Natural Language Processing

Lecture 14

Today's Outline

Exam

Semantics

Semantic Interpretation

- Syntax-Driven Compositional Analysis
- Idioms
- More Robust Syntax-Driven Methods

Semantics

Up until to now, we have not focussed at all on what things *mean*.

The approach to meaning that we'll follow here is based on the systematic creation of <u>meaning representations</u> – representations that bridge the gap from linguistic forms to knowledge of the world.

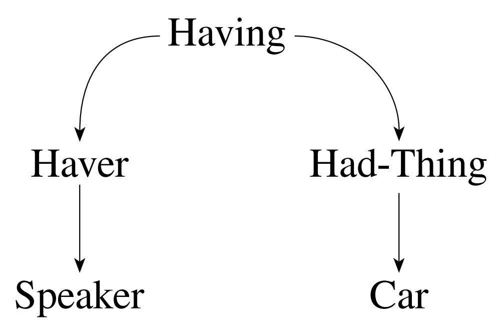
Meaning Representations

What can serve as a meaning representation . . . Well anything that serves the core practical purposes of a program that is doing semantic processing . . .

- answer questions
- determining truth
- making inference
- . . .

A Sampler of Meaning Representations

$$\exists x, y\, Having(x) \wedge Haver(Speaker, x) \wedge HadThing(y, x) \wedge Car(y)$$



$$\begin{array}{l} \mathit{Car} \\ \quad \Uparrow \textsc{poss-by} \\ \mathit{Speaker} \end{array}$$

```
Having
 Haver:  Speaker
 HadThing:  Car
```

Meaning Representations (cont.)

Consider the following examples:

- Is Denver north of Pueblo?
- Is Pueblo south of Denver?
- Denver is north of Pueblo.
- If I'm driving south on 125 from Cheyenne will I get to Denver or Pueblo first?
- If Denver is north of Colorado Springs and Colorado Springs is north of Pueblo, is Denver north of Pueblo?

Verifiability

Why are meaning representations needed?

Examples

- Does Spice Island serve vegetarian food?
- Serves(SpiceIsland,VegetarianFood)

Verifiability: The system's ability to compare representation to facts in KB.

Basic Assumptions

Focus on literal meaning

- conventional meanings of words
- ignore context

Ambiguity

I want to eat someplace that's close to Pitt.

Vagueness

Vagueness: Uncertainty; exact specifics of the entity not known

I want to eat Italian food.

Canonical Form

Inputs that mean the same thing have the same representation.

"Standardized" representation of meaning for (potentially) multiple utterances.

- Does Spice Island have vegetarian dishes?
- Do they have vegetarian food at Spice Island?
- Are vegetarian dishes served at Spice Island?
- Does Spice Island serve vegetarian fare?

Alternatives

- four different semantic representations
- store all possible meaning representations in KB

Inference

Inference: Draw valid conclusions based on the meaning representation of inputs and its store of background knowledge.

Can vegetarians eat at Spice Island?

Meaning Representations: FOPC

We'll make use of First Order Predicate Calculus (FOPC) as our meaning representation.

It is not ideal and doesn't do everything we want. But it is pretty close.

- supports the determination of truth
- supports the answering of questions (via variables)
- supports inference

FOPC Syntax

$$
\begin{array}{rcl}
\textit{Formula} & \rightarrow & \textit{AtomicFormula} \\
& | & \textit{Formula Connective Formula} \\
& | & \textit{Quantifier Variable}, \ldots \textit{ Formula} \\
& | & \neg \textit{ Formula} \\
& | & (\textit{Formula}) \\
\\
\textit{AtomicFormula} & \rightarrow & \textit{Predicate}(\textit{Term}, \ldots) \\
\\
\textit{Term} & \rightarrow & \textit{Function}(\textit{Term}, \ldots) \\
& | & \textit{Constant} \\
& | & \textit{Variable} \\
\\
\textit{Connective} & \rightarrow & \wedge \mid \vee \mid \Rightarrow \\
\textit{Quantifier} & \rightarrow & \forall \mid \exists \\
\textit{Constant} & \rightarrow & A \mid \textit{VegetarianFood} \mid Maharani \cdots \\
\textit{Variable} & \rightarrow & x \mid y \mid \cdots \\
\textit{Predicate} & \rightarrow & \textit{Serves} \mid Near \mid \cdots \\
\textit{Function} & \rightarrow & \textit{LocationOf} \mid \textit{CuisineOf} \mid \cdots
\end{array}
$$

Truth Table

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$
False	*False*	*True*	*False*	*False*	*True*
False	*True*	*True*	*False*	*True*	*True*
True	*False*	*False*	*False*	*True*	*False*
True	*True*	*False*	*True*	*True*	*True*

Meaning Structure of Language

This choice of FOPC isn't completely arbitrary or driven by the needs of applications ...

