NATURAL LANGUAGE PROCESSING

UNIT-2

Language Models, Lexical Semantics

feedback/corrections: vibha@pesu.pes.edu

VIBHA MASTI

Language Modeling

- Determine probability of given sequence of words
- NLG
- Statistical language model: probability distribution over all possible sentences/sequence of words

$$P(w_1, w_2, \ldots, w_m)$$

Language Models

Statistical LMs
- N grams
- HMMs
- linguistic rules

Neural LMs
- NNs

BAYES' THEOREM

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}$$

N-GRAMS

- Sequence of N tokens
- 2-grams: "I am", "am a", "a student"

Probabilistic LM

- Probability of word w given history h
- Sequence of n words: $w_1 \dots w_n$ or w_1^n
- Joint probability of seq : $P(w_1^n)$

$$P(w_1^n) = P(w_1)\, P(w_2 | w_1)\, P(w_3 | w_1^2) \dots P(w_n | w_1^{n-1})$$

$$P(w_1^n) = \prod_{i=1}^{n} P(w_i | w_1^{i-1}) = P(W)$$

- This joint probability too difficult to calculate
- Insted, n-grams

N-GRAMS LM

$$P(w_1^n) = \prod_{i=k+1}^{n} P(w_i | w_{i-k}^{i-1}) \quad \text{where } k = \text{no. of grams}$$

Eg: 3-gram or tri-gram model

W = word 1 word 2 word 3 word 4

$$P(W) = P(w_1^4) = \prod_{i=4}^{4} P(w_i | w_{i-3}^{i-1}) = P(w_4 | w_1^3)$$

- Assumption: $P(w_n|w_1^{n-1}) \approx P(w_n|w_{k+1}^{n-1})$

Markov Models

- Models that assume $P(q_i)$ depends only on prev states $i-k+1$ to $i-1$

Bigram MLE

$$P(w_n|w_{n-1}) = \frac{C(w_{n-1}\, w_n)}{C(w_{n-1}\, w)}$$

any word w

$$P(w_n|w_{n-1}) = \frac{C(w_{n-1}\, w_n)}{C(w_{n-1})}$$

unigram count of w_{n-1}

Q: Consider the corpus of

<s> I am here </s>
<s> Who am I </s>
<s> I would like to know </s>

Find $P(I|\langle s\rangle)$, $P(\langle /s\rangle|\text{here})$, $P(\text{know}|\text{like})$

$$P(I|\langle s\rangle) = \frac{C(\langle s\rangle\, I)}{C(\langle s\rangle)} = \frac{2}{3}$$

$$P(\langle /s\rangle|\text{here}) = \frac{C(\text{here}\ \langle /s\rangle)}{C(\text{here})} = \frac{1}{1}$$

$$P(\text{know} \mid \text{like}) = \frac{C(\text{like know})}{C(\text{like})} = \frac{0}{1}$$

Probability Estimate of a Sentence using Bigrams

$$P(\text{The water was salty}) = P(\text{water} \mid \text{the}) \times P(\text{was} \mid \text{water}) \times P(\text{salty} \mid \text{was})$$

Evaluation

1. Extrinsic: compare models
2. Intrinsic: cross entropy & perplexity

Perplexity

- Low is good

$$PP(w_1^n) = \sqrt[N]{\frac{1}{P(w_1 w_2 \ldots w_n)}}$$

- For bigrams

$$PP(w_1^n) = \sqrt[N]{\prod_{i=2}^{n} \frac{1}{P(w_i \mid w_{i-1})}}$$

Q: Let's say, A system has to recognize Operator, Sales, Technical Support and 30,000 names. If there are 1,20,000 words in total and word "operator", "Sales" and "Technical support" occurs 30,000 times each and 30,000 names each occur once only. Compute the perplexity in this scenario.

Length of sentence = 120,000

$$PP = \left(\frac{1}{4}^{30000} \times \frac{1}{4}^{30000} \times \frac{1}{4}^{30000} \times \frac{1}{120000}^{30000} \right)^{-\frac{1}{120000}}$$

$$PP = \left(\frac{1}{4}^{3} \times \frac{1}{120000} \right)^{-\frac{1}{4}}$$

$$= \left(1.302 \times 10^{-7}\right)^{-1/4} = 52.64$$

Q: Suppose there are 100 Characters in a language L. Let's say all characters are equally likely. Find the perplexity for a sequence of length N.

$$PP = \left(\frac{1}{100}^{N} \right)^{-\frac{1}{N}} = 100$$

Q: Suppose there are 3 characters in a language, and there is a Unigram Model. The prob for the 3 characters given by the model are P("A")=P("C")=0.25 and P("B")=0.5. What will be the perplexity for the sequence "AAA" and "ABC"? And what if the probability of the three characters are equally likely?

Model #1

$$PP(AAA) = \left(\frac{1}{4} \times \frac{1}{4} \times \frac{1}{4}\right)^{-\frac{1}{3}} = \left(\frac{1}{4}^{3}\right)^{-\frac{1}{3}} = 4$$

$$PP(ABC) = \left(\frac{1}{4} \times \frac{1}{4} \times \frac{1}{2}\right)^{-\frac{1}{3}} = 3.174$$

Model #2

$$P(AAA) = \left(\frac{1}{3}^{3}\right)^{-\frac{1}{3}} = 3$$

$$P(ABC) = \left(\frac{1}{3}^{3}\right)^{-\frac{1}{3}} = 3$$

Shannon Visualization Method

- Start with <s> and choose a random bigram (<s>, w) according to its probability
- Follow with a random bigram (w, v) according to its probability
- Keep going until a bigram (x, </s>) is generated
- String words together

<s> I
I want
want to
to eat
eat Chinese
Chinese food
food
</s>
I want to eat Chinese food

