Introduction to semantic parsing

Bill MacCartney

CS224U

14 May 2018

Full understanding?

- We're doing natural language understanding, right?
- Are we there yet? Do we fully understand?
  - With VSMs? Dependency parses? Relation extraction?
  - Arguably, all are steps toward NLU … but are they sufficient?
- How to generate complete, precise semantics?
  - *"Barack and Michelle Obama got married in 1992 in Chicago."*
  - *"No one may enter the building except policemen or firefighters."*

Logic games from LSAT (& old GRE)

Six sculptures — C, D, E, F, G, H — are to be exhibited in rooms 1, 2, and 3 of an art gallery.

- Sculptures C and E may not be exhibited in the same room.
- Sculptures D and G must be exhibited in the same room.
- If sculptures E and F are exhibited in the same room, no other sculpture may be exhibited in that room.
- At least one sculpture must be exhibited in each room, and no more than three sculptures may be exhibited in any room.

If sculpture D is exhibited in room 3 and sculptures E and F are exhibited in room 1, which of the following may be true?

A. Sculpture C is exhibited in room 1.
B. Sculpture H is exhibited in room 1.
C. Sculpture G is exhibited in room 2.
D. Sculptures C and H are exhibited in the same room.
E. Sculptures G and F are exhibited in the same room.

Travel reservations

Yes, hi, I need to book a flight for myself and my husband from SFO to Boston. Actually Oakland would be OK too. We need to fly out on Friday the 12th, and then I could come back on Sunday evening or Monday morning, but he won't return until Wednesday the 18th, because he's staying for business. No flights with more than one stop, and we don't want to fly on United because we hate their guts.

SHRDLU (Winograd 1972)

Find a block which is taller than the one you are holding and put it into the box.

OK.

How many blocks are not in the box?

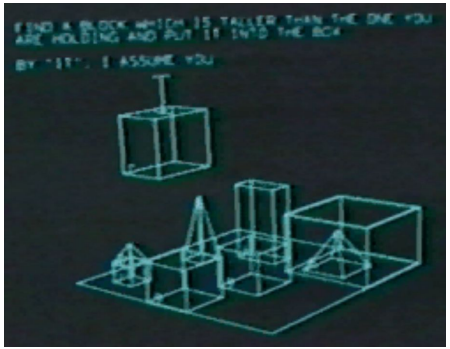
FOUR OF THEM.

Is at least one of them narrower than the one which I told you to pick up?

YES, THE RED CUBE.

http://youtube.com/watch?v=8SvD-lNg0TA

http://hci.stanford.edu/winograd/shrdlu/



CHAT-80

- Developed 1979-82 by Fernando Pereira & David Warren
- Proof-of-concept natural language interface to database
- Could answer questions about geography
- Implemented in Prolog
- Hand-built lexicon & grammar
- Highly influential NLIDB system

CHAT-80 demo

You can run Chat-80 yourself on the rice machines!

```
$ ssh rice.stanford.edu
$ cd /afs/ir/class/cs224u/chat-80/
$ module load sicstus
$ sicstus
? [load].
? hi.
? what is the capital of france?
```

Sample queries can be found at:

```
/afs/ir/class/cs224u/chat-80/demo
```

All the source code is there for your perusal as well.

Things you could ask CHAT-80

- Is there more than one country in each continent?
- What countries border Denmark?
- What are the countries from which a river flows into the Black_Sea?
- What is the total area of countries south of the Equator and not in Australasia?
- Which country bordering the Mediterranean borders a country that is bordered by a country whose population exceeds the population of India?
- How far is London from Paris?