Human languages

- display a basic predicate-argument structure
- make use of variables
- make use of quantifiers
- display a partially compositional semantics (sort of)

Predicate Argument Structure

It appears that languages make heavy use of a predicate-argument structure. In particular, there are words and constituents that prototypically act like predicates, and words and constituents that act like arguments.

Predicate-like elements

- verbs, VPs, prepositions, adjectives, some nouns

Argument-like elements

- nouns, nominals, NPs

Syntax

- NP give NP2 NP1
- NP give NP1 to NP2

Semantics

- give(x,y,z)

Example

John gave Mary a book.

$Giving(John, Mary, Book)$

Better

John gave Mary a book.

$\exists x, y Giving(x) \wedge Giver(John, x) \wedge Givee(Mary, x) \wedge Given(y, x) \wedge Isa(y, Book)$

This is the "right" way to do it.

Predicates

You can think of this as the Verb give contributing the predicate and the number and names of the arguments to the representation:

Ie. $Give$ means ...

$\exists w, x, y, z Giving(x) \wedge Giver(w, x) \wedge Givee(z, x) \wedge Given(y, x)$

The NPs in the sentence provided the values for the variables introduced by the named arguments to the Verb.

Earlier Examples

Is Denver north of Pueblo?

- $NorthOf(Denver, Pueblo)$

What is north of Denver?

- $\exists x NorthOf(x, Denver)$

If I'm driving ...

- $\forall x, y, z NorthOf(x, y) \wedge NorthOf(y, z) \Rightarrow NorthOf(x, z)$

Semantic Analysis

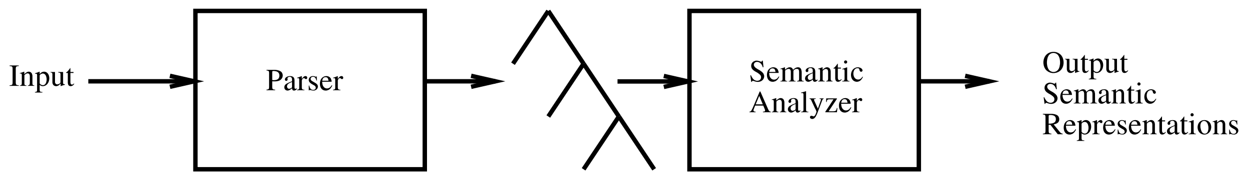
Semantic Analysis is the process of taking in some linguistic form and producing a meaning representation for it.

There are many ways of doing this, ranging from completely ad hoc domain specific methods to more theoretically founded but not quite useful methods.

Most methods rely in some way on a prior or concurrent syntactic analysis (parse).

We'll outline a compositional rule-to-rule approach.

A Simple Pipeline



Compositional Semantics

At the core of most methods is the principle of compositionality which states that the meaning of the whole is based on the meaning of parts.

What are the parts?

Compositional Parts

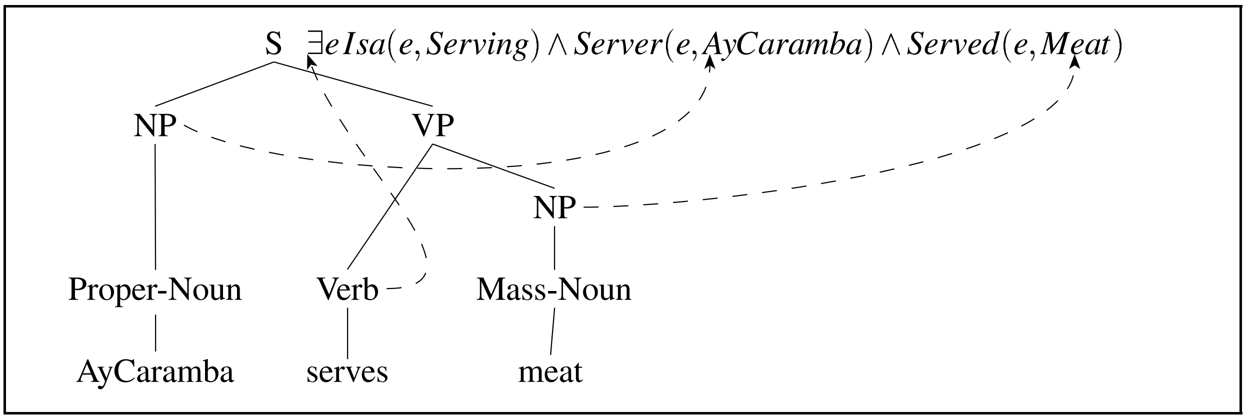
What are the parts?

