Learning compositional semantic theories

Christopher Potts

CS 244U: Natural language understanding
May 5

Plan

1. Review of learning to map to logical forms
2. Discussion of learning from denotations

Related materials

Readings

- Liang, Percy and Christopher Potts. 2014. Bringing machine learning and compositional semantics together.
- Liang, Percy; Michael I. Jordan; and Dan Klein. 2013. Learning dependency-based compositional semantics. *Computational Linguistics* 39(2): 389-446.

Code

- SEMPRE: Semantic Parsing with Execution
- UW Semantic Parsing Framework

Data

- Geoquery, Jobsquery, Restaurant Query
- Abstract Meaning Representation Bank
- WebQuestions and Free917
- CCGBank (Penn Treebank in CCG; syntax only)

Linguistic objects

$$\langle u, t, r, d \rangle$$

- u: the utterance (sequence of strings/words)
- t: the syntactic structure (tree structure)
- r: the semantic representation (a.k.a. logical form)
- d: the denotation (meaning)

(The denotation might under-represent or mis-represent the speaker's intended message. We'll return to that issue in the context of pragmatics.)

Example interpreted grammar

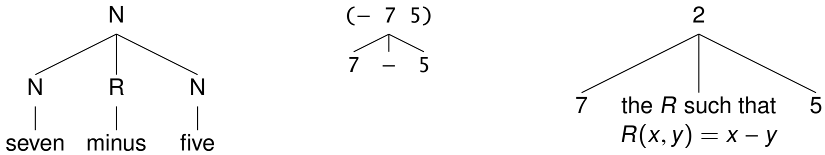
Syntax	Logical form	Denotation
N → one	1	1
N → two	2	2
$\vdots$	$\vdots$	$\vdots$
R → plus	$+$	the R such that $R(x, y) = x + y$
R → minus	$-$	the R such that $R(x, y) = x - y$
R → times	$\times$	the R such that $R(x, y) = x * y$
S → minus	$\neg$	the f such that $f(x) = -x$
N → S N	$\ulcorner \mathrm{S} \urcorner \ulcorner \mathrm{N} \urcorner$	$[\![\ulcorner \mathrm{S} \urcorner]\!]([\![\ulcorner \mathrm{N} \urcorner]\!])$
N → N_L R N_R	$(\ulcorner \mathrm{R} \urcorner \ulcorner \mathrm{N}_L \urcorner \ulcorner \mathrm{N}_R \urcorner)$	$[\![\ulcorner \mathrm{R} \urcorner]\!]([\![\ulcorner \mathrm{N}_L \urcorner]\!], [\![\ulcorner \mathrm{N}_R \urcorner]\!])$

Table: An illustrative grammar. $\ulcorner u \urcorner$ is the translation of syntactic expression u, and $[\![r]\!]$ is the denotation of semantic representation r. N is the CFG's start symbol. In the final rule, the L and R subscripts are meta-annotations to ensure deterministic translation and interpretation.

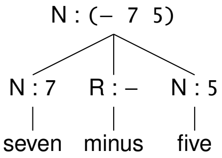
Examples

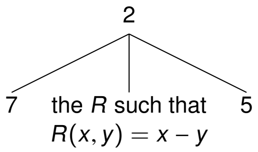
	Syntax	Logical form	Denotation
A.	seven minus five	`(− 7 5)`	2
B.	minus three plus one	`(+ ¬3 1)`	−2
C.	two minus two times two	`(× (− 2 2) 2)`	0
D.	two plus three plus four	`(+ 2 (+ 3 4))`	9

Examples

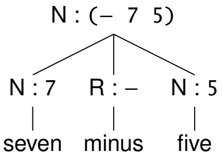


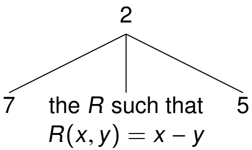
Examples

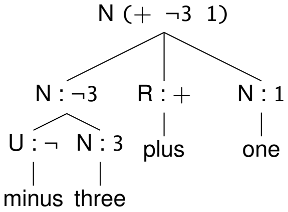


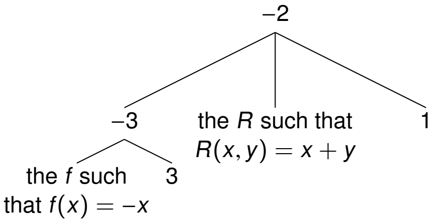


Examples


N : (− 7 5)
N : 7
R : −
N : 5
seven
minus
five

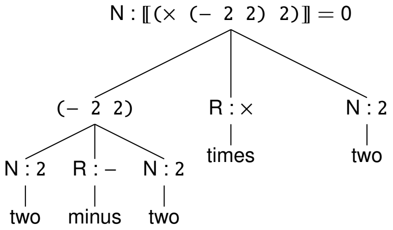

2
7
the R such that $R(x, y) = x - y$
5


N (+ ¬3 1)
N : ¬3
R : +
N : 1
U : ¬
N : 3
plus
one
minus
three

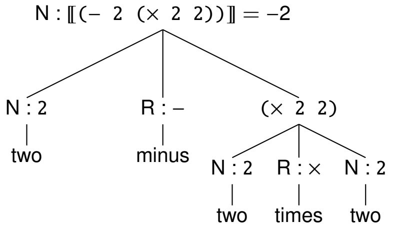

−2
−3
the R such that $R(x, y) = x + y$
1
the f such that $f(x) = -x$
3

Parsing and ambiguity

The grammar determines the candidate space; dynamic programming algorithms efficiently map us to that space.



$\textsc{Gen}(\textit{two minus two times two}) =$



Direct implementations

- Prominent recent examples: Bos 2005; Bos and Markert 2005
- Excel at inference (via theorem provers).
- Tend to be high precision, low recall — the analyst must anticipate every lexical item and every constructional quirk.