- **Probability sampling**

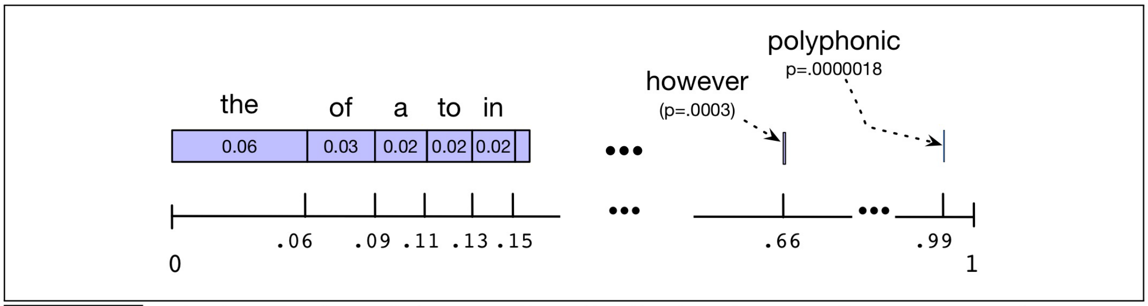


Figure 3.3 A visualization of the sampling distribution for sampling sentences by repeatedly sampling unigrams. The blue bar represents the frequency of each word. The number line shows the cumulative probabilities. If we choose a random number between 0 and 1, it will fall in an interval corresponding to some word. The expectation for the random number to fall in the larger intervals of one of the frequent words (*the*, *of*, *a*) is much higher than in the smaller interval of one of the rare words (*polyphonic*).

Dependence on Training Corpus

1 gram	–To him swallowed confess hear both. Which. Of save on trail for are ay device and rote life have –Hill he late speaks; or! a more to leg less first you enter
2 gram	–Why dost stand forth thy canopy, forsooth; he is this palpable hit the King Henry. Live king. Follow. –What means, sir. I confess she? then all sorts, he is trim, captain.
3 gram	–Fly, and will rid me these news of price. Therefore the sadness of parting, as they say, ’tis done. –This shall forbid it should be branded, if renown made it empty.
4 gram	–King Henry. What! I will go seek the traitor Gloucester. Exeunt some of the watch. A great banquet serv’d in; –It cannot be but so.

Figure 3.4 Eight sentences randomly generated from four n-grams computed from Shakespeare’s works. All characters were mapped to lower-case and punctuation marks were treated as words. Output is hand-corrected for capitalization to improve readability.

Problem with Probabilistic n-grams

- Overfitting
- Zeros (unseen words)

SMOOTHING

- Words follow zipfian distribution
- zero prob for OOV words

1. Laplace / Add-One Smoothing

- Modify MLE

For Unigrams

Normal: $P(w_i) = \frac{C(w_i)}{N}$

total word tokens in corpus

Laplace: $P_{laplace}(w_i) = \frac{C(w_i) + 1}{N + V}$

vocab size

For Bigrams

Normal: $P(w_i | w_{i-1}) = \frac{C(w_{i-1}\, w_i)}{C(w_{i-1})}$

Laplace: $P_{laplace}(w_i | w_{i-1}) = \frac{C(w_{i-1}\, w_i) + 1}{C(w_{i-1}) + V}$

unique bigrams

Q: Find Normal and Laplace MLEs

String	Count	Likelihood	Count+1	Add-1 Likelihood
xy a	100	100/300	101	101/326
xy b	0	0	1	1/326
xy c	0	0	1	1/326
xy d	200	200/300	201	201/326
⋮	⋮	⋮	⋮	⋮
xy y	0	0	1	1/326
xy z	0	0	1	1/326
Total xy ␣	300	300/300	326	326/326

No. of bigrams = 26

Reconstituted Counts

$$c^*(w_{i-1}\, w_i) = P_{laplace}(w_{i-1}\, w_i) * c(w_{i-1})$$

Discount

$$d_c = \frac{c^*}{c}$$

- For Laplace, discounts are disproportionate

2. Add-k Smoothing

$$P_{add\text{-}k}(w_{i-1} w_i) = \frac{C(w_{i-1} w_i) + k}{C(w_{i-1}) + kV}$$

- can re-write $kV = m$
- m phantom bigrams added
- $k = m\left(\frac{1}{V}\right)$ where each of the V bigrams equally likely

$$P_{add\text{-}k}(w_{i-1} w_i) = \frac{C(w_{i-1} w_i) + m\left(\frac{1}{V}\right)}{C(w_{i-1}) + m}$$

3. Unigram-Prior Smoothing

$$P_{unigram\text{-}prior}(w_{i-1} w_i) = \frac{C(w_{i-1} w_i) + m\,P(w_i)}{C(w_{i-1}) + m}$$

- $P(w_i)$ = unigram probability = $\frac{C(w_i)}{N}$

total word token occurrences in corpus

4. Good Turing Discounting

- Use count of words seen just once to estimate unseen words

- N_c : no. of tokens of frequency c
  N_1 : no. of tokens that occur only once

Q: Find N_1, N_2, N_3, N_4

Word	Freq
Ram	4
is	4
a	1
good	2
boy	1
and	1
at	1
studies	1

$N_1 = 5$
$N_2 = 1$
$N_3 = 0$
$N_4 = 2$

- Good Turing count

$$c^* = \frac{(c+1)\, N_{c+1}}{N_c}$$

- Probability of unseen word

$$P_{GT}(w_i) = \frac{N_1}{N}$$

Q: Suppose $N = 18$. Use normal and GT.

carp	Perch	Whitefish	trout	Salmon	eel
10	3	2	1	1	1

P(trout) = ?
P(bass) = ?

