自然语言处理

week-8

凌震华

2024年4月28日

Word Sense Disambiguation (词义消歧)

Word Sense Disambiguation

- Given
  - A word in context
  - A fixed inventory of potential word senses
  - Decide which sense of the word this is
- Why? Machine translation, QA, speech synthesis
- What set of senses?
  - English-to-Spanish MT: set of Spanish translations
  - Speech Synthesis: homographs (同形异义词) like *bass* and *bow*
  - In general: the senses in a thesaurus like WordNet

Two variants of WSD task

- Lexical Sample task
  - Small pre-selected set of target words (*line, plant*)
  - And inventory of senses for each word
  - **Supervised machine learning: train a classifier for each word**
- All-words task
  - Every word in an entire text
  - A lexicon with senses for each word
  - Data sparseness: can't train word-specific classifiers

WSD Methods

- Supervised Machine Learning
- Thesaurus/Dictionary Methods
- Semi-Supervised Learning

Supervised Machine Learning Approaches

- Supervised machine learning approach:
  - a training corpus of words tagged in context with their sense
  - used to train a classifier that can tag words in new text
- 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)

Inventory of sense tags for *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. . .

Supervised WSD 2: Get a corpus

- Lexical sample task:
  - *Line-hard-serve* corpus - 4000 examples of each
  - *Interest* corpus - 2369 sense-tagged examples
    http://www.d.umn.edu/~tpederse/data.html
- All words:
  - **Semantic concordance**: a corpus in which each open-class word is labeled with a sense from a specific dictionary/thesaurus.
    - SemCor: 234,000 words from Brown Corpus, manually tagged with WordNet senses
      https://www.sketchengine.eu/semcor-annotated-corpus/
    - SENSEVAL-3 competition corpora - 2081 tagged word tokens
      http://web.eecs.umich.edu/~mihalcea/senseval/senseval3/index.html

SemCor

<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 3: 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 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

- A simple representation for each observation (each instance of a target word)
  - **Vectors** of sets of feature/value pairs
  - Represented as an ordered list of values
  - These vectors represent, e.g., the window of words around the target

Two kinds of features in the vectors

- **Collocational** (搭配) features and **bag-of-words** (词袋) features
  - **Collocational**
    - 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 guitar and **bass** player 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
- guitar and bass player 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
- Counts of words occur within the window.
- First choose a vocabulary
- Then count how often each of those terms occurs in a given window
  - sometimes 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]

Supervised WSD 4: Classifier

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

Classification Methods: Supervised Machine Learning

- *Input:*
  - a word *w in a text window d (which we'll call a "document")*
  - a fixed set of classes $C = \{c_1, c_2, \ldots, c_J\}$
  - A training set of *m* hand-labeled text windows again called "documents" $(d_1,c_1),\ldots,(d_m,c_m)$
- *Output:*
  - a learned classifier $\gamma:d \rightarrow c$

Classification Methods: Supervised Machine Learning

- Any kind of classifier
  - Naive Bayes
  - Logistic regression
  - Neural Networks
  - Support-vector machines
  - k-Nearest Neighbors
  - …

