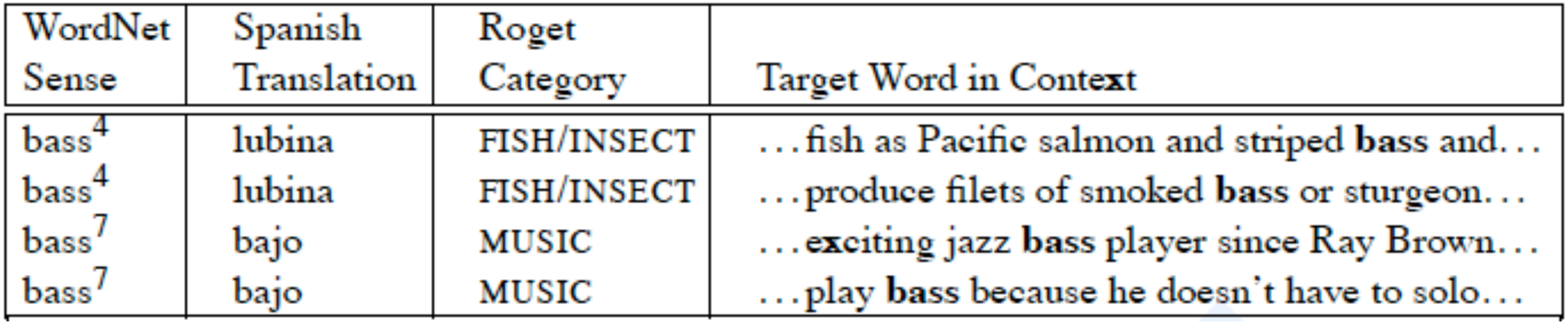
- Early work used semantic networks, frames, logical reasoning, or “expert system” methods for disambiguation based on contexts
- In the 90’s, emergence of corpus-based approaches and use of supervised machine learning techniques
- Much recent work focuses on minimizing need for annotation, semi- and non-supervised approaches, use of the Web.

Task description

WSD algorithm

- Basic form:
  - Input: word in context and sense inventory
  - Output: correct sense in that context
- What do we mean by context?
  - surrounding words/lemmas/POS tags/...?
  - context size?
- Sense Inventory?
  - Task dependent
    - set of translations for MT
    - set of homographs for speech synthesis
    - sense-tag inventory for automatic indexing of medical texts
  - As stand-alone, set of senses from thesaurus (e.g., Wordnet)

Example: *bass*



WordNet Sense	Spanish Translation	Roget Category	Target Word in Context
bass^4	lubina	FISH/INSECT	. . . fish as Pacific salmon and striped **bass** and. . .
bass^4	lubina	FISH/INSECT	. . . produce filets of smoked **bass** or sturgeon. . .
bass^7	bajo	MUSIC	. . . exciting jazz **bass** player since Ray Brown. . .
bass^7	bajo	MUSIC	. . . play **bass** because he doesn't have to solo. . .

Variants of the task

- Lexical sample WSD
  - small pre-selected set of target words is chosen, along with sense inventory for each word
  - a number of corpus instances (context sentences) are selected and labeled with correct sense
  - supervised machine learning techniques: instances are used to train word-specific classification algorithms
- All-words WSD
  - all content words are disambiguated by the system
  - similar to POS tagging, but with a much larger tagset (each lemma has its own sets)
  - data sparseness: no enough training data for each word
  - dictionary-based and bootstrapping techniques

Evaluation

Extrinsic evaluation

- Long term goal: improve performance in end-to-end application (e.g., MT, IE)
- Extrinsic evaluation (or task-based, end-to-end, in vivo evaluation)
  - Example: Word Sense Disambiguation for (Cross-Lingual) Information Retrieval
    - http://ixa2.si.ehu.es/clirwsd
- Extrinsic evaluation is difficult and time consuming, results may not generalize from one application to the other

