Learning From and About Context in Semantic Parsing

Luke Zettlemoyer
Computer Science & Engineering
University of Washington

joint work with Yoav Artzi, Michael Collins,
Tom Kwiatkowski, Sharon Goldwater, Mark Steedman
Mapping Sentences to Meaning

Texas borders Kansas.

\[ \text{next-to}(\text{TEX}, \text{KAN}) \]
Mapping Sentences to Meaning

What states border Texas?
\[ \lambda x. \text{state}(x) \land \text{next-to}(x, \text{TEX}) \]
Mapping Sentences to Meaning

What states border Texas?
\[ \lambda x. \text{state}(x) \land \text{next-to}(x, \text{TEX}) \]

Machine Learning Problem
Given: Many input, output pairs
Learn: A function that maps sentences to lambda-calculus expressions
More Examples

Input: What is the largest state?
Output: $\text{argmax}(\lambda x.\text{state}(x), \lambda y.\text{size}(y))$

Input: What states border the largest state?
Output: $\lambda z.\text{state}(z) \land \text{borders}(z,$
$\text{argmax}(\lambda x.\text{state}(x), \lambda y.\text{size}(y)))$

Input: What states border states that border states ... that border Texas?
Output: $\lambda x.\text{state}(x) \land \exists y.\text{state}(y) \land \exists z.\text{state}(z) \land ...$
$\land \text{borders}(x,y) \land \text{borders}(y,z) \land \text{borders}(z,\text{texas})$
Related Work

**Supervised semantic parsing with:**
- Machine Translation
- Inducting Logic Programming
- Prob. CFG Parsing
- Prob. PDA
- Support Vector Machines
- Perceptron-style Learning
- Higher-order unification

**Less supervision:**
- Question-Answers Semi-supervision
- Confidence-driven Unsupervised
- Learning from Natural Instructions
- Learning from Conversations

[Papineni et al. 2007; Wong, Mooney 2006, 2007; Matuszek et al. 2010]
[Zelle, Money 1996; Tang, Mooney 2000; Thompson, Mooney 2002]
[Miller et al. 2006; Ge, Mooney 2006]
[He, Young 2005, 2006]
[Kate, Mooney 2006; Nguyen et al. 2006]
[Zettlemoyer, Collins 2005, 2007]
[Kwiatkowski 2010, 2011]
[Clarke et al. 2010, Liang et al. 2011]
[Goldwasser et al. 2011]
[Goldwasser and Roth, 2011]
[Artzi and Zettlemoyer, 2011]
Talk Outline

- Representing and recovering meaning with CCG
- Learning context-dependent semantic analyses
- Learning from conversations with no meaning annotations
Parsing with CCGs

\[
\begin{align*}
S/N & \quad \lambda f. f \\
\lambda y. \lambda f. \lambda x. f(x) \wedge \text{from}(x, y) \\
(N \setminus N)/NP & \quad \lambda f. \lambda x. f(x) \wedge \text{from}(x, \text{SEA}) \\
NP & \quad \lambda f. \lambda x. f(x) \wedge \text{to}(x, \text{DEN}) \\
B & \quad \text{denver} \quad \text{to} \quad \text{DEN}
\end{align*}
\]
Probabilistic CCGs

Lexicon: \( \Lambda = \)

Parameters: \( \theta \)
Probabilistic CCGs

Probability distribution: sentence $x$, parse $y$, logical form $z$

- Log-linear model:
  \[
P(y, z|x; \theta, \Lambda) = \frac{e^{\theta \cdot \phi(x,y,z)}}{\sum_{(y',z')} e^{\theta \cdot \phi(x,y',z')}}
\]

- Parsing:
  \[
f(x) = \arg \max_z p(z|x; \theta, \Lambda)
\]
  where \[p(z|x; \theta, \Lambda) = \sum_y p(y, z|x; \theta, \Lambda)\]
List flights to Singapore.
\( \lambda x. \text{flight}(x) \land \text{to}(x, \text{SIN}) \)
A Multilingual Learning Algorithm

Key challenge: learn from data with different natural languages and meaning representations

English, logical-form:

NL: what states border texas
MR: $\lambda x. state(x) \land next_to(x, tex)$

Turkish, functional query language:

NL: texas a siniri olan eyaletler nelerdir
MR: answer(state(next_to_2(stateid tex)))

[Kwiatkowski 2010,2011]
A Multilingual Learning Algorithm

Key challenge: learn from data with different natural languages and meaning representations

English, logical-form:

NL: what states border texas  
MR: $\lambda x. \text{state}(x) \land \text{next_to}(x, \text{tex})$

Turkish, functional query language:

NL: texas a siniri olan eyaletler nelerdir  
MR: $\text{answer}(\text{state}(\text{next_to}_2(\text{stateid tex})))$

Approach: use higher-order unification to recursively split the input LF

[Kwiatkowski 2010,2011]
## Example Learned Lexical Entries

<table>
<thead>
<tr>
<th>English</th>
<th>Turkish</th>
</tr>
</thead>
<tbody>
<tr>
<td>population of ⊩ NP/NP : λx.population(x)</td>
<td>nedir ⊩ S\NP(NP</td>
</tr>
<tr>
<td>smallest ⊩ NP/(S</td>
<td>NP) : λf.arg min(y, f(y), size(y))</td>
</tr>
<tr>
<td>what ⊩ S\NP/(S</td>
<td>NP) : λfλx.f(x)</td>
</tr>
<tr>
<td>border ⊩ S</td>
<td>NP/NP : λxλy.next_to(y, x)</td>
</tr>
<tr>
<td>state ⊩ S\NP : λx.state(x)</td>
<td>kac tane ⊩ S\NP/(S</td>
</tr>
<tr>
<td>most ⊩ NP/(S</td>
<td>NP)(S</td>
</tr>
<tr>
<td>no ⊩ NP</td>
<td>NP/(NP</td>
</tr>
<tr>
<td>shuu ⊩ S\NP : λx.state(x)</td>
<td>que es la ⊩ S\NP/(NP</td>
</tr>
<tr>
<td>nan desu ka ⊩ S\NP(NP</td>
<td>NP) : λfλx.f(x)</td>
</tr>
<tr>
<td>wa ⊩ NP\NP(NP</td>
<td>NP) : λfλx.f(x)</td>
</tr>
<tr>
<td>ikutsu ⊩ NP(S</td>
<td>NP)(S</td>
</tr>
<tr>
<td>chiiki ⊩ NP\NP : λx.area(x)</td>
<td>mayores ⊩ S\NP(S</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Japanese</th>
<th>Spanish</th>
</tr>
</thead>
<tbody>
<tr>
<td>population of ⊩ NP/NP : λx.population(x)</td>
<td>nedir ⊩ S\NP(NP</td>
</tr>
<tr>
<td>smallest ⊩ NP/(S</td>
<td>NP) : λf.arg min(y, f(y), size(y))</td>
</tr>
<tr>
<td>what ⊩ S\NP/(S</td>
<td>NP) : λfλx.f(x)</td>
</tr>
<tr>
<td>border ⊩ S</td>
<td>NP/NP : λxλy.next_to(y, x)</td>
</tr>
<tr>
<td>state ⊩ S\NP : λx.state(x)</td>
<td>kac tane ⊩ S\NP/(S</td>
</tr>
<tr>
<td>most ⊩ NP/(S</td>
<td>NP)(S</td>
</tr>
<tr>
<td>no ⊩ NP</td>
<td>NP/(NP</td>
</tr>
<tr>
<td>shuu ⊩ S\NP : λx.state(x)</td>
<td>que es la ⊩ S\NP/(NP</td>
</tr>
<tr>
<td>nan desu ka ⊩ S\NP(NP</td>
<td>NP) : λfλx.f(x)</td>
</tr>
<tr>
<td>wa ⊩ NP\NP(NP</td>
<td>NP) : λfλx.f(x)</td>
</tr>
<tr>
<td>ikutsu ⊩ NP(S</td>
<td>NP)(S</td>
</tr>
<tr>
<td>chiiki ⊩ NP\NP : λx.area(x)</td>
<td>mayores ⊩ S\NP(S</td>
</tr>
</tbody>
</table>
Learning Context-dependent Mappings from Sentences to Logical Form