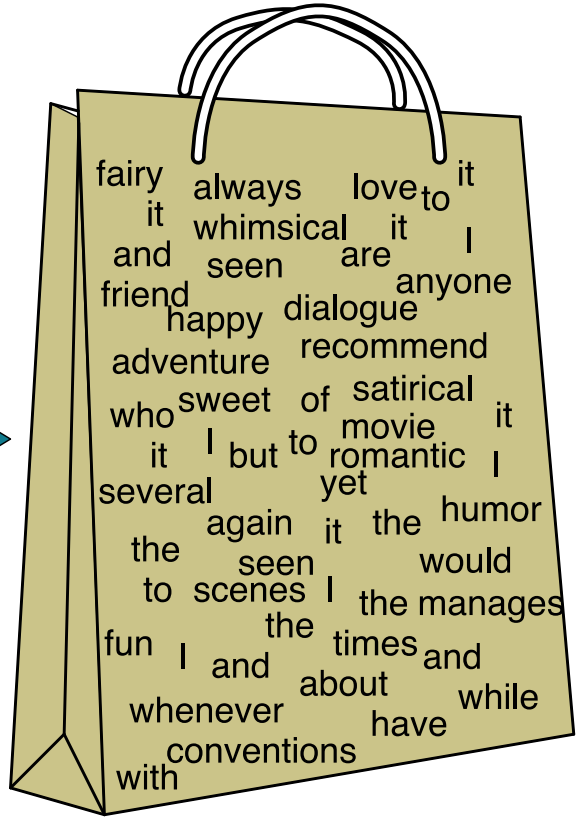
The Naive Bayes Classifier

Naive Bayes Intuition

- Simple ("naive") classification method based on Bayes rule
- Relies on very simple representation of document
  - **Bag of words**

The Bag of Words Representation

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!



it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
…	…

The Bag of Words Representation

γ (

seen	2
sweet	1
whimsical	1
recommend	1
happy	1
...	...

) = c

Bayes' Rule Applied to Documents and Classes

- For a document *d* and a class *c*

$$P(c|d) = \frac{P(d|c)P(c)}{P(d)}$$

Naive Bayes Classifier

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(c \mid d)$$

MAP is "maximum a posteriori" = most likely class

$$= \underset{c \in C}{\operatorname{argmax}} \frac{P(d \mid c)P(c)}{P(d)}$$

Bayes Rule

$$= \underset{c \in C}{\operatorname{argmax}} P(d \mid c)P(c)$$

Dropping the denominator

Naive Bayes Classifier

"Likelihood" "Prior"

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(d \mid c) P(c)$$

$$= \underset{c \in C}{\operatorname{argmax}} P(x_1, x_2, \ldots, x_n \mid c) P(c)$$

Document d represented as features x1..xn

Naive Bayes Classifier

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}}\, P(x_1, x_2, \ldots, x_n \mid c) P(c)$$

O(|X|^n•|C|) parameters

How often does this class occur?

Could only be estimated if a very, very large number of training examples was available.

We can just count the relative frequencies in a corpus

Multinomial Naive Bayes Independence Assumptions

$$P(x_1, x_2, ..., x_n \mid c)$$

- **Bag of Words assumption**: Assume position doesn't matter
- **Conditional Independence**: Assume the feature probabilities $P(x_i|c_j)$ are independent given the class *c*.

$$P(x_1, ..., x_n \mid c) = P(x_1 \mid c) \bullet P(x_2 \mid c) \bullet P(x_3 \mid c) \bullet ... \bullet P(x_n \mid c)$$

Multinomial Naive Bayes Classifier

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(x_1, x_2, ..., x_n \mid c) P(c)$$

$$c_{NB} = \underset{c \in C}{\operatorname{argmax}} P(c_j) \prod_{x \in X} P(x \mid c)$$

Applying Multinomial Naive Bayes Classifiers to Text Classification

positions ← all word positions in test document

$$c_{NB} = \underset{c_j \in C}{\operatorname{argmax}} P(c_j) \prod_{i \in positions} P(x_i \mid c_j)$$

Problems with multiplying lots of probs

- There's a problem with this:

$$c_{NB} = \underset{c_j \in C}{\operatorname{argmax}} P(c_j) \prod_{i \in positions} P(x_i \mid c_j)$$

- Multiplying lots of probabilities can result in floating-point underflow!

  .0006 * .0007 * .0009 * .01 * .5 * .000008….

- Idea: Use logs, because log(*ab*) = log(*a*) + log(*b*)
- We'll sum logs of probabilities instead of multiplying probabilities!

We actually do everything in log space

Instead of this: $$c_{NB} = \underset{c_j \in C}{\operatorname{argmax}} P(c_j) \prod_{i \in positions} P(x_i \mid c_j)$$

This: $$c_{\mathrm{NB}} = \underset{c_j \in C}{\operatorname{argmax}} \left[\log P(c_j) + \sum_{i \in \mathrm{positions}} \log P(x_i | c_j) \right]$$

Notes:

1) Taking log doesn't change the ranking of classes!