Intrinsic evalution

- Shorter term goal: evaluate WSD as a stand-alone system
- Intrinsic evaluation (or in vitro)
  - requires held-out data from the same sense-tagged corpora used for training (train/test methodology)
  - sense accuracy: percentage of words that receive the correct sense
- Standardized datasets and evaluation campaigns:
  - Lexical sample:
    - SENSEVAL-1, -2, -3: sense-labeled corpora for 34, 73, and 57 target words
  - All-words:
    - SemCor: 234,000 word subset of Brown corpus, manually tagged with WN senses
    - SENSEVAL-3: 5,000 tagged tokens from WSJ and Brown

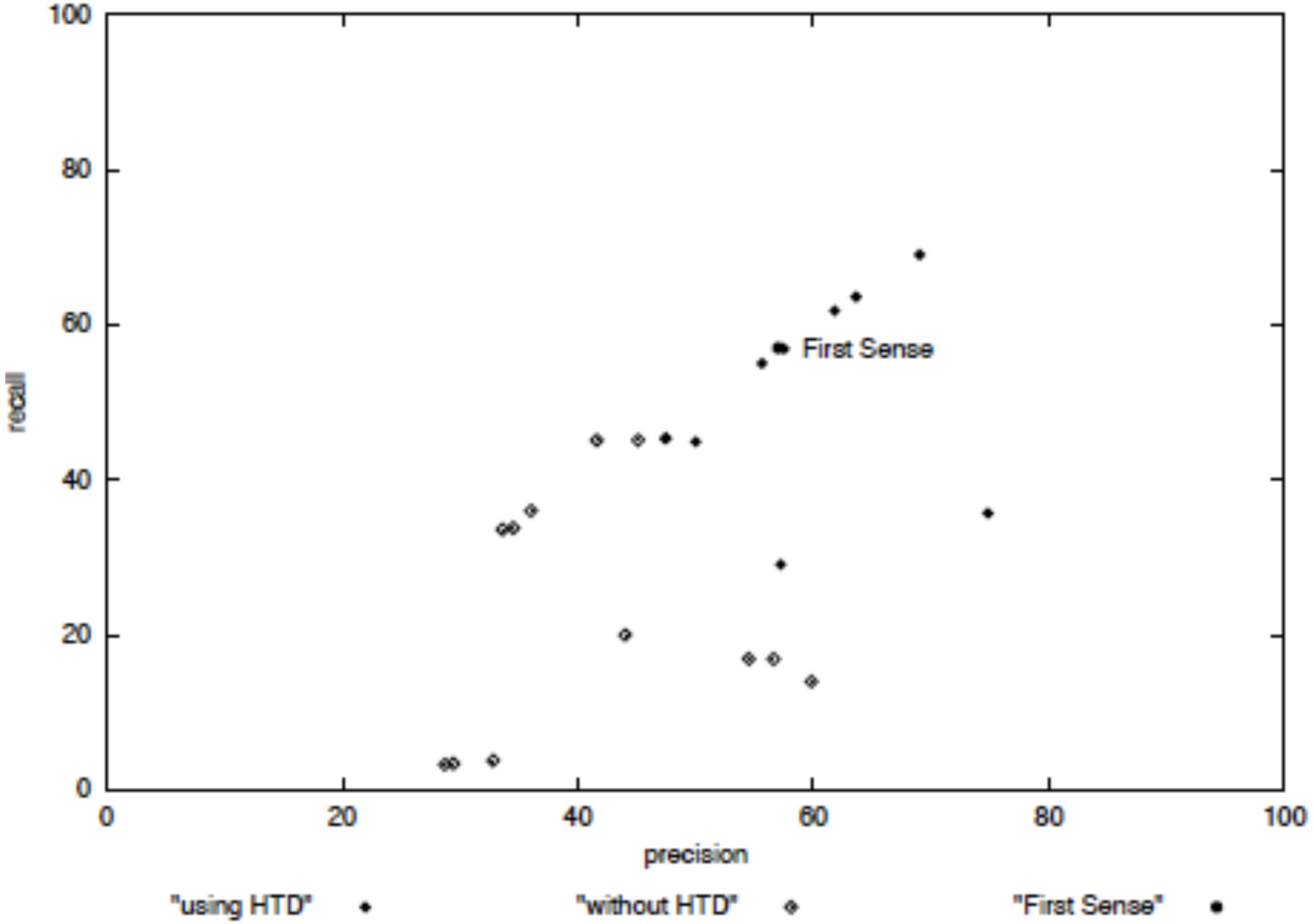
Excerpt from SEMCOR3

```
<s snum=1>
<wf cmd=ignore pos=DT>The</wf>
<wf cmd=done pos=JJ lemma=injured wnsn=1 lexsn=3:00:00::>injured</wf>
<wf cmd=done pos=JJ lemma=german wnsn=1 lexsn=3:01:00::>German</wf>
<wf cmd=done pos=NN lemma=veteran wnsn=2 lexsn=1:18:01::>veteran</wf>
<wf cmd=done pos=VB lemma=be wnsn=2 lexsn=2:42:06::>was</wf>
<wf cmd=ignore pos=DT>a</wf>
<wf cmd=done pos=JJ lemma=former wnsn=2 lexsn=5:00:01:past:00>former</wf>
<wf cmd=done pos=NN lemma=miner wnsn=1 lexsn=1:18:00::>miner</wf>
<punc>,</punc>
<wf cmd=done pos=JJ lemma=twenty-four wnsn=1 lexsn=5:00:00:cardinal:00>twenty-four</wf>
<wf cmd=done pos=NN lemma=year wnsn=1 lexsn=1:28:01::>years</wf>
<wf cmd=done pos=JJ lemma=old wnsn=1 lexsn=3:00:02::>old</wf>
<punc>,</punc>
<wf cmd=ignore pos=WP>who</wf>
<wf cmd=done pos=VBD ot=notag>had</wf>
<wf cmd=done pos=VBN ot=notag>been</wf>
<wf cmd=done pos=VB lemma=wound wnsn=1 lexsn=2:29:00::>wounded</wf>
<wf cmd=ignore pos=IN>by</wf>
<wf cmd=done pos=NN lemma=shrapnel wnsn=1 lexsn=1:06:00::>shrapnel</wf>
<wf cmd=done pos=RB ot=notag>in</wf>
<wf cmd=done pos=RB ot=notag>the</wf>
<wf cmd=done pos=NN lemma=back wnsn=2 lexsn=1:06:00::>back</wf>
<wf cmd=ignore pos=IN>of</wf>
<wf cmd=ignore pos=DT>the</wf>
<wf cmd=done pos=NN lemma=head wnsn=1 lexsn=1:08:00::>head</wf>
<punc>.</punc>
</s>
```