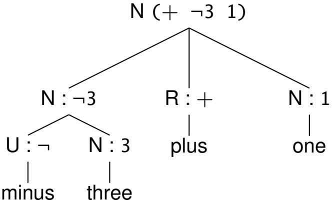
```
sem(7,

 [word(7001,'Mubarak'),word(7002,reviewed),word(7003,the),word(7004,blueprints),word(7005,for),word(7006,a),
  word(7007,number),word(7008,of),word(7009,other),word(7010,huge),word(7011,national),word(7012,projects),
  word(7013,','),word(7014,known),word(7015,as),word(7016,'Egypts'),word(7017,'21st'),word(7018,century),
  word(7019,project),word(7020,'.')],

 [pos(7001,'NNP'),pos(7002,'VBN'),pos(7003,'DT'),pos(7004,'NNS'),pos(7005,'IN'),pos(7006,'DT'),
  pos(7007,'NN'),pos(7008,'IN'),pos(7009,'JJ'),pos(7010,'JJ'),pos(7011,'JJ'),pos(7012,'NNS'),pos(7013,','),
  pos(7014,'VBN'),pos(7015,'IN'),pos(7016,'NNS'),pos(7017,'JJ'),pos(7018,'NN'),pos(7019,'NN'),pos(7021,'.')],

 alfa(nam,drs([7001:A],[7001:pred('Mubarak',[A]),7001:ne(A,'I-PER')]),
          alfa(def,drs([7003:B],[7004:pred(blueprint,[B])]),
                   merge(drs([7006:C],[7007:pred(number,[C])]),
                         merge(merge(drs([7009:D],[]),
                                     alfa(def,drs([0:E],[7010:pred(huge,[E]),7011:pred(national,[E]),
                                                         7012:pred(project,[E])]),
                                              drs([],[7009:not(drs([],[0:eq(D,E)])),
                                                      7010:pred(huge,[D]),7011:pred(national,[D]),
                                                      7012:pred(project,[D])]))),
                               drs([7014:F,7016:G,7002:H],[7008:pred(of,[C,D]),7014:pred(know,[F]),
                                                          7014:pred(patient,[F,C]),7016:pred(egypt,[G]),
                                                          7017:pred('21st',[G]),7017:ne(G,'I-DAT'),
                                                          7018:pred(century,[G]),7018:ne(G,'I-DAT'),
                                                          7019:pred(project,[G]),7015:pred(as,[F,G]),
                                                          7005:pred(for,[B,C]),7002:pred(review,[H]),
                                                          7002:pred(agent,[H,A]),7002:pred(patient,[H,B])])))))).
```

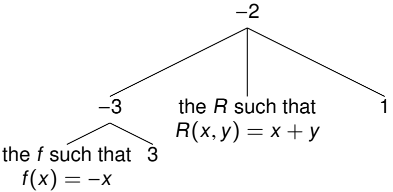
Figure: Prolog representation from Bos 2005: *Mubarak reviewed the blueprints for a number of other huge national projects, known as Egypt's 21st century project.*

Compositionality

Compositionality

The meaning of a phrase is a function of the meanings of its immediate syntactic constituents and the way they are combined.

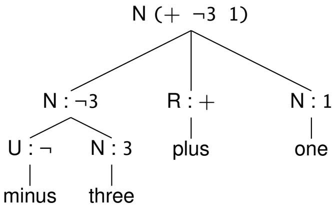


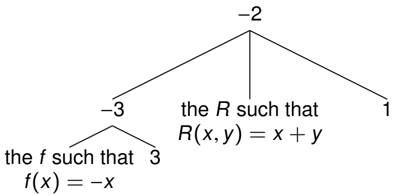


Compositionality

Compositionality

The meaning of a phrase is a function of the meanings of its immediate syntactic constituents and the way they are combined.





Liang and Potts (2014)

"the claim of compositionality is that being a semantic interpreter for a language L amounts to mastering the syntax of L, the lexical meanings of L, and the modes of semantic combination for L. This also suggests the outlines of a learning task."

Learning tasks

The grammar frames the task; different parts of it can be learned.

Syntax	Logical form	Denotation
N $\rightarrow$ one	1	1
N $\rightarrow$ two	2	2
$\vdots$	$\vdots$	$\vdots$
R $\rightarrow$ plus	$+$	the R such that $R(x, y) = x + y$
R $\rightarrow$ minus	$-$	the R such that $R(x, y) = x - y$
R $\rightarrow$ times	$\times$	the R such that $R(x, y) = x * y$
S $\rightarrow$ minus	$\neg$	the f such that $f(x) = -x$
N $\rightarrow$ S N	$\ulcorner\mathrm{S}\urcorner\ulcorner\mathrm{N}\urcorner$	$[\![\ulcorner\mathrm{S}\urcorner]\!]([\![\ulcorner\mathrm{N}\urcorner]\!])$
N $\rightarrow$ N_L R N_R	$(\ulcorner\mathrm{R}\urcorner\ \ulcorner\mathrm{N}_L\urcorner\ \ulcorner\mathrm{N}_R\urcorner)$	$[\![\ulcorner\mathrm{R}\urcorner]\!]([\![\ulcorner\mathrm{N}_L\urcorner]\!], [\![\ulcorner\mathrm{N}_R\urcorner]\!])$

- Parsing
- Semantic parsing
- Interpretive

Learning tasks

The grammar frames the task; different parts of it can be learned.

Syntax	Logical form	Denotation
N → one	1	1
N → two	2	2
⋮	⋮	⋮
R → plus	$+$	the R such that $R(x, y) = x + y$
R → minus	$-$	the R such that $R(x, y) = x - y$
R → times	$\times$	the R such that $R(x, y) = x * y$
S → minus	$\neg$	the f such that $f(x) = -x$
N → S N	$\ulcorner \mathrm{S} \urcorner \ulcorner \mathrm{N} \urcorner$	$[\![\ulcorner \mathrm{S} \urcorner]\!]([\![\ulcorner \mathrm{N} \urcorner]\!])$
N → N_L R N_R	$(\ulcorner \mathrm{R} \urcorner \ulcorner \mathrm{N}_L \urcorner \ulcorner \mathrm{N}_R \urcorner)$	$[\![\ulcorner \mathrm{R} \urcorner]\!]([\![\ulcorner \mathrm{N}_L \urcorner]\!], [\![\ulcorner \mathrm{N}_R \urcorner]\!])$

- Parsing
- Semantic parsing
- Interpretive

Learning tasks

The grammar frames the task; different parts of it can be learned.

Syntax	Logical form	Denotation
N → one	1	1
N → two	2	2
⋮	⋮	⋮
R → plus	$+$	the R such that $R(x, y) = x + y$
R → minus	$-$	the R such that $R(x, y) = x - y$
R → times	$\times$	the R such that $R(x, y) = x * y$
S → minus	$\neg$	the f such that $f(x) = -x$
N → S N	$\ulcorner S \urcorner \ulcorner N \urcorner$	$[\![\ulcorner S \urcorner]\!]([\![\ulcorner N \urcorner]\!])$
N → N_L R N_R	$(\ulcorner R \urcorner\ \ulcorner N_L \urcorner\ \ulcorner N_R \urcorner)$	$[\![\ulcorner R \urcorner]\!]([\![\ulcorner N_L \urcorner]\!], [\![\ulcorner N_R \urcorner]\!])$

- Parsing
- Semantic parsing
- Interpretive

Learning tasks

The grammar frames the task; different parts of it can be learned.