The CHAT-80 database

```
% Facts about countries.
% country(Country, Region, Latitude, Longitude,
%   Area(sqmiles), Population, Capital, Currency)
country(andorra, southern_europe, 42, -1, 179, 25000,
andorra_la_villa, franc_peseta).
country(angola, southern_africa, -12, -18, 481351, 5810000, luanda,
?).
country(argentina, south_america, -35, 66, 1072067, 23920000,
buenos_aires, peso).

capital(C,Cap) :- country(C,_,_,_,_,_,Cap,_).
```

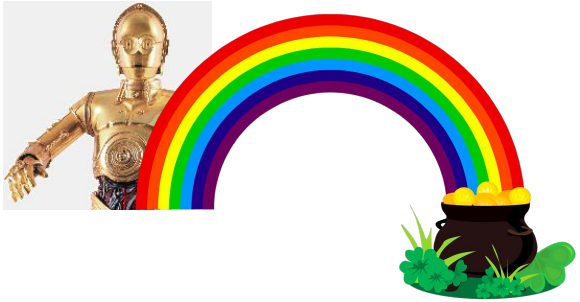
The CHAT-80 grammar

```
/* Sentences */
sentence(S) --> declarative(S), terminator(.) .
sentence(S) --> wh_question(S), terminator(?) .
sentence(S) --> yn_question(S), terminator(?) .
sentence(S) --> imperative(S), terminator(!) .

/* Noun Phrase */
np(np(Agmt,Pronoun,[]),Agmt,NPCase,def,_,Set,Nil) -->
  {is_pp(Set)},
  pers_pron(Pronoun,Agmt,Case),
  {empty(Nil), role(Case,decl,NPCase)}.

/* Prepositional Phrase */
pp(pp(Prep,Arg),Case,Set,Mask) -->
  prep(Prep),
  {prep_case(NPCase)},
  np(Arg,_,NPCase,_,Case,Set,Mask).
```

Precision vs. robustness

SHRDLU CHAT-80

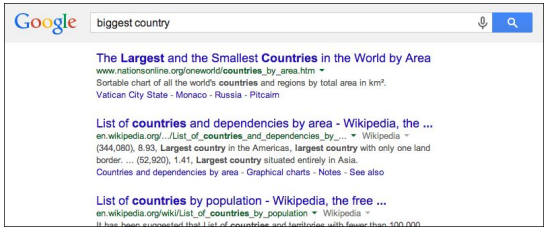
Precise, complete understanding



Brittle, narrow coverage

Robust, broad coverage

Fuzzy, partial understanding



Carbon emissions

Which country has the highest CO2 emissions?
What about highest per capita?
Which had the biggest increase over the last five years?
What fraction was from European countries?

Baseball statistics

Pitchers who have struck out four batters in one inning
Players who have stolen at least 100 bases in a season
Complete games with fewer than 90 pitches
Most home runs hit in one game

Voice commands

How do I get to the Ferry Building by bike
Book a table for four at Nopa on Friday after 9pm
Text my wife I’m going to be twenty minutes late
Add House of Cards to my Netflix queue at the top

Semantic parsing

If we want to understand natural language completely and precisely, we need to do semantic parsing.

That is, translate natural language into a formal meaning representation on which a machine can act.

First, we need to define our goal.

What should we choose as our target output representation of meaning?

Database queries

To facilitate data exploration and analysis, you might want to parse natural language into database queries:

which country had the highest carbon emissions last year

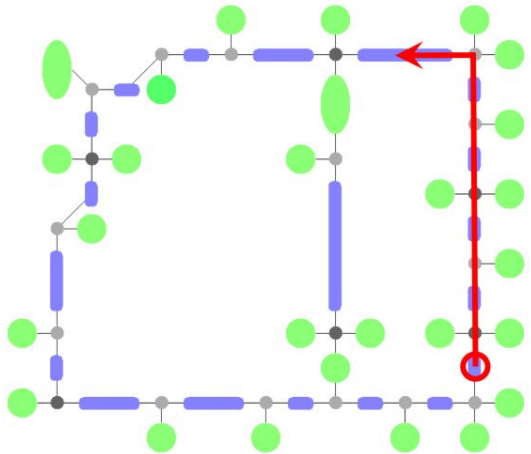
```
SELECT    country.name
FROM      country, co2_emissions
WHERE     country.id = co2_emissions.country_id
AND       co2_emissions.year = 2014
ORDER BY  co2_emissions.volume DESC
LIMIT     1;
```

