New Resources and Ideas for Semantic Parsing

Kyle Richardson

Institute for Natural Language Processing (IMS)

School of Computer Science, Electrical Engineering and Information Technology
University of Stuttgart

Collaborators: Jonas Kuhn (advisor, Stuttgart) and Jonathan Berant (work on "polyglot semantic parsing", Tel Aviv), last updated 24.9.2018
Main Topic: Semantic Parsing

- **Task**: mapping text to formal meaning representations (ex., from Herzig and Berant (2017)).

  **Text**: *Find an article with no more than two authors.*

  →

  **LF**: $\text{Type.Article} \sqcap R[\lambda x.\text{count}(	ext{AuthorOf}.x)] \leq 2$
Main Topic: Semantic Parsing

- **Task**: mapping text to formal meaning representations (ex., from Herzig and Berant (2017)).

**Text**: Find an article with no more than two authors.

\[ \text{LF: } \text{Type.Article} \sqcap R[\lambda x.\text{count(AuthorOf}.x)] \leq 2 \]

"Machines and programs which attempt to answer English question have existed for only about five years.... Attempts to build machine to test logical consistency date back to at least Roman Lull in the thirteenth century... **Only in recent years have attempts been made to translate mechanically from English into logical formalisms...”

Classical Natural Language Understanding (NLU)

- Conventional **pipeline model**: focus on capturing **deep inference** and entailment.

\[ \text{List samples that contain every major element} \]

\[ \text{database} \]

\[ [[\text{sem}]] = \{ \text{S10019, S10059, ...} \} \]

Lunar QA system of Woods (1973)
NLU model is a kind of compiler, involves a **transduction** from NL to a formal (usually logical) language.
Data-driven Semantic Parsing and NLU

1. Semantic Parsing
   List samples that contain every major element

2. Knowledge Representation
   (FOR EVERY X / MAJORELT : T;
    (FOR EVERY Y /
     SAMPLE : (CONTAINS Y X);
     (PRINTOUT Y)))

3. Reasoning

\[ \{sem\} = \{S10019, S10059, \ldots \} \]

- **Data-driven NLU**: Asks an empirical question: Can we learn NLU models from examples? Building a NL compiler by hand is hard....
Data-driven Semantic Parsing and NLU

1. Semantic Parsing

List samples that contain every major element

\[ \text{sem} = \{S10019, S10059, \ldots \} \]

2. Knowledge Representation

\[
\begin{align*}
\text{(for every } X / \text{ MAJORELT : } T; \\
\text{(for every } Y / \text{ SAMPLE : (contains } Y \ X); \\
\text{ (printout } Y)))
\end{align*}
\]

3. Reasoning

Semantics Parser Induction: Learn semantic parser (weighted transduction) from parallel text/meaning data, constrained SMT task.
Data-driven Semantic Parsing in a Nutshell

- **Training**
  - Parallel Training Set: \( D = \{(x_i, z_i)\}_{i}^{D} \) → Machine Learner

- **Testing**
  - Input: \( x \) → Semantic Parsing → \( z \) → reasoning → world
  - Decoding

**Desiderata**: robust and domain agnostic models that require **minimal** amounts of hand engineering and data supervision.
Data-driven Semantic Parsing in a Nutshell

Training

Parallel Training Set

\[ D = \{(x_i, z_i)\}_{i=1}^{D} \]

\[ \text{Machine Learner} \]

Testing

\[ \text{Testing} \]

\[ \text{input} \]

\[ \text{Semantic Parsing} \]

\[ \text{decoding} \]

\[ \text{world} \]

\[ \text{reasoning} \]

\[ \text{sem} \]

\[ \text{Desiderata}: \] robust and domain agnostic models that require minimal amounts of hand engineering and data supervision.
Data-driven Semantic Parsing in a Nutshell

- **Challenge 1: Getting data?**
  - 2014[LREC], 2017c[INLG], 2017b[ACL], 2017a[EMNLP]

- **Challenge 2: Missing Data?**
  - 2018[NAACL]

- **Challenge 3: Deficient LFs?**
  - 2012[COLING], 2016[TACL]

**Training**

Parallel Training Set:

\[ D = \left\{ (x_i, z_i) \right\}_{i}^{\left| D \right|} \]

- **Machine Learner**

**Testing**

- **Input** \( x \) → **Semantic Parsing** → **Decoding** → **Model** → **Output** \( z \)

**Desiderata:** robust and domain agnostic models that require minimal amounts of hand engineering and data supervision.
Desiderata: robust and domain agnostic models that require minimal amounts of hand engineering and data supervision.
challenge 1: Getting data?