Syntax	Logical form	Denotation
N $\rightarrow$ one	1	1
N $\rightarrow$ two	2	2
$\vdots$	$\vdots$	$\vdots$
R $\rightarrow$ plus	$+$	the R such that $R(x, y) = x + y$
R $\rightarrow$ minus	$-$	the R such that $R(x, y) = x - y$
R $\rightarrow$ times	$\times$	the R such that $R(x, y) = x * y$
S $\rightarrow$ minus	$\neg$	the f such that $f(x) = -x$
N $\rightarrow$ S N	$\ulcorner\mathrm{S}\urcorner\ulcorner\mathrm{N}\urcorner$	$[\![\ulcorner\mathrm{S}\urcorner]\!]([\![\ulcorner\mathrm{N}\urcorner]\!])$
N $\rightarrow$ N_L R N_R	$(\ulcorner\mathrm{R}\urcorner\ \ulcorner\mathrm{N}_L\urcorner\ \ulcorner\mathrm{N}_R\urcorner)$	$[\![\ulcorner\mathrm{R}\urcorner]\!]([\![\ulcorner\mathrm{N}_L\urcorner]\!], [\![\ulcorner\mathrm{N}_R\urcorner]\!])$

- Parsing
- Semantic parsing
- Interpretive

Learning tasks

The grammar frames the task; different parts of it can be learned.

Syntax	Logical form	Denotation
N → one	1	1
N → two	2	2
⋮	⋮	⋮
R → plus	$+$	the R such that $R(x, y) = x + y$
R → minus	$-$	the R such that $R(x, y) = x - y$
R → times	$\times$	the R such that $R(x, y) = x * y$
S → minus	$\neg$	the f such that $f(x) = -x$
N → S N	$\ulcorner S \urcorner \ulcorner N \urcorner$	$[\![\ulcorner S \urcorner]\!]([\![\ulcorner N \urcorner]\!])$
N → N_L R N_R	$(\ulcorner R \urcorner\ \ulcorner N_L \urcorner\ \ulcorner N_R \urcorner)$	$[\![\ulcorner R \urcorner]\!]([\![\ulcorner N_L \urcorner]\!], [\![\ulcorner N_R \urcorner]\!])$

- Parsing
- Semantic parsing
- Interpretive

Learning tasks

The grammar frames the task; different parts of it can be learned.

Syntax	Logical form	Denotation
N $\rightarrow$ one	1	1
N $\rightarrow$ two	2	2
$\vdots$	$\vdots$	$\vdots$
R $\rightarrow$ plus	$+$	the R such that $R(x, y) = x + y$
R $\rightarrow$ minus	$-$	the R such that $R(x, y) = x - y$
R $\rightarrow$ times	$\times$	the R such that $R(x, y) = x * y$
S $\rightarrow$ minus	$\neg$	the f such that $f(x) = -x$
N $\rightarrow$ S N	$\ulcorner$S$\urcorner\ulcorner$N$\urcorner$	$[\![\ulcorner S\urcorner]\!]([\![\ulcorner N\urcorner]\!])$
N $\rightarrow$ N_L R N_R	$(\ulcorner R\urcorner\ \ulcorner N_L\urcorner\ \ulcorner N_R\urcorner)$	$[\![\ulcorner R\urcorner]\!]([\![\ulcorner N_L\urcorner]\!], [\![\ulcorner N_R\urcorner]\!])$

- Parsing
- Semantic parsing
- Interpretive

Semantic parsing

$$\langle \boxed{u, t, r}, d \rangle$$

Pioneering work

- Logical: Woods et al. 1972; Warren and Pereira 1982
- Statistical: Zelle and Mooney 1996; Tang and Mooney 2001; Thompson and Mooney 2003; Zettlemoyer and Collins 2005

Basic formulation

	Utterance	Logical form
Train	seven minus five	`(− 7 5)`
	five minus seven	`(− 5 7)`
	three plus one	`(− 7 5)`
	minus three plus one	`(+ ¬3 1)`
	minus three plus one	`¬(+ 3 1)`
	two minus two times two	`(× (− 2 2) 2)`
	two minus two times two	`(− 2 (× 2 2))`
	two plus three plus four	`(+ 2 (+ 3 4))`
	⋮	
Test	three minus one	?
	three times one	?
	minus six times four	?
	one plus three plus five	?
	⋮	

Table: Data requirements.

Syntax	Logical form
N → one	1
N → one	2
	⋮
N → two	1
N → two	2
	⋮
R → plus	$+$
R → plus	$-$
R → plus	$\times$
R → minus	$+$
R → minus	$-$
R → minus	$\times$
R → times	$+$
R → times	$-$
R → times	$\times$
S → minus	$\neg$
N → S N	$\ulcorner S \urcorner \ulcorner N \urcorner$
N → N_L R N_R	$(\ulcorner R \urcorner \ulcorner N_L \urcorner \ulcorner N_R \urcorner)$

Table: Crude grammar.

Learning framework

1. Feature representations: $\phi(x, y) \in \mathbb{R}^d$
2. Scoring: $\text{Score}_{\mathbf{w}}(x, y) = \mathbf{w} \cdot \phi(x, y) = \sum_{j=1}^{d} w_j \phi(x, y)_j$
3. Multiclass hinge-loss objective function:

$$\min_{\mathbf{w} \in \mathbb{R}^d} \sum_{(x,y) \in \mathcal{D}} \max_{y' \in \textsc{Gen}(x)} [\text{Score}_{\mathbf{w}}(x, y') + c(y, y')] - \text{Score}_{\mathbf{w}}(x, y)$$

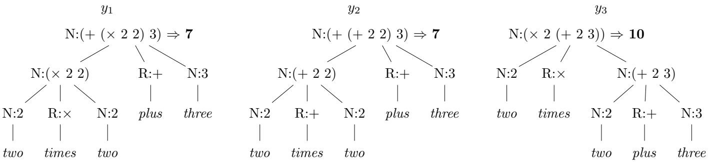
   where $\mathcal{D}$ is a set of (x, y) training examples and $c(a, b) = 1$ if $a \neq b$, else 0.

