Lexical Semantics
CSCI-GA.2590 – Lecture 7A

Ralph Grishman
Words and Senses

• Until now we have manipulated structures based on words

• But if we are really interested in the meaning of sentences, we must consider the *senses* of words
  • most words have several senses
  • frequently several words share a common sense
  • both are important for information extraction
Terminology

• multiple senses of a word
• polysemy (and homonymy for totally unrelated senses ("bank"))
• metonymy for certain types of regular, productive polysemy ("the White House", "Washington")
• zeugma (conjunction combining distinct senses) as test for polysemy ("serve")
• synonymy: when two words mean (more-or-less) the same thing
• hyponymy: X is the hyponym of Y if X denotes a more specific subclass of Y (X is the hyponym, Y is the hypernym)
WordNet

- large-scale database of lexical relations
- organized as graph whose nodes are synsets (synonym sets)
  - each synset consists of 1 or more word senses which are considered synonymous
  - fine-grained senses
- primary relation: hyponym / hypernym
- sense-annotated corpus SEMCOR
  - subset of Brown corpus
- available on Web
  - along with foreign-language Wordnets
Two basic tasks

Two basic tasks we will consider today:

• given an inventory of senses for a word, deciding which sense is used in a given context

• given a word, identifying other words that have a similar meaning [in a given context]
Word Sense Disambiguation

• process of identifying the sense of a word in context
• WSD evaluation: either using WordNet or coarser senses (e.g., main senses from a dictionary)
• local cues (Weaver): train a classifier using nearby words as features
• either treat words at specific positions relative to target word as separate features
• or put all words within a given window (e.g., 10 words wide) as a 'bag of words'
• simple demo for 'interest'
Simple supervised WSD algorithm: naive Bayes

• select sense
  
  \[ s' = \arg\max(s) \ P(s | F) = \arg\max(s) \ P(s) \ \prod_i P(f[i] | s) \]

  where \( F = \{f_1, f_2, \ldots \} \) is the set of context features
  
  – typically specific words in immediate context

• Maximum likelihood estimates for \( P(s) \) and \( P(f[i] | s) \)
  
  can be easily obtained by counting
  
  – some smoothing (e.g., add-one smoothing) is needed
  
  – works quite well at selecting best sense (not at estimating probabilities)

  – But needs substantial annotated training data for each word
Sources of training data for supervised methods

• SEMCOR and other hand-made WSD corpora
• dictionaries
  • Lesk algorithm: overlap of definition and context
• bitexts (parallel bilingual data)
• crowdsourcing
• Wikipedia links
  – treat alternative articles linked from the same word as alternative senses (Mihalcea NAACL 2007) articles provide lots of info for use by classifier
Wikification and Grounding

• We can extend the notion of disambiguating individual words to cover multi-word terms and names.
  – Wikipedia comes closest to providing an inventory of such concepts: people, places, classes of objects, ....
  – This has led to the process of *Wikification*: linking the phrases in a text to Wikipedia articles.

• *Wikification demo* ([UIUC](http://uiuc.edu))

• annual evaluation (for names) as part of NIST Text Analysis Conference
Local vs Global Disambiguation

• Local disambiguation
  – each mention (word, name, term) in an article is disambiguated separately based on context (other words in article)

• Global disambiguation:
  – take into account coherence of disambiguations across document
  – optimize sum of local disambiguation scores plus a term representing coherence of referents
    • coherence reflected in links between Wikipedia entries
  – relative importance of prominence, local features, and global coherence varies greatly
Using Coherence

Wikipedia entries

- Texas Rangers (lawmen)
- Texas Rangers (baseball team)
- NY Yankees (baseball team)

Major League Baseball

document: ?

... the Texas Rangers defeated the New York Yankees ...

3/3/15

NYU
Using Coherence

Wikipedia entries

- Texas Rangers (lawmen)
- Texas Rangers (baseball team)
- NY Yankees (baseball team)

document: ?