Baselines

- Baseline: performance we would get without much knowledge / with a simple approach
- Necessary for any Machine Learning experiment (how good is 70%?)
- Simplest and very powerful baseline: most frequent sense (first sense in WN)
  - skewed (Zipfian) distribution of senses in corpora
- But we need access to annotated data for every word in the dataset to estimate sense frequencies
- Another baseline: Lesk algorithm

First-sense baseline in “all-words” SENSEVAL-2



Finding MFS from corpus (McCarthy et al. 2003)

- MFS is a powerful baseline, but requires oracle (i.e., Wordnet or labeled corpus)
  - sense distribution varies depending on domain, genre, ...
- Idea: use distributional similarity from raw corpus and Wordnet to find MFS
  - Given a target word *w* with its WN senses *S* (e.g., *pipe: {pipe#1:tobacco pipe, pipe#2:tube of metal or plastic}*)
    - extract most contextually similar words N_w in corpus: e.g., *tube, cable, wire, tank, ...*
    - find sense of *n* in N_w that maximizes similarity with each possible senses of *w*: e.g., *sim(pipe#1,tube#3) = .3*, *sim(pipe#2,tube#1) = .6*
    - compute “prevalence score” for each sense of *w* by summing all WN similarity scores for words in N_w: *score(pipe#1) = 0.25*, *score(pipe#2) = .73...*
- McCarthy et al. experiment with various WN similarity measures: best results with JCN and Lesk
- They report F-score ~64% on SENSEVAL-2 nouns