4. Optimization:
   $\textsc{StochasticGradientDescent}(\mathcal{D}, T, \eta)$

```
Initialize w ← 0
Repeat T times
    for each (x, y) ∈ D (in random order)
        ỹ ← arg max_{y′∈Gen(x)} Score_w(x, y′) + c(y, y′)
        w ← w + η(φ(x, y) − φ(x, ỹ))
Return w
```

Example

(a) Candidates GEN(x) for utterance $x =$ *two times two plus three*



$\phi(x, y_1) = \boxed{\begin{array}{r} \text{R:}\times[\textit{times}]:\ 1 \\ \text{R:+}[\textit{plus}]:\ 1 \\ \text{top[R:+]}:\ 1 \end{array}}$ $\phi(x, y_2) = \boxed{\begin{array}{r} \text{R:+}[\textit{times}]:\ 1 \\ \text{R:+}[\textit{plus}]:\ 1 \\ \text{top[R:+]}:\ 1 \end{array}}$ $\phi(x, y_3) = \boxed{\begin{array}{r} \text{R:}\times[\textit{times}]:\ 1 \\ \text{R:+}[\textit{plus}]:\ 1 \\ \text{top[R:}\times\text{]}:\ 1 \end{array}}$

(b) Learning from logical forms (Section 4.1)

Iteration 1

$\mathbf{w} = \boxed{\begin{array}{r} \text{R:}\times[\textit{times}]:0 \\ \text{R:+}[\textit{times}]:0 \\ \text{R:+}[\textit{plus}]:0 \\ \text{top[R:+]}:0 \\ \text{top[R:}\times\text{]}:0 \end{array}}$ Scores: $[0, 0, 0]$, $y = y_1$, $\tilde{y} = y_3$ (tied with y_2) $\Rightarrow$

Iteration 2

$\mathbf{w} = \boxed{\begin{array}{r} \text{R:}\times[\textit{times}]:0 \\ \text{R:+}[\textit{times}]:0 \\ \text{R:+}[\textit{plus}]:0 \\ \text{top[R:+]}:1 \\ \text{top[R:}\times\text{]}:\text{-}1 \end{array}}$ Scores: $[1, 1, -1]$, $y = y_1$, $\tilde{y} = y_2$ (tied with y_1) $\Rightarrow$

Iteration 3

$\mathbf{w} = \boxed{\begin{array}{r} \text{R:}\times[\textit{times}]:1 \\ \text{R:+}[\textit{times}]:\text{-}1 \\ \text{R:+}[\textit{plus}]:0 \\ \text{top[R:+]}:1 \\ \text{top[R:}\times\text{]}:\text{-}1 \end{array}}$ Scores: $[2, 0, 0]$, $y = y_1$, $\tilde{y} = y_1$

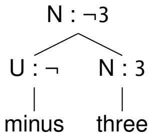
Derivational ambiguity

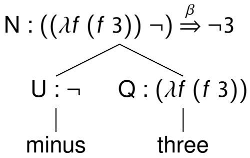
In the rich grammars of Zettlemoyer and Collins (2005, 2007) and others, a given logical expression might have multiple derivations.

Syntax	Logical form
N $\rightarrow$ one	1
N $\rightarrow$ two	2
	$\vdots$
R $\rightarrow$ plus	$+$
R $\rightarrow$ minus	$-$
R $\rightarrow$ times	$\times$
S $\rightarrow$ minus	$\neg$
N $\rightarrow$ S N	$\ulcorner S \urcorner \ulcorner N \urcorner$
N $\rightarrow$ N_L R N_R	$(\ulcorner R \urcorner\ \ulcorner N_L \urcorner\ \ulcorner N_R \urcorner)$
Q $\rightarrow$ n	$(\lambda f\ (f\ \ulcorner n \urcorner))$
N $\rightarrow$ U Q	$(\ulcorner Q \urcorner\ \ulcorner U \urcorner)$

Table: Grammar with type-lifting.

Training instance: (*minus three*, $\neg 3$)





(Beta-conversion $\stackrel{\beta}{\Rightarrow}$ is the syntactic counterpart of functional application.)

Derivations as latent variables

- The training instances are (u, r) pairs.
- Since r might have multiple derivations, derivations are latent variables.
- Zettlemoyer and Collins (2005, 2007) use log-linear latent variable models, but our earlier framework can accommodate them as well.
- Latent support vector machine objective:

$$\min_{\mathbf{w}\in\mathbb{R}^d} \sum_{(x,r)\in\mathcal{D}} \max_{y'\in\textsc{Gen}(x)} \left[\mathrm{Score}_{\mathbf{w}}(x, y') + c(r, \textsc{Root}(y'))\right] - \max_{y''\in\textsc{Gen}(x,r)} \mathrm{Score}_{\mathbf{w}}(x, y''),$$

where $\mathcal{D}$ is a set of (utterance, formula) pairs; $c(a, b) = 1$ if $a \neq b$, else 0; and $\textsc{Gen}(x, r) = \{y \in \textsc{Gen}(x) : \textsc{Root}(y) = r\}$

- Optimization:

$\textsc{StochasticGradientDescent}(\mathcal{D}, T, \eta)$

Initialize $\mathbf{w} \leftarrow \mathbf{0}$
Repeat T times
    for each $(x, r) \in \mathcal{D}$ (in random order)
        $y \leftarrow \arg\max_{y''\in\textsc{Gen}(x,r)} \mathrm{Score}_{\mathbf{w}}(x, y'')$
        $\tilde{y} \leftarrow \arg\max_{y'\in\textsc{Gen}(x)} \mathrm{Score}_{\mathbf{w}}(x, y') + c(y, y')$
        $\mathbf{w} \leftarrow \mathbf{w} + \eta(\phi(x, y) - \phi(x, \tilde{y}))$
Return $\mathbf{w}$

Taming the search space

The complexity issues trace to the fact that the size of $\textsc{Gen}(x)$ is expontential in the length of x.

- Variants of CKY parsing algorithms that track both syntactic and semantic information (Zettlemoyer 2009:Appendix A).
- Assume parts of the lexicon are known (function words, easily specified open-class items).
- Prune the lexicon during training, thereby keeping it small, thereby keeping $\textsc{Gen}(x)$ small (Zettlemoyer and Collins 2005).

High-level look at results

Paper	Recall (LFs)	Recall (Answers)
Zettlemoyer and Collins (2005)	79.3	–
Zettlemoyer and Collins (2007)	81.6	–
Kwiatkowksi et al. (2010)	88.2	–
Kwiatkowski et al. (2011)	88.6	–

Table: Results for the Geo880 test set (Zelle and Mooney 1996). For a fuller summary, see Liang et al. 2013:435.

