Semantic Parsing with Relaxed Hybrid Trees

Wei Lu

Singapoue University of Technology and Design

luwei@sutd.edu.sg

October 27, 2014
Overview

Background
  Semantic Parsing
  Related Works

Hybrid Trees

Relaxed Hybrid Trees
  Algorithms
  Features

Experiments

Conclusions
Semantic Parsing

Parsing complete natural language sentences into their corresponding complete semantic representations.

What are the semantic representations?

▶ $\lambda$-calculus expressions (with CCG)
▶ DRT representations
▶ Dependency-based semantics
▶ Forest, or DAG representations (AMR)
▶ Logical forms with tree structures

We focus on logical forms with tree structures.
Semantic Parsing

Task-independent
- Wong and Mooney (2006), Kate and Mooney (2006)
- Zettlemoyer and Collins (2005, 2007)
- Lu, Ng, Lee and Zettlemoyer (2008)

Task-specific
- Answering questions
  - Liang, Klein and Jordan (2010)
  - Clarke, Goldwasser, Chang and Roth (2010)
- Taking actions
  - Artzi and Zettlemoyer (2013)

We focus on task-independent semantic parsing.
Semantic Parsing

how many states do not have a river?

⇓

Query: \( \text{answer(Num)} \)

| Num: \( \text{count(State)} \)
| State: \( \text{exclude(State, State)} \)

State: \( \text{state(all)} \)  State: \( \text{loc(River)} \)

River: \( \text{river(all)} \)

⇓

\( \text{answer(count(exclude(state(all), loc(river(all))))}) \)

Each tree node is a semantic unit:

\( \tau_a: \alpha(\tau_b) \) or \( \tau_a: \alpha(\tau_b, \tau_c) \)
Goal: Learn to transform texts into semantic trees.

how many states do not have a river?

\[ \downarrow \]

**Query**: \( \text{answer}(\text{Num}) \)

**Num**: \( \text{count}(\text{State}) \)

**State**: \( \text{exclude}(\text{State}, \text{State}) \)

**State**: \( \text{state}(\text{all}) \) \hspace{1cm} **State**: \( \text{loc}(\text{River}) \)

**River**: \( \text{river}(\text{all}) \)
Semantic Parsing

Goal: Learn to transform texts into semantic trees.

how many states do not have a river?

⇓

QUERY: answer(Num)

| Num: count(State)
| State: exclude(State, State)

STATE: state(all)  STATE: loc(River)

RIVER: river(all)

Assumption: there exist joint representations of both!!
Joint Representations

Proposed in previous works:

- Synchronous CFG derivation trees
  - Wong and Mooney (2006, 2007)
- Recursive phrase-to-tree mappings
  - Kate and Mooney (2006)
- Semantically-augmented syntactic trees
  - Ge and Mooney (2005)
- CCG derivations
  - Zettlemoyer and Collins (2005, 2007)
  - Kwiatkowski, Zettlemoyer, Goldwater and Steedman (2010)
- Bayesian tree transducers
  - Jones, Goldwater and Johnson (2012)
- Hybrid trees
  - Lu, Ng, Lee and Zettlemoyer (2008)
  - Zhou, Xu and Qu (2013)
Hybrid Trees

QUERY: \textit{answer}(\textit{Num})

\textit{Num}: \textit{count}(\textit{State})

how many \hspace{1cm} \textit{State}: \textit{exclude}(\textit{State}, \textit{State})

\textit{State}: \textit{state}(all) \hspace{0.5cm} \text{do not} \hspace{0.5cm} \textit{State}: \textit{loc}(\textit{River})

states \hspace{1cm} \text{have} \hspace{1cm} \textit{River}: \textit{river}(all)

a river

- Internal nodes are \textit{semantic units}
- Leaf nodes are \textit{natural language words}
- Such hybrid trees are generated from an underlying generative process in a recursive manner.
Hybrid Trees – Generative Process

