Lecture 16:
More on Compositional Semantics, Verb Semantics

Julia Hockenmaier
juliahmr@illinois.edu
3324 Siebel Center

Combinatory Categorial Grammar (CCG)

Admin

Midterm:
Regrade requests for midterm accepted until Nov 9
Points available on Compass. 22 points = 100%

Project/Literature review proposals:
Due at the end of day on Monday on Compass
One page PDF (in LaTeX, not Word) is sufficient
Include your names and NetIDs
Include all references (ideally with hyperlinks)
Explain what you want to do and why.
Include a to-do list
For projects: describe what resources you have or need.
(Use existing datasets, don't annotate your own data)

CCG categories

Simple (atomic) categories: NP, S, PP

Complex categories (functions):
Return a result when combined with an argument

<table>
<thead>
<tr>
<th>VP, intransitive verb</th>
<th>S\NP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transitive verb</td>
<td>(S\NP)/NP</td>
</tr>
<tr>
<td>Adverb</td>
<td>(S\NP):(S\NP)</td>
</tr>
<tr>
<td>Prepositions</td>
<td>((S\NP):(S\NP))/NP, (NP\NP)/NP, PP/NP</td>
</tr>
</tbody>
</table>
CCG categories are functions

CCG has a few atomic categories, e.g.

S, NP, PP

All other CCG categories are functions:

\[
\begin{align*}
S & / \ NP \\
\text{Result} & \text{Dir. Argument}
\end{align*}
\]

Rules: Function application

\[
\begin{align*}
S & / \ NP \\
\text{Function} & \text{Argument}
\end{align*}
\]

\[
\begin{align*}
S & \cdot y \\
x & = x
\end{align*}
\]

\[
\begin{align*}
S & \cdot y \\
x & = x
\end{align*}
\]

\[
\begin{align*}
S & \cdot y \\
x & = x
\end{align*}
\]

\[
\begin{align*}
S & \cdot y \\
x & = x
\end{align*}
\]
A (C)CG derivation

Rules: Function Composition

\[
\begin{align*}
S/\text{NP} & \quad \text{1st Function} \\
\text{NP} & \quad \text{2nd Function}
\end{align*}
\]

\[
x \cdot y = \frac{x}{z}
\]

Rules: Type-Raising

Type-raising and composition

Type-raising: \( X \rightarrow T/(TX) \)
- Turns an argument into a function.

\[
\begin{align*}
\text{NP} & \quad \rightarrow \quad S/(S/\text{NP}) \quad \text{(subject)} \\
\text{NP} & \quad \rightarrow \quad (S/\text{NP})/(S/\text{NP}/\text{NP}) \quad \text{(object)}
\end{align*}
\]

Harmonic composition: \( X/Y \quad Y/Z \rightarrow X/Z \)
- Composes two functions (complex categories)

\[
\begin{align*}
(S/\text{NP})/\text{PP} & \quad \rightarrow \quad (S/\text{NP})/\text{NP} \\
S/(S/\text{NP}) & \quad \rightarrow \quad S/\text{NP}
\end{align*}
\]

Crossing function composition: \( X/Y \quad Y\backslash Z \rightarrow X\backslash Z \)
- Composes two functions (complex categories)

\[
\begin{align*}
(S/\text{NP})/S & \quad \rightarrow \quad (S/\text{NP})/\text{NP}
\end{align*}
\]
Type-raising and composition

Wh-movement (relative clause):

<table>
<thead>
<tr>
<th></th>
<th>the tapas</th>
<th>which</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP</td>
<td>(NP\NP)/(S/NP)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Mary</td>
<td>ordered</td>
</tr>
<tr>
<td></td>
<td>NP</td>
<td>(S/NP)/NP</td>
</tr>
<tr>
<td></td>
<td>S/(S\NP)</td>
<td>T</td>
</tr>
<tr>
<td></td>
<td>S/NP</td>
<td></td>
</tr>
</tbody>
</table>

Right-node raising:

<table>
<thead>
<tr>
<th></th>
<th>Mary</th>
<th>ordered</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP</td>
<td>(NP\NP)/(S/NP)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>S/(S\NP)</td>
<td>T</td>
</tr>
<tr>
<td></td>
<td>S/NP</td>
<td></td>
</tr>
<tr>
<td></td>
<td>S</td>
<td></td>
</tr>
</tbody>
</table>

Using Combinatory Categorial Grammar (CCG) to map sentences to predicate logic

\[ \lambda - \text{Expressions} \]

We often use \( \lambda \)-expressions to construct complex logical formulas:

- \( \lambda x.\varphi(\ldots x\ldots) \) is a function where \( x \) is a variable, and \( \varphi \) some FOL expression.

- \( \beta \)-reduction (called \( \lambda \)-reduction in textbook):
  Apply \( \lambda x.\varphi(\ldots x\ldots) \) to some argument \( a \):
  \( (\lambda x.\varphi(\ldots x\ldots)) a \Rightarrow \varphi(\ldots a\ldots) \)
  Replace all occurrences of \( x \) in \( \varphi(\ldots x\ldots) \) with \( a \)

- \( n\)-ary functions contain embedded \( \lambda \)-expressions:
  \( \lambda x.\lambda y.\lambda z.give(x,y,z) \)

\[ \text{CCG semantics} \]

Every syntactic constituent has a semantic interpretation:

Every lexical entry maps a word to a syntactic category and a corresponding semantic type:

John=\( (\text{NP}, \text{john'} ) \) Mary=\( (\text{NP}, \text{mary'} ) \)
loves: \( ((S\text{\textbackslash NP})/\text{NP} \lambda x.\lambda y.loloves(x,y)) \)

Every combinatory rule has a syntactic and a semantic part:

Function application:
\( \text{X}/\text{Y}:\lambda x.f(x) \quad \text{Y}:a \quad \Rightarrow \text{X}:f(a) \)

Function composition:
\( \text{X}/\text{Y}:\lambda x.f(x) \quad \text{Y}/\text{Z}:\lambda y.g(y) \quad \Rightarrow \text{X}/\text{Z}:\lambda z.f(\lambda y.g(y).z) \)

Type raising:
\( \text{X}:a \quad \Rightarrow \text{T}/(\text{T}\times\text{X}) \lambda f.f(a) \)
An example with semantics

\[
\begin{array}{c}
\text{John}
\end{array}
\begin{array}{c}
\text{sees}
\end{array}
\begin{array}{c}
\text{Mary}
\end{array}
\]

\[
\text{NP : John} \quad \begin{array}{c}
(S \backslash \text{NP}) / \text{NP}
\end{array}
\begin{array}{c}
\lambda x. \lambda y. \text{sees}(x, y)
\end{array}
\begin{array}{c}
\text{NP : Mary}
\end{array}
\]

\[
\begin{array}{c}
S \backslash \text{NP}
\end{array}
\begin{array}{c}
\lambda y. \text{sees}(\text{Mary}, y)
\end{array}
\begin{array}{c}
S
\end{array}
\begin{array}{c}
: \text{sees}(\text{Mary}, \text{John})
\end{array}
\]

Supplementary material: quantifier scope ambiguities in CCG

Quantifier scope ambiguity

"Every chef cooks a meal"

- **Interpretation A:** For every chef, there is a meal which he cooks.
  \[\forall x [\text{chef}(x) \rightarrow \exists y [\text{meal}(y) \land \text{cooks}(y, x)]]\]

- **Interpretation B:** There is some meal which every chef cooks.
  \[\exists y [\text{meal}(y) \land \forall x [\text{chef}(x) \rightarrow \text{cooks}(y, x)]]\]

**Interpretation A**
Understanding sentences

“Every chef cooks a meal”

\[
\forall x[\text{chef}(x) \rightarrow \exists y[\text{meal}(y) \wedge \text{cooks}(y, x)]]
\]

\[
\exists y[\text{meal}(y) \wedge \forall x[\text{chef}(x) \rightarrow \text{cooks}(y, x)]]
\]

We translate sentences into (first-order) predicate logic.

