Semantic Parsing with Combinatory Categorial Grammars

Yoav Artzi, Nicholas FitzGerald and Luke Zettlemoyer
University of Washington

ACL 2013 Tutorial
Sofia, Bulgaria
• What kind of data/supervision we can use?
• What do we need to learn?
Supervised Data

<table>
<thead>
<tr>
<th>show me</th>
<th>flights</th>
<th>to</th>
<th>Boston</th>
</tr>
</thead>
<tbody>
<tr>
<td>( S/N ) ( \lambda f. f )</td>
<td>( N ) ( \lambda x. \text{flight}(x) )</td>
<td>( PP/NP ) ( \lambda y. \lambda x. \text{to}(x, y) )</td>
<td>( NP ) ( \text{BOSTON} )</td>
</tr>
</tbody>
</table>

\[
\frac{PP}{\lambda x. \text{to}(x, \text{BOSTON})}
\]

\[
\frac{N \setminus N}{\lambda f. \lambda x. f(x) \land \text{to}(x, \text{BOSTON})}
\]

\[
\frac{N}{\lambda x. \text{flight}(x) \land \text{to}(x, \text{BOSTON})}
\]

\[
\frac{S}{\lambda x. \text{flight}(x) \land \text{to}(x, \text{BOSTON})}
\]
Supervised Data

<table>
<thead>
<tr>
<th>show me</th>
<th>flights</th>
<th>to Boston</th>
</tr>
</thead>
<tbody>
<tr>
<td>S/N</td>
<td>N</td>
<td>PP/NP</td>
</tr>
<tr>
<td>λf.f</td>
<td>λx.flight(x)</td>
<td>λy.λx.to(x, y)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>λx.to(x, BOSTON)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>N\N</td>
</tr>
<tr>
<td></td>
<td></td>
<td>λf.λx.f(x) ∧ to(x, BOSTON)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>N</td>
</tr>
<tr>
<td></td>
<td></td>
<td>λx.flight(x) ∧ to(x, BOSTON)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>S</td>
</tr>
<tr>
<td></td>
<td></td>
<td>λx.flight(x) ∧ to(x, BOSTON)</td>
</tr>
</tbody>
</table>
Supervised Data

Supervised learning is done from pairs of sentences and logical forms

Show me flights to Boston
\( \lambda x. \text{flight}(x) \land \text{to}(x, BOSTON) \)

I need a flight from baltimore to seattle
\( \lambda x. \text{flight}(x) \land \text{from}(x, BALTIMORE) \land \text{to}(x, SEATTLE) \)

what ground transportation is available in san francisco
\( \lambda x. \text{ground\_transport}(x) \land \text{to\_city}(x, SF) \)

[Zettlemoyer and Collins 2005; 2007]
Weak Supervision

• Logical form is latent
• “Labeling” requires less expertise
• Labels don’t uniquely determine correct logical forms
• Learning requires executing logical forms within a system and evaluating the result
Weak Supervision
Learning from Query Answers

What is the largest state that borders Texas?

New Mexico

[Clarke et al. 2010; Liang et al. 2011]
Weak Supervision
Learning from Query Answers

What is the largest state that borders Texas?

*New Mexico*

\[
\text{argmax}(\lambda x. \text{state}(x) \\
\land \text{border}(x, TX), \lambda y. \text{size}(y))
\]

\[
\text{argmax}(\lambda x. \text{river}(x) \\
\land \text{in}(x, TX), \lambda y. \text{size}(y))
\]

[Clarke et al. 2010; Liang et al. 2011]
Weak Supervision
Learning from Query Answers

What is the largest state that borders Texas?

*New Mexico*

$$\text{argmax}(\lambda x. \text{state}(x) \wedge \text{border}(x,TX), \lambda y. \text{size}(y))$$

$$\text{argmax}(\lambda x. \text{river}(x) \wedge \text{in}(x,TX), \lambda y. \text{size}(y))$$

[Clarke et al. 2010; Liang et al. 2011]
Weak Supervision
Learning from Query Answers

What is the largest state that borders Texas?

\[
\text{New Mexico}
\]

\[
\text{argmax}(\lambda x.\text{state}(x) \wedge \text{border}(x, TX), \lambda y.\text{size}(y))
\]

\[
\text{argmax}(\lambda x.\text{river}(x) \wedge \text{in}(x, TX), \lambda y.\text{size}(y))
\]

[Clarke et al. 2010; Liang et al. 2011]
Weak Supervision
Learning from Demonstrations

at the chair, move forward three steps past the sofa
Weak Supervision
Learning from Demonstrations

at the chair, move forward three steps past the sofa

Some examples from other domains:

• Sentences and labeled game states [Goldwasser and Roth 2011]
• Sentences and sets of physical objects [Matuszek et al. 2012]

[Chen and Mooney 2011; Kim and Mooney 2012; Artzi and Zettlemoyer 2013b]
• Structured perceptron
• A unified learning algorithm
• Supervised learning
• Weak supervision
Structured Perceptron

