Computational Semantics

Deep Processing for NLP
Ling 571
February 6, 2017

Roadmap

- Motivation: Dialog Systems
- Key challenges
- Meaning representation
  - Representational requirements
  - First-order logic
    - Syntax & Semantics
  - Representing compositional meaning

Dialogue Systems

- User: What do I have on Thursday?
- Parse:
  - (S
  - (Q-WH-Obj
  - (Whwd What)
  - (Aux do)
  - (NP (Pron I))
  - (VP/NP (V have)
  - (NP/NP *t*)
  - (PP (Prep on)
  - (NP (N Thursday))))))

Dialogue Systems

- Parser:
  - Yes, it’s grammatical!
  - Here’s the structure!
- System: Great, but what am I supposed to DO?!
- Need to associate meaning with structure

Dialogue Systems

- (S
- (Q-WH-Obj Action: check; cal: USER; Date:Thursday
- (Whwd What)
- (Aux do)
- (NP (Pron I)) Cal: USER
- (VP/NP (V have)
- (NP/NP *t*)
- (PP (Prep on)
- (NP (N Thursday)))))) Date: Thursday

Natural Language

- Syntax: Determine the structure of natural language input
- Semantics: Determine the meaning of natural language input

Tasks for Semantics

- Semantic interpretation required for many tasks
  - Answering questions
  - Following instructions in a software manual
  - Following a recipe

- Requires more than phonology, morphology, syntax
- Must link linguistic elements to world knowledge

Semantics is Complex

- Sentences have many entailments, presuppositions
- *Instead, the protests turned bloody, as anti-government crowds were confronted by what appeared to be a coordinated group of Mubarak supporters*.

Semantics is Complex

- Sentences have many entailments, presuppositions
- *Instead, the protests turned bloody, as anti-government crowds were confronted by what appeared to be a coordinated group of Mubarak supporters*.
  - The protests became bloody.

Semantics is Complex

- Sentences have many entailments, presuppositions
- *Instead, the protests turned bloody, as anti-government crowds were confronted by what appeared to be a coordinated group of Mubarak supporters*.
  - The protests became bloody.
  - The protests had been peaceful.

Semantics is Complex

- Sentences have many entailments, presuppositions
- *Instead, the protests turned bloody, as anti-government crowds were confronted by what appeared to be a coordinated group of Mubarak supporters*.
  - The protests became bloody.
  - The protests had been peaceful.
  - Crowds oppose the government.

Semantics is Complex

- Sentences have many entailments, presuppositions
- *Instead, the protests turned bloody, as anti-government crowds were confronted by what appeared to be a coordinated group of Mubarak supporters*.
  - The protests became bloody.
  - The protests had been peaceful.
  - Crowds oppose the government.
  - Some support Mubarak.

Semantics is Complex

- Sentences have many entailments, presuppositions
- *Instead, the protests turned bloody, as anti-government crowds were confronted by what appeared to be a coordinated group of Mubarak supporters*.
  - The protests became bloody.
  - The protests had been peaceful.
  - Crowds oppose the government.
  - Some support Mubarak.
  - There was a confrontation between two groups.
  - Anti-government crowds are not Mubarak supporters.
  - Etc..

Challenges in Semantics

- Semantic representation:
  - What is the appropriate formal language to express propositions in linguistic input?

Challenges in Semantics

- Semantic representation:
  - What is the appropriate formal language to express propositions in linguistic input?
    - E.g. predicate calculus
      - $\exists x\ (dog(x) \land disappear(x))$

Challenges in Semantics

- Semantic representation:
  - What is the appropriate formal language to express propositions in linguistic input?
    - E.g. predicate calculus
      - $\exists x.(dog(x) \wedge disappear(x))$

- Entailment:
  - What are all the valid conclusions that can be drawn from an utterance?

Challenges in Semantics

- Semantic representation:
  - What is the appropriate formal language to express propositions in linguistic input?
    - E.g. predicate calculus
      - $\exists x.(dog(x) \wedge disappear(x))$
- Entailment:
  - What are all the valid conclusions that can be drawn from an utterance?
    - 'Lincoln was assassinated' entails

Challenges in Semantics

- Semantic representation:
  - What is the appropriate formal language to express propositions in linguistic input?
    - E.g. predicate calculus
      - $\exists x.(dog(x) \land disappear(x))$

- Entailment:
  - What are all the valid conclusions that can be drawn from an utterance?
    - ‘Lincoln was assassinated’ entails ‘Lincoln is dead.’

