Natural Language Processing

Lecture 19: Compositional Semantics and Semantic Parsing
Meaning

• The last three lectures dealt with some issues around word meaning, primarily:
  • Relationships between words and word similarity
  • Word sense disambiguation
  • Predicates and their arguments
• Today: meaning of NL (English) sentences
Key Challenge of Meaning

• We actually say very little - much more is left unsaid, because it’s assumed to be widely known.

• Examples:
  • Grading assignments
  • Restaurant menus
  • Learning to use a new piece of software
Meaning Representation Languages

• Symbolic representation that does two jobs:
  • Conveys the meaning of a sentence
  • Represents (some part of) the world

• Today we’ll use first-order logic.
A MRL Should Be Able To ...

- Verify a query against a knowledge base
  - Do CMU students follow politics?
- Eliminate ambiguity
  - CMU students enjoy visiting Senators.
- Cope with vagueness
  - Sally heard the news.
- Cope with many ways of expressing the same meaning (canonical forms)
  - The candidate evaded the question.
  - The question was evaded by the candidate.
- Draw conclusions based on the knowledge base
  - Who could become the 45th president?
- Represent all of the meanings we care about
Model-Theoretic Semantics

- Model: a simplified representation of the world: objects, properties, relations (domain).
- Non-logical vocabulary
  - Each element denotes a well-defined part of the model
  - Such a mapping is called an interpretation
A Model

- **Domain**: Noah, Karen, Rebecca, Frederick, Green Mango, Casbah, Udipi, Thai, Mediterranean, Indian
- **Properties**: Green Mango and Udipi are crowded; Casbah is expensive
- **Relations**: Karen likes Green Mango, Frederick likes Casbah, everyone likes Udipi, Green Mango serves Thai, Casbah serves Mediterranean, and Udipi serves Indian
- n, k, r, f, g, c, u, t, m, i
- Crowded = \{g, u\}
- Expensive = \{c\}
- Likes = \{(k, g), (f, c), (n, u), (k, u), (r, u), (f, u)\}
- Serves = \{(g, t), (c, m), (u, i)\}
Some English

• Karen likes Green Mango and Frederick likes Casbah.
• Noah and Rebecca like the same restaurants.
• Noah likes expensive restaurants.
• Not everybody likes Green Mango.

• What we want is to be able to represent these statements in a way that lets us compare them to our model.

• Truth-conditional semantics: need operators and their meanings, given a particular model.
First-Order Logic

• **Terms** refer to elements of the domain: **constants**, **functions**, and **variables**
  - Noah, SpouseOf(Karen), X

• **Predicates** are used to refer to sets and relations
  - Serves(Casbah, Mediterranean)

• Logical connectives: $\land$ (and), $\lor$ (or), $\neg$ (not), $\Rightarrow$ (implies), ...

• Quantifiers ...
Quantifiers in FOL

• Two ways to use variables:
  • refer to one anonymous object from the domain (existential; \( \exists \); “there exists”)
  • refer to all objects in the domain (universal; \( \forall \); “for all”)

• a restaurant near CMU that serves Indian food
  \( \exists x \) Restaurant\( (x) \), Near\( (x, \text{CMU}) \), Serves\( (x, \text{Indian}) \)

• All expensive restaurants are far from campus
  \( \forall x \) Restaurant\( (x) \), Expensive\( (x) \) \( \Rightarrow \) \( \neg \)Near\( (x, \text{CMU}) \)
Extension: Lambda Notation

• A way of making anonymous functions.

• $\lambda x. \text{(some expression mentioning } x\text{)}$
  • Example: $\lambda x.\text{Near}(x, \text{CMU})$
  • Deeper example: $\lambda x.\lambda y.\text{Serves}(y, x)$

• Lambda reduction: substitute for the variable.
  • $(\lambda x.\text{Near}(x, \text{CMU}))(\text{Lulu’s Noodles})$
    becomes $\text{Near}(\text{Lulu’s Noodles, CMU})$
**Inference**

- Big idea: extend the knowledge base, or check some proposition against the knowledge base.