Recent developments and extensions

- Zettlemoyer and Collins (2007): grapping with messy data (ATIS travel-planning)
- Artzi and Zettlemoyer (2011): bootstrapping from machine-generated dialog systems
- Kwiatkowksi et al. (2010): learning (weights on) the modes of composition
- Matuszek et al. (2012b): mapping to a robot controller language
- Kwiatkowksi et al. (2010); Kwiatkowski et al. (2011): multilingual semantic parsing
- Cai and Yates (2013): question-answering with Freebase

Learning from denotations

$$\langle \boxed{u, t, r, d} \rangle$$

Pioneering work

- Psychological: see Frank et al. 2009 for models and references
- NLP: Clarke et al. (2010); Liang et al. (2011, 2013)

Motivations

Detailed Supervision
- doesn't scale up
- representation-dependent

What is the largest city in California?

$\Big\downarrow$ **expert**

$\texttt{argmax}(\{c : \texttt{city}(c) \wedge \texttt{loc}(c, \texttt{CA})\}, \texttt{population})$

Natural Supervision
- scales up
- representation-independent

What is the largest city in California?

$\Big\downarrow$ **non-expert**

Los Angeles

(Slide from Percy Liang)

Basic formulation

	Utterance	Denotation
Train	seven minus five	2
	five minus seven	−2
	three plus one	4
	minus three plus one	−2
	minus three plus one	−4
	two minus two times two	0
	two minus two times two	−2
	two plus three plus four	9
	⋮	
Test	three minus one	?
	three times one	?
	minus six times four	?
	one plus three plus five	?
	⋮	

Table: Data requirements.

Syntax	Logical form	Denotation
N → one	1	1
N → one	2	2
	⋮	
N → two	1	1
N → two	2	2
	⋮	
R → plus	$+$	addition
R → plus	$-$	subtraction
R → plus	$\times$	multiplication
R → minus	$+$	addition
R → minus	$-$	subtraction
R → minus	$\times$	multiplication
R → times	$+$	addition
R → times	$-$	subtraction
R → times	$\times$	multiplication
S → minus	$\neg$	negative
N → S N	$\ulcorner S \urcorner \ulcorner N \urcorner$	$[\![\ulcorner S \urcorner]\!]([\![\ulcorner N \urcorner]\!])$
N → N_L R N_R	$(\ulcorner R \urcorner\ \ulcorner N_L \urcorner\ \ulcorner N_R \urcorner)$	$[\![\ulcorner R \urcorner]\!]([\![\ulcorner N_L \urcorner]\!], [\![\ulcorner N_R \urcorner]\!])$

Table: Crude grammar.

Learning framework

1. Feature representations: $\phi(x, y) \in \mathbb{R}^d$
2. Scoring: $\text{Score}_{\mathbf{w}}(x, y) = \mathbf{w} \cdot \phi(x, y) = \sum_{j=1}^{d} w_j \phi(x, y)_j$
3. Latent support vector machine objective:

$$\min_{\mathbf{w} \in \mathbb{R}^d} \sum_{(x,d) \in \mathcal{D}} \max_{y' \in \textsc{Gen}(x)} [\text{Score}_{\mathbf{w}}(x, y') + c(d, [\![y']\!])] - \max_{y \in \textsc{Gen}(x,d)} \text{Score}_{\mathbf{w}}(x, y),$$

where $\textsc{Gen}(x, d) = \{y \in \textsc{Gen}(x) : [\![y]\!] = d\}$ is the set of logical forms that evaluate to denotation d.

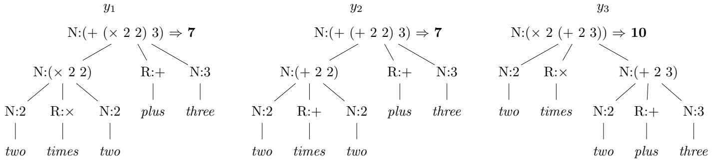
4. Optimization:

$\textsc{StochasticGradientDescent}(\mathcal{D}, T, \eta)$

```
Initialize w ← 0
Repeat T times
    for each (x, d) ∈ D (in random order)
        y ← arg max_{y'' ∈ GEN(x,d)} Score_w(x, y'')
        ỹ ← arg max_{y' ∈ GEN(x)} Score_w(x, y') + c(y, y')
        w ← w + η(φ(x, y) − φ(x, ỹ))
Return w
```

Example

(a) Candidates GEN(x) for utterance $x =$ *two times two plus three*



$\phi(x, y_1) = \boxed{\begin{array}{r} \text{R:}\times[\textit{times}]: 1 \\ \text{R:}+[\textit{plus}]: 1 \\ \text{top[R:}+]: 1 \end{array}}$ $\phi(x, y_2) = \boxed{\begin{array}{r} \text{R:}+[\textit{times}]: 1 \\ \text{R:}+[\textit{plus}]: 1 \\ \text{top[R:}+]: 1 \end{array}}$ $\phi(x, y_3) = \boxed{\begin{array}{r} \text{R:}\times[\textit{times}]: 1 \\ \text{R:}+[\textit{plus}]: 1 \\ \text{top[R:}\times]: 1 \end{array}}$

(c) Learning from denotations (Section 4.2)

Iteration 1

$\mathbf{w} = \boxed{\begin{array}{r} \text{R:}\times[\textit{times}]: 0 \\ \text{R:}+[\textit{times}]: 0 \\ \text{R:}+[\textit{plus}]: 0 \\ \text{top[R:}+]: 0 \\ \text{top[R:}\times]: 0 \end{array}}$

Scores: $[0, 0, 0]$
GEN$(x, d) = \{y_1, y_2\}$
$y = y_1$ (tied with y_2)
$\tilde{y} = y_3$ (tied with y_2)

$\Rightarrow$

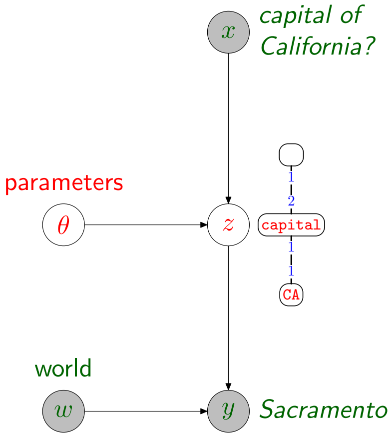
Iteration 2

$\mathbf{w} = \boxed{\begin{array}{r} \text{R:}\times[\textit{times}]: 0 \\ \text{R:}+[\textit{times}]: 0 \\ \text{R:}+[\textit{plus}]: 0 \\ \text{top[R:}+]: 1 \\ \text{top[R:}\times]: \text{-}1 \end{array}}$

Scores: $[1, 1, -1]$
GEN$(x, d) = \{y_1, y_2\}$
$y = y_1$ (tied with y_2)
$\tilde{y} = y_1$ (tied with y_2)

Not pictured: possibility of features on denotations!