Robot control

For a robot control application, you might want a custom-designed procedural language:

Go to the third junction and take a left.

```
(do-sequentially
  (do-n-times 3
    (do-sequentially
      (move-to forward-loc)
      (do-until
        (junction current-loc)
        (move-to forward-loc))))
  (turn-left))
```



Intents and arguments

For smartphone voice commands, you might want relatively simple meaning representations, with *intents* and *arguments*:

directions to SF by train

```
(TravelQuery
  (Destination /m/0d6lp)
  (Mode TRANSIT))
```

angelina jolie net worth

```
(FactoidQuery
  (Entity /m/0f4vbz)
  (Attribute /person/net_worth))
```

weather friday austin tx

```
(WeatherQuery
  (Location /m/0vzm)
  (Date 2013-12-13))
```

text my wife on my way

```
(SendMessage
  (Recipient 0x31cbf492)
  (MessageType SMS)
  (Subject "on my way"))
```

play sunny by boney m

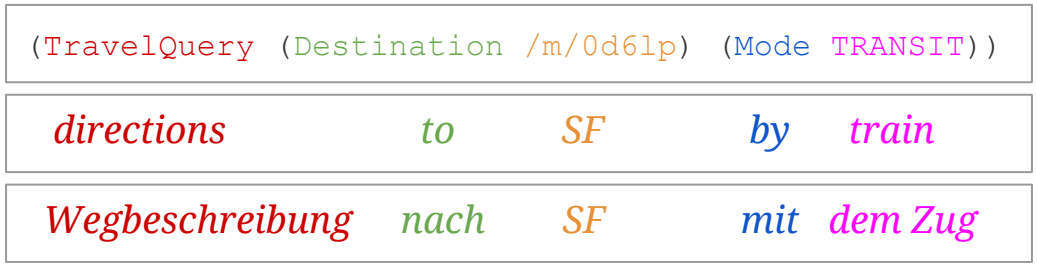
```
(PlayMedia
  (MediaType MUSIC)
  (SongTitle "sunny")
  (MusicArtist /m/017mh))
```

is REI open on sunday

```
(LocalQuery
  (QueryType OPENING_HOURS)
  (Location /m/02nx4d)
  (Date 2013-12-15))
```

Semantic parsing vs. machine translation

- Both involve translating from one semantic representation into another.
- Both involve complex structures, often related in complex ways.
- Some techniques are transferable: co-occurrence analysis, alignment, ...
- But in machine translation, the target semantic representation is *not* machine-readable! Rather, it is human-readable.



SippyCup

SippyCup is a simple semantic parser,
written in Python,
created purely for didactic purposes.

The design favors simplicity and readability
over efficiency and performance.

The goal is to make semantic parsing look easy!

SippyCup units

Unit 0	Introduction to semantic parsing
Unit 1	Natural language arithmetic “two times three plus four”
Unit 2	Travel queries “driving directions to williamsburg virginia”
Unit 3	Geographical queries “how many states border the largest state”

Outline of an approach

- Similar to much recent academic work in semantic parsing
  - Luke Zettlemoyer, Percy Liang, and many others
- Define possible syntactic structures using a context-free grammar
- Construct semantics bottom-up, following syntactic structure
- Score parses with a log-linear model learned from training data
- Leverage annotators for names, numbers, places, times, …
- Use grammar induction to avoid manual grammar engineering