Regular:

$P(\text{trout}) = 1/18$

$P(\text{bass}) = 0$

$N_1 = 3$

$N_2 = 1$

$N_3 = 1$

$N_{10} = 1$

Good Turing:

$$P_{GT}(\text{bass}) = \frac{N_1}{N} = \frac{3}{18}$$

$$c^*(\text{trout}) = \frac{(c+1)\ N_{c+1}}{N_c} = \frac{2N_2}{N_1} = \frac{2\times 1}{3} = \frac{2}{3}$$

$$P_{GT}(\text{trout}) = \frac{2/3}{18} = \frac{1}{27}$$

Q: Compute GT probabilities of test data (bigrams)

Corpus (Training data):

The following represents the corpus of words:

cats chase rats

cats meow

rats chatter

cats chase birds

rats sleep

Test Data

rats chase birds

$P(\text{rats chase birds})$

$$= P(\text{rats}|\langle s\rangle) * P(\text{chase}|\text{rats}) * P(\text{birds}|\text{chase}) * P(\langle /s\rangle|\text{birds})$$

$N_1 = 10$
$N_2 = 2$
$N_3 = 1$

Bigram	Count
<s> cats	3
cats chase	2
chase rats	1
rats </s>	1
cats meow	1
meow </s>	1
<s> rats	2
rats chatter	1
chatter </s>	1
chase birds	1
birds </s>	1
rats sleep	1
sleep </s>	1
	17

Unigram	Count
cats	3
chase	2
rats	1
<s>	5
</s>	5
meow	1
rats	2
chatter	1
birds	1
sleep	1
	22

① $c^*(\langle s\rangle\ \text{rats}) = \dfrac{3 \times N_3}{N_2} = \dfrac{3\times 1}{2} = \dfrac{3}{2}$

$P_{GT}(\langle s\rangle\ \text{rats}) = \dfrac{3/2}{} = \dfrac{3}{}$

② $P_{GT}(\text{rats chase}) = \dfrac{N_1}{N} = \dfrac{10}{17}$

③ $c^*(\text{chase birds}) = \dfrac{2 N_2}{N_1} = \dfrac{2\times 2}{10} = \dfrac{2}{5}$

5. Katz Backoff

- Try $P(w_i \mid w_{i-2} w_{i-1})$
- If 0, backoff to $P(w_i \mid w_{i-1})$
- If 0, backoff to $P(w_i)$
- Bigram version

$$P_{Katz}(w_i \mid w_{i-1}) = \begin{cases} P_{GT}(w_i \mid w_{i-1}), & C(w_{i-1}, w_i) > 0 \\ \alpha(w_{i-1}) P_{GT}(w_i), & \text{otherwise} \end{cases}$$

$$\alpha(y) = \frac{1 - \sum \text{discounted prob of bigrams starting with } y}{\sum \text{prob of } w \text{ of all unobserved bigrams } (y,w)}$$

$$= \frac{1 - \sum_{w : c(yw) > 0} P_{GT}(w \mid y)}{\sum_{w : c(yw) = 0} P_{GT}(w)}$$

6. Class-Based Backoff

- Back off to the class, not the n-1 gram
- $c(\text{dog} \mid \text{friendly}) = c(\text{noun} \mid \text{friendly})$

7. Linear Interpolation

$$\hat{P}(w_i | w_{i-2} w_{i-1}) = \lambda_1 P(w_i | w_{i-2} w_{i-1}) + \lambda_2 P(w_i | w_{i-1}) + \lambda_3 P(w_i)$$

$$\sum_i \lambda_i = 1$$

8. Kneser-Ney Smoothing

- Absolute discounting
- Subtract 0.75 for all, 0.5 for bigrams of count 1

$$P_{KN}(w_i | w_{i-1}) = \frac{\max(C(w_{i-1} w_i) - d, 0)}{\sum_v C(w_{i-1} v)} + \lambda(w_{i-1}) P_{continuation}(w_i)$$

bigrams starting with w_{i-1}

novel continuations

$$P_{continuation}(w_i) = \frac{|\{w_{i-1} : C(w_{i-1} w_i) > 0\}|}{|\{(w_{j-1}, w_j) : C(w_{j-1} w_j) > 0\}|}$$

bigram types

$$\lambda(w_{i-1}) = \frac{d}{C(w_{i-1})} |\{w : C(w_{i-1} w) > 0\}|$$

discount

word types that can follow w_{i-1}

Q: Paul is running
Mary is running
Nick is cycling
They are running

$P_{cont}(is) = ?$
$P_{cont}(running) = ?$
$P_{KN}(running \mid is) = ?$

$d = 1$

No. of bigrams (unique) = 7

$$P_{continuation}(is) = \frac{\text{unique words preceding is}}{\text{unique bigrams}}$$

$$= \frac{3}{7}$$

$$P_{continuation}(running) = \frac{2}{7}$$

$$P_{KN}(running \mid is) = \frac{2-1}{3} + \lambda(is)\, P_{cont}(running)$$

$$= \frac{1}{3} + \lambda(is) \times \frac{2}{7}$$

$$\lambda(is) = \frac{1}{3} \times 2 = \frac{2}{3}$$

$$P_{KN}(running \mid is) = \frac{1}{3} + \frac{2}{3} \times \frac{2}{7} = 0.524$$

9. Stupid Backoff

$$S(w_i \mid w_{i-k+1}^{i-1}) = \begin{cases} \dfrac{c(w_{i-k+1}^{i})}{c(w_{i-k+1}^{i-1})}, & \text{count}(w_{i-k+1}^{i}) > 0 \\ 0.4\, S(w_i \mid w_{i-k+2}^{i-1}), & \text{otherwise} \end{cases}$$

$$S(w_i) = \frac{c(w_i)}{N}$$

WORD SENSES

- Lexical semantics: study of word meanings
- Word sense: discrete representation of one meaning of a word
  - eg: bank: river
    bank: finance
- Word form: inflected word as it appears in text
  - eg: beautify, beautiful, beautifully are verb, adj and adv forms
- Gloss: human-readable meaning representations of senses