- **Forward chaining** with modus ponens:
  - given $\alpha$ and $\alpha \Rightarrow \beta$, we know $\beta$.

- **Backward chaining** takes a query $\beta$ and looks for propositions $\alpha$ and $\alpha \Rightarrow \beta$ that would prove $\beta$.
  - Not the same as backward reasoning (abduction).
  - Used by Prolog

- Both are sound, neither is complete.
Lots More To Say About MRLs!

- See chapter 17 for more about:
  - Representing events and states in FOL
  - Dealing with optional arguments (e.g., “eat”)
  - Representing time
  - Non-FOL approaches to meaning
First-Order Worlds, Then and Now

• Interest in this topic waned during the 1990s and 2000s.

• It’s come back, with the rise of semi-structured databases like Wikipedia.
  • Lay contributors to these databases may be helping us to solve the knowledge acquisition problem.

• Also, lots of research on using NLP, information extraction, and machine learning to grow and improve knowledge bases from free text data.
  • “Read the Web” project here at CMU.
Connecting Syntax and Semantics
Semantic Analysis

• Goal: transform a NL statement into MRL (today, FOL).
• We’re assuming a very literal, inference-free version of meaning!
• Sometimes called “semantic parsing.”
Compositionality

- The meaning of an NL phrase is determined by combining the meaning of its sub-parts.
- There are obvious exceptions (“hot dog,” “straw man,” “New York,” etc.).
- Big idea: start with parse tree, build semantics on top using FOL with $\lambda$-expressions.
An Example

• Noah likes expensive restaurants.

• ∀x Restaurant(x), Expensive(x) ⇒ Likes(Noah, x)
An Example

- Noah likes expensive restaurants.

\[ \forall x \text{ Restaurant}(x), \text{ Expensive}(x) \Rightarrow \text{ Likes}(\text{Noah}, x) \]
An Example

• Noah likes expensive restaurants.

• $\forall x \text{ Restaurant}(x), \text{Expensive}(x) \Rightarrow \text{Likes}(\text{Noah}, x)$
An Example

• Noah likes expensive restaurants.

• $\forall x \text{ Restaurant}(x), \text{Expensive}(x) \Rightarrow \text{Likes}(\text{Noah}, x)$
An Example

• Noah likes expensive restaurants.

• $\forall x \text{ Restaurant}(x), \text{Expensive}(x) \Rightarrow \text{Likes}(\text{Noah, x})$
An Example

- Noah likes expensive restaurants.
- $\forall x \, \text{Restaurant}(x), \text{Expensive}(x) \Rightarrow \text{Likes}(\text{Noah}, x)$
An Example

• Noah likes expensive restaurants.

• $\forall x$ Restaurant($x$), Expensive($x$) $\Rightarrow$ Likes(Noah, $x$)
An Example

• Noah likes expensive restaurants.

• $\forall x \text{ Restaurant}(x), \text{ Expensive}(x) \Rightarrow \text{ Likes}(\text{Noah}, x)$
An Example

- Noah likes expensive restaurants.
- $\forall x \text{ Restaurant}(x), \text{ Expensive}(x) \Rightarrow \text{ Likes}(\text{Noah}, x)$

$S \quad \forall x \text{ Expensive}(x) \land \text{ Restaurant}(x) \Rightarrow \text{ Likes}(\text{Noah}, x)$

$VP \quad \lambda y.\forall x \text{ Expensive}(x) \land \text{ Restaurant}(x) \Rightarrow \text{ Likes}(y, x)$

$NP \quad \lambda f.\lambda y.\forall x f(x) \Rightarrow \text{ Likes}(y, x)$

$NP \quad \lambda x. \text{ Expensive}(x) \land \text{ Restaurant}(x)$

$NNP \quad \text{Noah}$

$VBZ \quad \text{likes}$

$JJ \quad \text{expensive}$

$NNS \quad \text{restaurants}$

- Noah likes expensive restaurants.
- $\forall x \text{ Restaurant}(x), \text{ Expensive}(x) \Rightarrow \text{ Likes}(\text{Noah}, x)$
Alternative (Following SLP)