Query: $answer(\text{Num})$
Hybrid Trees – Generative Process

**QUERY:** $\text{answer(Num)}$

```
          QUERY : answer(Num)
                  |
                  X_w
```
Hybrid Trees – Generative Process

Query: \textit{answer(Num)}

\texttt{Num}: \textit{count(State)} \quad ?
Hybrid Trees – Generative Process

**QUERY**: $answer(Num)$

$Num : count(State)$

$wx$
Hybrid Trees – Generative Process

Query: \( \text{answer}(\text{Num}) \)

\text{Num}: \text{count}(\text{State})

how many \text{State}: \text{exclude}(\text{State}_1\text{State}_2)

\text{XwY}
Hybrid Trees – Generative Process

QUERY: \textit{answer(Num)}

\textbf{Num: count(State)}

how many

\textbf{State: exclude(State, State)}

\textbf{State: state(all) do not State: loc(River)}

w

wX
Hybrid Trees – Generative Process

QUERY : $answer(Num)$

$Num : count(State)$

how many

$State : exclude(State, State)$

$State : state(all)$

do not

$State : loc(River)$

states

have $River : river(all)$

w
Hybrid Trees – Generative Process

Such a process:

- models the joint probability $P(m, h, w)$.
- gives the correspondences between the semantic units and natural language words (lexicon).
- produces one of the many possible hybrid trees.
Find the best hybrid tree using the EM (inside-outside) algorithm.
Hybrid Trees

Interesting Properties

- Language-independent
- Integrated lexicon acquisition and semantic parsing
- Efficient (we’ve developed a training algorithm with a time complexity that is cubic in the number of words, and linear in the number of semantic units.)

Limitations

- Unable to capture long-distance dependencies
- Unable to incorporate rich features
Hybrid Trees – Limitations

**QUERY:** \(\text{answer}(\text{Num})\)

**Num:** \(\text{count}(\text{State})\)

How many

**State:** \(\text{exclude}(\text{State}, \text{State})\)

**State:** \(\text{state}(\text{all})\) do not

**State:** \(\text{loc}(\text{River})\)

States have \(\text{River}: \text{river}(\text{all})\)

A river

\[P(m, h^*_1, w_1) = 0.00079...\]

**QUERY:** \(\text{answer}(\text{Num})\)

**Num:** \(\text{count}(\text{State})\)

How many

**State:** \(\text{exclude}(\text{State}, \text{State})\)

**State:** \(\text{state}(\text{all})\)

**State:** \(\text{loc}(\text{River})\)

States have \(\text{River}: \text{river}(\text{all})\) rivers

No

\[P(m, h^*_2, w_2) = 0.000000000000000045...\]
Relaxed Hybrid Trees

Idea #1: Hybrid trees are relaxed to capture unbounded long-distance dependencies.

QUERY: \( \text{answer(Num)} \)
how many states have no rivers?

\( \text{Num: count(State)} \)
how many states have no rivers

\( \text{State: exclude(State, State)} \)
how many states have no rivers

\( \text{State: state(all)} \)
states

\( \text{State: loc(River)} \)
have no rivers

\( \text{River: river(all)} \)
rivers
Relaxed Hybrid Trees

Idea #2: Moving from the generative model to the (latent-variable) discriminative model to support flexible features:

\[
P(m|w) = \frac{\sum_h P(m, h, w)}{\sum_{m',h'} P(m', h', w)}
\]

where

\[
P(m, h, w) \propto \exp \left( w \cdot \Phi(m, h, w) \right)
\]
Algorithms – Training

\[ P(m|w) = \frac{\sum_h P(m, h, w)}{\sum_{m', h'} P(m', h', w)} \]

where

\[ P(m, h, w) \propto \exp(w \cdot \Phi(m, h, w)) \]