Relation Between Senses

1. **Homonyms:** identical orthographic (spelling) form
   unrelated meaning
   ↳ bank - finance
   ↳ bank - river

2. **Polysemes:** related but distinct sences
   ↳ bank - finance
   ↳ bank - blood

3. **Metonyms:** usage of one term as a "stand-in" for another
   close association
   ↳ the US government
   ↳ Washington

4. **Synonyms:** similar senses (no perfect synonyms)
   ↳ vehicle
   ↳ automobile

5. **Homophones:** same pronunciation, different orthography, different senses
   ↳ wood
   ↳ would

6. **Homographs:** different pronunciation, same orthography, different senses
   ↳ minute (my-nute: small / tiny)
   ↳ minute (min-ut: 60 seconds)

7. **Hyponyms:** one sense is a hyponym of another if it is more specific than the second
   - ↳ mango hyponym of fruit
   - ↳ car hyponym of vehicle

8. **Hypernyms:** opposite of hyponyms
   - ↳ fruit hypernym of mango
   - ↳ vehicle hypernym of car

9. **Meronyms:** a word is part of another
   - ↳ wheel meronym of car

10. **Holonym:** opposite of meronym
    - ↳ car holonym of wheel

Zeugma test for senses

- Check if a word has different senses

  1. Which flights serve breakfast?
  2. Which flights serve London?

- Create conjuction

  Which flights serve breakfast and London?

- If it sounds weird, distinct senses

WordNet

- Lexical knowledge base
- Concepts in semantic network
- Psycholinguistic theory

- Syntagmatic & paradigmatic

pur, furry

animal, mammal

Componential Semantics

- Disambiguate words using features
- Hard to design
- Eg: features – (furry, carnivorous, heavy, domesticable)

  for cat – (Y, Y, N, Y)

  for tiger – (Y, Y, Y, N)

Relational Semantics

- Synonymy, antonymy, gradation (word to word)
- Hypernymy, hyponymy, meronymy, holonymy, entailment, troponymy (synset to synset)

Synset

- Unordered set of roughly synonymous words
- Eg: chump^1 : gullible person

This sense of "chump" is shared by 9 words:

`chump`^{1}, `fool`^{2}, `gull`^{1}, `mark`^{9}, `patsy`^{1}, `fall guy`^{1}, `sucker`^{1}, `soft touch`^{1}, `mug`^{2}

DSF Format of Synset

ID :: 121

CATEGORY :: NOUN

CONCEPT :: अपने से छोटों के प्रति हृदय में उठनेवाला प्रेम

EXAMPLE :: “चाचा नेहरू को बच्चों से बहुत ही स्नेह था”

SYNSET :: स्नेह,नेह,लगाव,ममता

WordNet Relations

Relation	Also Called	Definition	Example
Hypernym	Superordinate	From concepts to superordinates	*breakfast*^{1} $\rightarrow$ *meal*^{1}
Hyponym	Subordinate	From concepts to subtypes	*meal*^{1} $\rightarrow$ *lunch*^{1}
Instance Hypernym	Instance	From instances to their concepts	*Austen*^{1} $\rightarrow$ *author*^{1}
Instance Hyponym	Has-Instance	From concepts to concept instances	*composer*^{1} $\rightarrow$ *Bach*^{1}
Member Meronym	Has-Member	From groups to their members	*faculty*^{2} $\rightarrow$ *professor*^{1}
Member Holonym	Member-Of	From members to their groups	*copilot*^{1} $\rightarrow$ *crew*^{1}
Part Meronym	Has-Part	From wholes to parts	*table*^{2} $\rightarrow$ *leg*^{3}
Part Holonym	Part-Of	From parts to wholes	*course*^{7} $\rightarrow$ *meal*^{1}
Substance Meronym		From substances to their subparts	*water*^{1} $\rightarrow$ *oxygen*^{1}
Substance Holonym		From parts of substances to wholes	*gin*^{1} $\rightarrow$ *martini*^{1}
Antonym		Semantic opposition between lemmas	*leader*^{1} $\Longleftrightarrow$ *follower*^{1}
Derivationally Related Form		Lemmas w/same morphological root	*destruction*^{1} $\Longleftrightarrow$ *destroy*

Figure C.2 Noun relations in WordNet.

Relation	Definition	Example
Hypernym	From events to superordinate events	*fly*^{9} $\rightarrow$ *travel*^{5}
Troponym	From events to subordinate event (often via specific manner)	*walk*^{1} $\rightarrow$ *stroll*^{1}
Entails	From verbs (events) to the verbs (events) they entail	*snore*^{1} $\rightarrow$ *sleep*^{1}
Antonym	Semantic opposition between lemmas	*increase*^{1} $\Longleftrightarrow$ *decrease*^{1}
Derivationally Related Form	Lemmas with same morphological root	*destroy*^{1} $\Longleftrightarrow$ *destruction*^{1}

Figure C.3 Verb relations in WordNet.