[Zettlemoyer & Collins, ACL 2009]
Context-dependent Analysis

Show me flights from New York to Singapore.

Which of those are nonstop?

Show me the cheapest one.

What about connecting?
Context-dependent Analysis

Show me flights from New York to Singapore.
\( \lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SIN}) \)

Which of those are nonstop?

Show me the cheapest one.

What about connecting?
Context-dependent Analysis

Show me flights from New York to Singapore.
\[ \lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SIN}) \]

Which of those are nonstop?
\[ \lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SIN}) \land \text{nonstop}(x) \]

Show me the cheapest one.

What about connecting?
Show me flights from New York to Singapore.
\( \lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SIN}) \)

Which of those are nonstop?
\( \lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SIN}) \land \text{nonstop}(x) \)

Show me the cheapest one.
\( \text{argmax}(\lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SIN}) \land \text{nonstop}(x),
   \lambda y. \text{cost}(y)) \)

What about connecting?
Context-dependent Analysis

Show me flights from New York to Singapore.
\[ \lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SIN}) \]

Which of those are nonstop?
\[ \lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SIN}) \land \text{nonstop}(x) \]

Show me the cheapest one.
\[ \text{argmax}(\lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SIN}) \land \text{nonstop}(x),
\quad \lambda y. \text{cost}(y)) \]

What about connecting?
\[ \text{argmax}(\lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SIN}) \land \text{connect}(x),
\quad \lambda y. \text{cost}(y)) \]
A Supervised Learning Problem

Each training example:

a sequence of sentences and logical forms

- **Show me flights from New York to Seattle.**
  \[
  \lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SEA})
  \]

- **List ones from Newark on Friday.**
  \[
  \lambda x. \text{flight}(x) \land \text{from}(x, \text{NEW}) \land \text{to}(x, \text{SEA}) \land \text{day}(x, \text{FRI})
  \]

- **Show me the cheapest.**
  \[
  \text{argmax}(\lambda x. \text{flight}(x) \land \text{from}(x, \text{NEW}) \land \text{to}(x, \text{SEA}) \land \text{day}(x, \text{FRI}),
  \lambda y. \text{cost}(y))
  \]
A Supervised Learning Problem

**Goal:** Find a function $F$

$$\lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SEA})$$

$$\lambda x. \text{flight}(x) \land \text{to}(x, \text{SEA}) \land \text{from}(x, \text{NEW}) \land \text{day}(x, \text{FRI})$$

Show me the cheapest?

$$\arg \max (\lambda x. \text{flight}(x) \land \text{from}(x, \text{NEW}) \land \text{to}(x, \text{SEA}) \land \text{day}(x, \text{FRI}), \lambda y. \text{cost}(y))$$

[Zettlemoyer & Collins, 2009]
A Supervised Learning Problem

Goal: Find a function $F$

Show me the cheapest?

$$\text{argmax}\left(\lambda x.\text{flight}(x) \land \text{from}(x, \text{NEW}) \land \text{to}(x, \text{SEA}) \land \text{day}(x, \text{FRI}),\right.
\lambda y.\text{cost}(y))$$

Key Challenges:

• Structured input and output (lambda calculus)
• Hidden variables (only annotate final logical forms)

[Zettlemoyer & Collins, 2009]
An Example Analysis

Show me flights from New York to Seattle.
$\lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SEA})$

List ones from Newark on Friday.
Show me flights from New York to Seattle.
\[\lambda x.\text{flight}(x) \land \text{from}(x,\text{NYC}) \land \text{to}(x,\text{SEA})\]

List ones from Newark on Friday.
Show me flights from New York to Seattle.
\( \lambda x. \text{flight}(x) \land \text{from}(x,\text{NYC}) \land \text{to}(x,\text{SEA}) \)

List ones from Newark on Friday.
\( \lambda x. \text{flight}(x) \land \text{to}(x,\text{SEA}) \land \text{from}(x,\text{NEW}) \land \text{day}(x,\text{FRI}) \)
An Example Analysis

Show me flights from New York to Seattle.
\[ \lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SEA}) \]

List ones from Newark on Friday.
\[ \lambda x. \text{flight}(x) \land \text{to}(x, \text{SEA}) \land \text{from}(x, \text{NEW}) \land \text{day}(x, \text{FRI}) \]
An Example Analysis

Context:

\[ \lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SEA}) \]

Current sentence:

List ones from Newark on Friday.
An Example Analysis

Context: \[ \lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SEA}) \]

Current sentence: List ones from Newark on Friday.

**Step 1:** Context-independent parse
An Example Analysis

Context:

\[ \lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SEA}) \]

Current sentence:

List ones from Newark on Friday.

\[ \lambda x. \text{!f}(x) \land \text{from}(x, \text{NEW}) \land \text{day}(x, \text{FRI}) \]

Step 1: Context-independent parse
An Example Analysis

Context:

\( \lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SEA}) \)

Current sentence:

List \boxed{ones} \text{ from Newark on Friday.}

\( \lambda x. \text{!} f(x) \land \text{from}(x, \text{NEW}) \land \text{day}(x, \text{FRI}) \)

**Step 1:** Context-independent parse
An Example Analysis

Context:

\[ \lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SEA}) \]

Current sentence:

List ones from Newark on Friday.

\[ \lambda x. \text{!}f(x) \land \text{from}(x, \text{NEW}) \land \text{day}(x, \text{FRI}) \]

**Step 1:** Context-independent parse
An Example Analysis

Step 1: Context-independent parse

Step 2: Resolve reference
An Example Analysis

Step 1: Context-independent parse
Step 2: Resolve reference
An Example Analysis

Step 1: Context-independent parse

Step 2: Resolve reference
Derivations

Three step process:
• **Step 1**: Context-independent parsing
• **Step 2**: Resolve all references
• **Step 3**: Optionally, perform an elaboration

\[
\lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \\
\land \text{to}(x, \text{SEA})
\]

\[
\lambda x. \text{flight}(x) \land \text{to}(x, \text{SEA}) \\
\land \text{from}(x, \text{NEW}) \\
\land \text{day}(x, \text{FRI})
\]

List ones from Newark on Friday.

\[
\lambda x. \text{!f}(x) \land \text{from}(x, \text{NEW}) \\
\land \text{day}(x, \text{FRI})
\]
Derivations

Three step process:

Step 1: Context-independent parsing
• Step 2: Resolve all references
• Step 3: Optionally, perform an elaboration

$\lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SEA})$

List ones from Newark on Friday.

