Outline of today's lecture

Compositional semantics: the construction of meaning based on syntax and morphology.

Lecture 6: Compositional semantics
- Compositional semantics using lambda calculus
- Logical representations in broad coverage grammars
- Simple semantics in feature structures
- Dependency structures
- Inference
- Recognising Textual Entailment task

Sentence meaning as logical form

Kitty chased Rover.
Rover was chased by Kitty.

Logical form (simplified!):

$$\text{chase}'(k, r)$$

k and r are constants (*Kitty* and *Rover*), chase$'$ is the predicate corresponding to *chase*.

- Sentence structure conveys some meaning: obtained by syntactic representation plus rules of semantic composition.
- Principle of Compositionality: meaning of each whole phrase derivable from meaning of its parts.

Semantic composition rules are non-trivial

Ordinary pronouns contribute to the semantics:

It barked.
$\exists x[bark'(x) \wedge \text{PRON}(x)]$

Pleonastic pronouns don't:

It rained.
$rain'$

Similar syntactic structures may have different meanings.
Different syntactic structures may have the same meaning:

Kim seems to sleep.
It seems that Kim sleeps.

Differences in presentation but not in truth conditions.

Lambda calculus and composition

- One semantic composition rule per syntax rule.
- `S -> NP VP`
  $VP'(NP')$
- Rover barks:
  VP *bark* is $\lambda x[\text{bark}'(x)]$
  NP *Rover* is r
  $\lambda x[\text{bark}'(x)](r) = \text{bark}'(r)$

Transitive verbs

Kitty chases Rover

- Transitive verbs: two arguments (NOTE the order)
  $\lambda x[\lambda y[\text{chase}'(y, x)]]$
- `VP -> Vtrans NP`
  Vtrans′(NP′)
- $\lambda x \lambda y[\text{chase}'(y, x)](r) = \lambda y[\text{chase}'(y, r)]$
- `S -> NP VP`
  VP′(NP′)
- $\lambda y[\text{chase}'(y, r)](k) = \text{chase}'(k, r)]$

Grammar fragment using lambda calculus

```
S -> NP VP
```
$VP'(NP')$

```
VP -> Vtrans NP
```
$Vtrans'(NP')$

```
VP -> Vintrans
```
$Vintrans'$

```
Vtrans -> chases
```
$\lambda x \lambda y[chase'(y, x)]$

```
Vintrans -> barks
```
$\lambda z[bark'(z)]$

```
Vintrans -> sleeps
```
$\lambda w[sleep'(w)]$

```
NP -> Kitty
```
k

Beyond toy examples . . .

- Use first order logic where possible (e.g., event variables).
- However, First Order Predicate Calculus (FOPC) is sometimes inadequate: e.g., *most*, *may*, *believe*.
- Quantifier scoping multiplies analyses:
  Every cat chased some dog:
  $\forall x[\text{cat}'(x) \implies \exists y[\text{dog}'(y) \wedge \text{chase}'(x, y)]]$
  $\exists y[\text{dog}'(y) \wedge \forall x[\text{cat}'(x) \implies \wedge \text{chase}'(x, y)]]$
- Often no straightforward logical analysis
  e.g., Bare plurals such as *Ducks lay eggs*.
- Non-compositional phrases (multiword expressions): e.g., *red tape* meaning bureaucracy.

Event variables

- Allow first order treatment of adverbs and PPs modifying verbs by reifying the event.
- Rover barked.
- instead of $\text{bark}'(r)$ we have $\exists e[\text{bark}'(e, r)]$
- Rover barked loudly.
- $\exists e[\text{bark}'(e, r) \wedge \text{loud}'(e)]$
- There was an event of Rover barking and that event was loud.