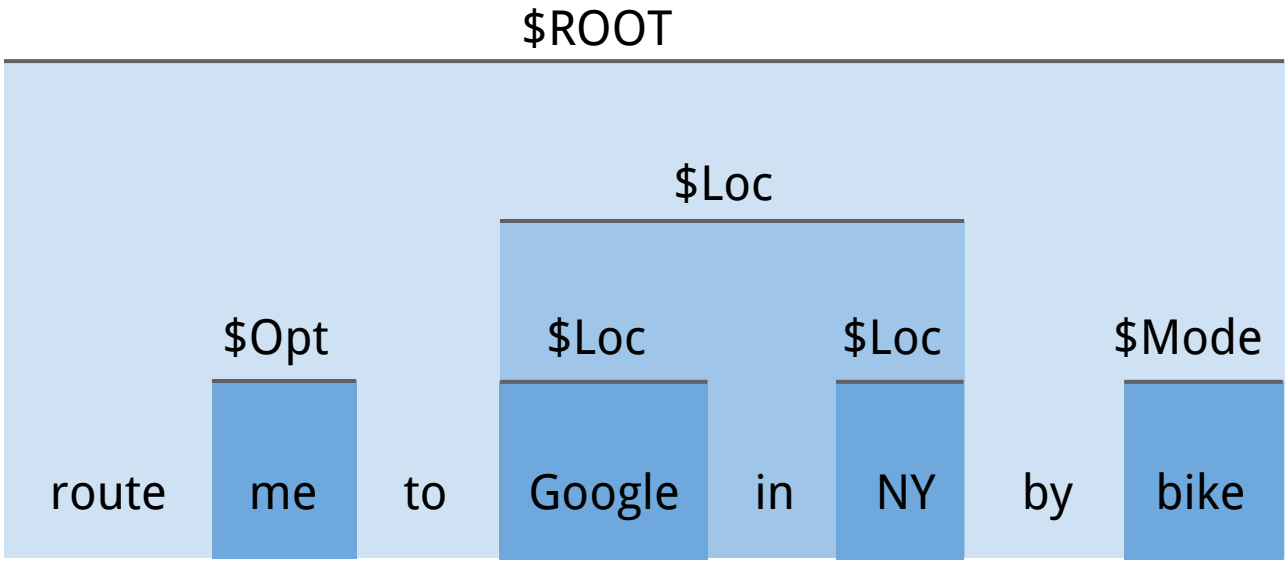
Context-free grammar

The *syntactic* part of the grammar is a fairly conventional CFG:

```
$Loc → Google
$Loc → NY
$Loc → $Loc in $Loc
$Opt → me
$Mode → bike
$Mode → car
$ROOT → route ($Opt)? to $Loc by $Mode
```

Usually *not* deterministic: many possible derivations per input.