Ceilings

- Ceiling or upper-bound for performance: human inter-annotator agreement (e.g., kappa measure)
- All-word corpora using WordNet: $A_0 \approx 0.75 - 0.8$
- More coarse-grained sense distinctions: $A_0 \approx 0.9$
- Much better agreement for homonymous than polysemous words

Pseudo-words

- Building hand-labeled test sets is both expensive and time consuming (even more data for supervised WSD)
- Another possibility: create ambigious words by concatenating two randomly picked words (e.g., *banana* and *door*) into a pseudo-word (*banana-door*)
- The correct sense is defined the original word
- Same techniques and evaluation can be used
- But unrealistic: easier than average ambiguous words
  - Real polysemy is not like *banana-doors*
  - Need to find more subtle ways to create pseudowords

Supervised WSD

Supervised method

- Collect training data where a given input is associated with a given outcome (or class) from a set of outcomes (classes)
  - lexical sample where words (=inputs) are hand labeled with senses (=classes)
- Select a set of features to represent the input
  - co-occurrences, collocations, POS tags, grammatical relations, ...
- Convert training instances into feature vectors
- Apply a machine learning algorithm to induce a classifier: set of weights associated the features
- Apply classifier to test instances to assign class (here, sense)

Features for WSD

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 ?” *(Weaver, 1955)*

1. Collocational features

- A collocation is a word or phrase in a position-specific relationship to a target word
- Collocation encodes information about words located to the left or right of the target word:
  - *“An electric guitar and bass player stand off to one side, ... “*
- Collocation feature vector, extracted from a window of two words to the right and left of the target word, made up of the lemma and their POS:
  - $[w_{i-2}, POS_{i-2}, w_{i-1}, POS_{i-1}, w_{i+1}, POS_{i+1}, w_{i+2}, POS_{i+2}]$
- Would yield the following vector:
  - [guitar, NN, and, CC, player, NN, stand, VB]
- Collocational features are effective at capturing local lexical and grammatical information that help isolating specific sense

2. Bag-of-word features

- Another way to model neighboring context
- Bag-of-words are unordered set of words
- In simplest case, they are represented by binary feature vectors
- Typically, only words from pre-selected vocabulary are used: stop words removed, freq. threshold. E.g., 12 most frequent words found with *bass* in WSJ
  - [*fishing,big,sound,player,fly,rod,pound,double,runs,playing,guitar,band*]
  - [0,0,0,1,0,0,0,0,0,0,1,0]
- Bag-of-words are effective at capturing the general topic of the discourse in which the target word occurs

Supervised learning algorithms

- Many machine learning algorithms have been successfully applied to the problem of WSD
  - Decision Lists
  - Decision Trees
  - Naive Bayes Classifiers
  - Perceptrons
  - Neural Networks
  - Log Linear Models
  - Support Vector Machines
  - …

Naive Bayes Classifier

- Choosing best sense $\hat{s}$ out of possible senses *S* for a feature vector $\vec{f}$ amounts to choosing the most probable sense given $\vec{f}$:

$$\hat{s} = \arg\max_{s \in S} P(s|\vec{f})$$

- Impossible to collect reasonable statistics for this: 2^n possible binary feature vectors for a vocabulary of *n* words!

