CMSC 723: Computational Linguistics I — Session #11

Word Sense Disambiguation

Jimmy Lin
The iSchool
University of Maryland

Wednesday, November 11, 2009

Material drawn from slides by Saif Mohammad and Bonnie Dorr
Progression of the Course

- Words
  - Finite-state morphology
  - Part-of-speech tagging (TBL + HMM)

- Structure
  - CFGs + parsing (CKY, Earley)
  - N-gram language models

- Meaning!
Today’s Agenda

- Word sense disambiguation
- Beyond lexical semantics
  - Semantic attachments to syntax
  - Shallow semantics: PropBank
Word Sense Disambiguation
Recap: Word Sense

From WordNet:

**Noun**
- `{pipe, tobacco pipe}` (a tube with a small bowl at one end; used for smoking tobacco)
- `{pipe, pipage, piping}` (a long tube made of metal or plastic that is used to carry water or oil or gas etc.)
- `{pipe, tube}` (a hollow cylindrical shape)
- `{pipe}` (a tubular wind instrument)
- `{organ pipe, pipe, pipework}` (the flues and stops on a pipe organ)

**Verb**
- `{shriek, shrill, pipe up, pipe}` (utter a shrill cry)
- `{pipe}` (transport by pipeline) “pipe oil, water, and gas into the desert”
- `{pipe}` (play on a pipe) “pipe a tune”
- `{pipe}` (trim with piping) “pipe the skirt”
Word Sense Disambiguation

- Task: automatically select the correct sense of a word
  - Lexical sample
  - All-words

- Theoretically useful for many applications:
  - Semantic similarity (remember from last time?)
  - Information retrieval
  - Machine translation
  - ...

- Solution in search of a problem? Why?
How big is the problem?

- Most words in English have only one sense
  - 62% in Longman’s Dictionary of Contemporary English
  - 79% in WordNet
- But the others tend to have several senses
  - Average of 3.83 in LDOCE
  - Average of 2.96 in WordNet
- Ambiguous words are more frequently used
  - In the British National Corpus, 84% of instances have more than one sense
- Some senses are more frequent than others
Ground Truth

- Which sense inventory do we use?
- Issues there?
- Application specificity?
Corpora

- Lexical sample
  - *line-hard-serve* corpus (4k sense-tagged examples)
  - *interest corpus* (2,369 sense-tagged examples)
  - ...

- All-words
  - SemanticCorpus (234k words, subset of Brown Corpus)
  - Senseval-3 (2081 tagged content words from 5k total words)
  - ...

- Observations about the size?
Evaluation

- Intrinsic
  - Measure accuracy of sense selection wrt ground truth

- Extrinsic
  - Integrate WSD as part of a bigger end-to-end system, e.g., machine translation or information retrieval
  - Compare ±WSD
Baseline + Upper Bound

- **Baseline**: most frequent sense
  - Equivalent to “take first sense” in WordNet
  - Does surprisingly well!
  - **62% accuracy in this case!**

<table>
<thead>
<tr>
<th>Freq</th>
<th>Synset</th>
<th>Gloss</th>
</tr>
</thead>
<tbody>
<tr>
<td>338</td>
<td>plant(^1), works, industrial plant</td>
<td>buildings for carrying on industrial labor</td>
</tr>
<tr>
<td>207</td>
<td>plant(^2), flora, plant life</td>
<td>a living organism lacking the power of locomotion</td>
</tr>
<tr>
<td>2</td>
<td>plant(^3)</td>
<td>something planted secretly for discovery by another</td>
</tr>
<tr>
<td>0</td>
<td>plant(^4)</td>
<td>an actor situated in the audience whose acting is rehearsed but seems spontaneous to the audience</td>
</tr>
</tbody>
</table>

- **Upper bound**:
  - Fine-grained WordNet sense: 75-80% human agreement
  - Coarser-grained inventories: 90% human agreement possible

- What does this mean?
WSD Approaches

- Depending on use of manually created knowledge sources
  - Knowledge-lean
  - Knowledge-rich
- Depending on use of labeled data
  - Supervised
  - Semi- or minimally supervised
  - Unsupervised
Lesk’s Algorithm

- Intuition: note word overlap between context and dictionary entries
  - Unsupervised, but knowledge rich

The **bank** can guarantee **deposits** will eventually cover future tuition costs because it invests in adjustable-rate **mortgage** securities.