Every (declarative) sentence corresponds to a proposition, which can be true or false.

To summarize...

But...

… what can we do with these representations?

Being able to translate a sentence into predicate logic is not enough, unless we also know what these predicates mean.

Semantics joke (B. Partee): The meaning of life is life’

Compositional formal semantics tells us how to fit together pieces of meaning, but doesn’t have much to say about the meaning of the basic pieces (i.e. lexical semantics)

… how do we put together meaning representations of multiple sentences?

We need to consider discourse (there are approaches within formal semantics, e.g. Discourse Representation Theory)

… Do we really need a complete analysis of each sentence?

This is pretty brittle (it’s easy to make a parsing mistake)

Can we get a more shallow analysis?
Semantic Role Labeling/Verb Semantics

What do verbs mean?

Verbs describe events or states (‘eventualities’):

- Tom broke the window with a rock.
- The window broke.
- The window was broken by Tom by a rock.

We want to translate verbs to predicates. But: a naive translation (e.g. subject = first argument, object = second argument, etc.) does not capture the differences in meaning:

- break(Tom, window, rock)
- break(window)
- break(window, Tom)
- break(window, rock)

Semantic/Thematic roles

Verbs describe events or states (‘eventualities’):

- Tom broke the window with a rock.
- The window broke.
- The window was broken by Tom by a rock.

**Thematic roles** refer to participants of these events:

- **Agent** (who performed the action): Tom
- **Patient** (who was the action performed on): window
- **Tool/Instrument** (what was used to perform the action): rock

Semantic/thematic roles (agent, patient) are different from grammatical roles (subject or object).

The inventory of thematic roles

We need to define an inventory of thematic roles

To create systems that can identify thematic roles automatically, we need to create labeled training data.

It is difficult to give a formal definition of thematic roles that generalizes across all verbs.
PropBank and FrameNet

Proposition Bank (PropBank):

Very coarse argument roles (arg0, arg1,…), used for all verbs (but interpretation depends on the specific verb)
- Arg0 = proto-agent
- Arg1 = proto-patient
- Arg2,…: specific to each verb
- ArgM-TMP/LOC/…: temporal/locative/… modifiers

FrameNet:

Verbs fall into classes that define different kinds of frames (change-position-on-a-scale frame: rise, increase,…). Each frame has its own set of “frame elements” (thematic roles)

PropBank

agree.01 Arg0: Agreer Arg1: Proposition
Arg2: Other entity agreeing
[Arg0 The group] agreed [Arg1 it wouldn’t make an offer]
[Arg0 John] agrees with [Arg2 Mary]

fall.01 Arg1: patient/thing falling Arg2: extent/amount fallen
Arg3: start point Arg4: end point
[Arg1 Sales] fell [Arg4 to $251 million]
[Arg1 Junk bonds] fell [Arg2 by 5%]

Semantic role labeling: Recover the semantic roles of verbs (nowadays typically PropBank-style)
- Machine learning; trained on PropBank
- Syntactic parses provide useful information

Diathesis Alternations

Active/passive alternation:
- Tom broke the window with a rock. (active voice)
- The window was broken by Tom by a rock. (passive voice)

Causative alternation:
- Tom broke the window. (‘causative’; active voice)
- The window broke. (‘anticausative’/’inchoative’; active voice)

Dative alternation
- Tom gave the gift to Mary.
- Tom gave Mary the gift.

Locative alternation:
- Jessica loaded boxes into the wagon.
- Jessica loaded the wagon with boxes.

Verb classes

Verbs with similar meanings undergo the same syntactic alternations, and have the same set of thematic roles (Beth Levin, 1993)

VerbNet (verbs.colorado.edu; Kipper et al., 2008)

A large database of verbs, their thematic roles and their alternations