Computational Semantics

CS224N 2005
Christopher Manning

(Borrowed some slides from Mary Daly and Jim Martin)

Why study computational semantics?

- Because everyone has been wanting me to talk about this all course!
- Obvious high-level applications
  - Summarization
  - Translation
  - Question answering
  - Information access
  - Talking to your pet robot
  - Speech user interfaces

The next generation of intelligent applications need deeper semantics than we have seen so far
- Often you must understand well to be able to act

Shallow vs. deep semantics

- We can do more than one might have thought without deep linguistic analysis
- But we can’t do everything we would like:
  - Not all tasks can ignore higher structure
  - Unsuitable if new text must be generated
  - Unsuitable if machine must act rather than relying on user to interpret material written by the author of the document
  - You get what you pay for:
    - Cheap, fast, low-level techniques are appropriate in domains where speed and volume are more important than accuracy
    - More computationally-expensive, higher-level techniques are appropriate when high-quality results are required

MSN Search: Which is the largest African country?

MSN Search: Which countries does the Danube flow through?

MSN Search: What are the capitals of the countries bordering the Baltic?
Precise semantics. An early example: Chat-80

- Developed between 1979 and 1982 by Fernando Pereira and David Warren; became Pereira's dissertation
- Proof-of-concept natural language interface to database system
- Used in projects: e.g. Shoptalk (Colen et al. 1989), a natural language and graphical interface for decision support in manufacturing
- Even used in an AppliedNLP-2000 conference paper! [Asking about train routes and schedules]
- Available in /usr/local/src directory
  - Need stellus prog: /usr/sweet/bin/stellus

The CHAT-80 Database

Facts about countries:
- % country(Country, Region, Latitude, Longitude,
  % Area (sqmiles), Population, Capital, Currency)
country(Andorra, southerneurope, 42.1, 1.779,
25000, andorra-la-vila, franc, peseta).
country(Angola, southernafrica, -12.1, 18.481351,
5810000, luanda, $).
country(Argentina, southamerica, -35.66, 1072067,
23920000, buenosaires, peso).
capital(C, Cap) :- country(C, _, _, _, Cap, _).

Chat-80 trace (illegibly small)

Question: What is the capital of Australia?

Parse: 3+5*6

<table>
<thead>
<tr>
<th>np</th>
<th>head</th>
<th>det(head)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>capital</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pp</td>
</tr>
<tr>
<td></td>
<td></td>
<td>prep</td>
</tr>
<tr>
<td></td>
<td></td>
<td>np</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3+5*6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>whili</td>
</tr>
<tr>
<td></td>
<td></td>
<td>verb(take, pres -frn, s, pres)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>dir</td>
</tr>
<tr>
<td></td>
<td></td>
<td>np</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3+5*6</td>
</tr>
</tbody>
</table>

Semantics: 0.0 sec.
answer[fill] = capitalAustralia, II

Programming Language Interpreter

- What is meaning of 3+5*6?
- First parse it into 3+(5*6)
- Now give a meaning to each node in the tree (bottom-up)
More complex meanings

- How about 3+5\times x?
- Don’t know x at compile time
- “Meaning” at a node is a piece of code, not a number
- Form is “rule-to-rule” translation
  - We provide a way to form the semantics of each parent in terms of the semantics of the children

What Counts as Understanding?

- A somewhat difficult philosophical question
- We understand if we can respond appropriately
  - “Throw a axe at daw”
- We understand statement if we can determine its truth
- We understand statement if we can use it to answer questions (similar to above – require reasoning)
  - Easy: John ate pizza. What was eaten by John?
- Understanding is the ability to translate
  - English to Chinese (require deep understanding)
  - English to logic (deeply the definition method)
- All humans are mortal = ∀x[Human(x) ⊃ Mortal(x)]
- We assume we have logic-manipulating rules to tell us how to act, draw conclusions, answer questions ...