Probabilistic formulation



Semantic Parsing: $p(z \mid x, \theta)$
(probabilistic)

Interpretation: $p(y \mid z, w)$
(deterministic)

(Slide from Percy Liang)

EM-style learning

Objective Function:

$$\underbrace{p(y \mid z, w)}_{\textbf{Interpretation}} \; \underbrace{p(z \mid x, \theta)}_{\textbf{Semantic parsing}}$$

(Slide from Percy Liang)

EM-style learning

Objective Function:

$$\max_{\theta} \quad \underbrace{p(y \mid z, w)}_{\textbf{Interpretation}} \, \underbrace{p(z \mid x, \theta)}_{\textbf{Semantic parsing}}$$

(Slide from Percy Liang)

EM-style learning

Objective Function:

$$\max_{\theta} \sum_{z} \underbrace{p(y \mid z, w)}_{\textbf{Interpretation}} \underbrace{p(z \mid x, \theta)}_{\textbf{Semantic parsing}}$$

(Slide from Percy Liang)

EM-style learning

Objective Function:

$$\max_\theta \sum_z \underbrace{p(y \mid z, w)}_{\textbf{Interpretation}} \underbrace{p(z \mid x, \theta)}_{\textbf{Semantic parsing}}$$

EM-like Algorithm:

parameters θ

$(0, 0, \ldots, 0)$

(Slide from Percy Liang)

EM-style learning

Objective Function:

$$\max_{\theta} \sum_{z} p(y \mid z, w)\, p(z \mid x, \theta)$$

Interpretation **Semantic parsing**

EM-like Algorithm:

parameters θ

$(0, 0, \ldots, 0)$ $\xrightarrow{\text{enumerate/score DCS trees}}$

(Slide from Percy Liang)

EM-style learning

Objective Function:

$$\max_{\theta} \sum_{z} p(y \mid z, w)\, p(z \mid x, \theta)$$

Interpretation **Semantic parsing**

EM-like Algorithm:

parameters θ

$(0, 0, \ldots, 0)$

$\xrightarrow{\text{enumerate/score DCS trees}}$

k-best list

tree1 ✗
tree2 ✗
tree3 ✓
tree4 ✗
tree5 ✗

(Slide from Percy Liang)

EM-style learning

Objective Function:

$$\max_\theta \sum_z \underbrace{p(y \mid z, w)}_{\textbf{Interpretation}} \underbrace{p(z \mid x, \theta)}_{\textbf{Semantic parsing}}$$

EM-like Algorithm:

parameters θ

$(0.2, -1.3, \ldots, 0.7)$

$\xrightarrow{\text{enumerate/score DCS trees}}$

$\xleftarrow[\text{numerical optimization (L-BFGS)}]{}$

k-best list

tree1 ✗
tree2 ✗
tree3 ✓
tree4 ✗
tree5 ✗

(Slide from Percy Liang)

EM-style learning

Objective Function:

$$\max_{\theta} \sum_{z} p(y \mid z, w)\, p(z \mid x, \theta)$$

Interpretation **Semantic parsing**

EM-like Algorithm:

parameters θ

$(0.2, -1.3, \ldots, 0.7)$

$\xrightarrow{\text{enumerate/score DCS trees}}$

$\xleftarrow[\text{numerical optimization (L-BFGS)}]{}$

k-best list

- `tree3` ✓
- `tree8` ✓
- `tree6` ✗
- `tree2` ✗
- `tree4` ✗

(Slide from Percy Liang)

EM-style learning

Objective Function:

$$\max_{\theta} \sum_{z} \underbrace{p(y \mid z, w)}_{\textbf{Interpretation}} \, \underbrace{p(z \mid x, \theta)}_{\textbf{Semantic parsing}}$$

EM-like Algorithm:

parameters θ

$(0.3, -1.4, \ldots, 0.6)$

enumerate/score DCS trees →

← numerical optimization (L-BFGS)

k-best list

- `tree3` ✓
- `tree8` ✓
- `tree6` ✗
- `tree2` ✗
- `tree4` ✗

(Slide from Percy Liang)

EM-style learning

Objective Function:

$$\max_{\theta} \sum_{z} p(y \mid z, w)\, p(z \mid x, \theta)$$

Interpretation **Semantic parsing**

EM-like Algorithm:

parameters θ

$(0.3, -1.4, \ldots, 0.6)$

enumerate/score DCS trees →

← numerical optimization (L-BFGS)

k-best list

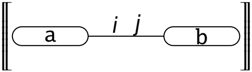
- `tree3` ✓
- `tree8` ✓
- `tree2` ✗
- `tree4` ✗
- `tree9` ✗

(Slide from Percy Liang)

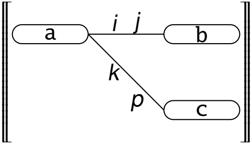
Basic Dependency-based Compositional Semantics (DCS)

A sub-logic of the full version in Liang et al. 2013:§2.5:

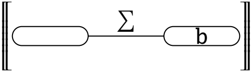
$$[\![\mathrm{P}_n]\!] = \left\{\langle x_1, \ldots, x_n\rangle, \ldots\right\}$$



$$= \left\{x \in [\![a]\!] : x_i = y_j \text{ for some } y \in [\![b]\!]\right\}$$

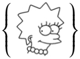


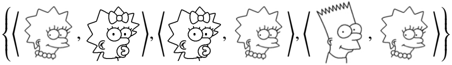
$$= \begin{array}{l}\left\{x \in [\![a]\!] : x_i = y_j \text{ for some } y \in [\![b]\!]\right\} \\ \cap \\ \left\{x \in [\![a]\!] : x_k = z_p \text{ for some } z \in [\![b]\!]\right\}\end{array}$$