- words and syntactic constituents

What does it mean for a part (word or constituents) to have a meaning?

Example

AyCaramba serves meat.



Augmented Rules

We'll accomplish this by attaching semantic rules to our CFG rules just as we did with unification.

Abstractly . . .

- $A \rightarrow \alpha_1 \ldots \alpha_n \; \{f(\alpha_j.sem, \ldots, \alpha_k.sem)\}$

Augmented Rules

Back to our example ...

- $ProperNoun \rightarrow AyCaramba\ \{AyCaramba\}$
- $MassNoun \rightarrow meat\ \{Meat\}$
- $NP \rightarrow ProperNoun\ \{ProperNoun.sem\}$
- $NP \rightarrow MassNoun\ \{MassNoun.sem\}$
- $Verb \rightarrow serves\ \{\exists e, x, y Isa(e, Serving) \wedge Server(e, x) \wedge Served(e, y)\}$

Lambda Forms

We've been speaking informally for a while now as if Verbs, and similar objects, were functions. The arguments to these functions are the various constituents in the tree. Therefore the semantics of the Verb should encode the incorporation of its arguments. We can use lambda forms to do that:

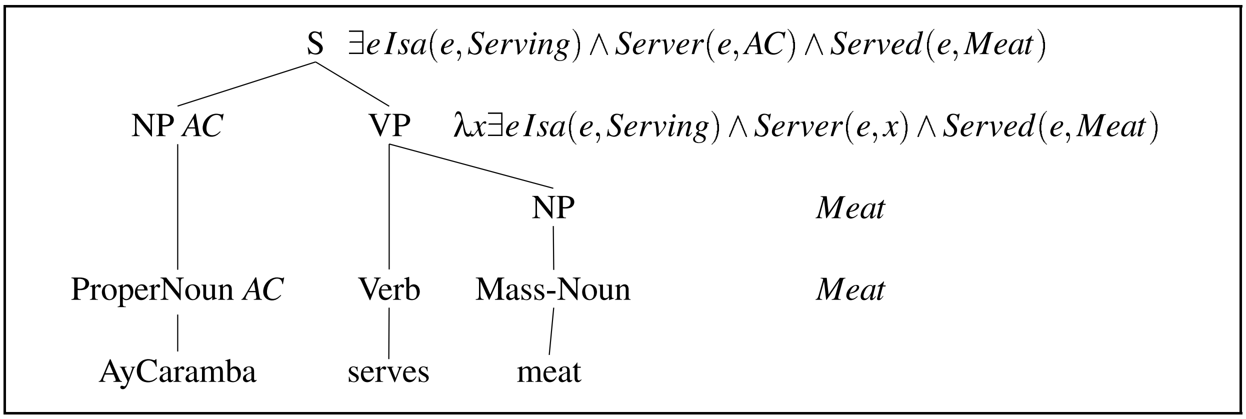
- $\lambda x P(x)$
- $\lambda x P(x)(A)$
- $P(A)$

Lambda Forms (cont.)

- $Verb \rightarrow serves\ \{\lambda x \lambda y \exists e Isa(e, Serving) \wedge Server(e, y) \wedge Served(e, x)\}$
- $S \rightarrow NPVP\{VP.sem(NP.sem)\}$
- $VP \rightarrow VerbNP\{Verb.sem(NP.sem)\}$

Lambda Forms (cont.)

AyCaramba serves meat.



Another Example

How about ...

- *A restaurant serves meat.*

Should be ...

- $\exists e, x Isa(e, Serving) \wedge Server(e, x) \wedge Served(e, Meat) \wedge Isa(x, Restaurant)$

Another Example (cont.)

Let's assume that the interpretation of "a restaurant" is

- $\exists x Isa(x, Restaurant) \wedge Isa(x, Restaurant)$

Then how can we get this into the expression

- $\exists e, x Isa(e, Serving) \wedge Server(e, x) \wedge Served(e, Meat) \wedge Isa(x, Restaurant)$

Another Example (cont.)

The λ application in the S rule does not do it correctly. It produces something like ...

- $\exists e Isa(e, Serving) \wedge Server(e, \exists x Isa(x, Restaurant)) \wedge Served(e, Meat)$

The problem is that the semantic representation of subject NP has parts that need to be distributed or incorporated in various ways into the representation of the VP and hence the S.

Complex Terms

The solution is make the representation of the NP into a kind of object with parts that are accessible.