Simple compositional semantics in feature structures

- An alternative to lambda calculus
- Semantics is built up along with syntax
- Subcategorization ‘slot’ filling instantiates syntax
- Formally equivalent to logical representations (below: predicate calculus with no quantifiers)
- Alternative FS encodings possible

Example *they like fish*:
with semantics
$\text{pron}(x) \wedge (\text{like\_v}(x, y) \wedge \text{fish\_n}(y))$

Feature structure encoding of semantics

$$
\begin{bmatrix}
\text{PRED} & \textbf{and} \\
\text{ARG1} & \begin{bmatrix} \text{PRED} & \textbf{pron} \\ \text{ARG1} & \boxed{1} \end{bmatrix} \\
\text{ARG2} & \begin{bmatrix}
\text{PRED} & \textbf{and} \\
\text{ARG1} & \begin{bmatrix} \text{PRED} & \textbf{like\_v} \\ \text{ARG1} & \boxed{1} \\ \text{ARG2} & \boxed{2} \end{bmatrix} \\
\text{ARG2} & \begin{bmatrix} \text{PRED} & \textbf{fish\_n} \\ \text{ARG1} & \boxed{2} \end{bmatrix}
\end{bmatrix}
\end{bmatrix}
$$

$\text{pron}(x) \wedge (\text{like\_v}(x, y) \wedge \text{fish\_n}(y))$

Noun entry

fish

HEAD	[CAT **n** ; AGR [ ]]
OBJ	**fld**
SUBJ	**fld**
SEM	[INDEX [1] ; PRED **fish_n** ; ARG1 [1]]

- Corresponds to fish(x) where the INDEX points to the characteristic variable of the noun (that is x).
  The INDEX is unambiguous here, but
  e.g., picture(x, y) $\wedge$ sheep(y)
  picture of sheep

Noun entry

fish

$$\begin{bmatrix} \text{HEAD} & \begin{bmatrix} \text{CAT} & \textbf{n} \\ \text{AGR} & [\,] \end{bmatrix} \\ \text{OBJ} & \textbf{fld} \\ \text{SUBJ} & \textbf{fld} \\ \text{SEM} & \begin{bmatrix} \text{INDEX} & \boxed{1} \\ \text{PRED} & \textbf{fish\_n} \\ \text{ARG1} & \boxed{1} \end{bmatrix} \end{bmatrix}$$

- Corresponds to fish(x) where the INDEX points to the characteristic variable of the noun (that is x).
  The INDEX is unambiguous here, but
  e.g., picture(x, y) $\wedge$ sheep(y)
  picture of sheep

Verb entry

like

- HEAD
  - CAT **v**
  - AGR **pl**
- OBJ
  - HEAD
    - CAT **n**
  - OBJ **fld**
  - SEM
    - INDEX [2]
- SUBJ
  - HEAD
    - CAT **n**
  - SEM
    - INDEX [1]
- SEM
  - PRED **like_v**
  - ARG1 [1]
  - ARG2 [2]

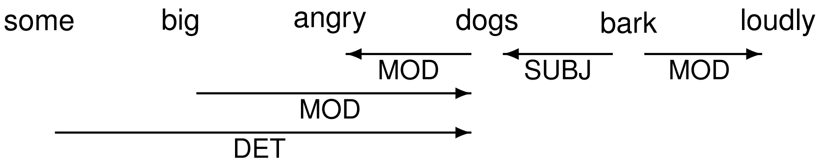
Verb-object rule

$$
\begin{bmatrix}
\text{HEAD} & \boxed{1} \\
\text{OBJ} & \textbf{fld} \\
\text{SUBJ} & \boxed{3} \\
\text{SEM} & \begin{bmatrix} \text{PRED} & \textbf{and} \\ \text{ARG1} & \boxed{4} \\ \text{ARG2} & \boxed{5} \end{bmatrix}
\end{bmatrix}
\rightarrow
\begin{bmatrix}
\text{HEAD} & \boxed{1} \\
\text{OBJ} & \boxed{2} \\
\text{SUBJ} & \boxed{3} \\
\text{SEM} & \boxed{4}
\end{bmatrix}
, \boxed{2}
\begin{bmatrix}
\text{OBJ} & \textbf{fld} \\
\text{SEM} & \boxed{5}
\end{bmatrix}
$$