$\lambda x. \text{flight}(x) \land \neg \text{from}(x, \text{NEW}) \land \text{day}(x, \text{FRI})$

$\lambda x. \text{flight}(x) \land \text{to}(x, \text{SEA}) \land \text{from}(x, \text{NEW}) \land \text{day}(x, \text{FRI})$
Step 1: Referential lexical items

List ones from Newark on Friday.

\[ \lambda x. \neg f(x) \land from(x, NEW) \land day(x, FRI) \]
Step 1: Referential lexical items

List ones from Newark on Friday.

\[ \lambda x. f(x) \land \text{from}(x, \text{NEW}) \land \text{day}(x, \text{FRI}) \]

**First extension:**
Add referential lexical items

<table>
<thead>
<tr>
<th>ones</th>
<th>N</th>
<th>( \lambda x. f(x) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>it</td>
<td>NP</td>
<td>!e</td>
</tr>
<tr>
<td></td>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>
Step 1: Type-shifting operations

Second extension:
Add type-shifting operators for elliptical expressions

\[
\text{the cheapest} \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad
\]
Step 1: Type-shifting operations

Second extension:
Add type-shifting operators for elliptical expressions

\[
\lambda g. \text{argmin}(g, \lambda y. \text{cost}(y))
\]
Step 1: Type-shifting operations

Second extension:
Add type-shifting operators for elliptical expressions

\[ \text{the cheapest} \]
\[
\text{NP/N}\]
\[
\lambda g. \text{argmin}(g, \lambda y. \text{cost}(y))\]
\[
\text{NP}\]
\[
\text{argmin}(\lambda x. !f(x), \lambda y. \text{cost}(y))\]

\[ A/B : g \quad \Rightarrow \quad A : g(\lambda x. !f(x)) \]
where \( g \) is a function with input type \( \langle e, t \rangle \)
Derivations

Three step process:
- **Step 1**: Context-independent parsing
- **Step 2**: Resolve all references
- **Step 3**: Optionally, perform an elaboration
Step 2: Resolving References

For each reference:
1. Select a (sub)expression from the context
2. Substitute into current analysis

\[ \lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SEA}) \]

\[ \lambda x. \text{!f}(x) \land \text{from}(x, \text{NEW}) \land \text{day}(x, \text{FRI}) \]

List ones from Newark on Friday.

\[ \lambda x. \text{flight}(x) \land \text{to}(x, \text{SEA}) \land \text{from}(x, \text{NEW}) \land \text{day}(x, \text{FRI}) \]
Step 2: Selecting from Context

For each logical form in context, enumerate $e$ and $<e, t>$ type subexpressions:

Context:

$$\lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SEA})$$

$$\lambda x. \text{flight}(x) \land \text{to}(x, \text{SEA}) \land \text{from}(x, \text{NEW}) \land \text{day}(x, \text{FRI})$$

$$\text{argmax}(\lambda x. \text{flight}(x) \land \text{to}(x, \text{SEA}) \land \text{from}(x, \text{BOS}), \lambda y. \text{depart}(y))$$
Step 2: Selecting from Context

For each logical form in context, enumerate $e$ and $<e, t>$ type subexpressions:

Context:

\[
\lambda x.\text{flight}(x) \land \text{from}(x, \text{NYC}) \\
\land \text{to}(x, \text{SEA})
\]

\[
\lambda x.\text{flight}(x) \land \text{to}(x, \text{SEA}) \\
\land \text{from}(x, \text{NEW}) \land \text{day}(x, \text{FRI})
\]

\[
\text{argmax}(\lambda x.\text{flight}(x) \land \text{to}(x, \text{SEA}) \\
\land \text{from}(x, \text{BOS}), \\
\lambda y.\text{depart}(y))
\]
Step 2: Selecting from Context

For each logical form in context, enumerate $e$ and $<e, t>$ type subexpressions:

Context:

\[
\lambda x.\text{flight}(x) \land \text{from}(x,\text{NYC}) \land \text{to}(x,\text{SEA})
\]

\[
\lambda x.\text{flight}(x) \land \text{to}(x,\text{SEA}) \land \text{from}(x,\text{NEW}) \land \text{day}(x,\text{FRI})
\]

\[
\text{argmax}(\lambda x.\text{flight}(x) \land \text{to}(x,\text{SEA}) \land \text{from}(x,\text{BOS}), \lambda y.\text{depart}(y))
\]
Step 2: Selecting from Context

For each logical form in context, enumerate $e$ and $<e,t>$ type subexpressions:

Context:

$\lambda x.\text{flight}(x) \land \text{from}(x,\text{NYC}) \land \text{to}(x,\text{SEA})$

$\lambda x.\text{flight}(x) \land \text{to}(x,\text{SEA}) \land \text{from}(x,\text{NEW}) \land \text{day}(x,\text{FRI})$

$\text{argmax}(\lambda x.\text{flight}(x) \land \text{to}(x,\text{SEA}) \land \text{from}(x,\text{BOS}),$
$\quad \lambda y.\text{depart}(y))$

$\lambda x.\text{flight}(x) \land \text{from}(x,\text{NYC}) \land \text{to}(x,\text{SEA})$

$\text{SEA}$

$\text{NYC}$
Step 2: Selecting from Context

For each logical form in context, enumerate $e$ and $<e, t>$ type subexpressions:

Context:

\[
\lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SEA})
\]

\[
\lambda x. \text{flight}(x) \land \text{to}(x, \text{SEA}) \land \text{from}(x, \text{NEW}) \land \text{day}(x, \text{FRI})
\]

\[
\text{argmax}(\lambda x. \text{flight}(x) \land \text{to}(x, \text{SEA}) \land \text{from}(x, \text{BOS}), \lambda y. \text{depart}(y))
\]

For each logical form in context, enumerate $e$ and $<e, t>$ type subexpressions:
Step 2: Selecting from Context

For each logical form in context, enumerate $e$ and $<e, t>$ type subexpressions:

Context:

$$\lambda x.\text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SEA})$$
$$\lambda x.\text{flight}(x) \land \text{to}(x, \text{SEA}) \land \text{from}(x, \text{NEW}) \land \text{day}(x, \text{FRI})$$
$$\text{argmax}(\lambda x.\text{flight}(x) \land \text{to}(x, \text{SEA}) \land \text{from}(x, \text{BOS}),$$
$$\lambda y.\text{depart}(y))$$
Step 2: Resolving References