Word Sense Disambiguation

1. Supervised Learning Approach

- Training corpus: words tagged in context with sense
- choose WS of word
- SemCor

(a) Baseline

- Choose most frequent sense always

(b) Feature-based WSD

↳ Collocational features

↳ BOW features

- Naive Bayes classifier WSD
- Choose best sense $\hat{s}$ from possible S for a feature vector $\vec{f}$ of a word

$$\hat{s} = \underset{s \in S}{\operatorname{argmax}} \; P(s \mid \vec{f})$$

$$= \underset{s \in S}{\operatorname{argmax}} \; \frac{P(\vec{f} \mid s)\, P(s)}{P(\vec{f})}$$

constant

- Assume each feature conditionally independent from the other

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

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

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

probability of that sense for the word

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

how often a feature f_j occurs for sense s

Q: If a collocational feature $[w_{i-2} = guitar]$ occurred 3 times for sense bass', sense bass' occurred 60 times in training, and bass occurred 70 times in training,

$P(f_j \mid s) = ?$

$P(s) = ?$

$$P(f_j \mid s) = \frac{count(f_j, s)}{count(s)} = \frac{3}{60} = 0.05$$

$$P(s) = \frac{count(s, w)}{count(w)} = \frac{60}{70} = 0.86$$

(c) Lesk Algorithm

- Corpora like SemCor expensive
- Knowledge-based method
- Baseline
- Choose sense whose gloss shares most words with target word's neighbourhood

Algorithm

1. Retrieve sense definitions of words
2. Determine definition overlap
3. Choose senses with highest overlap

Q:

Example: Disambiguate PINE and CONE

- PINE
  - 1. kinds of evergreen tree with needle-shaped leaves
  - 2. waste away through sorrow or illness
- CONE
  - 1. solid body which narrow to a point.
  - 2. something of this shape whether solid or hollow
  - 3. fruit of certain evergreen trees

PINE #1 ∩ CONE #1 = 0
PINE #1 ∩ CONE #2 = 0
PINE #1 ∩ CONE #3 = 2
PINE #2 ∩ CONE #1 = 0
PINE #2 ∩ CONE #2 = 0
PINE #2 ∩ CONE #3 = 0

(ignoring stopwords and performing stemming)