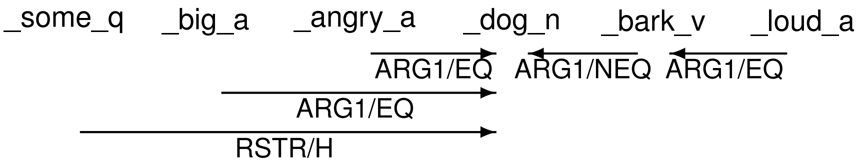
- As in Lecture 5: object of the verb (DTR2) ‘fills’ the OBJ slot
- New: semantics on the mother is the ‘and’ of the semantics of the dtrs

Dependency structure

- Alternative to syntax trees and to logical representation for 'who does what to whom'.
- Relate words to each other via labelled directed arcs (dependencies).
- May be syntactic or semantic.



Semantic dependencies



Equivalent to:

$$\text{\_some\_q}\ (x, \text{\_big\_a}(e1,x) \wedge \text{\_angry\_a}(e2,x) \wedge \text{\_dog\_n}(x), \text{\_bark\_v}(e3,x) \wedge \text{\_loud\_a}(e4,e3))$$

which in this case can be converted into FOPC:

$$\exists x\ [\ \text{\_big\_a}(e1,x) \wedge \text{\_angry\_a}(e2,x) \wedge \text{\_dog\_n}(x) \wedge \text{\_bark\_v}(e3,x) \wedge \text{\_loud\_a}(e4,e3)\ ]$$

Natural language inference

- Inference on a knowledge base: convert natural language expression to KB expression, valid inference according to KB.
  - \+ Precise
  - \+ Formally verifiable
  - \+ Disambiguation using KB state
  - \- Limited domain, requires KB to be formally encodable
- Language-based inference: does one utterance follow from another?
  - \+ Unlimited domain
  - +/- Human judgement
  - -/+ Approximate/imprecise
- Both approaches may use logical form of utterance.

Lexical meaning and meaning postulates

- Some inferences validated on logical representation directly, most require lexical meaning.
- meaning postulates: e.g.,

$$\forall x[\text{bachelor}'(x) \rightarrow \text{man}'(x) \wedge \text{unmarried}'(x)]$$

- usable with compositional semantics and theorem provers
- e.g. from ‘Kim is a bachelor’, we can construct the LF bachelor′(Kim) and then deduce unmarried′(Kim)
- Problematic in general, OK for narrow domains or micro-worlds.

Recognising Textual Entailment (RTE) shared tasks

T: The girl was found in Drummondville earlier this month.

H: The girl was discovered in Drummondville.

- DATA: pairs of text (T) and hypothesis (H). H may or may not follow from T.
- TASK: label TRUE (if follows) or FALSE (if doesn't follow), according to human judgements.

RTE using logical forms

- T sentence has logical form T′, H sentence has logical form H′
- If $T' \implies H'$ conclude TRUE, otherwise conclude FALSE.

T The girl was found in Drummondville earlier this month.

T′ $\exists x, u, e[\text{girl}'(x) \wedge \text{find}'(e, u, x) \wedge \text{in}'(e, \text{Drummondville}) \wedge \text{earlier-this-month}'(e)]$

H The girl was discovered in Drummondville.

H′ $\exists x, u, e[\text{girl}'(x) \wedge \text{discover}'(e, u, x) \wedge \text{in}'(e, \text{Drummondville})]$

MP $[\text{find}'(x, y, z) \implies \text{discover}'(x, y, z)]$

- So $T' \implies H'$ and we conclude TRUE

More complex examples

T: Four Venezuelan firefighters who were traveling to a training course in Texas were killed when their sport utility vehicle drifted onto the shoulder of a highway and struck a parked truck.

H: Four firefighters were killed in a car accident.

Systems using logical inference are not robust to missing information: simpler techniques can be effective (partly because of choice of hypotheses in RTE).

More examples

T: Clinton's book is not a big seller here.
H: Clinton's book is a big seller.

T: After the war the city was briefly occupied by the Allies and then was returned to the Dutch.
H: After the war, the city was returned to the Dutch.

T: Lyon is actually the gastronomic capital of France.
H: Lyon is the capital of France.

Next time ...

- Lexical semantics and semantic relations
- Word sense ambiguity