Computation of the term \( \sum_h P(m, h, w) \) can be done using a similar algorithm as the one used in the generative model (if the features are defined in a certain manner).
Algorithms – Training

\[ P(m|w) = \frac{\sum_h P(m, h, w)}{\sum_{m', h'} P(m', h', w)} \]

where

\[ P(m, h, w) \propto \exp(w \cdot \Phi(m, h, w)) \]

Computation of the term \( \sum_{m', h'} P(m', h', w) \) involves dynamic programming over a packed forest representation of all possible semantic trees. This algorithm is similar to the one used in Lu and Ng (2011) for a generation task.
Algorithms – Training

\[
P(m|w) = \frac{\sum_h P(m, h, w)}{\sum_{m', h'} P(m', h', w)}
\]

where

\[
P(m, h, w) \propto \exp (w \cdot \Phi(m, h, w))
\]

Gradients of the objective function can be computed analogously. We use L-BFGS for learning model parameters.
Computation of the marginal is expensive:

\[ m^* = \max_m P(m|w) = \max_m \sum_h P(m, h|w) \]

We instead do the following:

\[ m^* = \max_m \max_h P(m, h|w) = \max_{m, h} P(m, h, w) \]

Computation of this can be done with a similar dynamic programming algorithm used for training.
Features

\[ \text{STATE} : \text{loc}(\text{River}) \]
\[ \text{have no} \]
\[ \text{have no rivers} \]
\[ \text{RIVER} : \text{river}(\text{all}) \]
\[ \text{rivers} \]

Local features

- Unigram/bigram features
  - Unigram: \textit{STATE} : \textit{loc}(\textit{River}) + have
  - Bigram: \textit{STATE} : \textit{loc}(\textit{River}) + have no

- Character-level features
  - Char: \textit{RIVER} : \textit{river}(\textit{all}) + rive
  - Char: \textit{RIVER} : \textit{river}(\textit{all}) + river
  - Char: \textit{RIVER} : \textit{river}(\textit{all}) + rivers
Features

Span (long-distance) features

- Unigram/bigram/trigram features
  - Span: \texttt{STATE : loc(RIVER)+have}
  - Span: \texttt{STATE : loc(RIVER)+have no}
  - Span: \texttt{STATE : loc(RIVER)+have no rivers}
Experiments

- Data: multilingual Geoquery dataset (Jones et al 2012).
- The same experimental setup used in previous works.

Hybrid Trees vs. Relaxed Hybrid Trees (RHT), when all features are used.

<table>
<thead>
<tr>
<th>System</th>
<th>English Acc.</th>
<th>English F1</th>
<th>Thai Acc.</th>
<th>Thai F1</th>
<th>German Acc.</th>
<th>German F1</th>
<th>Greek Acc.</th>
<th>Greek F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hybrid Tree+</td>
<td>76.8</td>
<td>81.0</td>
<td>73.6</td>
<td>76.7</td>
<td>62.1</td>
<td>68.5</td>
<td>69.3</td>
<td>74.6</td>
</tr>
<tr>
<td>RHT (all features)</td>
<td>83.6</td>
<td>83.6</td>
<td>79.3</td>
<td>79.3</td>
<td>74.3</td>
<td>74.3</td>
<td>78.2</td>
<td>78.2</td>
</tr>
</tbody>
</table>
Experiments

Other systems vs. Relaxed Hybrid Trees (RHT), when all features are used.