- First, apply Bayes' rule:

$$\hat{s} = \arg\max_{s \in S} \frac{P(\vec{f}|s)P(s)}{P(\vec{f})}$$

- Then, (naively) assume that features are conditionally independent given the word sense:

$$P(\vec{f}|s) \approx \prod_{j=1}^{n} P(f_j|s)$$

Training the Naive Bayes Classifier

- Training the NB classifier means estimating each of the following probabilities:
  - Prior probability of each sense *P(s)*:

$$P(s_i) = \frac{count(s_i, w_j)}{count(w_j)}$$

  - Individual feature ("likelihood") probabilities *P(f_j|s)*:

$$P(f_j|s) = \frac{count(f_j, s)}{count(s)}$$

- Smoothing (e.g., add-1, add-k) needed to avoid null probabilities

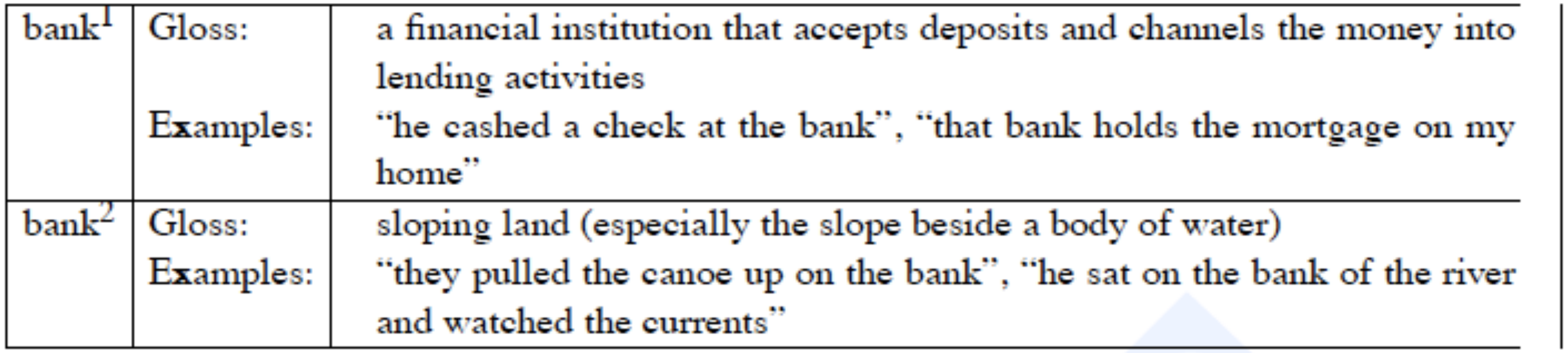
Conclusions

- Supervised ML methods give the best performance for sense disambiguation
- But labeled training data is expensive and limited, and supervised methods fail on unseen words
- Different ways to get indirect supervision from other sources:
  - use of dictionary or thesaurus
  - combine small amount of labeled data with unlabeled data

Thesaurus-based WSD

Intuition

- Use dictionary/thesaurus as corpus:
  - word definitions/examples = training data
- Example:
  - The bank can guarantee deposits will eventually cover future tuition costs because it invests in adjustable-rate mortgage securities.