Challenges in Semantics

- Reference: How do linguistic expressions link to objects/concepts in the real world?
  - 'the dog' , 'the evening star', 'the Superbowl'

Challenges in Semantics

- Reference: How do linguistic expressions link to objects/concepts in the real world?
  - 'the dog' , 'the evening star', 'the Superbowl'

- Compositionality: How can we derive the meaning of a unit from its parts?
  - How do syntactic structure and semantic composition relate?
    - 'rubber duck' vs 'rubber chicken'

Challenges in Semantics

- Reference: How do linguistic expressions link to objects/concepts in the real world?
  - 'the dog' , 'the evening star', 'the Superbowl'

- Compositionality: How can we derive the meaning of a unit from its parts?
  - How do syntactic structure and semantic composition relate?
    - 'rubber duck' vs 'rubber chicken'
    - 'kick the bucket'

Tasks in Computational Semantics

- Computational semantics aims to extract, interpret, and reason about the meaning of NL utterances, and includes:
  - Defining a **meaning representation**

Tasks in Computational Semantics

- Computational semantics aims to extract, interpret, and reason about the meaning of NL utterances, and includes:
  - Defining a **meaning representation**
  - Developing techniques for **semantic analysis,** to convert NL strings to meaning representations

Tasks in Computational Semantics

- Computational semantics aims to extract, interpret, and reason about the meaning of NL utterances, and includes:
  - Defining a **meaning representation**
  - Developing techniques for **semantic analysis,** to convert NL strings to meaning representations
  - Developing methods for reasoning about these representations and performing inference from them

NLP Semantics Tasks

- Tasks:
  - Semantic similarity: words, texts
  - Semantic role labeling
  - Semantic analysis
  - “Semantic parsing”
  - Recognizing textual entailment
  - Sentiment Analysis

Complexity of Computational Semantics

- Requires:

Complexity of Computational Semantics

- Requires:
  - Knowledge of language: words, syntax, relationships b/t structure and meaning, composition procedures

Complexity of Computational Semantics

- Requires:
  - Knowledge of language: words, syntax, relationships b/t structure and meaning, composition procedures
  - Knowledge of the world: what are the objects that we refer to, how do they relate, what are their properties?

Complexity of Computational Semantics

- Requires:
  - Knowledge of language: words, syntax, relationships b/t structure and meaning, composition procedures
  - Knowledge of the world: what are the objects that we refer to, how do they relate, what are their properties?
  - Reasoning: Given a representation and a world, what new conclusions – bits of meaning – can we infer?

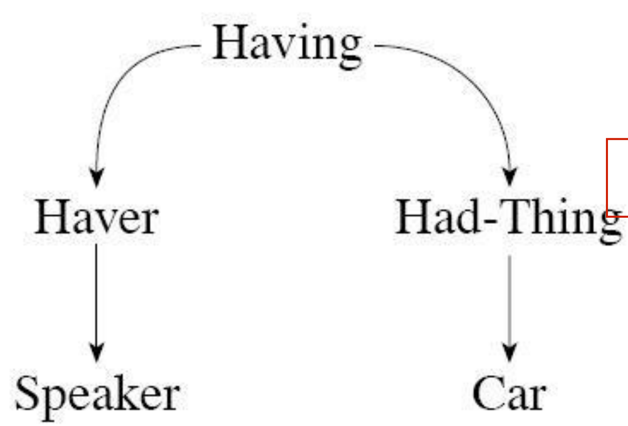
Complexity of Computational Semantics

- Requires:
  - Knowledge of language: words, syntax, relationships b/t structure and meaning, composition procedures
  - Knowledge of the world: what are the objects that we refer to, how do they relate, what are their properties?
  - Reasoning: Given a representation and a world, what new conclusions – bits of meaning – can we infer?
- Effectively AI-complete
  - Need representation, reasoning, world model, etc

Representing Meaning

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

First-order Logic



Semantic Network

Conceptual Dependency

Car
$\Uparrow$ POSS-BY
$Speaker$

Frame-Based

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

Meaning Representations

- All consist of structures from set of symbols
  - Representational vocabulary

Meaning Representations

- All consist of structures from set of symbols
  - Representational vocabulary
- Symbol structures correspond to:
  - Objects
  - Properties of objects
  - Relations among objects

Meaning Representations

- All consist of structures from set of symbols
  - Representational vocabulary
- Symbol structures correspond to:
  - Objects
  - Properties of objects
  - Relations among objects
- Can be viewed as:

Meaning Representations

- All consist of structures from set of symbols
  - Representational vocabulary
- Symbol structures correspond to:
  - Objects
  - Properties of objects
  - Relations among objects
- Can be viewed as:
  - Representation of meaning of linguistic input

Meaning Representations

- All consist of structures from set of symbols
  - Representational vocabulary
- Symbol structures correspond to:
  - Objects
  - Properties of objects
  - Relations among objects
- Can be viewed as:
  - Representation of meaning of linguistic input
  - Representation of state of world
- Here we focus on **literal** meaning

Representational Requirements

- Verifiability
- Unambiguous representations
- Canonical Form
- Inference and Variables
- Expressiveness

Representational Requirements

- Verifiability
  - Can compare representation of sentence to KB model
- Unambiguous representations
- Canonical Form
- Inference and Variables
- Expressiveness

Representational Requirements

- Verifiability
  - Can compare representation of sentence to KB model
- Unambiguous representations
  - Semantic representation itself is unambiguous
- Canonical Form
- Inference and Variables
- Expressiveness

Representational Requirements

- Verifiability
  - Can compare representation of sentence to KB model
- Unambiguous representations
  - Semantic representation itself is unambiguous
- Canonical Form
  - Alternate expressions of same meaning map to same rep
- Inference and Variables
- Expressiveness

Representational Requirements

- Verifiability
  - Can compare representation of sentence to KB model
- Unambiguous representations
  - Semantic representation itself is unambiguous
- Canonical Form
  - Alternate expressions of same meaning map to same rep
- Inference and Variables
  - Way to draw valid conclusions from semantics and KB
- Expressiveness

Representational Requirements

- Verifiability
  - Can compare representation of sentence to KB model
- Unambiguous representations
  - Semantic representation itself is unambiguous
- Canonical Form
  - Alternate expressions of same meaning map to same rep
- Inference and Variables
  - Way to draw valid conclusions from semantics and KB
- Expressiveness
  - Represent any natural language utterance

Meaning Structure of Language

- Human languages
  - Display basic predicate-argument structure
  - Employ variables
  - Employ quantifiers
  - Exhibit a (partially) compositional semantics

Predicate-Argument Structure

- Represent concepts and relationships
- Words behave like predicates:

Predicate-Argument Structure

- Represent concepts and relationships
- Words behave like predicates:
  - Verbs, Adj, Adv:
    - Book(John,United); Non-stop(Flight)
- Some words behave like arguments:

Predicate-Argument Structure

- Represent concepts and relationships
- Words behave like predicates:
  - Verbs, Adj, Adv:
    - Book(John,United); Non-stop(Flight)
- Some words behave like arguments:
  - Nouns: Book(John,United); Non-stop(Flight)

Predicate-Argument Structure

- Represent concepts and relationships
- Words behave like predicates:
  - Verbs, Adj, Adv:
    - Book(John,United); Non-stop(Flight)
- Some words behave like arguments:
  - Nouns: Book(John,United); Non-stop(Flight)
- Subcategorization frames indicate:
  - Number, Syntactic category, order of args

First-Order Logic

- Meaning representation:
  - Provides sound computational basis for verifiability, inference, expressiveness
- Supports determination of propositional truth
- Supports compositionality of meaning
- Supports inference
- Supports generalization through variables

First-Order Logic

- FOL **terms:**
  - **Constants:** specific objects in world;
    - *A, B, John*
    - Refer to exactly one object; objects referred to by many

First-Order Logic

- FOL **terms:**
  - **Constants:** specific objects in world;
    - *A, B, John*
    - Refer to exactly one object; objects referred to by many
  - **Functions:** concepts refer to objects, e.g. SFO's loc
    - *LocationOf(SFO)*
    - Refer to objects, avoid using constants

First-Order Logic

- FOL **terms:**
  - **Constants:** specific objects in world;
    - *A, B, John*
    - Refer to exactly one object; objects referred to by many
  - **Functions:** concepts refer to objects, e.g. SFO's loc
    - *LocationOf(SFO)*
    - Refer to objects, avoid using constants
  - **Variables:**
    - *x, e*

FOL Representation

- **Predicates:**
  - Relations among objects
    - *United serves Chicago.* ➔➔
    - *Serves(United, Chicago)*
    - *United is an airline.* ➔➔
    - *Airline(United)*

FOL Representation

- **Predicates:**
  - Relations among objects
    - *United serves Chicago.* ➔➔
    - *Serves(United, Chicago)*
    - *United is an airline.* ➔➔
    - *Airline(United)*
- **Logical connectives:**
  - Allow compositionality of meaning
    - *Maharani serves vegetarian food and is cheap.*