Lecture Plan

- Today:
  - Look at some sentences and phrases
  - What would be reasonable logical representations for them?
  - Get some idea of compositional semantics
  - A couple of other semantic approaches (quickly)
  - Semantic grammars
  - Semantic role labeling

- Next Wednesday:
  - How can we build these representations?
  - Another course (somewhere in AI, hopefully):
    - How can we reason with these representations?
  - Last lectures: discourse, dialog, and QA

Logic: Some Preliminaries

Three major kinds of objects

1. Booklets
   - Roughly, the semantic values of sentences
2. Individuals/Entities (Ind)
   - Values of NPs, i.e., objects
   - Maybe also other types of entities, like times
3. Functions of various types
   - A function returning a boolean is called a “predicate” – e.g., Has(s, p)
   - A predicate defines a set of individuals that satisfy it
   - More complex functions return other functions
   - Some functions take other functions as arguments
   - (Higher order functions)

Logic: Lambda Terms

- Lambda terms:
  - A way of writing “anonymous functions”
  - No function header or function name
  - But defines the key thing: behavior of the function
  - Just as we can talk about 3 without naming it “x”
  - Let square = λp.p^2
  - Equivalent to int square(p) return p^2
  - But we can talk about λp.p^2 without naming it
  - Format of a lambda term: λ variable . expression
**Logic: Multiple Arguments**
- All lambda terms have one argument
- But we can fake multiple arguments...
- Suppose we want to write $\text{times}(5, 6)$
- Remember: square can be written as $\lambda x.\text{square}(x)$
- Similarly, times is equivalent to $\lambda x.\text{times}(x, y)$

Claim that $\text{times}(5, 6)$ means same as $\text{times}(5 \cdot 6)$
- $\text{times}(5) = (\lambda x.\text{times}(x, y))(5) = \text{times}(5 \cdot 5)$
- If this function weren't anonymous, what would we call it?
- $\text{times}(5)(6) = \text{times}(5 \cdot 6)$
- Referred to as “currying”

**Logic: Interesting Constants**
- We have “constants” that name some of the entities and functions (e.g., $\text{times}$)
  - George W. Bush - an entity
  - red - a predicate on entities
  - holds of just the red entities: $\text{red}(x)$ is true if $x$ is red
  - loves - a predicate on 2 entities
  - loves(“George W. Bush”, “Laura Bush”)
  - Question: What does loves(“Laura Bush”) denote?
  - Constants used to define meanings of words
  - Meanings of phrases will be built from the constants

**Logic: Interesting Constants**
- Generalized Quantifiers
  - most - a predicate on 2 predicates on entities
    - $\text{most}(\text{pig}, \text{big}) = \text{most pigs are big}$
    - $\text{equilemtly, most}(\text{pig}(x), \text{big}(x))$ returns true if most of the things satisfying the first predicate also satisfy the second predicate
  - Similarly for other quantifiers
    - $\text{all}(\text{pig}, \text{big})$ $\equiv \text{equiv to } \forall x \text{pig}(x) \Rightarrow \text{big}(x)$
    - $\text{ex}(\text{pig}, \text{big})$ $\equiv \text{equiv to } \exists x \text{pig}(x) \text{ AND big}(x)$
    - can even make complex quantifiers from English phrases:
      - “between 12 and 75”, “a majority of”, “all but the smallest 2”

**Quantifier Order**
- Groucho Marx celebrates quantifier order ambiguity:
  - “In this country a woman gives birth every 15 min. Our job is to find that woman and stop her.”
    - $\exists \text{woman}(\forall 15 \text{min gives birth during woman, 15min})$
    - $\forall 15 \text{min} (\exists \text{woman gives birth during 15min, woman})$
    - Surprisingly, both are possible in natural language!
    - Which is the joke meaning?
      - “where it’s always the same woman”

**Compositional Semantics**
- We’ve discussed what semantic representations should look like.
- But how do we get them from sentences??
  - First - parse to get a syntax tree.
  - Second - look up the semantics for each word.
  - Third - build the semantics for each constituent
    - Work from the bottom up
      - The syntax tree is a “recipe” for how to do it
    - Principle of Compositionality
      - The meaning of a whole is derived from the meanings of the parts, via composition rules