The class with highest probability also has highest log probability!

2) It's a linear model:

Just a max of a sum of weights: a **linear** function of the inputs

So naive bayes is a **linear classifier**

Learning the Multinomial Naive Bayes Model

- First attempt: maximum likelihood estimates
  - simply use the frequencies in the data

$$\hat{P}(c_j) = \frac{N_{c_j}}{N_{total}}$$

$$\hat{P}(w_i \mid c_j) = \frac{count(w_i, c_j)}{\sum_{w \in V} count(w, c_j)}$$

Parameter estimation

$$\hat{P}(w_i \mid c_j) = \frac{count(w_i, c_j)}{\sum_{w \in V} count(w, c_j)}$$

fraction of times word w_i appears among all words in documents of topic c_j

- Create mega-document for topic *j* by concatenating all docs in this topic
  - Use frequency of *w* in mega-document

Problem with Maximum Likelihood

- What if we have seen no training documents with the word ***fantastic*** and classified in the topic **positive** (***thumbs-up***)?

$$\hat{P}(\text{"fantastic"} \mid \text{positive}) = \frac{count(\text{"fantastic"}, \text{positive})}{\sum_{w \in V} count(w, \text{positive})} = 0$$

- Zero probabilities cause c_{MAP}=0, no matter the other evidence!

$$c_{MAP} = \text{argmax}_c \hat{P}(c) \prod_i \hat{P}(x_i \mid c)$$

Laplace (add-1) smoothing for Naive Bayes

$$\hat{P}(w_i \mid c) = \frac{count(w_i, c) + 1}{\sum_{w \in V} \left(count(w, c) + 1 \right)}$$

$$= \frac{count(w_i, c) + 1}{\left(\sum_{w \in V} count(w, c) \right) + |V|}$$

Multinomial Naive Bayes: Learning

- From training corpus, extract *Vocabulary*
- Calculate $P(c_j)$ terms
  - For each c_j in C do $docs_j \leftarrow$ all docs with class $=c_j$

$$P(c_j) \leftarrow \frac{|docs_j|}{|\text{total \# documents}|}$$

- Calculate $P(w_k \mid c_j)$ terms
  - $Text_j \leftarrow$ single doc containing all $docs_j$
  - For each word w_k in *Vocabulary*

    $n_k \leftarrow$ # of occurrences of w_k in $Text_j$

$$P(w_k \mid c_j) \leftarrow \frac{n_k + \alpha}{n + \alpha |Vocabulary|}$$

Unknown words

- What about unknown words
  - that appear in our test data
  - but not in our training data or vocabulary?
- We **ignore** them
  - Remove them from the test document!
  - Pretend they weren't there!
  - Don't include any probability for them at all!
- Why don't we build an unknown word model?
  - It doesn't help: knowing which class has more unknown words is not generally helpful!

Stop words

- Some systems ignore stop words
  - **Stop words:** very frequent words like *the* and *a*.
    - Sort the vocabulary by word frequency in training set
    - Call the top 10 or 50 words the **stopword list**.
    - Remove all stop words from both training and test sets
      - As if they were never there!
- But removing stop words doesn't usually help
  - So in practice most NB algorithms use **all** words and **don't** use stopword lists

Let's do a worked sentiment example!

	Cat	Documents
Training	-	just plain boring
	-	entirely predictable and lacks energy
	-	no surprises and very few laughs
	+	very powerful
	+	the most fun film of the summer
Test	?	predictable with no fun

A worked sentiment example with add-1 smoothing

	Cat	Documents
Training	-	just plain boring
	-	entirely predictable and lacks energy
	-	no surprises and very few laughs
	+	very powerful
	+	the most fun film of the summer
Test	?	predictable ~~with~~ no fun