FOL Representation

- **Predicates:**
  - Relations among objects
    - *United serves Chicago.* ➔➔
    - *Serves(United, Chicago)*
    - *United is an airline.* ➔➔
    - *Airline(United)*
- **Logical connectives:**
  - Allow compositionality of meaning
    - *Frontier serves Seattle and is cheap.*
    - $Serves(Frontier, Seattle) \land Cheap(Frontier)$

Variables & Quantifiers

- Variables refer to:

Variables & Quantifiers

- Variables refer to:
  - Anonymous objects

Variables & Quantifiers

- Variables refer to:
  - Anonymous objects
  - All objects in some collection
- Quantifiers:

Variables & Quantifiers

- Variables refer to:
  - Anonymous objects
  - All objects in some collection
- Quantifiers:
  - $\exists$: existential quantifier: "there exists"
    - Indefinite NP, one such object for truth
    - A non-stop flight that serves Pittsburgh

$$\exists x Flight(x) \wedge Serves(x, Pittsburgh) \wedge Non-stop(x)$$

Variables & Quantifiers

- Variables refer to:
  - Anonymous objects
  - All objects in some collection
- Quantifiers:
  - $\exists$: existential quantifier: "there exists"
    - Indefinite NP, one such object for truth
    - A non-stop flight that serves Pittsburgh

$$\exists x Flight(x) \wedge Serves(x, Pittsburgh) \wedge Non-stop(x)$$

  - $\forall$: universal quantifier: "for all"
    - All flights include beverages.

$$\forall x Flight(x) \Rightarrow Includes(x, beverages)$$

FOL Syntax Summary

Fornzula	-----+	*AtomicFornu!a*
		Forrnula Connective Forrnula
		Quantifier Variable . .. Forrnula
		-, *For,nula*
		(Formula)
AtonzicForrnula	-----+	*Predicate(Terrn ...)*
Ternz	-----+	*Function(Ter,n ...)*
		C'onstant
		Variable
C'onnective	---+	*I* ǀ V ǀ ==>
Quantifier	-----+	*I* ǀ 3
:onstant	-----+	*A* ǀ *VegetarianFood* ǀ *lvlaharani* · · ·
Vririable	-----+	*x* ǀ *y* ǀ . . .
Predicate	-----+	*5erves* ǀ *Near* ǀ . . .
Function	-----+	*LocationOJ* ǀ *c·uisine(Jj* ǀ

Compositionality

- **Compositionality**: The meaning of a complex expression is a function of the meaning of its parts and the rules for their combination.
  - Formal languages are compositional.
  - Natural language meaning is largely, though not fully, compositional, but much more complex.
    - How can we derive things like loves(John, Mary) from John, loves(x,y), and Mary?

Lambda Expressions

- Lambda (λ) notation: (Church, 1940)
  - Just like lambda in Python, Scheme, etc
  - Allows abstraction over FOL formulas
    - Supports compositionality

Lambda Expressions

- Lambda (λ) notation: (Church, 1940)
  - Just like lambda in Python, Scheme, etc
  - Allows abstraction over FOL formulas
    - Supports compositionality

  - Form: λ + variable + FOL expression
    - E.g. $\lambda x.P(x)$ "Function taking x to P(x)"

Lambda Expressions

- Lambda (λ) notation: (Church, 1940)
  - Just like lambda in Python, Scheme, etc
  - Allows abstraction over FOL formulas
    - Supports compositionality

  - Form: λ + variable + FOL expression
    - E.g. $\lambda x.P(x)$ "Function taking x to P(x)"

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

λ-Reduction

- λ-reduction: Apply λ-expression to logical term
  - Binds formal parameter to term

$$\lambda x.P(x)$$

λ-Reduction

- λ-reduction: Apply λ-expression to logical term
  - Binds formal parameter to term

$$\lambda x.P(x)$$

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

λ-Reduction

- λ-reduction: Apply λ-expression to logical term
  - Binds formal parameter to term