For each reference:
• Select an expression from the context
• Substitute into current analysis
Derivations

Three step process:
• Step 1: Context-independent parsing
• Step 2: Resolve all references
• Step 3: Optionally, perform an elaboration
Step 3: Elaboration operations

Show me the latest flight from New York to Seattle.
argmax(\(\lambda x.\) flight(\(x\)) \& from(\(x,\) NYC) \& to(\(x,\) SEA),
    \(\lambda y.\) time(\(y\)))
on Friday
Step 3: Elaboration operations

Show me the latest flight from New York to Seattle.
argmax(\(\lambda x.\text{flight}(x) \land \text{from}(x,NYC) \land \text{to}(x,SEA),\)
\(\lambda y.\text{time}(y))\)

on Friday
argmax(\(\lambda x.\text{flight}(x) \land \text{from}(x,NYC) \land \text{to}(x,SEA) \land \text{day}(x,FRI),\)
\(\lambda y.\text{time}(y))\)
Step 3: Elaboration operations

\[
\text{on Friday}
\]

\[
\lambda x. \text{day}(x, \text{FRI})
\]
Step 3: Elaboration operations

\[
\arg\max (\lambda x. \text{flight}(x) \land \text{to}(x, \text{SEA}) \land \text{from}(x, \text{NYC}), \\
\lambda y. \text{time}(y))
\]

on Friday

\[
\lambda x. \text{day}(x, \text{FRI})
\]

\[
\lambda f. \arg\max (\lambda x. \text{flight}(x) \land \text{to}(x, \text{SEA}) \land \text{from}(x, \text{NYC}) \land f(x), \\
\lambda y. \text{time}(y))
\]
Step 3: Elaboration operations

\[
\text{argmax}(\lambda x. \text{flight}(x) \land \text{to}(x, \text{SEA}) \land \text{from}(x, \text{NYC}),
\lambda y. \text{time}(y))
\]

on Friday

\[
\lambda x. \text{day}(x, \text{FRI})
\]
Step 3: Elaboration operations

\[ \text{argmax}(\lambda x. \text{flight}(x) \land \text{to}(x, \text{SEA}) \land \text{from}(x, \text{NYC}), \lambda y. \text{time}(y)) \]

on Friday

\[ \lambda x. \text{day}(x, \text{FRI}) \]

\[ \lambda f. \text{argmax}(\lambda x. \text{flight}(x) \land \text{to}(x, \text{SEA}) \land \text{from}(x, \text{NYC}) \land f(x), \lambda y. \text{time}(y)) \]

\[ \text{argmax}(\lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SEA}) \land \text{day}(x, \text{FRI}), \lambda y. \text{time}(y)) \]
Step 3: Elaboration operations

\[
\text{argmax}(\lambda x. \text{flight}(x) \land \text{to}(x, \text{SEA}) \land \text{from}(x, \text{NYC}), \\
\lambda y. \text{time}(y))
\]

on Friday

\[
\lambda x. \text{day}(x, \text{FRI})
\]

\[
\lambda f. \text{argmax}(\lambda x. \text{flight}(x) \land \text{to}(x, \text{SEA}) \land \text{from}(x, \text{NYC}) \land f(x), \\
\lambda y. \text{time}(y))
\]

Possible elaborations:

- Potentially expand any embedded variable
- Can do deletions on elaboration function
Derivations

Three step process:

• Step 1: Context-independent parsing
• Step 2: Resolve all references
• Step 3: Optionally, perform an elaboration
Scoring Derivations

List ones from Newark on Friday.

\[
\lambda x. \neg f(x) \land \text{from}(x,\text{NEW}) \land \text{day}(x,\text{FRI})
\]
Scoring Derivations

Weighted linear model:
• Introduce features: $f(d)$
• Compute scores for derivations: $w \cdot f(d)$
Features for Derivations: \( f(d) \)

\[
\lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \\
\quad \land \text{to}(x, \text{SEA}) \\
\lambda x. \text{flight}(x) \land \text{to}(x, \text{SEA}) \\
\lambda x. \text{flight}(x) \land \text{to}(x, \text{SEA}) \land \text{from}(x, \text{NEW}) \land \text{day}(x, \text{FRI}) \\
\lambda x. !f(x) \land \text{from}(x, \text{NEW}) \land \text{day}(x, \text{FRI})
\]

List ones from Newark on Friday.

Parsing features: set from Zettlemoyer and Collins (2007)

Context features:

- Distance indicators, for integers \((0,1,2,...)\)
- Copy indicators, for all predicates \(\{\text{flight, from, to, ...}\}\)
- Deletion indicators, for all pairs of predicates \(\{(\text{from, flight}), (\text{from, from}), (\text{from, to}), ...\}\)
Features for Derivations: $f(d)$

Parsing features: set from Zettlemoyer and Collins (2007)

Context features:

- Distance indicators, for integers $(0, 1, 2, ...)$
- Copy indicators, for all predicates $\{\text{flight, from, to, ...}\}$
- Deletion indicators, for all pairs of predicates $\{(\text{from, flight}), (\text{from, from}), (\text{from, to}), ...\}$

List ones from Newark on Friday.

\[ \lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SEA}) \]

\[ \lambda x. \text{flight}(x) \land \text{to}(x, \text{SEA}) \]

\[ \lambda x. \text{flight}(x) \land \text{to}(x, \text{SEA}) \land \text{from}(x, \text{NEW}) \land \text{day}(x, \text{FRI}) \]

\[ \lambda x. !f(x) \land \text{from}(x, \text{NEW}) \land \text{day}(x, \text{FRI}) \]
Features for Derivations: $f(d)$

 Parsing features: set from Zettlemoyer and Collins (2007)

 Context features:

- Distance indicators, for integers (0,1,2,...)
- Copy indicators, for all predicates \{flight, from, to, ...\}
- Deletion indicators, for all pairs of predicates
  \{(from, flight), (from, from), (from, to), ...\}
Features for Derivations: \( f(d) \)

\[
\lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SEA})
\]

List ones from Newark on Friday.

\[
\lambda x. \neg f(x) \land \text{from}(x, \text{NEW}) \land \text{day}(x, \text{FRI})
\]

\[
\lambda x. \text{flight}(x) \land \text{to}(x, \text{SEA}) \land \text{from}(x, \text{NEW}) \land \text{day}(x, \text{FRI})
\]

Parsing features: set from Zettlemoyer and Collins (2007)

Context features:

- Distance indicators, for integers \((0, 1, 2, \ldots)\)
- Copy indicators, for all predicates \{flight, from, to, \ldots\}
- Deletion indicators, for all pairs of predicates
  \[((from, flight), (from, from), (from, to), \ldots)\]
Features for Derivations: \( f(d) \)

\[
\lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SEA})
\]