Q: Disambiguate bank with sentences

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"

```
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 C.9 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*.

(d) Corpus Lesk Algorithm

- Needs SemCor-like labelled corpus
- Every sense has a signature
- Signature contains all words from sentences that contain that sense
- Choose sense with most word overlap b/w context and signature
- Weigh each word with its inverse document frequency

no. of docs

$$IDF_i = \log\left(\frac{N_{doc}}{nd_i}\right)$$

no. of docs with word i

$$\text{score}(\text{sense}_i, \text{context}_j) = \sum_{w \in \text{overlap}(\text{signature}_i, \text{context}_j)} IDF_w$$

(e) Graph-Based

- WordNet: senses are nodes
  relations (meronymy etc.) are edges

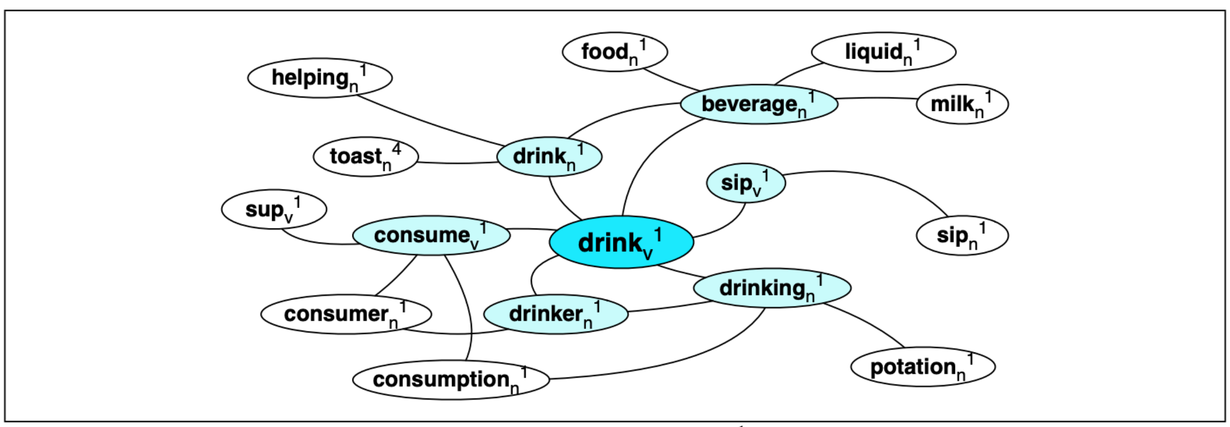


Figure C.10 Part of the WordNet graph around drink$_v^1$, after Navigli and Lapata (2010).

- Using for WSD: add target word & context into graph with directed edges to each of their senses

- Eg: "She drank some milk"

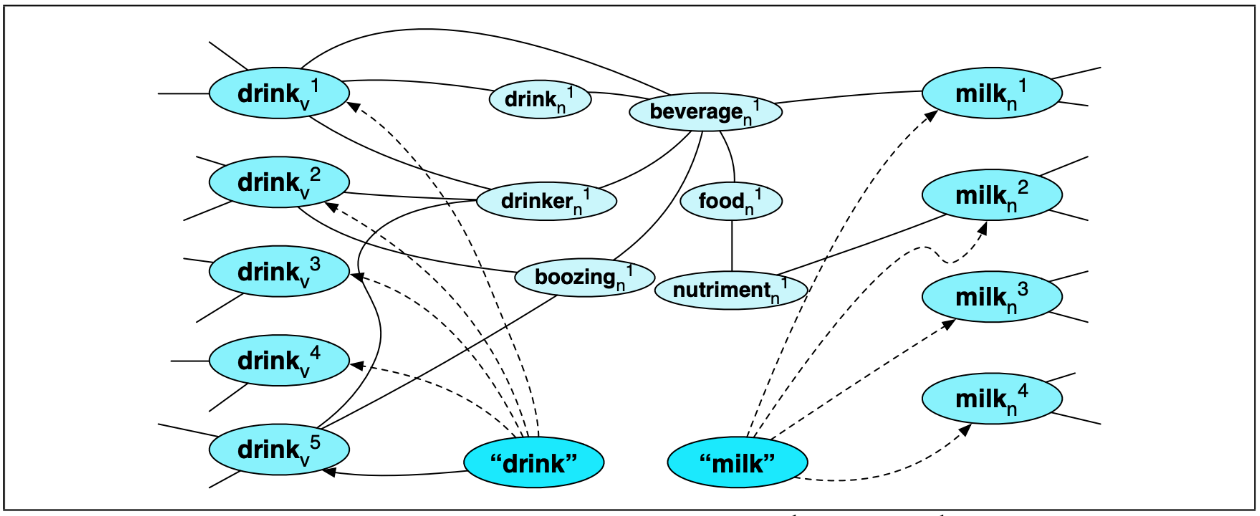


Figure C.11 Part of the WordNet graph between drink$_v^1$ and milk$_n^1$, for disambiguating a sentence like *She drank some milk*, adapted from Navigli and Lapata (2010).

- Correct sense: using centrality measures
  - ↳ degree
  - ↳ personalized page rank

(f) Semi-supervised WSD: Bootstrapping - Yarowsky

- Seed labelled corpus Λ_0
- Large unlabelled corpus V_0
- Algorithm
  1. Train classifier on Λ_0
  2. Label unlabelled corpus V_0
  3. Select k most confident labels and add to labelled set, called Λ_1
  4. Repeat until low error rate or all tagged

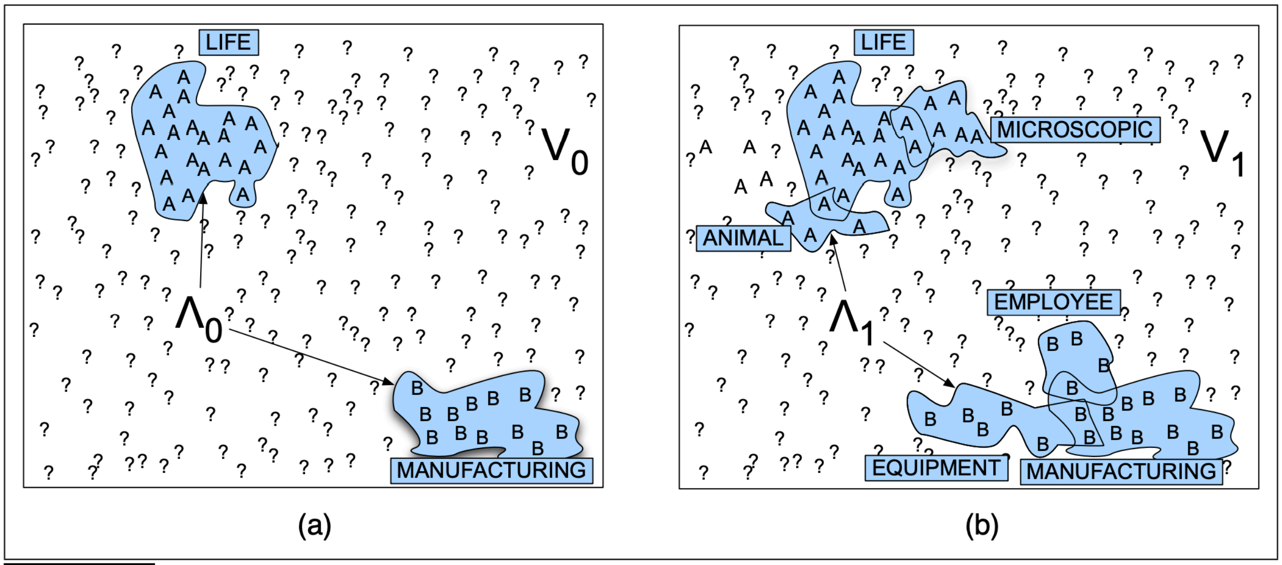


Figure C.12 The Yarowsky algorithm disambiguating "plant" at two stages; "?" indicates an unlabeled observation, A and B are observations labeled as SENSE-A or SENSE-B. The initial stage (a) shows only seed sentences Λ_0 labeled by collocates ("life" and "manufacturing"). An intermediate stage is shown 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).

Yarowsky Heuristics

- To label initial seed

1. One sense per collocation
2. One sense per disclosure

- Eg: bass — play
  bass — fish

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,

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...

Figure C.13 Samples of *bass* sentences extracted from the WSJ by using the simple correlates *play* and *fish*.

(g) Unsupervised - Word Sense Induction

- Clustering over word embeddings

Most algorithms for word sense induction use some sort of clustering over word embeddings. (The earliest algorithms, due to Schütze (Schütze 1992, Schütze 1998), represented each word as a context vector of bag-of-words features $\vec{c}$.) Then in training, we use three steps.