1. Prior from training:

$$\hat{P}(c_j) = \frac{N_{c_j}}{N_{total}}$$

P(-) = 3/5
P(+) = 2/5

2. Drop "with"

3. Likelihoods from training:

$$p(w_i|c) = \frac{count(w_i, c) + 1}{(\sum_{w \in V} count(w, c)) + |V|}$$

$$P(\text{“predictable”}|-) = \frac{1+1}{14+20} \qquad P(\text{“predictable”}|+) = \frac{0+1}{9+20}$$

$$P(\text{“no”}|-) = \frac{1+1}{14+20} \qquad P(\text{“no”}|+) = \frac{0+1}{9+20}$$

$$P(\text{“fun”}|-) = \frac{0+1}{14+20} \qquad P(\text{“fun”}|+) = \frac{1+1}{9+20}$$

4. Scoring the test set:

$$P(-)P(S|-) = \frac{3}{5} \times \frac{2 \times 2 \times 1}{34^3} = 6.1 \times 10^{-5}$$

$$P(+)P(S|+) = \frac{2}{5} \times \frac{1 \times 1 \times 2}{29^3} = 3.2 \times 10^{-5}$$

Optimizing for sentiment analysis

For tasks like sentiment, word **occurrence** seems to be more important than word **frequency**.

- The occurrence of the word *fantastic* tells us a lot
- The fact that it occurs 5 times may not tell us much more

Binary multinominal naive bayes, or **binary NB**

- Clip our word counts at 1

Binary Multinomial Naive Bayes: Learning

- From training corpus, extract *Vocabulary*
- Calculate $P(c_j)$ terms
  - For each c_j in C do

    $docs_j \leftarrow$ all docs with class $=c_j$

$$P(c_j) \leftarrow \frac{|docs_j|}{|\text{total \# documents}|}$$

- Calculate $P(w_k \mid c_j)$ terms
  - Remove duplicates in each doc:
    - For each word type w in doc_j
      - Retain only a single instance of w
  - $Text_j \leftarrow$ single doc containing all $docs_j$
  - For each word w_k in *Vocabulary*

    $n_k \leftarrow$ # of occurrences of w_k in $Text_j$

$$P(w_k \mid c_j) \leftarrow \frac{n_k + \alpha}{n + \alpha |Vocabulary|}$$

Binary Multinomial Naive Bayes on a test document *d*

- First remove all duplicate words from *d*
- Then compute NB using the same equation:

$$c_{NB} = \underset{c_j \in C}{\operatorname{argmax}} P(c_j) \prod_{i \in positions} P(w_i \mid c_j)$$

Binary multinominal naive Bayes

Four original documents:

- − it was pathetic the worst part was the boxing scenes
- − no plot twists or great scenes
- + and satire and great plot twists
- + great scenes great film

After per-document binarization:

- − it was pathetic the worst part boxing scenes
- − no plot twists or great scenes
- + and satire great plot twists
- + great scenes film

	NB Counts		Binary Counts	
	+	−	+	−
and	2	0	1	0
boxing	0	1	0	1
film	1	0	1	0
great	3	1	2	1
it	0	1	0	1
no	0	1	0	1
or	0	1	0	1
part	0	1	0	1
pathetic	0	1	0	1
plot	1	1	1	1
satire	1	0	1	0
scenes	1	2	1	2
the	0	2	0	1
twists	1	1	1	1
was	0	2	0	1
worst	0	1	0	1

Counts can still be 2! Binarization is within-doc!

Applying Naive Bayes to WSD

- P(c) is the prior probability of that sense
  - Counting in a labeled training set.

- P(w|c) conditional probability of a word given a particular sense
  - P(w|c) = count(w,c)/count(c)

- We get both of these from a tagged corpus like SemCor

$$\hat{P}(c) = \frac{N_c}{N}$$

$$\hat{P}(w \mid c) = \frac{count(w,c)+1}{count(c)+|V|}$$

	Doc	Words	Class
Training	1	fish smoked fish	f
	2	fish line	f
	3	fish haul smoked	f
	4	guitar jazz line	g
Test	5	line guitar jazz jazz	?