List ones from Newark on Friday.

\[
\lambda x. \neg f(x) \land \text{from}(x, \text{NEW}) \land \text{day}(x, \text{FRI})
\]

\[
\lambda x. \text{flight}(x) \land \text{to}(x, \text{SEA}) \land \text{from}(x, \text{NEW}) \land \text{day}(x, \text{FRI})
\]

**Parsing features:** set from Zettlemoyer and Collins (2007)

**Context features:**

- Distance indicators, for integers \((0, 1, 2, ...)
- Copy indicators, for all predicates \(\{\text{flight, from, to, ...}\}
- Deletion indicators, for all pairs of predicates

\(\{(\text{from, flight}), (\text{from, from}), (\text{from, to}), ...\}\)
Features for Derivations: $f(d)$

**Parsing features:** set from Zettlemoyer and Collins (2007)

**Context features:**

- Distance indicators, for integers (0,1,2,...)
- Copy indicators, for all predicates \{flight, from, to, ...\}
- Deletion indicators, for all pairs of predicates \{(from, flight), (from, from), (from, to), ...\}

List ones from Newark on Friday.

$$\lambda x. \text{flight}(x) \land \text{to}(x, \text{SEA})$$

$$\lambda x. \text{flight}(x) \land \text{to}(x, \text{SEA}) \land \text{from}(x, \text{NEW}) \land \text{day}(x, \text{FRI})$$
Features for Derivations: $f(d)$

$\lambda x.\text{flight}(x) \land \text{from}(x,\text{NYC}) \land \text{to}(x,\text{SEA})$

List ones from Newark on Friday.

$\lambda x.\text{!f}(x) \land \text{from}(x,\text{NEW}) \land \text{day}(x,\text{FRI})$

$\lambda x.\text{flight}(x) \land \text{to}(x,\text{SEA})$

$\lambda x.\text{flight}(x) \land \text{to}(x,\text{SEA}) \land \text{from}(x,\text{NEW}) \land \text{day}(x,\text{FRI})$

Parsing features: set from Zettlemoyer and Collins (2007)

Context features:

• Distance indicators, for integers $(0,1,2,...)$
• Copy indicators, for all predicates $\{\text{flight, from, to, ...}\}$
• Deletion indicators, for all pairs of predicates $
  \{\text{(from, flight), (from, from), (from, to), ...}\}$
Inference and Learning

Two computations:

• Best derivation:
  \[ d^* = \arg \max_d w \cdot f(d) \]

• Best derivation with final logical form \( z \):
  \[ d' = \arg \max_{d \text{ s.t. } L(d)=z} w \cdot f(d) \]

We use a beam search algorithm.
Inference and Learning

**Two computations:**

- Best derivation:
  \[ d^* = \arg \max_d w \cdot f(d) \]

- Best derivation with final logical form \( z \):
  \[ d' = \arg \max_{d \text{ s.t. } L(d)=z} w \cdot f(d) \]

  We use a beam search algorithm.

**Learning:**

- Hidden variable version of the structured perceptron algorithm [Liang et al., 2006] [Zettlemoyer & Collins, 2007]
**Inputs:** Training set \( \{I_i \mid i = 1 \ldots n\} \) of interactions. Each interaction \( I = \{(w_{i,j}, z_{i,j}) \mid j = 1 \ldots n_i\} \) is a sequence of sentences and logical forms. Initial parameters \( w \). Number of iterations \( T \).

**Output:** Parameters \( w \).
**Inputs:** Training set \( \{I_i \mid i = 1\ldots n\} \) of interactions. Each interaction \( I = \{(w_{i,j}, z_{i,j}) \mid j = 1\ldots n_i\} \) is a sequence of sentences and logical forms. Initial parameters \( w \). Number of iterations \( T \).

**Computation:**

For \( t = 1\ldots T, i = 1\ldots n \) : (Iterate interactions)

- Set \( C = \{\} \) (Reset Context)
- For \( j = 1\ldots n_i \) : (Iterate training examples)

**Output:** Parameters \( w \).
**Inputs:** Training set \( \{I_i \mid i = 1 \ldots n\} \) of interactions. Each interaction \( I = \{(w_{i,j}, z_{i,j}) \mid j = 1 \ldots n_i\} \) is a sequence of sentences and logical forms. Initial parameters \( w \). Number of iterations \( T \).

**Computation:**

For \( t = 1 \ldots T, i = 1 \ldots n \) : (Iterate interactions)
- Set \( C = \{\} \) (Reset Context)
- For \( j = 1 \ldots n_i \) : (Iterate training examples)

**Step 3:** Update context: Append \( z_{i,j} \) to \( C \)

**Output:** Parameters \( w \).
Inputs: Training set \( \{I_i \mid i = 1...n\} \) of interactions. Each interaction \( I = \{(w_{i,j}, z_{i,j}) \mid j = 1...n_i\} \) is a sequence of sentences and logical forms. Initial parameters \( w \). Number of iterations \( T \).

Computation:

For \( t = 1...T, i = 1...n \) : (Iterate interactions)

Set \( C = \{\} \) (Reset Context)

For \( j = 1...n_i \) : (Iterate training examples)

Step 1: Check Correctness
  
  - Find best analysis: \( d^* = \arg \max_{d} w \cdot f(d) \)
  - If correct: \( L(d^*) = z_{i,j} \), go to the Step 3.

  
  
  
  
  
  Step 3: Update context: Append \( z_{i,j} \) to \( C \)

Output: Parameters \( w \).
Inputs: Training set \( \{I_i \mid i = 1 \ldots n\} \) of interactions. Each interaction 
\( I = \{(w_{i,j}, z_{i,j}) \mid j = 1 \ldots n_i\} \) is a sequence of sentences and logical forms. Initial 
parameters \( w \). Number of iterations \( T \).

Computation:

For \( t = 1 \ldots T \), \( i = 1 \ldots n \) : (Iterate interactions)

Set \( C = \{\} \) (Reset Context)

For \( j = 1 \ldots n_i \) : (Iterate training examples)

Step 1: Check Correctness

- Find best analysis: \( d^* = \arg \max_d w \cdot f(d) \)
- If correct: \( L(d^*) = z_{i,j} \), go to the Step 3.

Step 2: Update Parameters

- Find best correct analysis \( d' = \arg \max_d \quad \text{s.t. } L(d) = z_{i,j} \)
- Update parameters: \( w = w + f(d') - f(d^*) \)

Step 3: Update context: Append \( z_{i,j} \) to \( C \)

Output: Parameters \( w \).
**Inputs:** Training set \( \{I_i \mid i = 1 \ldots n\} \) of interactions. Each interaction \( I = \{(w_{i,j}, z_{i,j}) \mid j = 1 \ldots n_i\} \) is a sequence of sentences and logical forms. Initial parameters \( w \). Number of iterations \( T \).