**A simple grammar of English**
- In Definite Clause Grammar, DCG (form as in Prolog)

```
sentence -> noun_phrase, verb_phrase.
noun_phrase -> proper_noun.
noun_phrase -> determiner, noun.
verb_phrase -> verb, noun_phrase.

Proper_noun -> [John] verb -> [ate]
Proper_noun -> [Mary] verb -> [kissed]
Determiner -> [the] noun -> [cake]
Determiner -> [a] noun -> [lion]
```
Extending the grammar to check number agreement between subjects and verbs

\[
S \rightarrow \text{NP}(\text{Num}), \text{VP}(\text{Num})..
\]

\[
\text{NP}(\text{Num}) \rightarrow \text{Proper noun}(\text{Num}),\text{det}(\text{Num}),\text{noun}(\text{Num}).
\]

\[
\text{VP}(\text{Num}) \rightarrow \text{verb}(\text{Num}),\text{noun phrase}(\text{Num}).
\]

\[
\text{Proper noun}(\text{Num}) \rightarrow \text{[Mary]},\text{noun}(\text{Num}) \rightarrow \text{[John]},\text{det}(\text{Num}) \rightarrow \text{[the]},\text{det}(\text{Num}) \rightarrow \text{[a]}.\text{verb}(\text{Num}) \rightarrow \text{[eats]},\text{verb}(\text{Num}) \rightarrow \text{[eat]}.\]

A simple DCG grammar with semantics

\[
\text{sentence}(\text{SMeaning}) \rightarrow \text{noun phrase}(\text{NPMeaning}),\text{verb phrase}(\text{VPMeaning}),\{\text{combine}(\text{NPMeaning},\text{VPMeaning},\text{SMeaning})\}.
\]

\[
\text{verb phrase}(\text{VPMeaning}) \rightarrow \text{verb}(\text{Vmeaning}),\text{noun phrase}(\text{NPMeaning}),\{\text{combine}(\text{NPMeaning},\text{VMeaning},\text{VPMeaning})\}.
\]

\[
\text{noun phrase}(\text{NPMeaning}) \rightarrow \text{name}(\text{NPMeaning}).\text{name}(\text{ohn}) \rightarrow \text{[ohn]},\text{verb}(\text{ohy},\text{loves}(\text{x},\text{y})) \rightarrow \text{[loves]}
\]

\[
\text{name}(\text{Mary}) \rightarrow \text{[mary]},\text{verb}(\text{ohy},\text{loves}(\text{x},\text{y})) \rightarrow \text{[loves]}
\]

Combine(\text{X, Y, Z}) \rightarrow \text{apply}(\text{Y, X, Z})

Parse tree with associated semantics

![Parse tree](image)

Augmented CFG Rules

- We can also accomplish this just by attaching semantic formation rules to our syntactic CFG rules

\[
A \rightarrow \alpha_1 ... \alpha_n \{ f(\alpha_1, \text{sem}, ..., \alpha_n, \text{sem}) \}
\]

- This should be read as the semantics we attach to A can be computed from some function applied to the semantics of A's parts.
- The functions/operations permitted in the semantic rules are restricted, falling into two classes
  - Pass the semantics of a daughter up unchanged to the mother
  - Apply (as a function) the semantics of one of the daughters of a node to the semantics of the other daughters

How do things get more complex? (The former GRE analytic section)

- Six sculptures C, D, E, F, G, H are to be exhibited in rooms 1, 2, and 3 of an art gallery.
  - Sculpture C and E may not be exhibited in the same room.
  - Sculptures D and G must be exhibited in the same room.
  - If sculptures E and F are exhibited in the same room, no other sculpture may be exhibited in that room.
  - At least one sculpture must be exhibited in each room, and no more than three sculptures may be exhibited in any room.
  - If sculpture D is exhibited in room 3 and sculptures E and F are exhibited in room 1, which of the following may be true?
    - A. Sculpture C is exhibited in room 1.
    - B. Sculpture H is exhibited in room 1.
    - C. Sculpture G is exhibited in room 2.
    - D. Sculptures C and H are exhibited in the same room.
    - E. Sculptures G and F are exhibited in the same room.

Scope Needs to be Resolved!