<table>
<thead>
<tr>
<th>System</th>
<th>English Acc.</th>
<th>English F1</th>
<th>Thai Acc.</th>
<th>Thai F1</th>
<th>German Acc.</th>
<th>German F1</th>
<th>Greek Acc.</th>
<th>Greek F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wasp</td>
<td>71.1</td>
<td>77.7</td>
<td>71.4</td>
<td>75.0</td>
<td>65.7</td>
<td>74.9</td>
<td>70.7</td>
<td><strong>78.6</strong></td>
</tr>
<tr>
<td>Hybrid Tree+</td>
<td>76.8</td>
<td>81.0</td>
<td>73.6</td>
<td>76.7</td>
<td>62.1</td>
<td>68.5</td>
<td>69.3</td>
<td>74.6</td>
</tr>
<tr>
<td>UBL-s</td>
<td>82.1</td>
<td>82.1</td>
<td>66.4</td>
<td>66.4</td>
<td><strong>75.0</strong></td>
<td><strong>75.0</strong></td>
<td>73.6</td>
<td>73.7</td>
</tr>
<tr>
<td>TreeTrans</td>
<td>79.3</td>
<td>79.3</td>
<td>78.2</td>
<td>78.2</td>
<td>74.6</td>
<td>74.6</td>
<td>75.4</td>
<td>75.4</td>
</tr>
<tr>
<td>RHT (all features)</td>
<td><strong>83.6</strong></td>
<td><strong>83.6</strong></td>
<td><strong>79.3</strong></td>
<td><strong>79.3</strong></td>
<td>74.3</td>
<td>74.3</td>
<td><strong>78.2</strong></td>
<td>78.2</td>
</tr>
</tbody>
</table>
## Experiments

### Effect of different features.

<table>
<thead>
<tr>
<th>System</th>
<th>English Acc.</th>
<th>English F1</th>
<th>Thai Acc.</th>
<th>Thai F1</th>
<th>German Acc.</th>
<th>German F1</th>
<th>Greek Acc.</th>
<th>Greek F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>WASP</td>
<td>71.1</td>
<td>77.7</td>
<td>71.4</td>
<td>75.0</td>
<td>65.7</td>
<td>74.9</td>
<td>70.7</td>
<td>78.6</td>
</tr>
<tr>
<td>Hybrid Tree+</td>
<td>76.8</td>
<td>81.0</td>
<td>73.6</td>
<td>76.7</td>
<td>62.1</td>
<td>68.5</td>
<td>69.3</td>
<td>74.6</td>
</tr>
<tr>
<td>UBL-s</td>
<td>82.1</td>
<td>82.1</td>
<td>66.4</td>
<td>66.4</td>
<td>75.0</td>
<td>75.0</td>
<td>73.6</td>
<td>73.7</td>
</tr>
<tr>
<td>TreeTrans</td>
<td>79.3</td>
<td>79.3</td>
<td>78.2</td>
<td>78.2</td>
<td>74.6</td>
<td>74.6</td>
<td>75.4</td>
<td>75.4</td>
</tr>
<tr>
<td>RHT (all features)</td>
<td>83.6</td>
<td>83.6</td>
<td>79.3</td>
<td>79.3</td>
<td>74.3</td>
<td>74.3</td>
<td>78.2</td>
<td>78.2</td>
</tr>
<tr>
<td>no local features</td>
<td>81.4</td>
<td>81.4</td>
<td>78.2</td>
<td>78.2</td>
<td>74.3</td>
<td>74.3</td>
<td>75.7</td>
<td>75.7</td>
</tr>
<tr>
<td>no char features</td>
<td>79.6</td>
<td>79.6</td>
<td>82.1</td>
<td>82.1</td>
<td>73.6</td>
<td>73.6</td>
<td>76.1</td>
<td>76.1</td>
</tr>
<tr>
<td>no span features</td>
<td>81.1</td>
<td>81.1</td>
<td>77.9</td>
<td>77.9</td>
<td>78.2</td>
<td>78.2</td>
<td>78.9</td>
<td>78.9</td>
</tr>
</tbody>
</table>
Novel \textit{relaxed hybrid trees} for better task-independent semantic parsing.

- Performs integrated lexicon acquisition and semantic parsing.
- Captures unbounded long-distance dependencies.
- Supports efficient algorithms for training and inference.
- Easy to incorporate additional features.

Future...

- Incorporate more language-specific features
- Incorporate distributional semantics
Thank You

Code available:
http://statnlp.org/research/sp/