**Computation:**

For \( t = 1 \ldots T, i = 1 \ldots n : \) (Iterate interactions)
- Set \( C = \{\} \) (Reset Context)
- For \( j = 1 \ldots n_i : \) (Iterate training examples)

  **Step 1: Check Correctness**
  - Find best analysis: \( d^* = \arg \max_d w \cdot f(d) \)
  - If correct: \( L(d^*) = z_{i,j} \), go to the Step 3.

  **Step 2: Update Parameters**
  - Find best correct analysis: \( d' = \arg \max_{d'} w \cdot f(d') \)
  - Update parameters: \( w = w + f(d') - f(d^*) \)

  **Step 3: Update context:** Append \( z_{i,j} \) to \( C \)

**Output:** Parameters \( w \).
Inputs: Training set \( \{I_i \mid i = 1 \ldots n\} \) of interactions. Each interaction \( I = \{(w_{i,j}, z_{i,j}) \mid j = 1 \ldots n_i\} \) is a sequence of sentences and logical forms. Initial parameters \( w \). Number of iterations \( T \).

Computation:

For \( t = 1 \ldots T, i = 1 \ldots n \): (Iterate interactions)
Set \( C = \{\} \) (Reset Context)
For \( j = 1 \ldots n_i \): (Iterate training examples)

Step 1: Check Correctness
- Find best analysis: \( d^* = \arg \max_d w \cdot f(d) \)
- If correct: \( L(d^*) = z_{i,j} \), go to the Step 3.

Step 2: Update Parameters
- Find best correct analysis: \( d' = \arg \max_{d' \text{s.t. } L(d') = z_{i,j}} w \cdot f(d') \)
- Update parameters: \( w = w + f(d') - f(d^*) \)

Step 3: Update context: Append \( z_{i,j} \) to \( C \)

Output: Parameters \( w \).
Evaluation

- **Domain**: ATIS travel database queries
  - 399 training interactions (3813 sentences)
  - 127 test interactions (826 sentences)
- **Comparison**: previous state-of-the-art [Miller et al. 1996]
  - requires full annotation of all syntactic, semantic, and context-resolution decisions
  - decision tree learning
Miller et al. [1996]

Step 1: Semantic parsing

Step 2: Select frame and fill slot values

Step 3: Optionally copy slot values from previous frames
Evaluation

- **Domain**: ATIS travel database queries
  - 399 training interactions (3813 sentences)
  - 127 test interactions (826 sentences)
- **Comparison**: previous state-of-the-art [Miller et al. 1996]
- **Metric**: accuracy recovering fully correct meanings
Evaluation

- **Domain:** ATIS travel database queries
  - 399 training interactions (3813 sentences)
  - 127 test interactions (826 sentences)

- **Comparison:** previous state-of-the-art [Miller et al. 1996]

- **Metric:** accuracy recovering fully correct meanings

- **Results:**
  - improved accuracy: 78.4% => 83.7%
  - less engineering effort: only annotated final meanings
Varying the Length of a Context Window $M$

ATIS Development Set:

<table>
<thead>
<tr>
<th>Context Length</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M=0$</td>
<td>45.4</td>
</tr>
<tr>
<td>$M=1$</td>
<td>79.8</td>
</tr>
<tr>
<td>$M=2$</td>
<td>81.0</td>
</tr>
<tr>
<td>$M=3$</td>
<td>82.1</td>
</tr>
<tr>
<td>$M=4$</td>
<td>81.6</td>
</tr>
<tr>
<td>$M=10$</td>
<td>81.4</td>
</tr>
</tbody>
</table>
Example Learned Feature Weights

Negative weights:
- Distance features: (1,2,3,...)

Positive weights:
- Copy features: flight, from, to
- Deletion features: (from, from), (nonstop, connect), (during-day, time)
Summary

Show me the cheapest?

\[
\lambda x.\text{flight}(x) \land \text{from}(x, \text{NYC}) \\
\land \text{to}(x, \text{SEA}) \\
\lambda x.\text{flight}(x) \land \text{to}(x, \text{SEA}) \\
\land \text{from}(x, \text{NEW}) \land \text{day}(x, \text{FRI})
\]

\[
\text{argmax}(\lambda x.\text{flight}(x) \land \text{from}(x, \text{NEW}) \land \text{to}(x, \text{SEA}) \land \text{day}(x, \text{FRI}), \\
\lambda y.\text{cost}(y))
\]

Key challenges:

- Structured input and output, hidden structure not annotated

Solution:

- Analysis: two-stage approach
- Learn: how to incorporate meaning from the context
Bootstrapping Semantic Parsers from Conversations

[Artzi & Zettlemoyer, EMNLP 2011]
Task Oriented Dialog

SYSTEM  how can i help you ?
USER  i ' d like to go from richmond to atlantic city on june seventh
SYSTEM  leaving richmond on thursday, june seventh arriving in atlantic city leaving at what time ?
USER  eight a. m.
SYSTEM  leaving in the morning what airline would you prefer ?
USER  u. s. air
SYSTEM  i currently have you leaving richmond on thursday, june seventh in the morning arriving in atlantic city flying on u. s. air . is this correct ?
USER  yes
SYSTEM  o . k . then . where would you like to go next ?
USER  richmond
[conversation continues]
Dialog and Semantics

- Many dialog systems use semantic parsers.
- Supervised learning of such parsers is expensive.
- Can computers learn through raw conversations?
Learning from Conversations

• Human-machine dialogs aim to achieve a goal.
• A well-defined goal allows the system to recover from failures.
how can i help you?
i would like to fly from atlanta georgia to london england
on september twenty fourth in the early evening i would
like to return on october first departing from london in
the late morning
System Recovery

SYSTEM: how can i help you?
USER: i would like to fly from atlanta georgia to london england on september twenty fourth in the early evening i would like to return on october first departing from london in the late morning

System fails
System Recovery

SYSTEM: how can i help you?
USER: i would like to fly from atlanta georgia to london england on september twenty fourth in the early evening i would like to return on october first departing from london in the late morning
SYSTEM: leaving what city?
how can i help you?

i would like to fly from atlanta georgia to london england on september twenty fourth in the early evening i would like to return on october first departing from london in the late morning

leaving what city?

atlanta georgia
how can i help you?
i would like to fly from atlanta georgia to london england on september twenty fourth in the early evening i would like to return on october first departing from london in the late morning.

leaving what city?

leaving atlanta. going to which city?

atlanta georgia
how can i help you?

i would like to fly from atlanta georgia to london england on september twenty fourth in the early evening i would like to return on october first departing from london in the late morning

leaving what city?

atlanta georgia

leaving atlanta. going to which city?

london
USER: I would like to fly from Atlanta, Georgia to London, England on September twenty-fourth in the early evening. I would like to return on October first, departing from London in the late morning.

SYSTEM: Leaving what city?

USER: Atlanta, Georgia

SYSTEM: Leaving Atlanta. Going to which city?