Complex term

- $\langle$ *Quantifier variable body* $\rangle$
- $\exists x Isa(x, Restaurant)$

Back to the Example

- $\exists e Isa(e, Serving) \wedge Server(e, \langle \exists x Isa(x, Restaurant) \rangle) \wedge Served(e, Meat)$

To get from this to a correct FOL form ...

- P($\langle$ *Quantifier variable body* $\rangle$) $\Rightarrow$ *Quantifier Variable body Connective P(variable)*
- $Server(e, \langle \exists x, Isa(x, Restaurant) \rangle) \Rightarrow \exists x Isa(x, Restaurant) \wedge Server(e, x)$

Quantifier Scoping Ambiguity

Problem: What happens if there is more than one complex term?

- *Every restaurant has a menu.*
- $\exists e Isa(e, Having) \wedge Haver(e, \langle \forall x Isa(x, Restaurant) \rangle) \wedge Had(e, \langle \exists y Isa(y, Menu) \rangle)$

Quantifier Scoping Ambiguity (cont.)

Is it . . .

- $\forall x Restaurant(x) \Rightarrow \exists e, y Having(e) \wedge Haver(e, x) \wedge Isa(y, Menu) \wedge Had(e, y)$

or is it . . .

- $\exists y Isa(y, Menu) \wedge \forall x Isa(x, Restaurnat) \Rightarrow \exists e Having(e) \wedge Haver(e, x) \wedge Had(e, y)$

Doing Compositional Semantics

Incorporating semantics into a grammar involves two separate but intertwined processes:

1. Figuring out the right representation for a single constituent based on the parts of that constituent.
2. Figuring out the right representation for a category of constituents based on the other grammar rules that make use of that constituent.

Semantic Analysis

Ok. So we now have a bunch of function-like semantic attachments incorporated into the grammar. How do we do semantic analysis?

There are two possibilities:

1. Just as we did with unification–alter the Early parser so that when completed constituents (dot at the end of the rule) are created, the semantic function attached to the rule is applied and a meaning representation is created and stored with that state.
2. Let the parser run to completion and walk the resulting tree running the semantic attachments from the bottom-up.

Semantic Analysis (cont.)

- $S \rightarrow NP\ \ VP\ \{VP.sem(NP.sem)\}$

The VP.sem is stored in the state representing the VP, and the NP.sem is stored with the state for the NP. When this rule is completed you go get the VP.sem (it better be a lambda), go get the NP.sem, and apply the VP.sem to the Np.sem and store the result in the S.sem.

Integrated Semantic Analysis

In the integrated style, semantic analysis is fully integrated into the parser. As fragments of the input are parsed semantic fragments are created. This can be used to block ambiguous representations.

- *I want to eat someplace near campus.*

Integrated Semantic Analysis (cont.)

But . . . it also means that you're performing semantic analysis on orphaned constituents that play no role in a final parse. Hence a pipelined semantic analysis might be better.

Question

Is it really necessary to specify the semantic attachment for this rule and rules like it?

- $S \rightarrow NP\ \ VP\ \{VP.sem(NP.sem)\}$

Question (cont.)

Is it really necessary to specify the semantic attachment for all the rules in grammar?

No, you should be able to figure out the right thing to do based on the types of the semantic objects attached to the constituents of the rule . . .

So if you have two parts on the right-hand side and one is a function-like thing and one is an argument-like thing what should you do?

Using Earley

In an Earley parser, semantic attachments mean creating a meaning representation for states that are complete when they are processed (in Completer, typically).

Typically, unification rules are used to make sure that only the right rules are applied in the right circumstances.

Typically, it also involves a fair amount of duplication for all of the possible variations of verb argument number, type and expected location in the input.

Example

Tell

- John told Mary.
- John told a story.
- John told Mary a story.
- John told Mary to go home.
- ...

All these would require separate lexical entries with separate lambda attachments.

Non-Compositional Language

Unfortunately, there are lots of examples of non-compositional inputs – inputs where the final meaning is not derived from the meanings of the component parts alone.

- jokes
- irony
- indirect requests
- metaphor
- idioms
- . . .

Kick the Bucket

$VP \rightarrow kick\ the\ bucket\ \{\lambda x Die(x)\}$

Tip of the iceberg

Coupons are just *the tip of the iceberg.*

The SEC's allegations are only *the tip of the iceberg.*

Coronary bypass surgery, hip replacement and intensive-care units are but *the tip of the iceberg.*

BUT

And that's but *the tip of Mrs. Ford's iceberg.*

These comments describe only *the tip of a 1,000-page iceberg.*

The 10 employees represent *the merest tip of the iceberg.*

Robust Variants

Semantic Grammars

Information Extraction and Finite-State Methods

Semantic Grammars

As we've seen, a problem with syntactically motivated grammars is that they often don't served semantic purposes very well. One solution is to fight to make them work (as in the first part of Chapter 15).