- Simple additive updates
  - Only requires efficient decoding \((\text{argmax})\)
  - Closely related to maxent and other feature rich models
  - Provably finds linear separator in finite updates, if one exists
- Challenge: learning with hidden variables
Structured Perceptron

Data: \{(x_i, y_i) : i = 1 \ldots n\}

For \( t = 1 \ldots T \):

For \( i = 1 \ldots n \):

\[ y^* \leftarrow \arg \max_y \langle \theta, \Phi(x_i, y) \rangle \]

If \( y^* \neq y_i \):

\[ \theta \leftarrow \theta + \Phi(x_i, y_i) - \Phi(x_i, y^*) \]
One Derivation of the Perceptron

Log-linear model: \( p(y|x) = \frac{e^{w \cdot f(x,y)}}{\sum_{y'} e^{w \cdot f(x,y')}} \)

Step 1: Differentiate, to maximize data log-likelihood

\[
\text{update} = \sum_i f(x_i, y_i) - E_{p(y|x_i)} f(x_i, y)
\]

Step 2: Use online, stochastic gradient updates, for example \( i \):

\[
\text{update}_i = f(x_i, y_i) - E_{p(y|x_i)} f(x_i, y)
\]

Step 3: Replace expectations with maxes (Viterbi approx.)

\[
\text{update}_i = f(x_i, y_i) - f(x_i, y^*) \quad \text{where} \quad y^* = \arg \max_y w \cdot f(x_i, y)
\]
The Perceptron with Hidden Variables

Log-linear model: \[ p(y|x) = \sum_h p(y, h|x) \quad p(y, h|x) = \frac{e^{w \cdot f(x, h, y)}}{\sum_{y', h'} e^{w \cdot f(x, h', y')}} \]

Step 1: Differentiate marginal, to maximize data log-likelihood
\[
\text{update} = \sum_i E_p(h|y_i, x_i) [f(x_i, h, y_i)] - E_p(y, h|x_i) [f(x_i, h, y)]
\]

Step 2: Use online, stochastic gradient updates, for example \( i \):
\[
\text{update}_i = E_p(y_i, h|x_i) [f(x_i, h, y_i)] - E_p(y, h|x_i) [f(x_i, h, y)]
\]

Step 3: Replace expectations with maxes (Viterbi approx.)
\[
\text{update}_i = f(x_i, h', y_i) - f(x_i, h^*, y^*) \quad \text{where}
\]
\[
y^*, h^* = \arg \max_{y, h} w \cdot f(x_i, h, y) \quad \text{and} \quad h' = \arg \max_h w \cdot f(x_i, h, y_i)
\]
Hidden Variable Perceptron

Data: \[ \{(x_i, y_i) : i = 1 \ldots n\} \]

For \( t = 1 \ldots T \):

For \( i = 1 \ldots n \):

\[ y^*, h^* \leftarrow \arg \max_{y,h} \langle \theta, \Phi(x_i, h, y) \rangle \]

If \( y^* \neq y_i \):

\[ h' \leftarrow \arg \max_h \langle \theta, \Phi(x_i, h, y_i) \rangle \]

\[ \theta \leftarrow \theta + \Phi(x_i, h', y_i) - \Phi(x_i, h^*, y^*) \]

[Liang et al. 2006; Zettlemoyer and Collins 2007]
Hidden Variable Perceptron

- No known convergence guarantees
  - Log-linear version is non-convex
- Simple and easy to implement
  - Works well with careful initialization
- Modifications for semantic parsing
  - Lots of different hidden information
  - Can add a margin constraint, do probabilistic version, etc.
Learning Choices

**Validation Function**

\[ \mathcal{V} : \mathcal{Y} \rightarrow \{ t, f \} \]

- Indicates correctness of a parse \( y \)
- Varying \( \mathcal{V} \) allows for differing forms of supervision

**Lexical Generation Procedure**

\[ GENLEX (x, \mathcal{V}; \Lambda, \theta) \]

- Given:
  - sentence \( x \)
  - validation function \( \mathcal{V} \)
  - lexicon \( \Lambda \)
  - parameters \( \theta \)
- Produce a overly general set of lexical entries
Initialize \( \theta \) using \( \Lambda_0 \), \( \Lambda \leftarrow \Lambda_0 \)

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

**Step 1:** (Lexical generation)

a. Set \( \lambda_G \leftarrow GENLEX(x_i, \mathcal{V}_i; \Lambda, \theta) \),
   \( \lambda \leftarrow \Lambda \cup \lambda_G \)

b. Let \( Y \) be the \( k \) highest scoring parses from \( GEN(x_i; \lambda) \)

c. Select lexical entries from the highest scoring valid parses:
   \( \lambda_i \leftarrow \bigcup_{y \in MAXV_i(Y; \theta)} LEX(y) \)

d. Update lexicon: \( \Lambda \leftarrow \Lambda \cup \lambda_i \)

**Step 2:** (Update parameters)

**Output:** Parameters \( \theta \) and lexicon \( \Lambda \)
Unification-based

$\text{GENLEX}(x, z; \Lambda, \theta)$

I want a flight to Boston

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

1. Find highest scoring correct parse
2. Find splits that most increases score
3. Return new lexical entries

Iteration 2