<table>
<thead>
<tr>
<th>WordNet</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>bank</strong>¹</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>bank</strong>²</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
Lesk’s Algorithm

- Simplest implementation:
  - Count overlapping content words between glosses and context

- Lots of variants:
  - Include the examples in dictionary definitions
  - Include hypernyms and hyponyms
  - Give more weight to larger overlaps (e.g., bigrams)
  - Give extra weight to infrequent words (e.g., $idf$ weighting)
  - ...

- Works reasonably well!
Supervised WSD: NLP meets ML

- WSD as a supervised classification task
  - Train a separate classifier for each word
- Three components of a machine learning problem:
  - Training data (corpora)
  - Representations (features)
  - Learning method (algorithm, model)
**Supervised Classification**

- **Training Data**
  - \( \text{label}_1 \)
  - \( \text{label}_2 \)
  - \( \text{label}_3 \)
  - \( \text{label}_4 \)

- **Representation Function**

- **Classifier**

  - \( \text{label}_1 \)?
  - \( \text{label}_2 \)?
  - \( \text{label}_3 \)?
  - \( \text{label}_4 \)?

- **Unlabeled Document**
Three Laws of Machine Learning

- Thou shalt not mingle training data with test data
- Thou shalt not mingle training data with test data
- Thou shalt not mingle training data with test data

But what do you do if you need more test data?
Features

- Possible features
  - POS and surface form of the word itself
  - Surrounding words and POS tag
  - Positional information of surrounding words and POS tags
  - Same as above, but with $n$-grams
  - Grammatical information
  - ...

- Richness of the features?
  - Richer features = ML algorithm does less of the work
  - More impoverished features = ML algorithm does more of the work
Classifiers

Once we cast the WSD problem as supervised classification, many learning techniques are possible:

- Naïve Bayes (the thing to try first)
- Decision lists
- Decision trees
- MaxEnt
- Support vector machines
- Nearest neighbor methods
- ...
Classifiers Tradeoffs

- Which classifier should I use?
- It depends:
  - Number of features
  - Types of features
  - Number of possible values for a feature
  - Noise
  - ...
- General advice:
  - Start with Naïve Bayes
  - Use decision trees/lists if you want to understand what the classifier is doing
  - SVMs often give state of the art performance
  - MaxEnt methods also work well
Naïve Bayes

- Pick the sense that is most probable given the context
  - Context represented by feature vector
    \[ \hat{s} = \arg \max_{s \in S} P(s | \vec{f}) \]
  - By Bayes’ Theorem:
    \[ \hat{s} = \arg \max_{s \in S} \frac{P(\vec{f} | s)P(s)}{P(\vec{f})} \]  
    We can ignore this term… why?

- Problem: data sparsity!
The “Naïve” Part

- Feature vectors are too sparse to estimate directly:

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

- So… assume features are conditionally independent given the word sense
- This is naïve because?

- Putting everything together:

\[ \hat{s} = \arg \max_{s \in S} P(s) \prod_{s \in S} P(f_j \mid s) \]
Naïve Bayes: Training

- How do we estimate the probability distributions?

\[
\hat{s} = \arg \max_{s \in S} P(s) \prod_{j=1}^{n} P(f_j \mid s)
\]

- Maximum-Likelihood Estimates (MLE):

\[
P(s_i) = \frac{\text{count}(s_i, w_j)}{\text{count}(w_j)}
\]

\[
P(f_j \mid s) = \frac{\text{count}(f_j, s)}{\text{count}(s)}
\]

- What else do we need to do?