USER: London

SYSTEM: Arriving in London, England. What date would you like to depart Atlanta?
System Recovery

USER: i would like to fly from atlanta georgia to london england on september twenty fourth in the early evening i would like to return on october first departing from london in the late morning

SYSTEM: leaving what city?
USER: atlanta georgia

SYSTEM: leaving atlanta. going to which city?
USER: london

SYSTEM: arriving in london england. what date would you like to depart atlanta?
USER: september twenty fourth in the early evening

[conversation continues]
USER: i would like to fly from atlanta georgia to london england on september twenty fourth in the early evening i would like to return on october first departing from london in the late morning

SYSTEM: leaving what city? (ASK: \( \lambda x. \text{from}(fl, x) \))

USER: atlanta georgia

SYSTEM: leaving atlanta. (CONFIRM: \( \text{from}(fl, ATL) \)) going to which city? (ASK: \( \lambda x. \text{to}(fl, x) \))

USER: london

[conversation continues]
how can i help you? (OPEN_TASK)

i would like to fly from atlanta georgia to london england on september twenty fourth in the early evening i would like to return on october first departing from london in the late morning

leaving what city? (ASK: λx.from(fl, x))

atlanta georgia

leaving atlanta. (CONFIRM: from(fl, ATL)) going to which city? (ASK: λx.to(fl, x))

london
Automatically Annotated Logs

**SYSTEM:** how can i help you? (OPEN_TASK)

**USER:** i would like to fly from atlanta georgia to london england on september twenty fourth in the early evening i would like to return on october first departing from london in the late morning

**SYSTEM:** leaving what city? (ASK: \( \lambda x. \text{from}(fl, x) \))

**USER:** atlanta georgia

**SYSTEM:** leaving atlanta. (CONFIRM: from(fl, ATL)) going to which city? (ASK: \( \lambda x. \text{to}(fl, x) \))

**USER:** london

[conversation continues]

System utterance

Natural Language
i would like to fly from atlanta georgia to london england on september twenty fourth in the early evening i would like to return on october first departing from london in the late morning

leaving what city? leaving atlanta. going to which city? london
i would like to fly from atlanta georgia to london england on september twenty fourth in the early evening i would like to return on october first departing from london in the late morning

leaving what city? (ASK: \( \lambda x. \text{from}(fl, x) \))

leaving atlanta. (CONFIRM: \text{from}(fl, ATL)) going to which city? (ASK: \( \lambda x. \text{to}(fl, x) \))

london
Automatically Annotated Logs

**SYSTEM:** how can i help you? *(OPEN_TASK)*

**USER:** i would like to fly from atlanta georgia to london england on september twenty fourth in the early evening i would like to return on october first departing from london in the late morning

**SYSTEM:** leaving what city? *(ASK: \(\lambda x.\text{from}(fl, x)\))*

**USER:** atlanta georgia

**SYSTEM:** leaving atlanta. *(CONFIRM: from(fl, ATL))* going to which city? *(ASK: \(\lambda x.\text{to}(fl, x)\))*

**USER:** london

[conversation continues]

System utterance

### Natural Language

### Speech Act

### Logical Meaning

### Conversational Objects
how can i help you? (OPEN_TASK)
i would like to fly from atlanta georgia to london england on september twenty fourth in the early evening i would like to return on october first departing from london in the late morning

leaving what city? (ASK: λx.\textit{from}(fl, x))

atlanta georgia

leaving atlanta. (CONFIRM: \textit{from}(fl, ATL)) going to which city? (ASK: λx.\textit{to}(fl, x))
london

System utterance

Natural Language Speech Act Logical Meaning Conversational Objects
how can i help you? (OPEN_TASK)

i would like to fly from atlanta georgia to london england on september twenty fourth in the early evening i would like to return on october first departing from london in the late morning

leaving what city? (ASK: λx. from(fl, x))

atlanta georgia

leaving atlanta. (CONFIRM: from(fl, ATL)) going to which city? (ASK: λx. to(fl, x))

london

[conversation continues]
Supervised Case

i want to fly from new york to los angeles

\[ \lambda x. \text{from}(x, \text{NYC}) \land \text{to}(x, \text{LA}) \]
Loss Signal Instead of Labels

\[ \mathcal{L} : \langle z, C \rangle \rightarrow \mathbb{R} \]

I want to fly from New York to Los Angeles.
Loss Signal Instead of Labels

\[ \mathcal{L} : \langle z, C \rangle \rightarrow \mathbb{R} \]

i want to fly from new york to los angeles

\[ \lambda x. \text{from}(x, \text{NYC}) \land \text{to}(x, \text{LA}) \]

\[ \lambda x. \text{from}(x, \text{NYC}) \land \text{from}(x, \text{LA}) \]

\[ \lambda x. \text{from}(x, \text{LA}) \]

\[ \lambda x[]. \text{to}(x[0], \text{NYC}) \land \text{from}(x[1], \text{LA}) \]

\[ \ldots \]
Loss Signal Instead of Labels

\[ \mathcal{L} : \langle z, C \rangle \rightarrow \mathbb{R} \]

i want to fly from new york to los angeles

1. -1.8 \( \lambda x. \text{from}(x, NYC) \land \text{to}(x, LA) \)

2. -0.9 \( \lambda x. \text{from}(x, LA) \)

3. 1.5 \( \lambda x[]. \text{to}(x[0], NYC) \land \text{from}(x[1], LA) \)

\[ \cdots \]
Computing Conversation Loss

1. Select the relevant subset of system utterances
2. Align the logical expression to content from selected system utterances
3. Score the alignment
how can i help you (OPEN_TASK)
i want to fly from new york to los angeles
from new york. (CONFIRM: from(fl, NYC)) to where? (ASK: \lambda x. to(fl, x))
los angeles
flying to los angeles. (CONFIRM: to(fl, LA)) where do you want to go from los angeles? ? (ASK: \lambda x. to(fl2, x) \land from(fl2, LA))
back to new york
from los angeles to new york (CONFIRM: from(fl2, LA) \land to(fl2, NYC))
i have a direct flight from new york to los angeles for $480. ok? (OFFER: direct (fl) \land from(fl, NYC) \land to(fl, LA) \land cost(fl, 480))
[conversation continues]
USER: i want to fly from new york to los angeles

SYSTEM: from new york. (CONFIRM: from(fl, NYC)) to where? (ASK: \(\lambda x.\text{to}(fl, x)\))

USER: los angeles

SYSTEM: flying to los angeles. (CONFIRM: to(fl, LA)) where do you want to go from los angeles? ? (ASK: \(\lambda x.\text{to}(fl2, x)\land \text{from}(fl2, LA)\))

USER: back to new york

SYSTEM: from los angeles to new york (CONFIRM: from(fl2, LA)\land \text{to}(fl2, NYC))

SYSTEM: i have a direct flight from new york to los angeles for $480. ok? (OFFER: direct(fl)\land \text{from}(fl, NYC)\land \text{to}(fl, LA)\land \text{cost}(fl, 480))