- Noah likes expensive restaurants.
- $\forall x \text{ Restaurant}(x), \text{ Expensive}(x) \rightarrow \text{ Likes}(\text{Noah}, x)$

$S \rightarrow \text{ NP VP } \{ \text{ NP.sem(VP.sem)} \}$
Quantifier Scope Ambiguity

- Every man loves a woman.

\[
\forall u \text{ Man}(u) \Rightarrow \exists x \text{ Woman}(x) \land \text{ Loves}(u, x)
\]

\[
\exists x \text{ Woman}(x) \land \forall u \text{ Man}(u) \Rightarrow \text{ Loves}(u, x)
\]
This Isn’t Quite Right!

• “Every man loves a woman” really is ambiguous.
  • A seat was available for every customer
  • A toll free number was available for every customer
  • A secretary phoned up each director
  • A letter was sent to each customer
• This gives only one of the two meanings.
• One approach is to delay the quantifier processing until the end, then permit any ordering.
Matching Syntax and Semantics

• Combinatorial Categorial Grammar (CCG)

• Five grammar rules (only)
  • Forward application \( A/B + B = A \)
  • Backward application: \( B + A\backslash B = A \)
  • Composition: \( A/B + B/C = A/C \)
  • Conjunction: \( A \text{ CONJ } A' = A \)
  • Type Raising \( A = X/(X\backslash A) \)
CCG Parsing

John = np
Mary = np
likes = (s\np)/np

Forward application
X/Y Y => X

Backward application
Y X\Y => X

Thus

John  likes  Mary
np    (s\np)/np   np
       -------- Forward
s\np
       -------- Backward
s
CCG Parsing

a, the np/n
old n/n
in (np\np)/np
man, ball, park n
kicked (s\np)/np

the old man kicked a ball in the park
np/n n/n n (s\np)/np np/n n (np\np)/np np/n n

n
np

n
np

np

np\np

np

s\np

s
CCG Parsing and Semantics

\[ A/B:S + B:T = A:S.T \]
\[ B:T + A\backslash B:S = A:S.T \]

John np:j
walks (s\np):lambda X walks(X)

John \hspace{1cm} \text{walks}
np:j s\np:lambda X walks(X)
\hspace{1cm} \text{-------------------}
\hspace{1cm} s : \text{walks}(j)

\[ B:T + A\backslash B:S = A:S \cdot T \]
np:j + s\np:lambda X walks(X)
s : lambda X walks(X) . j
s : \text{walks}(j)
CCG Parsing and Semantics

John np:j
Mary np:m
likes (s\np)/np: lambda Y lambda X likes(X,Y)

John np:j (s\np)/np: lambda Y lambda X likes(X,Y) m

---------------------------------
s\np: lambda X likes(X,m)

---------------------------------
s likes(j,m)

lambda Y lambda X likes(X,Y) . m
lambda X likes(X,m)

lambda X likes(X,m) . j
likes(j,m)
Probabilistic CCGs

• Derive lexical entries from data
  • Find which entries allow parsing (constrained)

• From data with logical forms
  • Find out possible parses that derived those forms
  • But needs sentence → logical form training data

• But can work on targeted domains
SEMAFOR

• Semantic parser (Das et al 2014)
• Uses FrameNET to identify frames
• Fills in roles for a sentence

But there still are n’t enough ringers to ring more than six of the eight bells.

Frame  
LU

<table>
<thead>
<tr>
<th>Frame</th>
<th>LU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agent</td>
<td>N.m</td>
</tr>
<tr>
<td>Sound_maker</td>
<td></td>
</tr>
<tr>
<td>Item</td>
<td>Enabled_situation</td>
</tr>
<tr>
<td>Entity</td>
<td></td>
</tr>
<tr>
<td>NOISE MAKERS</td>
<td>bell.n</td>
</tr>
<tr>
<td>CAUSE TO MAKE NOISE</td>
<td>ring.v</td>
</tr>
<tr>
<td>SUFFICIENCY</td>
<td>enough.a</td>
</tr>
<tr>
<td>EXISTENCE</td>
<td>there be.v</td>
</tr>
</tbody>
</table>