Priors:

$P(f) = \frac{3}{4}$

$P(g) = \frac{1}{4}$

V = {fish, smoked, line, haul, guitar, jazz}

Choosing a class:

$P(f|d5) \propto 3/4 * 2/14 * (1/14)^2 * 1/14$

≈ 0.00003

Conditional Probabilities:

P(line|*f*) = (1+1) / (8+6) = 2/14

P(guitar|*f*) = (0+1) / (8+6) = 1/14

P(jazz|*f*) = (0+1) / (8+6) = 1/14

P(line|*g*) = (1+1) / (3+6) = 2/9

P(guitar|*g*) = (1+1) / (3+6) = 2/9

P(jazz|*g*) = (1+1) / (3+6) = 2/9

$P(g|d5) \propto 1/4 * 2/9 * (2/9)^2 * 2/9$

≈ 0.0006

Decision Lists: another popular method

- A case statement….

Rule		Sense
fish within window	⇒	**bass**^{1}
striped bass	⇒	**bass**^{1}
guitar within window	⇒	**bass**^{2}
bass player	⇒	**bass**^{2}
piano within window	⇒	**bass**^{2}
tenor within window	⇒	**bass**^{2}
sea bass	⇒	**bass**^{1}
play/V *bass*	⇒	**bass**^{2}
river within window	⇒	**bass**^{1}
violin within window	⇒	**bass**^{2}
salmon within window	⇒	**bass**^{1}
on bass	⇒	**bass**^{2}
bass are	⇒	**bass**^{1}

Learning Decision Lists

- Restrict the lists to rules that test a single feature (1-decisionlist rules)
- Evaluate each possible test and rank them based on how well they work.
- Glue the top-N tests together and call that your decision list.

How to rank the tests?

- On a binary (homonymy) distinction used the following metric to rank the tests

$$\frac{P(\text{Sense}_1 \mid Feature)}{P(\text{Sense}_2 \mid Feature)}$$

- Tells us how "discriminative" the feature is
- Often used as log-likelihood

  |log P(Sense1|Feature)/P(Sense2|Feature)|

WSD Evaluations and baselines

- Best evaluation: **extrinsic（‘end-to-end’，‘task-based’）evaluation**
  - Embed WSD algorithm in a task and see if you can do the task better!
- What we often do for convenience: **intrinsic evaluation**
  - Exact match **sense accuracy**
    - % of words tagged identically with the human-manual sense tags
  - Usually evaluate using **held-out data** from same labeled corpus
- Baselines
  - Most frequent sense
  - The Lesk algorithm

Ceiling

- Human inter-annotator agreement
  - Compare annotations of two humans
  - On same data
  - Given same tagging guidelines
- Human agreements on all-words corpora with WordNet style senses
  - 75%-80%

Most Frequent Sense

- WordNet senses are ordered in frequency order
- So "most frequent sense" in WordNet = "take the first sense"
- Sense frequencies come from the *SemCor* corpus

Freq	Synset	Gloss
338	plant[1], works, industrial plant	buildings for carrying on industrial labor
207	plant[2], flora, plant life	a living organism lacking the power of locomotion
2	plant[3]	something planted secretly for discovery by another
0	plant[4]	an actor situated in the audience whose acting is rehearsed but seems spontaneous to the audience

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"

Corpus Lesk

- 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" (注记) of a sense.
- Choose sense with most word overlap between context and signature.

Corpus Lesk: IDF weighting

- Instead of just removing function words
  - Weigh each word by its `promiscuity' 混杂 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**

Corpus Lesk: IDF weighting

- Weigh each overlapping word by **inverse document frequency**
  - N is the total number of documents
  - df_i = "document frequency of word *i*"
  - = # of documents with word *i*

$$\text{idf}_i = \log\left(\frac{N}{df_i}\right)$$

$$score(sense_i, context_j) = \sum_{w \in overlap(signature_i, context_j)} \text{idf}_w$$

Semi-Supervised Learning

Problem:

- supervised and dictionary-based approaches require large hand-built resources
- What if you don't have so much training data?

Solution: Bootstrapping (自举法)

- Generalize from a very small hand-labeled seed-set.