1. For each token w_i of word w in a corpus, compute a context vector $\vec{c}$.
2. Use a **clustering algorithm** to **cluster** these word-token context vectors $\vec{c}$ into a predefined number of groups or clusters. Each cluster defines a sense of w.
3. Compute the **vector centroid** of each cluster. Each vector centroid $\vec{s}_j$ is a **sense vector** representing that sense of w.

Since this is an unsupervised algorithm, we don't have names for each of these "senses" of w; we just refer to the jth sense of w.

- Disambiguation
  1. Compute context vector $\vec{c}$ for token w_i
  2. Retrieve all sense vectors $\vec{s}_j$ for w
  3. Assign w_i to closest $\vec{s}_j$

- Requirements:
  1. Clustering algo
  2. Distance metric

- Evaluation
  ↳ extrensic (best to do)
  ↳ intrinsic (against gold std)

WORD SIMILARITY

1. Thesaurus based
2. Distributional algorithms
   - similar distributions in corpus

- Relatedness $\neq$ similarity
  - antonyms are highly related

1. Thesaurus based

(a) Path-based

- Shorter path b/w senses in thesaurus hierarchy, more similar
- Measure no. of edges

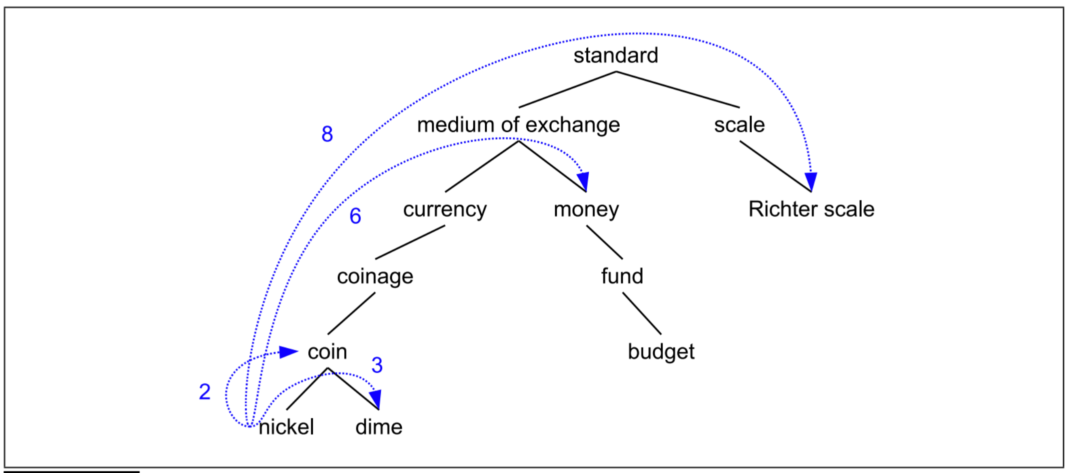


Figure C.5 A fragment of the WordNet hypernym hierarchy, showing path lengths (number of edges plus 1) from *nickel* to *coin* (2), *dime* (3), *money* (6), and *Richter scale* (8).

$$\text{pathlen}(c_1, c_2) = 1 + \text{edges in shortest path}$$

- Similarity

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

- Log similarity

$$\text{sim}_{\text{path}}(c_1, c_2) = -\log(\text{pathlen}(c_1, c_2))$$

- Similarity b/w words (not senses)

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

Q: Find sim_{path}(nickel, coin), sim_{path}(coinage, Richter scale)

$$\text{sim}_{\text{path}}(\text{nickel, coin}) = \frac{1}{1+1} = 0.5$$

$$\text{sim}_{\text{path}}(\text{coinage, Richter scale}) = \frac{1}{1+5} \approx 0.167$$

(b) Information content

- $P(c)$ = probability that a random word in corpus is instance of concept c
- $P(\text{root}) = 1$ (root concept subsumes all)

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

words under concept c

total words

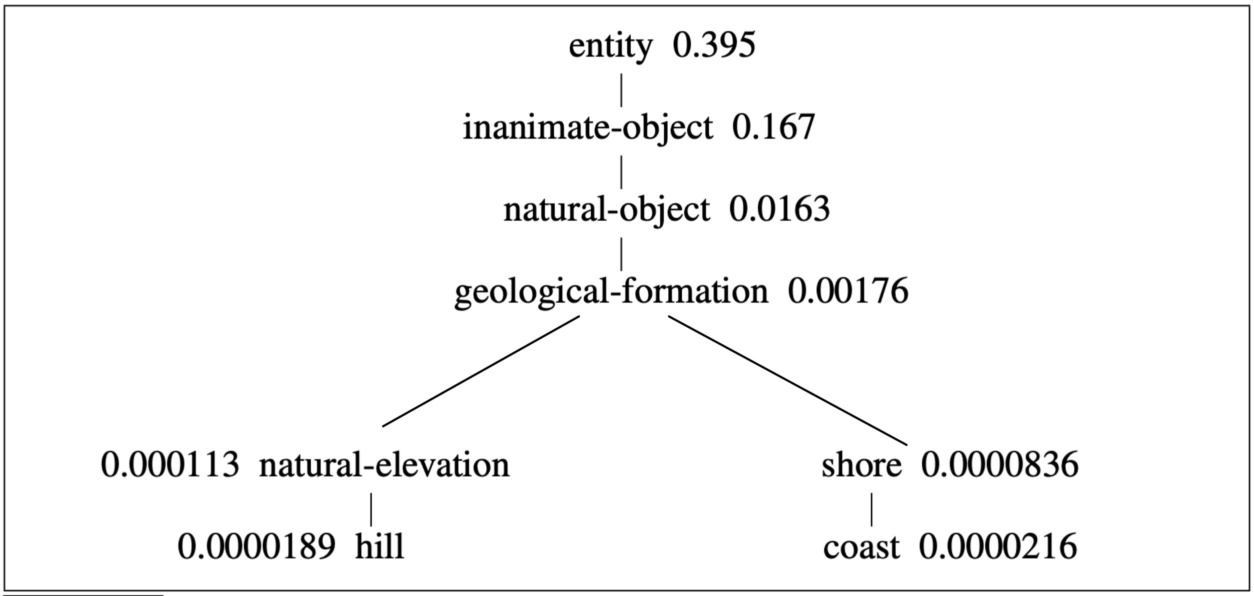


Figure C.6 A fragment of the WordNet hierarchy, showing the probability $P(c)$ attached to each content, adapted from a figure from Lin (1998).

- Information content of a concept

$$IC = -\log P(c)$$

- Lowest common subsumer

$$LCS(c_1, c_2) = \text{lowest node that subsumes } c_1 \text{ \& } c_2$$

Q: Find LCS of coinage & money (pg 31)

LCS = medium of exchange

Resnik Similarity

$$sim_{resnik}(c_1, c_2) = -\log P(LCS(c_1, c_2))$$

Lin Similarity

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

Q: Find Lin similarity of hill and coast