Well, how well does it work? (later...)
Decision List

- Ordered list of tests (equivalent to “case” statement):
- Example decision list, discriminating between bass (fish) and bass (music):

<table>
<thead>
<tr>
<th>Context</th>
<th>Sense</th>
</tr>
</thead>
<tbody>
<tr>
<td>fish in ±k words</td>
<td>FISH</td>
</tr>
<tr>
<td>striped bass</td>
<td>FISH</td>
</tr>
<tr>
<td>guitar in ±k words</td>
<td>MUSIC</td>
</tr>
<tr>
<td>bass player</td>
<td>MUSIC</td>
</tr>
<tr>
<td>piano in ±k words</td>
<td>MUSIC</td>
</tr>
<tr>
<td>sea bass</td>
<td>FISH</td>
</tr>
<tr>
<td>play bass</td>
<td>MUSIC</td>
</tr>
<tr>
<td>river in ±k words</td>
<td>FISH</td>
</tr>
<tr>
<td>on bass</td>
<td>MUSIC</td>
</tr>
<tr>
<td>bass are</td>
<td>FISH</td>
</tr>
</tbody>
</table>
Building Decision Lists

- Simple algorithm:
  - Compute how discriminative each feature is:
    \[ \log \left( \frac{P(S_1 | f_i)}{P(S_2 | f_i)} \right) \]
  - Create ordered list of tests from these values

- Limitation?

- How do you build \( n \)-way classifiers from binary classifiers?
  - One vs. rest (sequential vs. parallel)
  - Another learning problem

Well, how well does it work? (later...)
Decision Trees

- Instead of a list, imagine a tree...

<table>
<thead>
<tr>
<th>Context</th>
<th>Sense</th>
</tr>
</thead>
<tbody>
<tr>
<td>fish in ±k words</td>
<td>FISH</td>
</tr>
<tr>
<td>striped bass</td>
<td>FISH</td>
</tr>
<tr>
<td>guitar in ±k words</td>
<td>MUSIC</td>
</tr>
<tr>
<td>bass player</td>
<td>MUSIC</td>
</tr>
<tr>
<td>piano in ±k words</td>
<td>MUSIC</td>
</tr>
<tr>
<td>sea bass</td>
<td>FISH</td>
</tr>
<tr>
<td>play bass</td>
<td>MUSIC</td>
</tr>
<tr>
<td>river in ±k words</td>
<td>FISH</td>
</tr>
<tr>
<td>on bass</td>
<td>MUSIC</td>
</tr>
<tr>
<td>bass are</td>
<td>FISH</td>
</tr>
</tbody>
</table>

- Fish in ±k words
  - Yes
  - No

- Striped bass
  - Yes
  - No

- Guitar in ±k words
  - Yes
  - No

- Music
  - ...
Using Decision Trees

- Given an instance (= list of feature values)
  - Start at the root
  - At each interior node, check feature value
  - Follow corresponding branch based on the test
  - When a leaf node is reached, return its category

Decision tree material drawn from slides by Ed Loper
Building Decision Trees

- Basic idea: build tree top down, recursively partitioning the training data at each step
  - At each node, try to split the training data on a feature (could be binary or otherwise)

- What features should we split on?
  - Small decision tree desired
  - Pick the feature that gives the most information about the category

- Example: 20 questions
  - I’m thinking of a number from 1 to 1,000
  - You can ask any yes no question
  - What question would you ask?
Evaluating Splits via Entropy

- Entropy of a set of events $E$:
  
  $$H(E) = -\sum_{c \in C} P(c) \log_2 P(c)$$

  - Where $P(c)$ is the probability that an event in $E$ has category $c$

- How much information does a feature give us about the category (sense)?
  
  - $H(E)$ = entropy of event set $E$
  - $H(E|f)$ = expected entropy of event set $E$ once we know the value of feature $f$
  - Information Gain: $G(E, f) = H(E) - H(E|f)$ = amount of new information provided by feature $f$

- Split on feature that maximizes information gain

Well, how well does it work? (later...
WSD Accuracy

- Generally:
  - Naïve Bayes provides a reasonable baseline: ~70%
  - Decision lists and decision trees slightly lower
  - State of the art is slightly higher

- However:
  - Accuracy depends on actual word, sense inventory, amount of training data, number of features, etc.
  - Remember caveat about baseline and upper bound
Minimally Supervised WSD

- But annotations are expensive!