Bootstrapping

- For `bass`
  - Rely on “One sense per collocation” 一个搭配一个词义 rule
    - A word reoccurring in collocation with the same word will almost surely have the same sense.
  - the word `play` occurs with the music sense of bass
  - the word `fish` occurs with the fish sense of bass

Sentences extracting using "fish" and "play"

We need more good teachers – right now, there are only a half a dozen who can **play** the free **bass** with ease. An electric guitar and **bass play**er stand off to one side, not really part of the scene, just as a sort of nod to gringo expectations perhaps.
The researchers said the worms spend part of their life cycle in such **fish** as Pacific salmon and striped **bass** and Pacific rockfish or snapper. And it all started when **fish**ermen decided the striped **bass** in Lake Mead were too skinny.

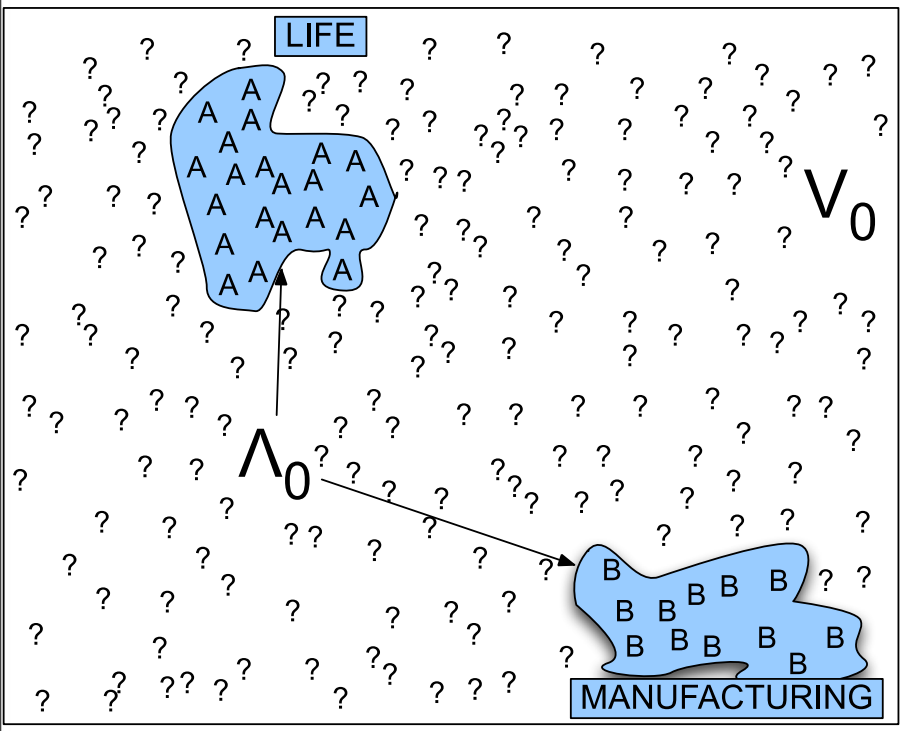
Summary: generating seeds

1) Hand labeling
2) “One sense per collocation”:
   - A word reoccurring in collocation with the same word will almost surely have the same sense.
3) “One sense per discourse”:
   - The sense of a word is highly consistent within a document - Yarowsky (1995)
   - (At least for non-function words, and especially topic-specific words)