LCS(hill, coast) = geological information

$$\text{sim}_{\text{Lin}} = \frac{2 \times \log P(\text{geological information})}{\log P(\text{hill}) + \log P(\text{coast})}$$

$$= \frac{2 \log (0.00176)}{\log(0.0000189) + \log(0.0000216)}$$

$$= 0.587$$

Extended Lesk

- *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 occurs in both glosses, Extended Lesk adds in a score of n^2 (the relation is non-linear because of the Zipfian relationship between lengths of phrases and their corpus frequencies; longer overlaps are rare, so they should be weighted more heavily). Here, the overlapping phrases are *paper* and *specially prepared*, for a total similarity score of $1^2 + 2^2 = 5$.

Summary

$$
\begin{aligned}
\text{sim}_{\text{path}}(c_1,c_2) &= \frac{1}{\text{pathlen}(c_1,c_2)} \\
\text{sim}_{\text{Resnik}}(c_1,c_2) &= -\log P(\text{LCS}(c_1,c_2)) \\
\text{sim}_{\text{Lin}}(c_1,c_2) &= \frac{2\times\log P(LCS(c_1,c_2))}{\log P(c_1)+\log P(c_2)} \\
\text{sim}_{\text{JC}}(c_1,c_2) &= \frac{1}{2\times\log P(\text{LCS}(c_1,c_2)) - (\log P(c_1)+\log P(c_2))} \\
\text{sim}_{\text{eLesk}}(c_1,c_2) &= \sum_{r,q\in\text{RELS}} \text{overlap}(\text{gloss}(r(c_1)),\text{gloss}(q(c_2)))
\end{aligned}
$$

Figure C.7 Five thesaurus-based (and dictionary-based) similarity measures.

LEXICONS

Scherer Typology of Affective States

- **Emotion**: brief organically synchronized … evaluation of a major event
  - *angry, sad, joyful, fearful, ashamed, proud, elated*
- **Mood**: diffuse non-caused low-intensity long-duration change in subjective feeling
  - *cheerful, gloomy, irritable, listless, depressed, buoyant*
- **Interpersonal stances**: affective stance toward another person in a specific interaction
  - *friendly, flirtatious, distant, cold, warm, supportive, contemptuous*
- **Attitudes**: enduring, affectively colored beliefs, dispositions towards objects or persons
  - *liking, loving, hating, valuing, desiring*
- **Personality traits**: stable personality dispositions and typical behavior tendencies
  - *nervous, anxious, reckless, morose, hostile, jealous*

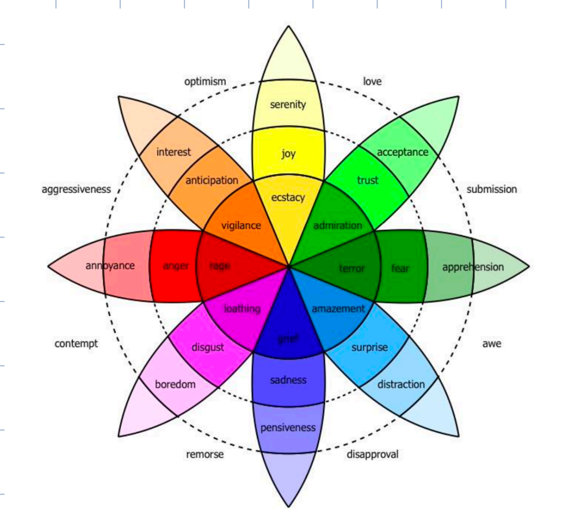
① Emotion

(a) Ekman's 6 basic emotions

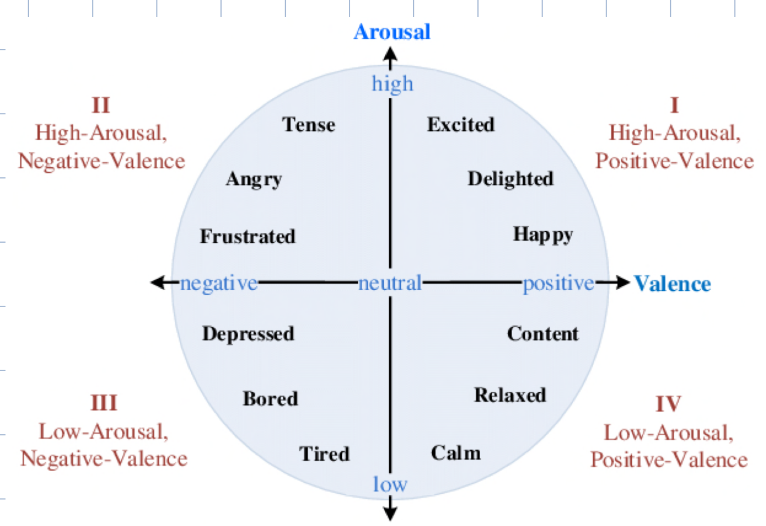
- Surprise
- Happiness
- Anger
- Fear
- Disgust
- Sadness

(b) Plutchik's wheel of emotion

- 8 basic



(c) VAD - Valence Arousal Dominance



Valence		Arousal		Dominance	
vacation	.840	enraged	.962	powerful	.991
delightful	.918	party	.840	authority	.935
whistle	.653	organized	.337	saxophone	.482
consolation	.408	effortless	.120	discouraged	.0090
torture	.115	napping	.046	weak	.045

Figure 21.4 Samples of the values of selected words on the three emotional dimensions from Mohammad (2018a).

- VAD lexicon: best-worst scoring