- “Bootstrapping” or co-training (Yarowsky 1995)
  - Start with (small) seed, learn decision list
  - Use decision list to label rest of corpus
  - Retain “confident” labels, treat as annotated data to learn new decision list
  - Repeat…

- Heuristics (derived from observation):
  - One sense per discourse
  - One sense per collocation
One Sense per Discourse

- A word tends to preserve its meaning across all its occurrences in a given discourse.

Evaluation:
- 8 words with two-way ambiguity, e.g. plant, crane, etc.
- 98% of the two-word occurrences in the same discourse carry the same meaning.

The grain of salt: accuracy depends on granularity:
- Performance of “one sense per discourse” measured on SemCor is approximately 70%.
One Sense per Collocation

- A word tends to preserve its meaning when used in the same collocation
  - Strong for adjacent collocations
  - Weaker as the distance between words increases

- Evaluation:
  - 97% precision on words with two-way ambiguity

- Again, accuracy depends on granularity:
  - 70% precision on SemCor words
Yarowsky’s Method: Example

- Disambiguating plant (industrial sense) vs. plant (living thing sense)

- Think of seed features for each sense
  - Industrial sense: co-occurring with “manufacturing”
  - Living thing sense: co-occurring with “life”

- Use “one sense per collocation” to build initial decision list classifier

- Treat results as annotated data, train new decision list classifier, iterate...
used to strain microscopic plant life from the zonal distribution of plant life.

Close-up studies of plant life and natural too rapid growth of aquatic plant life in water the proliferation of plant and animal life establishment phase of the plant virus life cycle that divide life into plant and animal kingdom many dangers to plant and animal life mammals. Animal and plant life are delicately automated manufacturing plant in Fremont vast manufacturing plant and distribution chemical manufacturing plant, producing viscose keep a manufacturing plant profitable without computer manufacturing plant and adjacent discovered at a St. Louis plant manufacturing copper manufacturing plant found that they copper wire manufacturing plant, for example cement manufacturing plant in Alpena

vinyl chloride monomer plant, which is molecules found in plant and animal tissue Nissan car and truck plant in Japan is and Golgi apparatus of plant and animal cells union responses to plant closures.
cell types found in the plant kingdom are company said the plant is still operating

Although thousands of plant and animal species animal rather than plant tissues can be
Intermediate state
Final state
Yarowsky’s Method: Stopping

- Stop when:
  - Error on training data is less than a threshold
  - No more training data is covered

- Use final decision list for WSD
Yarowsky’s Method: Discussion

- Advantages:
  - Accuracy is about as good as a supervised algorithm
  - Bootstrapping: far less manual effort

- Disadvantages:
  - Seeds may be tricky to construct
  - Works only for coarse-grained sense distinctions
  - Snowballing error with co-training

- Recent extension: now apply this to the web!
WSD with Parallel Text

- But annotations are expensive!
- What’s the “proper” sense inventory?
  - How fine or coarse grained?
  - Application specific?
- Observation: multiple senses translate to different words in other languages!
  - A “bill” in English may be a “pico” (bird jaw) in or a “cuenta” (invoice) in Spanish
  - Use the foreign language as the sense inventory!
  - Added bonus: annotations for free! (Byproduct of word-alignment process in machine translation)
Beyond Lexical Semantics
Syntax-Semantics Pipeline

Inputs → Syntactic Analysis → Syntactic Structures → Semantic Analysis → Meaning Representations

Example: FOPL
Semantic Attachments

- Basic idea:
  - Associate $\lambda$-expressions with lexical items
  - At branching node, apply semantics of one child to another (based on syntactic rule)

- Refresher in $\lambda$-calculus...
Augmenting Syntactic Rules

<table>
<thead>
<tr>
<th>Grammar Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S \rightarrow NP\ VP$</td>
</tr>
<tr>
<td>$NP \rightarrow Det\ Nominal$</td>
</tr>
<tr>
<td>$NP \rightarrow ProperNoun$</td>
</tr>
<tr>
<td>$Nominal \rightarrow Noun$</td>
</tr>
<tr>
<td>$VP \rightarrow Verb$</td>
</tr>
<tr>
<td>$VP \rightarrow Verb\ NP$</td>
</tr>
<tr>
<td>$Det \rightarrow every$</td>
</tr>
<tr>
<td>$Det \rightarrow a$</td>
</tr>
<tr>
<td>$Noun \rightarrow restaurant$</td>
</tr>
<tr>
<td>$ProperNoun \rightarrow Matthew$</td>
</tr>
<tr>
<td>$ProperNoun \rightarrow Franco$</td>
</tr>
<tr>
<td>$ProperNoun \rightarrow Frasca$</td>
</tr>
<tr>
<td>$Verb \rightarrow closed$</td>
</tr>
<tr>
<td>$Verb \rightarrow opened$</td>
</tr>
</tbody>
</table>
Semantic Analysis: Example