[conversation continues]
how can i help you (OPEN_TASK)

i want to fly from new york to los angeles

from new york. (CONFIRM: from(fl, NYC) to where? (ASK: \( \lambda x. to(fl, x) \))

los angeles

flying to los angeles (CONFIRM: to(fl, LA)) where do you want to go from los angeles? ? (ASK: \( \lambda x. to(fl2, x) \) \& from(fl2, LA))

back to new york

from los angeles to new york (CONFIRM: from(fl2, LA) \& to(fl2, NYC))

i have a direct flight from new york to los angeles for $480. ok? (OFFER: direct (fl) \& from(fl, NYC) \& to(fl, LA))

[conversation continues]
Extract Properties

**SYSTEM:** how can i help you (OPEN_TASK)

**USER:** i want to fly from new york to los angeles

**SYSTEM:** from new york. (CONFIRM: from(fl, NYC) to where? (ASK: λx.to(fl, x))

**USER:** los angeles

**SYSTEM:** flying to los angeles (CONFIRM: to(fl, LA)) where do you want to go from los angeles? ? (ASK: λx.to(fl2, x)\from(fl2, LA))

**USER:** back to new york

**SYSTEM:** from los angeles to new york (CONFIRM: from(fl2, LA)\to(fl2, NYC))

**SYSTEM:** i have a direct flight from new york to los angeles for $480. ok? (OFFER: direct(fl) \& from(fl, NYC) \& to(fl, LA) \& cost(fl, 480))

<table>
<thead>
<tr>
<th>Property</th>
<th>Loss</th>
</tr>
</thead>
<tbody>
<tr>
<td>(from, fl, NYC)</td>
<td>-0.5</td>
</tr>
<tr>
<td>(to, fl, LA)</td>
<td>-0.83</td>
</tr>
<tr>
<td>(from, fl2, LA)</td>
<td>-0.16</td>
</tr>
<tr>
<td>(to, fl2, NYC)</td>
<td>-0.16</td>
</tr>
</tbody>
</table>

[conversation continues]
Comparing Two Candidates

**USER:** i want to fly from new york to los angeles

\[
\lambda x. \text{from}(x, \text{NYC}) \land \text{to}(x, \text{LA})
\]

Alignment: \( x \rightarrow fl \)

\[
\text{Loss} = -0.5 - 0.83 = -1.33
\]

<table>
<thead>
<tr>
<th>Property</th>
<th>Loss</th>
</tr>
</thead>
<tbody>
<tr>
<td>(from, fl, NYC)</td>
<td>-0.5</td>
</tr>
<tr>
<td>(to, fl, LA)</td>
<td>-0.83</td>
</tr>
<tr>
<td>(from, fl2, LA)</td>
<td>-0.16</td>
</tr>
<tr>
<td>(to, fl2, NYC)</td>
<td>-0.16</td>
</tr>
</tbody>
</table>

\[
\lambda x[\cdot]. \text{from}(x[0], \text{NYC}) \land \text{from}(x[1], \text{LA})
\]

Alignment: \( x[0] \rightarrow fl, x[1] \rightarrow fl2 \)

\[
\text{Loss} = -0.5 - 0.16 = -0.66
\]
Comparing Two Candidates

i want to fly from new york to los angeles

\[ \lambda x. \text{from}(x, \text{NYC}) \land \text{to}(x, \text{LA}) \]

Alignment: \( x \rightarrow fl \)

\[ \text{Loss} = -0.5 - 0.83 = -1.33 \]

\[ \lambda x[]. \text{from}(x[0], \text{NYC}) \land \text{from}(x[1], \text{LA}) \]

Alignment: \( x[0] \rightarrow fl, x[1] \rightarrow fl2 \)

\[ \text{Loss} = -0.5 - 0.16 = -0.66 \]
Learning Algorithm

• Online
• Loss-driven
• 2 steps:
  – Lexical generation
  – Parameter update [Singh-Miller and Collins 2007]
DARPA Communicator

- Raw conversational logs [Walker et al. 2002]
- Annotated system utterances

<table>
<thead>
<tr>
<th>Conversations</th>
<th>Lucent</th>
<th></th>
<th>BBN</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Train</td>
<td>Test</td>
<td>Train</td>
<td>Test</td>
</tr>
<tr>
<td>Conversations</td>
<td>144</td>
<td>70</td>
<td>71</td>
<td>91</td>
</tr>
<tr>
<td>Context Independent Utterances</td>
<td>208</td>
<td>96</td>
<td>67</td>
<td>67</td>
</tr>
</tbody>
</table>
Results: Evaluation Data

F1 Measure

- Lucent Exact Match
- Lucent Partial Credit
- BBN Exact Match
- BBN Partial Credit

No Conversation Baseline
Our Approach
Supervised
Contributions

• Learning to do context-dependent semantic parsing
  – Complex, hidden variable problem
  – Learn distribution over derivations

• Learning from conversation logs
  – Loss driven, requiring no annotations
  – First step towards autonomous, self-improving dialog systems
Current Work: Self-learning dialog systems

Goal: learn full dialog semantic parser the produces context-dependent analyses

**USER:** i would like to fly from atlanta georgia to london england on september twenty fourth in the early evening i would like to return on october first departing from london in the late morning

**SYSTEM:** leaving what city? (ASK:λx.from(fl, x))

**USER:** atlanta georgia.

**SYSTEM:** leaving atlanta. (CONF:from(fl, ATL)) going to which city? (ASK:λx.to(fl, x))

**USER** london

**SYSTEM:** arriving in (CONF:to(fl, LON)) what date would to depart Atlanta? (ASK:λx.from(fl, ATL) ∧ depart- date(fl, x))

**USER** september twenty fourth in the early evening

[conversation continues]
Current Work: Mapping Speech to Logical From

Uh, flights New York to Singapore, sure

ACCEPT: $\lambda x. \text{flight}(x) \land \text{from}(x, \text{NYC}) \land \text{to}(x, \text{SIN})$
Challenge: Learn to sportscast, given only text and the game log

Purple10 is rushing down the field with only three defenders.
Purple10 passes out front to Purple9 near the side.
Purple9 passes back to Purple10 in the middle.
Purple10 again has a good chance to score a goal here.
Purple10 dribbles toward the goal.
Pink3 tries to stay in front of Purple10.
Purple10 passes to Purple9 on the side while getting open.

....
A Joint Model for Naming Objects

These are the ones that are not blue

\[ \lambda x. \neg \text{color}(x, \text{blue}) \]
Future: General language use in grounded settings

Conversational interaction in simulated environments:

- Can gather user input: *Which printer do you want to use?*
- Can help with learning: *Can you show me how to X?*

Learning through explanation in robotic environments:

- Can we teach the robot to play?
  - *This is a pawn.*
  - *Pawns can move forward one square at a time.*
  - *unless it is the first move, then they can ...*
Learning About and From Context in Semantic Parsing

special thanks to
Yoav Artzi, Tom Kwiatkowski, Sharon Goldwater, Mark Steedman, Adrienne Wang, Mark Yatskar

for more info:
http://www.cs.washington.edu/homes/lsz/