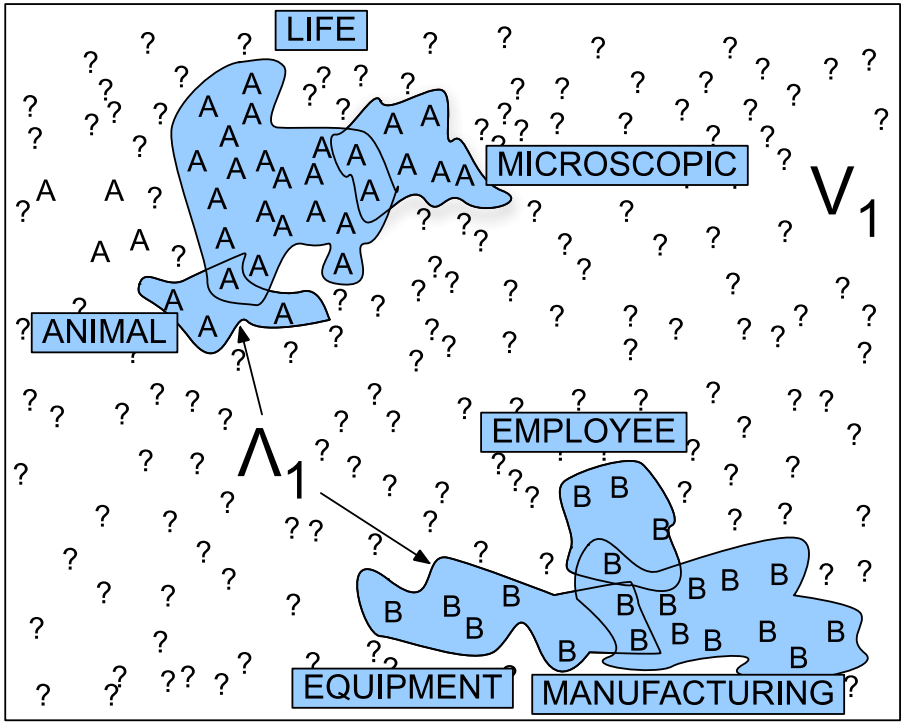
Steps in Yarowsky Algm

1) Train Classifier of seed set
   a. Yarowsky used decision list
2) Apply classifier to entire sample and compute weight of classification for each case
3) Add examples with high weights to seed set
4) Use one-sense-per-discourse constraint to filter new examples:
   a. If several examples have been annotated as sense A, extend to all examples of the word in the discourse
5) Go back to step 1

Yarowsky bootstrapping algorithm for the word "plant"



(a)



(b)

Seed set senses for plant: A vs B

Example of Classifier

LogL	Collocation	Sense
8.10	*plant* life	→ A
7.58	manufacturing *plant*	→ B
7.39	life (within +-2-10 words)	→ A
7.20	manufacturing (in +- 2-10 words)	→ B
6.27	animal (within +-2-10 words)	→ A
4.70	equipment (within +-2-10 words)	→ B
4.39	employee (within +-2-10 words)	→ B
4.30	assembly *plant*	→ B
4.10	*plant* closure	→ B
3.52	*plant* species	→ A
3.48	automate (within +-10 words)	→ B
3.45	microscopic *plant*	→ A
	...	

Collocation rules ordered by log-likelihood. First rule that applies, wins

Recent Approaches

- Unsupervised graph-based WSD
  - Connectivity of senses is exploited
- Many new methods in Senseval/Semeval conferences
- Multi-lingual WSD using BabelNet sense inventory
- Deep learning and neural networks

Problems

- Given these general ML approaches, how many classifiers do I need to perform WSD robustly
  - One for each ambiguous word in the language
- How do you decide what set of tags/labels/senses to use for a given word?
  - Depends on the application

WordNet Bass

- Tagging with this set of senses is an impossibly hard task that’ s probably overkill for any realistic application

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)

Summary of WSD

- Word Sense Disambiguation: choosing correct sense in context
- Applications: MT, QA, etc.
- Three classes of Methods
  - Supervised Machine Learning: Naive Bayes classifier
  - Thesaurus/Dictionary Methods
  - Semi-Supervised Learning
- Main intuition
  - There is lots of information in a word's context
  - Simple algorithms based just on word counts can be surprisingly good

WSD Performance

- Varies widely depending on how difficult the disambiguation task is
- Accuracies of over 90% are commonly reported on some of the classic, often fairly easy, WSD tasks (pike, star, interest)
- Senseval brought careful evaluation of difficult WSD (many senses, different POS)
- Senseval 1: more fine grained senses, wider range of types:
  - Overall: about 75% accuracy
  - Nouns: about 80% accuracy
  - Verbs: about 70% accuracy