An alternative is to work with grammars that closely match the semantics. Such grammars often go by the name of <u>semantic grammars</u>.

Example

I want to go to eat Italian food.

We can dump the notion of sentence and parse looking for something closer to what users say in this domain.

- *InfoRequest* $\rightarrow$ *User want to go to eat Foodtype*
- *Foodtype* $\rightarrow$ *Italian food*

Information Extraction

A different kind of work-around is called for with what are called <u>information extraction</u> tasks.

In these tasks, a system is asked to deal with arbitrary running written text and to form shallow attribute-value meaning representations suitable for insertion into a database.

Information Extraction Examples

Scanning newspapers/newswires for financial events of interest

Scanning websites for price/sales information of interest

Scanning radio/tv broadcasts

These tasks may involve extended discourse and typically involve text with long sentences, complex syntax, large vocabularies, multiple writer/speakers, ...

Robustness

Unfortunately, there are practical problems with using syntactically motivated grammars here as well...

Large broad coverage context grammars for English are hard to come by.

The degree of ambiguity in them makes them nearly unusable for arbitrary text.

When you don't get 50 parses, you end up with none.

Parsing is too slow for many current commercial applications.

Robustness (cont.)

Unfortunately, the semantic grammar trick does not scale well to the information extraction domain. A small domain specific grammar can not be used to parse/analyze text from an unrestricted domain.

So we can stay with a syntax-driven approach but fall back from context-free methods to finite state methods.

Finite-State Methods

One approach is to go a cascaded finite-state approach.

In this approach, a series of finite-state transducers are applied to the input. At each stage, some small aspect of syntax/semantics is captured and enclosed so it is available for the next stage.

The end result is typically a simple database scheme with slots to be filled in for the information of interest. The slots can be filled either with strings from the input text or symbols from some small alphabet.

FASTUS Example

Bridgestone Sports Co. said Friday it has set up a joint venture in Taiwan with a local concern and a Japanese trading house to produce golf clubs to be shipped to Japan.

The joint venture, Bridgestone Sports Taiwan Co., capitalized at 20 million new Taiwan dollars, will start production in January 1990 with production of 20,000 iron and "metal wood" clubs a month.

FASTUS Example (cont.)

TIE-UP-1:

Relationship:	TIE-UP
Entities:	“Bridgestone Sports Co.”
	“a local concern”
	“a Japanese trading house”
Joint Venture Company	“Bridgestone Sports Taiwan Co.”
Activity	ACTIVITY-1
Amount	NT$20000000

ACTIVITY-1:

Company	“Bridgestone Sports Taiwan Co.”
Product	“iron and “metal wood” clubs”
Start Date	DURING: January 1990

Cascades

Work from the input up, doing the easy things first (like tagging).

No.	Step	Description
1	**Tokens:**	Transfer an input stream of characters into a token sequence.
2	**Complex Words:**	Recognize multi-word phrases, numbers, and proper names.
3	**Basic phrases:**	Segment sentences into noun groups, verb groups, and particles.
4	**Complex phrases:**	Identify complex noun groups and complex verb groups.
5	**Semantic Patterns:**	Identify semantic entities and events and insert into templates.
6	**Merging:**	Merge references to the same entity or event from different parts of the text.

Key Point

What about material that is not of interest or can not be processed?

Skipped. Just isn't written out the tape for the next level of processing.

Some systems even use a filter to identify sentences that are not relevant to the task and remove them to begin with.

For Next Time

Chapter 16

Review: Meaning Representations

The approach to meaning that we'll present is based on the systematic creation of meaning representations–representations that bridge that gap from linguistic forms to knowledge of the world.

Review: Semantic Analysis

Semantic analysis is the process of taking in some linguistic form and producing a meaning representation for it.

Most methods rely in some way on a prior or concurrent syntactic analysis (parse).

We'll outline a compositional rule-to-rule approach.

Review: Compositional Semantics

At the core of most methods is the principle compositionality which states that the meaning of the whole is based on the meaning of parts.

The parts are the words and syntactic constituents specified by the grammar.

Review: Semantic Attachments

Given semantic attachments to context free grammar rules there are two approaches possible:

1. Alter a syntactic parser so that when completed constituents are created, the semantic function attached to the rule is applied and a meaning representation is created and stored with that state.
2. Let the parser run to completion and walk the resulting tree running the semantic attachments from the bottom-up.

Acknowledgements

Some of this material was taken from the course web pages of James Martin and Philip Resnik.