... the Texas Rangers defeated the New York Yankees ...
Using Coherence

Wikipedia entries

Texas Rangers (lawmen)

Texas Rangers (baseball team)

NY Yankees (baseball team)

Major League Baseball

links in Wikipedia

document:

... the Texas Rangers defeated the New York Yankees ...
Supervised vs. Semi-supervised

- problem: training some classifiers (such as WSD) needs lots of labeled data
  - supervised learners: all data labeled

- alternative: semi-supervised learners
  - some labeled data ("seed") + lots of unlabeled data
Bootstrapping: a semi-supervised learner

Basic idea of bootstrapping:
• start with a small set of labeled seeds $L$ and a large set of unlabeled examples $U$

repeat
• train classifier $C$ on $L$
• apply $C$ to $U$
• identify examples with most confident labels; remove them from $U$ and add them (with labels) to $L$
Bootstrapping WSD

Premises:

• one sense per discourse (document)

• one sense per collocation
example

“bass” as fish or musical term
example

bass

catch bass

bass

play bass

bass

catch bass

play bass
example

• label initial examples

- bass
  - fish
  - catch bass

- bass
  - music
  - play bass

- bass
  - catch bass

- play bass
example

• label other instances in same document

- bass
  - fish
  - catch bass
    - fish
  - catch bass

- bass
  - music
  - play bass
    - music
  - play bass
example

• learn collocations: catch ... → fish; play ... → music
example

• label other instances of collocations

- bass
- fish
- catch bass
- fish

- bass
- music
- play bass
- music

- catch bass
- fish

- play bass
- music
Identifying semantically similar words

- using WordNet (or similar ontologies)
- using distributional analysis of corpora
Using WordNet

• Simplest measures of semantic similarity based on WordNet: path length:
  
  longer path ➔ less similar

  mammals
    • felines
      • cats
      • tigers
    • apes
      • gorillas
      • humans
Using WordNet

• path length ignores differences in degrees of generalization in different hyponym relations:

\[
\text{mammals} \quad \vdash \quad \text{cats} \quad \text{people}
\]

*a cat’s view of the world (cats and people are similar)*
Information Content

• $P(c) = \text{probability that a word in a corpus is an instance of the concept (matches the synset } c \text{ or one of its hyponyms)}$

• Information content of a concept
  \[ IC(c) = -\log P(c) \]

• If $LCS(c_1, c_2)$ is the *lowest common subsumer* of $c_1$ and $c_2$, the JC distance between $c_1$ and $c_2$ is
  \[ IC(c_1) + IC(c_2) - 2 \cdot IC(LCS(c_1, c_2)) \]
Similarity metric from corpora

• Basic idea: characterize words by their contexts; words sharing more contexts are more similar

• Contexts can either be defined in terms of adjacency or dependency (syntactic relations)

• Given a word $w$ and a context feature $f$, define pointwise mutual information $\text{PMI}$:

$$\text{PMI}(w,f) = \log \left( \frac{P(w,f)}{P(w)P(f)} \right)$$
Given a list of contexts (words left and right) we can compute a context vector for each word.

The similarity of two vectors \( v \) and \( w \) (representing two words) can be computed in many ways; a standard way is using the cosine (normalized dot product):

\[
\text{sim}_{\text{cosine}} = \sum v_i \times w_i / (|v| \times |w|)
\]

See the Thesaurus demo by Patrick Pantel.
Clusters

• By applying clustering methods we have an unsupervised way of creating semantic word classes.
Word Embeddings

• In many NLP applications which look for specific words, we would prefer a soft match (between 0 and 1, reflecting semantic similarity) to a hard match (0 or 1)

• Can we use context vectors?
• Can we use context vectors?
• In principle, yes, but
  – very large (> 10^5 words, > 10^10 entries)
  – sparse matrix representation not convenient for neural networks
  – sparse ➔ context vectors rarely overlap
• Want to reduce dimensionality
Word embeddings

- A low dimension
- real valued
- distributed

representation of a word, computed from its distribution in a corpus

NLP analysis pipeline will operate on these vectors
Producing word embeddings

- Dimensionality reduction methods can be applied to the full co-occurrence matrix
- Neural network models can produce word embeddings
  - great strides in efficiency of word embedding generators in the last few years
  - skip-grams now widely used
Skip-Grams

- Given current word, build model to predict a few immediately preceding and following words
  - Captures local context
- Use log-linear models (for efficient training)
- Train with gradient descent
- Can build WE’s from a 6 GW corpus in < 1 day on a cluster (about 100 cores)
Word similarity

• Distributional similarity effectively captured in compact representation
  • 100-200 element d.p. vector (< 2 KB / word)
  • cosine metric between vectors provides good measure of similarity
Features from Embeddings

How to use information from word embeddings in a feature-based system?

• directly: use components of vector as features
• via clusters
  • cluster words based on similarity of embeddings
  • use cluster membership as features
• via prototypes
  • select prototypical terms for task
  • feature\_i (w) = \text{sim}(w, t_i) > \tau