$$\lambda x.P(x)$$

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

$$P(A)$$

- Equivalent to function application

Nested λ -Reduction

- Lambda expression as body of another

$$\lambda x.\lambda y.Near(x, y)$$

Nested λ-Reduction

- Lambda expression as body of another

$$\lambda x.\lambda y.Near(x, y)$$

$$\lambda x.\lambda y.Near(x, y)(Midway)$$

Nested λ-Reduction

- Lambda expression as body of another

$$\lambda x.\lambda y.Near(x, y)$$

$$\lambda x.\lambda y.Near(x, y)(Midway)$$

$$\lambda y.Near(Midway, y)$$

Nested λ -Reduction

- Lambda expression as body of another

$$\lambda x.\lambda y.Near(x, y)$$

$$\lambda x.\lambda y.Near(x, y)(Midway)$$

$$\lambda y.Near(Midway, y)$$

$$\lambda y.Near(Midway, y)(Chicago)$$

Nested λ-Reduction

- Lambda expression as body of another

$$\lambda x.\lambda y.Near(x, y)$$

$$\lambda x.\lambda y.Near(x, y)(Midway)$$

$$\lambda y.Near(Midway, y)$$

$$\lambda y.Near(Midway, y)(Chicago)$$

$$Near(Midway, Chicago)$$

Lambda Expressions

- Curying;
  - Converting multi-argument predicates to sequence of single argument predicates

  - Why?

Lambda Expressions

- Currying;
  - Converting multi-argument predicates to sequence of single argument predicates
  - Why?
    - Incrementally accumulates multiple arguments spread over different parts of parse tree

Semantics of Meaning Rep.

- Model-theoretic approach:
  - FOL terms (objects): denote elements in a domain
  - Atomic formulas are:
    - If properties, sets of domain elements
    - If relations, sets of tuples of elements
- Formulas based on logical operators:

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*

- Compositionality provided by lambda expressions

Inference

- Standard AI-type logical inference procedures
  - Modus Ponens
  - Forward-chaining, Backward Chaining
  - Abduction
  - Resolution
  - Etc,..
- We'll assume we have a prover

Representing Events

- Initially, single predicate with some arguments
  - Serves(United,Houston)
  - Assume # ags = # elements in subcategorization frame

Representing Events

- Initially, single predicate with some arguments
  - Serves(United,Houston)
  - Assume # ags = # elements in subcategorization frame
- Example:
  - The flight arrived.
  - The flight arrived in Seattle
  - The flight arrived in Seattle on Saturday.
  - The flight arrived on Saturday.
  - The flight arrived in Seattle from SFO.
  - The flight arrived in Seattle from SFO on Saturday.

Events

- Issues?

Events

- Issues?
  - Arity – how can we deal with different #s of arguments?

Neo-Davidsonian Events

- Neo-Davidsonian representation:
  - Distill event to single argument for event itself
  - Everything else is additional predication

$$\exists e Arriving(e) \wedge Arrived(e, Flight) \wedge Location(e, SEA) \wedge ArrivalDay(e, Saturday)$$

- Pros:

Neo-Davidsonian Events

- Neo-Davidsonian representation:
  - Distill event to single argument for event itself
  - Everything else is additional predication

$$\exists e Arriving(e) \wedge Arrived(e, Flight) \wedge Location(e, SEA) \wedge ArrivalDay(e, Saturday)$$

- Pros:
  - No fixed argument structure
    - Dynamically add predicates as necessary

Neo-Davidsonian Events

- Neo-Davidsonian representation:
  - Distill event to single argument for event itself
  - Everything else is additional predication

$$\exists e Arriving(e) \wedge Arrived(e, Flight) \wedge Location(e, SEA) \wedge ArrivalDay(e, Saturday)$$

- Pros:
  - No fixed argument structure
    - Dynamically add predicates as necessary
  - No extra roles

Neo-Davidsonian Events

- Neo-Davidsonian representation:
  - Distill event to single argument for event itself
  - Everything else is additional predication

$$\exists e Arriving(e) \wedge Arrived(e, Flight) \wedge Location(e, SEA) \wedge ArrivalDay(e, Saturday)$$

- Pros:
  - No fixed argument structure
    - Dynamically add predicates as necessary
  - No extra roles
  - Logical connections can be derived

Meaning Representation for Computational Semantics

- Requirements:
  - Verifiability, Unambiguous representation, Canonical Form, Inference, Variables, Expressiveness
- Solution:
  - First-Order Logic
    - Structure
    - Semantics
    - Event Representation
- Next: Semantic Analysis
  - Deriving a meaning representation for an input

Summary

- First-order logic can be used as a meaning representation language for natural language
- Principle of compositionality: the meaning of a complex expression is a function of the meaning of its parts
- λ-expressions can be used to compute meaning representations from syntactic trees based on the principle of compositionality
- In the next section, we will look at a syntax-driven approach to semantic analysis in more detail