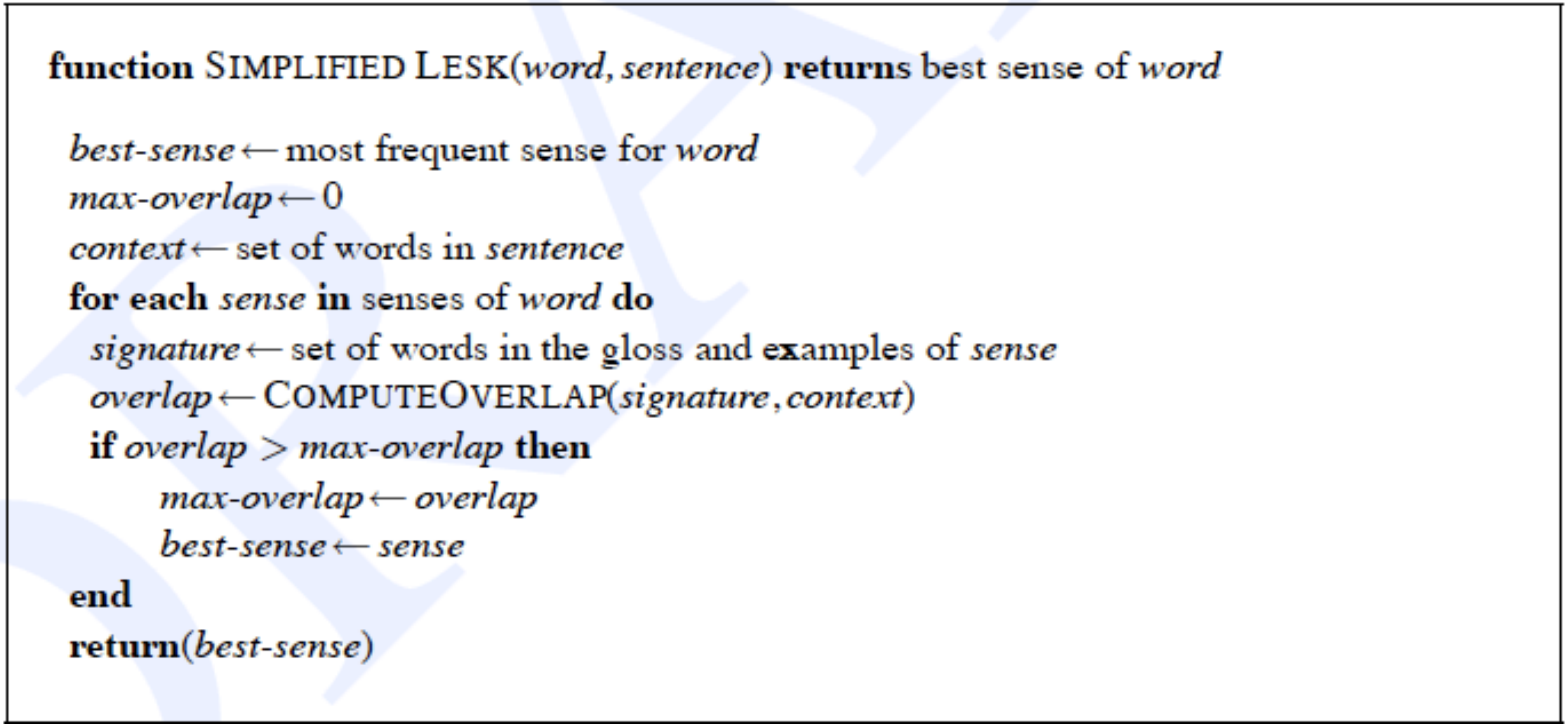
bank^1	Gloss:	a financial institution that accepts deposits and channels the money into lending activities
	Examples:	"he cashed a check at the bank", "that bank holds the mortgage on my home"
bank^2	Gloss:	sloping land (especially the slope beside a body of water)
	Examples:	"they pulled the canoe up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

- Sense **bank**^{1} has two non-stopwords overlapping with test context, while **bank**^{2} has zero => **bank**^{1} should be chosen

Signatures

- Set of words that characterize a given sense of a target word
- Signatures are extracted from dictionaries, thesauri, tagged corpora, ...
- In our example:
  - signature for **bank**^{1}: financial, institution, accept, deposit, channel, money, lending, activity, cash, check, hold, mortgage, home
  - signature for **bank**^{2}: sloping, land, body, water, pull, canoe, bank, sit, river

Simplified Lesk algorithm (Kilgariff & Rosenzweig, 2000)



```
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)
```