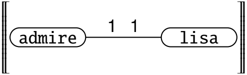


$$= \left\{[\![b]\!]\right\}$$

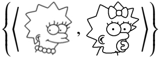
Basic DCS examples

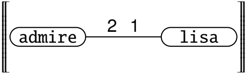
$[\![\texttt{lisa}]\!] =$ 

$[\![\texttt{admire}]\!] =$ 

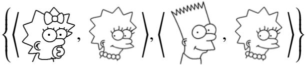


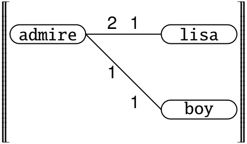
$= \{x \in [\![\texttt{admire}]\!] : x_1 = y_1 \text{ for some } y \in$  $\}$

$=$ 

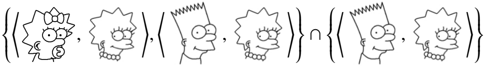


$= \{x \in [\![\texttt{admire}]\!] : x_2 = y_1 \text{ for some } y \in$  $\}$

$=$ 



$= \{x \in [\![admire]\!] : x_2 = y_1, y \in [\![lisa]\!]\} \cap \{x \in [\![admire]\!] : x_1 = z_1, z \in [\![boy]\!]\}$

$=$ 

DCS, mark/execute, and scope ambiguity

Some river traverses every city.

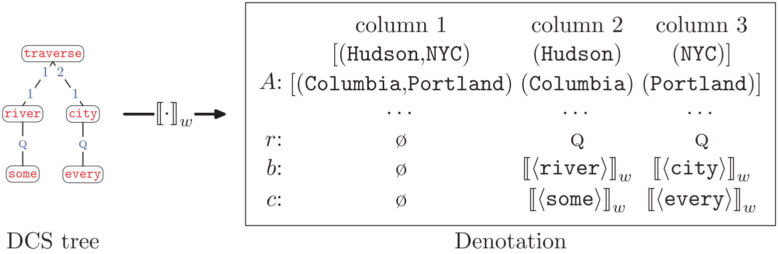


Figure 15
Denotation of Figure 8(c) before the execute relation is applied.

- Execute $\mathbf{x}_{12}$ processes column 3, then column 2: wide-scope *some river*
- Execute $\mathbf{x}_{21}$ processes column 2, then column 3: wide-scope *every city*

See also Percy's slides from last year.

Lambda DCS (Liang 2013)

Lambda DCS	Lambda DCS type	Lambda expression
$\mathtt{a}$	e	$\lambda x\ (x = \mathtt{a})$
$\mathtt{R}$	$\langle e, \langle e, t\rangle\rangle$	$\lambda x\ (\lambda y\ \mathtt{R}(x, y))$
$\mathtt{R.a}$	$\langle e, t\rangle$	$\lambda x\ \exists y\ (\mathtt{R}(x, y) \wedge \mathtt{a}(y))$
$\mathtt{P} \sqcap \mathtt{Q}$	$\langle e, t\rangle$	$\lambda x\ (\mathtt{P}(x) \wedge \mathtt{Q}(x))$
$\mathtt{P} \sqcup \mathtt{Q}$	$\langle e, t\rangle$	$\lambda x\ (\mathtt{P}(x) \vee \mathtt{Q}(x))$
$\neg\mathtt{P}$	$\langle e, t\rangle$	$\lambda x\ \neg\mathtt{P}(x)$
$\mu x\ (\mathrm{R.S.}x)$	$\langle e, t\rangle$	$\lambda x\ \exists y\ (\mathtt{R}(x, y) \wedge \mathtt{S}(y, x))$
	$\vdots$	

Table: Language definition.

Lambda DCS (Liang 2013)

Lambda DCS	Lambda DCS type	Lambda expression
$\mathtt{a}$	e	$\lambda x\ (x = \mathtt{a})$
$\mathtt{R}$	$\langle e, \langle e, t\rangle\rangle$	$\lambda x\ (\lambda y\ \mathtt{R}(x, y))$
$\mathtt{R.a}$	$\langle e, t\rangle$	$\lambda x\ \exists y\ (\mathtt{R}(x, y) \wedge \mathtt{a}(y))$
$\mathtt{P} \sqcap \mathtt{Q}$	$\langle e, t\rangle$	$\lambda x\ (\mathtt{P}(x) \wedge \mathtt{Q}(x))$
$\mathtt{P} \sqcup \mathtt{Q}$	$\langle e, t\rangle$	$\lambda x\ (\mathtt{P}(x) \vee \mathtt{Q}(x))$
$\neg\mathtt{P}$	$\langle e, t\rangle$	$\lambda x\ \neg\mathtt{P}(x)$
$\mu x\ (\mathrm{R.S.}x)$	$\langle e, t\rangle$	$\lambda x\ \exists y\ (\mathtt{R}(x, y) \wedge \mathtt{S}(y, x))$
	$\vdots$	

Table: Language definition.

Lambda DCS	Lambda expression
$\mathtt{peru}$	$\lambda x\ (x = \mathtt{peru})$
$\mathtt{Birthplace}$	$\lambda x\ (\lambda y\ \mathtt{Birthplace}(x, y))$
$\mathtt{Birthplace.peru}$	$\lambda x\ \exists y\ (\mathtt{Birthplace}(x, y) \wedge \mathtt{peru}(y))$
$\mathtt{Birthplace.peru} \sqcap \mathtt{Linguist}$	$\lambda x\ (\mathtt{Birthplace.peru}(x) \wedge \mathtt{Linguist}(x))$
$\mu x\ (\mathtt{Student.Influenced.}x)$	$\lambda x\ \exists y\ (\mathtt{Student}(x, y) \wedge \mathtt{Influenced}(y, x))$

Table: Examples.

High-level look at results

Paper	Recall (LFs)	Recall (Answers)
Zettlemoyer and Collins (2005)	79.3	–
Zettlemoyer and Collins (2007)	81.6	–
Kwiatkowksi et al. (2010)	88.2	–
Kwiatkowski et al. (2011)	88.6	–
Liang et al. (2011, 2013)	–	87.9
Liang et al. (2011, 2013) with L^+	–	91.4

Table: Results for the Geo880 test set (Zelle and Mooney 1996). For a fuller summary, see Liang et al. 2013:435. 'L^+' here involves 22 pre-specified training instances for semantically complex predicates like `size`.

Recent developments and extensions

- Learning from large databases: Clarke et al. 2010; Berant et al. 2013; Berant and Liang 2014; Kwiatkowski et al. 2013.
- Computer programming tasks: Kushman and Barzilay 2013; Lei et al. 2013
- Computer games: Branavan et al. 2010, 2011
- Learning via perception: Matuszek et al. 2012a; Tellex et al. 2011; Krishnamurthy and Kollar 2013

References I

Artzi, Yoav and Luke S. Zettlemoyer. 2011. Bootstrapping semantic parsers from conversations. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, 421–432. Edinburgh: ACL. URL http://www.aclweb.org/anthology/D11-1039.