\[ \forall x \, \text{Restaurant}(x) \Rightarrow \exists e \, \text{Closed}(e) \land \text{ClosedThing}(e, x) \]

\[ \lambda Q. \forall x \, \text{Restaurant}(x) \Rightarrow Q(x) \]

\[ \lambda x. \exists e \, \text{Closed}(e) \land \text{ClosedThing}(e, x) \]

\[ \lambda P. \lambda Q. \forall x P(x) \Rightarrow Q(x) \]

\[ \lambda x. \text{Restaurant}(x) \]

NP \rightarrow \text{Det Nominal} \quad \{ \text{Det.sem}(\text{Nominal.sem}) \}

\[ \lambda P. \lambda Q. \forall x P(x) \Rightarrow Q(x) (\lambda x. \text{Restaurant}(x)) \]

\[ \lambda Q. \forall x \lambda x. \text{Restaurant}(x)(x) \Rightarrow Q(x) \]

\[ \lambda Q. \forall x \, \text{Restaurant}(x) \Rightarrow Q(x) \]
Complexities

- Oh, there are many…
- Classic problem: quantifier scoping
  - Every restaurant has a menu
- Issues with this style of semantic analysis?
Semantics in NLP Today

- Can be characterized as “shallow semantics”
- Verbs denote events
  - Represent as “frames”
- Nouns (in general) participate in events
  - Types of event participants = “slots” or “roles”
  - Event participants themselves = “slot fillers”
  - Depending on the linguistic theory, roles may have special names: agent, theme, etc.
- Semantic analysis: semantic role labeling
  - Automatically identify the event type (i.e., frame)
  - Automatically identify event participants and the role that each plays (i.e., label the semantic role)
What works in NLP?

- POS-annotated corpora
- Tree-annotated corpora: Penn Treebank
- Role-annotated corpora: Proposition Bank (PropBank)
PropBank: Two Examples

- **agree.01**
  - Arg0: Agreer
  - Arg1: Proposition
  - Arg2: Other entity agreeing
  - Example: \([\text{Arg0 John}]\) agrees \([\text{Arg2 with Mary}]\) \([\text{Arg1 on everything}]\)

- **fall.01**
  - Arg1: Logical subject, patient, thing falling
  - Arg2: Extent, amount fallen
  - Arg3: Start point
  - Arg4: End point
  - Example: \([\text{Arg1 Sales}]\) fell \([\text{Arg4 to $251.2 million}]\) \([\text{Arg3 from $278.7 million}]\)
How do we do it?

- Short answer: supervised machine learning
- One approach: classification of each tree constituent
  - Features can be words, phrase type, linear position, tree position, etc.
  - Apply standard machine learning algorithms
Recap of Today’s Topics

- Word sense disambiguation
- Beyond lexical semantics
  - Semantic attachments to syntax
  - Shallow semantics: PropBank
The Complete Picture

- Speech Recognition
- Morphological Analysis
- Parsing
- Semantic Analysis
- Reasoning, Planning
- Speech Synthesis
- Morphological Realization
- Syntactic Realization
- Utterance Planning
- Phonology
- Morphology
- Syntax
- Semantics
- Reasoning

Phonology → Morphology → Syntax → Semantics → Reasoning

Speech Recognition → Morphological Analysis → Parsing → Semantic Analysis → Reasoning, Planning

Speech Synthesis ← Morphological Realization ← Syntactic Realization ← Utterance Planning ← Reasoning
The Home Stretch

- Next week: MapReduce and large-data processing
- No classes Thanksgiving week!
- December: two guest lectures by Ken Church