Discussion

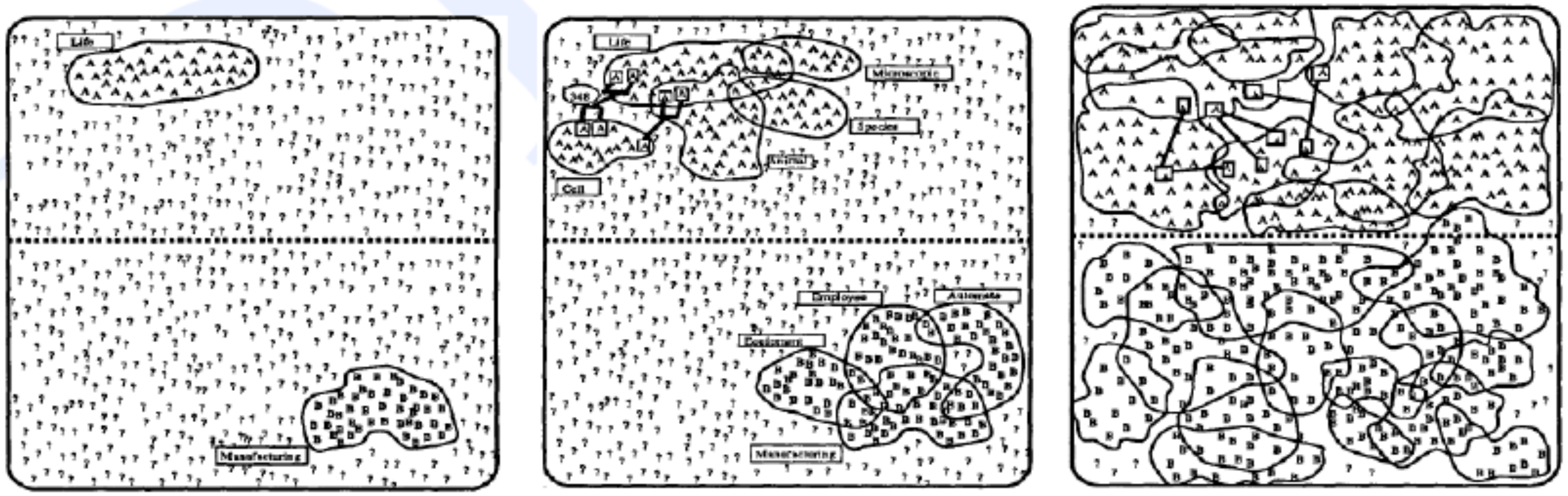
- Right intuition: words that appear in dictionary definitions and examples are relevant to a given sense
- Problem: data sparseness
  - dictionary entries short and may not overlap, not always examples
  - Lesk algorithm currently used as baseline (58% on SENSEVAL-2, with backoff to MFS)
- Many possible extensions:
  - include additional, related words in signatures
  - apply weight to each overlapping word (e.g., IDF)
- And dictionary-derived features can be used in standard supervised approaches

Semi-supervised WSD

Introduction

- Both supervised WSD and thesaurus-based WSD require large hand-built resources
- Instead, we can use semi-supervised learning:
  - Small set of labeled data ("seeds"), combined with:
  - Large set of unlabeled data
- Bootstrapping algorithm of Yarowsky (1995):
  - Train classifier on small seedset L0 of sense-labeled instances
  - Apply classifier on large unlabeled data U0
  - Select labeled examples from U0 that classifier is most confident about and add them to L0 (=L1)
  - Repeat until low error-rate is reached or no example is added

The Yarowsky algorithm (Yarowsky, 1995)



(a) (b) (c)

- Yarowksky algorithm disambiguating *plant*
  - (a) seed sentences labeled by collocates: "life" and "manufacturing"
  - (b) more collocates have been discovered: e.g., "equipment", "microscopic", ...
  - (c) final stage

Picking good seeds

- Importance of initial set of labeled data for the whole approach to be successful
- We can start by hand-labeling... pffffff
- Yarowsky (1995) uses two heuristics to automatically select seeds:
    - One sense per collocation: words that are strongly associated with a sense tend not to occur with other senses (e.g., "play" for "bass")
    - One sense per discourse: multiple instances of a word in a discourse tend to have same sense (e.g., "bass" in discourse about music)

Loose ends

- The task as currently defined does not allow for generalization over different words: learning is word-specific
  - number of classes = number of senses
  - need training data for every sense of every word
  - most words have low frequency (Zipf 's law)
  - no chance with unknown words
- this wouldn't be a problem if word sense alternation were like bank^1 − bank^2 (homonymy). . .
- ... but many alternations are systematic! (regular polysemy, metonymy, metaphor)