Berant, Jonathan; Andrew Chou; Roy Frostig; and Percy Liang. 2013. Semantic parsing on Freebase from question–answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 1533–1544. Stroudsburg, PA: ACL. URL http://www.aclweb.org/anthology/D13-1160.

Berant, Jonathan and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*.

Bos, Johan. 2005. Towards wide-coverage semantic interpretation. In *Proceedings of Sixth International Workshop on Computational Semantics*, 42–53.

Bos, Johan and Katja Markert. 2005. Recognising textual entailment with logical inference. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, 628–635. Stroudsburg, PA: ACL.

Branavan, S.R.K; David Silver; and Regina Barzilay. 2011. Learning to win by reading manuals in a monte-carlo framework. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 268–277. Portland, Oregon, USA: ACL. URL http://www.aclweb.org/anthology/P11-1028.

Branavan, S.R.K.; Luke Zettlemoyer; and Regina Barzilay. 2010. Reading between the lines: Learning to map high-level instructions to commands. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, 1268–1277. Uppsala, Sweden: ACL. URL http://www.aclweb.org/anthology/P10-1129.

Cai, Qingqing and Alexander Yates. 2013. Large-scale semantic parsing via schema matching and lexicon extension. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 423–433. Sofia, Bulgaria: ACL. URL http://www.aclweb.org/anthology/P13-1042.

Clarke, James; Dan Goldwasser; Ming-Wei Chang; and Dan Roth. 2010. Driving semantic parsing from the world's response. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, 18–27. Uppsala, Sweden: ACL. URL http://www.aclweb.org/anthology/W10-2903.

Frank, Michael C.; Noah D. Goodman; and Joshua B. Tenenbaum. 2009. Using speakers' referential intentions to model early cross-situational word learning. *Psychological Science* 20(5):578–585.

Krishnamurthy, Jayant and Thomas Kollar. 2013. Jointly learning to parse and perceive: Connecting natural language to the physical world. *Transactions of the Association for Computational Linguistics* 1:193–206.

Kushman, Nate and Regina Barzilay. 2013. Using semantic unification to generate regular expressions from natural language. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 826–836. Atlanta, Georgia: ACL. URL http://www.aclweb.org/anthology/N13-1103.

Kwiatkowksi, Tom; Luke S. Zettlemoyer; Sharon Goldwater; and Mark Steedman. 2010. Inducing probabilistic CCG grammars from logical form with higher-order unification. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, 1223–1233. Cambridge, MA: Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D10-1119.

References II

Kwiatkowski, Tom; Eunsol Choi; Yoav Artzi; and Luke Zettlemoyer. 2013. Scaling semantic parsers with on-the-fly ontology matching. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 1545–1556. Seattle, Washington: Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D13-1161.

Kwiatkowski, Tom; Luke S. Zettlemoyer; Sharon Goldwater; and Mark Steedman. 2011. Lexical generalization in CCG grammar induction for semantic parsing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 1512–1523. Stroudsburg, PA, USA: ACL.

Lei, Tao; Fan Long; Regina Barzilay; and Martin Rinard. 2013. From natural language specifications to program input parsers. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1294–1303. Sofia, Bulgaria: ACL. URL http://www.aclweb.org/anthology/P13-1127.

Liang, Percy. 2013. Lambda dependency-based compositional semantics. Technical Report arXiv:1309.4408, Stanford.

Liang, Percy; Michael Jordan; and Dan Klein. 2011. Learning dependency-based compositional semantics. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 590–599. Portland, Oregon, USA: Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P11-1060.

Liang, Percy; Michael I. Jordan; and Dan Klein. 2013. Learning dependency-based compositional semantics. *Computational Linguistics* 39(2):389–446. doi:\bibinfo{doi}{10.1162/COLI_a_00127}.

Liang, Percy and Christopher Potts. 2014. Bringing machine learning and compositional semantics together. Submitted to the Annual Review of Linguistics.

Matuszek, Cynthia; Nicholas FitzGerald; Luke S. Zettlemoyer; Liefeng Bo; and Dieter Fox. 2012a. A joint model of language and perception for grounded attribute learning. In John Langford and Joelle Pineau, eds., *Proceedings of the 29th International Conference on Machine Learning*, 1671–1678. Omnipress.

Matuszek, Cynthia; Even Herbst; Luke S. Zettlemoyer; and Dieter Fox. 2012b. Learning to parse natural language commands to a robot control system. In *Proceedings of the 13th International Symposium on Experimental Robotics*.

Tang, Lappoon R. and Raymond J. Mooney. 2001. Using multiple clause constructors in inductive logic programming for semantic parsing. In *Proceedings of the 12th European Conference on Machine Learning*, 466–477. London, UK, UK: Springer-Verlag.

Tellex, Stefanie; Thomas Kollar; Steven Dickerson; Matthew R. Walter; Ashis Gopal Banerjee; Seth Teller; and Nicholas Roy. 2011. Understanding natural language commands for robotic navigation and mobile manipulation. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, 1507–1514. San Francisco: AAAI Press.

Thompson, Cynthia A. and Raymond J. Mooney. 2003. Acquiring word-meaning mappings for natural language interfaces. *Journal of Artificial Intelligence Research* 18(1):1–44.

Warren, David H. D. and Fernando C. N. Pereira. 1982. An efficient easily adaptable system for interpreting natural language queries. *American Journal of Computational Linguistics* 8(3–4):110–122.

Woods, William A; Ronald M Kaplan; and Bonnie Nash-Webber. 1972. The lunar sciences natural language information system: Final report. Technical Report, BBN Report 2378, Bolt Beranek and Newman Inc.

Zelle, John M. and Raymond J. Mooney. 1996. Learning to parse database queries using inductive logic programming. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence – Volume 2*, 1050–1055. AAAI Press.

References III

Zettlemoyer, Luke S. 2009. *Learning to Map Sentences to Logical Form*. Ph.D. thesis, MIT.

Zettlemoyer, Luke S. and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Proceedings of the Twenty First Conference on Uncertainty in Artificial Intelligence*.

Zettlemoyer, Luke S. and Michael Collins. 2007. Online learning of relaxed CCG grammars for parsing to logical form. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, 678–687. Prague, Czech Republic: Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D/D07/D07-1071.