Example parse



Parsing algorithm

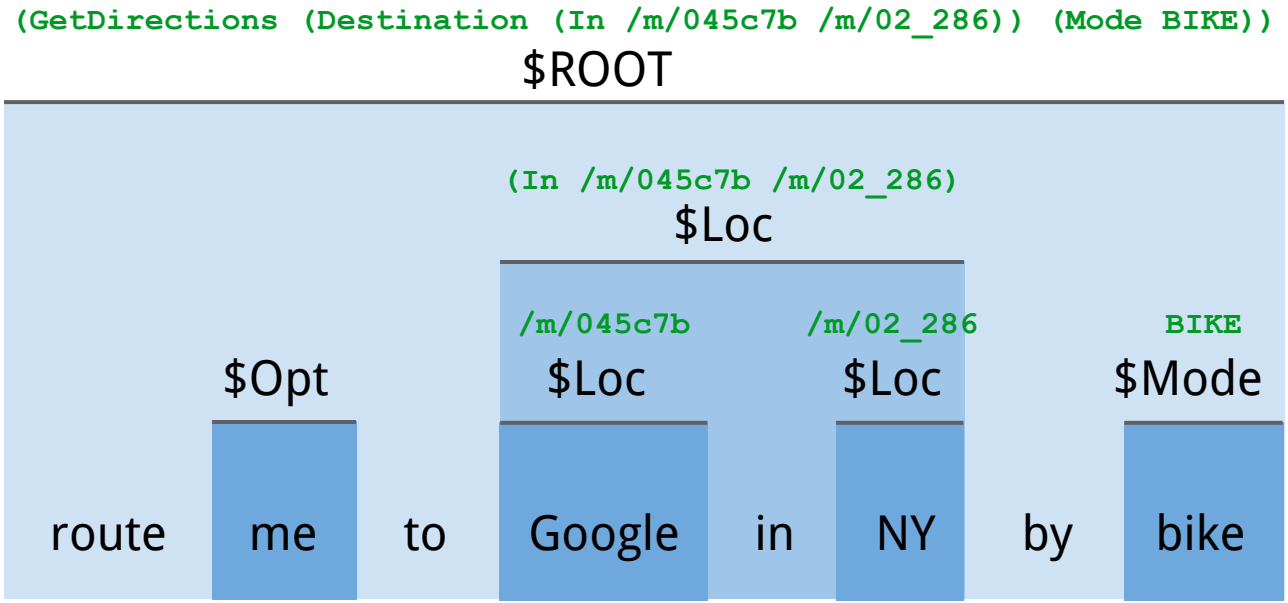
- An adaptation of the CYK chart parsing algorithm
  - (A nice example of dynamic programming)
- Rewrite the grammar so that all rules are binary (or unary)
- Consider every span of tokens in input, bottom-up
- Consider all ways of splitting span into two sub-spans
- Consider all grammar rules whose RHSes match the sub-spans
- Record the categories (LHSes) we can construct for the span

Semantic attachments to CFG rules

Little programs which compute semantic interpretation bottom-up

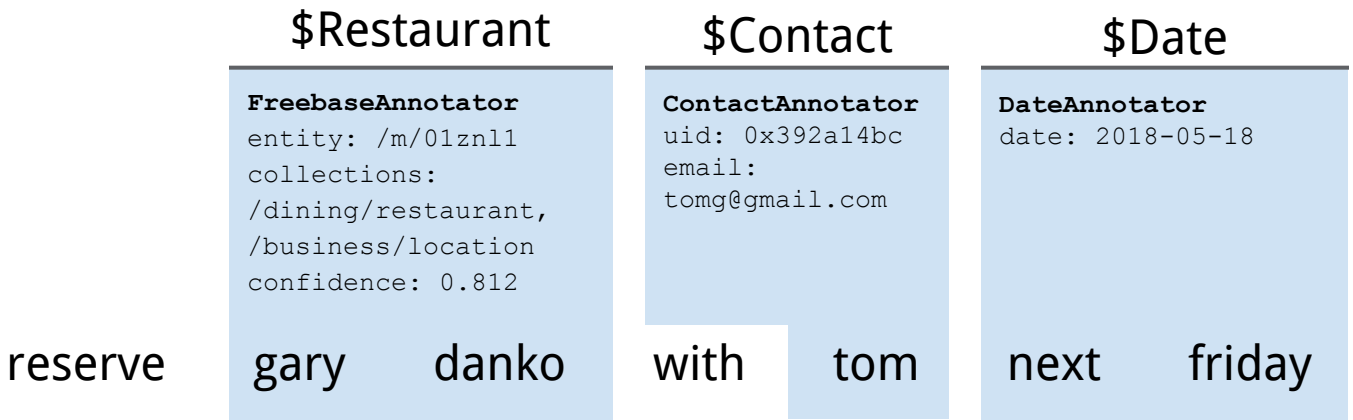
```
$Loc → Google [/m/045c7b]
$Loc → NY [/m/02_286]
$Loc → $Loc in $Loc [(In $1 $2)]
$Opt → me []
$Mode → bike [BIKE]
$Mode → car [CAR]
$ROOT → route ($Opt)? to $Loc by $Mode
  [(GetDirections (Destination $2) (Mode $3))]
```

Example parse, now with semantics!