- At least one sculpture must be exhibited in each room.
- The same sculpture in each room?
- No more than three sculptures may be exhibited in any room.
- Reading 1: Every room, there are no more than three sculptures exhibited in it.
- Reading 2: Only three or less are exhibited (the rest not shown).
- Reading 3: Only certain sets of three or less may be exhibited in any room (for the other there are restrictions in a known room).
- Some readings will be ruled out by being uninformative or by contradicting other statements.
- Otherwise, we must be content with distributions over scope-resolved semantic forms.
Semantic Grammars

- A problem with traditional linguistic grammars is that they don’t necessarily reflect the semantics in a straightforward way.
- You can deal with this by...
  - Fighting with the grammar
  - Complex lambda and complex terms, etc.
  - Rewriting the grammar to reflect the semantics
  - And in the process give up on some syntactic niceties
  - Known as “semantic grammar”
  - Simple idea, damn name

Semantic Grammar

- The term semantic grammar refers to the motivation for the grammar rules
  - The technology (skin CCG rules with a set of terminals) is the same as we’ve been using
  - The good thing about them is that you get exactly the semantic rules you need
  - The bad thing is that you need to develop a new grammar for each new domain
  - Typically used in conversational agents in constrained domains
    - Limited vocabulary
    - Limited grammatical complexity
    - Syntactic parsing can often produce all that’s needed for semantic interpretation even in the face of “ungrammatical” input - write fragment rules

Lifer Semantic Grammars

- Example domain — accessing DB of US Navy ships
  - $ \rightarrow \text{present the <attribute> of <ship>}$
  - $\text{present} \rightarrow \text{what} \mid \text{if you tell me}$
  - $\text{attribute} \rightarrow \text{length} \mid \text{beam} \mid \text{class}$
  - $\text{ship} \rightarrow \text{the <shipname>}$
  - $\text{shipname} \rightarrow \text{kennedy} \mid \text{enterprise}$
  - $\text{class} \rightarrow \text{kitty hawk} \mid \text{bluevale}$
- Example inputs recognized by above grammar:
  - *Can you tell me the class of the Enterprise?*
  - *What is the length of fifty kennedy class ships?*
  - Many categories are not “true” syntactic categories
  - Words are recognized by their context rather than category (e.g., class)
  - Recognition is strongly directed
  - Strong direction useful for error detection and correction
    - e.g., “present, I want the entire, and please, very, very, very specific language”
  - Error in database or in transcription of database causes errors

Semantic Grammars Summary

- Advantages:
  - Efficient recognition of limited domain input
  - Absence of overall grammar allows pattern-matching possibilities for idioms, etc.
  - No separate interpretation phase
  - Strength of top-down constraints allows powerfulipsis mechanisms
  - What is the length of the Kennedy? The Kittyhawk?
- Disadvantages:
  - Different grammar required for each new domain
  - Lack of overall syntax can lead to “spotty” grammar coverage
  - E.g., missing possessive *“attribute of ship to subject’s”
  - “attribute” doesn’t imply finding in “rank of officer”
  - Difficult to develop grammar past a certain size
  - Suffers from fragility

Semantic Role Labeling

- Semantic roles appear in different positions:
  - $\text{subject} \rightarrow \text{The company to} \ldots \text{offer} \ldots \text{a 15% to 20% stake (to the public)}$
  - $\text{object} \rightarrow \text{Sotheby’s} \ldots \text{offered} \ldots \text{the Darrance heir’s}
    \text{more than black guarantees}$
  - $\text{aux of c} \rightarrow \text{an amendment} \ldots \text{offered} \ldots \text{by Rep. Peter DeFazio}$
  - $\text{aux of e} \rightarrow \text{Subcontractors will} \ldots \text{not be offered} \ldots \text{a settlement}$
- Approach is to build classifiers (e.g., maxent models) that work over parse tree features to determine semantic roles of verb arguments
  - Gelade and Jurafsky (2002); Toutanova et al. (2005)

Semantic Role Labeling: Applications

- Question Answering
  - Q: When was Napoleon defeated
    - Look for:
      - [Napoleon] [defeat-synset] [NEG] [defeat-arg1] [ANS]
- Machine Translation
  - English \rightarrow French
  - $\text{subject} \rightarrow \text{The little boy}$
  - $\text{object} \rightarrow \text{a boy who kicked}$
  - $\text{aux of c} \rightarrow \text{the red ball}$
  - $\text{aux of e} \rightarrow \text{knows}$
- Document Summarization
  - Predicates and Heads of Roles summarize content
- Information Extraction