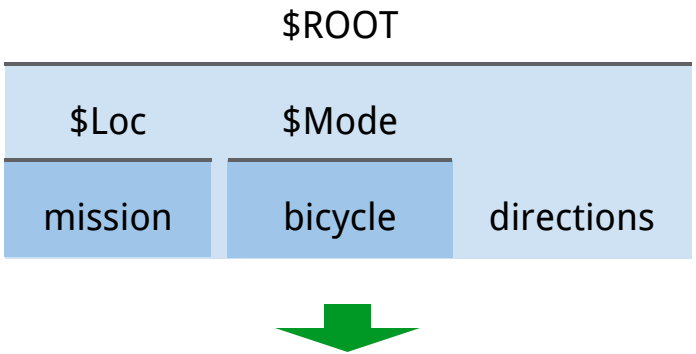
Annotators

- Don’t want a million rules like: **`$Loc → NY [/m/02_286]`**
- Instead, leverage intelligence of special-purpose annotators

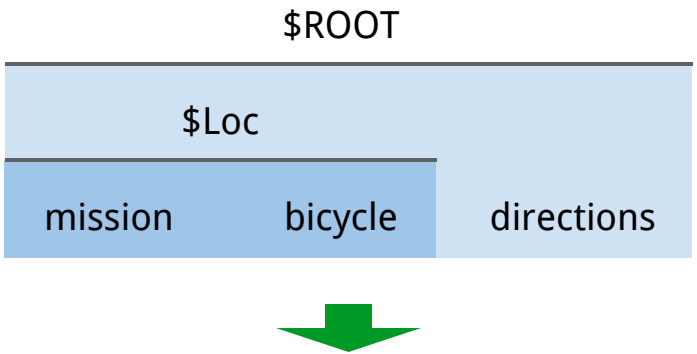


Ambiguity

When grammar supports multiple interpretations, how to choose?




```
(GetDirections
  (Destination /m/02ll7r)
  (Mode BIKE))
```




```
(GetDirections
  (Destination /g/1tfmcgjy))
```

Scoring model

- A log-linear model to score alternative derivations (parses)
- Features from input *x*, semantic yield *y*, and derivation *z*
  - E.g., co-occurrence of *"to"* in input and `Destination` in semantics
  - E.g., occurrence of specific CFG rules or categories in derivation
  - E.g., confidence score from an annotator

$$score(x, z) = features(x, z)^\top \theta$$

$$p(z|x, \theta) = \frac{e^{score(x,z)}}{\sum_{z' \in Z(x)} e^{score(x,z')}}$$

Learning

- Estimate parameters using EM-style training (Liang et al., 2011)
- Assume we have training data $(x_1, y_1), \ldots, (x_n, y_n)$
- Sum out latent derivations in n-best list

$$\operatorname*{argmax}_{\theta} \sum_{z} p(y|z)\, p(z|x, \theta)$$

- Update using Stochastic Gradient Descent (SGD)

Grammar induction

- Next: where do the grammar rules come from?
- For small domains, we can write the rules manually
- But for large, complex domains, this doesn't scale!
- Want to *induce* rules automatically from training data
- One strategy: reduce grammar induction to learning
  - Generate *all* possible rules (given all possible words, semantics)
  - Then use standard learning methods to train rule weights
  - But, exponential blowup ⇒ requires pruning strategies

Data

- Of course, the whole enterprise depends upon having data
- Ideally, *lots* of data — as usual, more data trumps cleverer models
- Cf. Halevy, Norvig, & Pereira (2009),
  “The Unreasonable Effectiveness of Data”
- Problem: examples annotated with semantics are expensive
- Solution: learn from indirect supervision, using the results of *evaluating* the semantics (e.g., learning from denotations)

Recap

We map queries into structured representations of meaning using:

- Context-free grammars with semantic attachments
- Log-linear scoring models, with weights learned from data
- Leverage annotators for entities, contacts, dates, …
- Methods for inducing grammar rules from data
- Lots and lots and lots of data!