Training

Parallel Training Set

\[ D = \left\{ (x_i, z_i) \right\}_{i}^{\left| D \right|} \]

→ Machine Learner

\[ \ldots \]

model
Learning from LFs: Pairs of text \( x \) and logical forms \( z \), \( D = \{(x, z)_i\}_{i=1}^n \),
learn \( \text{sem}: x \rightarrow z \)

Modularity: Study the translation independent of other semantic issues.
Semantic Parsing and Parallel Data

What state has the largest population?

\[ z = \text{argmax} (\lambda x. \text{(state } x) \lambda x. \text{(population } x)) \]

- **Learning from LFs**: Pairs of text \( x \) and logical forms \( z \), \( D = \{(x, z)\}_i^n \), learn \( \text{sem} : x \rightarrow z \)

- **Modularity**: Study the translation independent of other semantic issues.

- **Underlying Challenge**: Finding parallel data tends to require considerable hand engineering effort (cf. Wang et al. (2015)).
Returns the greater of two long values
@param a an argument
@param b another argument
@return the larger of a and b
@see java.lang.Long#MAX_VALUE

```java
public static Long max(long a, long b)
```

**Source Code Documentation:** High-level descriptions of internal software functionality paired with code.
Source Code and API Documentation

```
* Returns the greater of two long values
*
* @param a an argument
* @param b another argument
* @return the larger of a and b
* @see java.lang.Long#MAX_VALUE
*/

public static Long max(long a, long b)
```

- **Source Code Documentation:** High-level descriptions of internal software functionality paired with code.
- **Idea:** Treat as a parallel corpus (Allamanis et al., 2015; Gu et al., 2016; Iyer et al., 2016), or synthetic semantic parsing dataset.
Tight coupling between high-level text and code, easy to extract text/code pairs automatically.

* Returns the greater of two long values
* @param a an argument
* @param b another argument
* @return the larger of a and b
* @see java.lang.Long#MAX

public static Long max(long a, long b)

(extraction)

(text) Returns the greater...
(code) lang.Math long max( long... )

(ns ... clojure.core)

(defn random-sample
  "Returns items from coll with random probability of prob (0.0 - 1.0)"
  ([prob] ...) ([prob coll] ...))

(extraction)

(text) Returns items from coll...
(code) (core.random-sample prob...)
Source Code as a Parallel Corpus

- Tight coupling between high-level text and code, easy to extract text/code pairs automatically.

```
* Returns the greater of two long values
* @param a an argument
* @param b another argument
* @return the larger of a and b
* @see java.lang.Long#MAX_VALUE
*/
public static Long max(long a, long b)
```

<table>
<thead>
<tr>
<th>text</th>
<th>Returns the greater...</th>
</tr>
</thead>
<tbody>
<tr>
<td>code</td>
<td>lang.Math long max( long... )</td>
</tr>
</tbody>
</table>

```
(ns ... clojure.core)

(defn random-sample
  "Returns items from coll with random probability of prob (0.0 - 1.0)"
  ([prob] ...)
  ([prob coll] ...))
```

<table>
<thead>
<tr>
<th>text</th>
<th>Returns items from coll...</th>
</tr>
</thead>
<tbody>
<tr>
<td>code</td>
<td>(core.random-sample prob...)</td>
</tr>
</tbody>
</table>

- **Function signatures:** Header-like representations, containing function name, arguments, return value, namespace.

```
Signature ::= lang Math long max ( long a, long b )
```

```
namespace class return name named/typed arguments
```
## Resource 1: Standard Library Documentation (Stdlib)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Pairs</th>
<th>#Symbols</th>
<th>#Words</th>
<th>Vocab.</th>
<th>Example Pairs (x, z)</th>
</tr>
</thead>
</table>
| Java     | 7,183  | 4,072    | 82,696   | 3,721  | x: Compares this Calendar to the specified Object.  
z: boolean util.Calendar.equals(Object obj)                                                                                                        |
| Ruby     | 6,885  | 3,803    | 67,274   | 5,131  | x: Computes the arc tangent given y and x.  
z: Math.atan2(y,x) → Float                                                                                                                                 |
| PHP<sub>en</sub> | 6,611 | 8,308 | 68,921    | 4,874  | x: Delete an entry in the archive using its name.  
z: bool ZipArchive::deleteName(string $name)                                                                                                        |
| Python   | 3,085  | 3,991    | 27,012   | 2,768  | x: Remove the specific filter from this handler.  
z: logging.Filterer.removeFilter(filter)                                                                                                          |
| Elisp    | 2,089  | 1,883    | 30,248   | 2,644  | x: Returns the total height of the window.  
z: (window-total-height window round)                                                                                                               |
| Geoquery | 880    | 167      | 6,663    | 279    | x: What is the tallest mountain in America?  
z: (highest(mountain(loc_2(countryid usa))))                                                                                                        |

- Documentation for 16 APIs, 10 programming languages, 7 natural languages, from Richardson and Kuhn (2017b).
  
  - **Advantages:** zero annotation, highly multilingual, relatively large.
### Resource 1: Standard Library Documentation (Stdlib)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Pairs</th>
<th>#Symbols</th>
<th>#Words</th>
<th>Vocab.</th>
<th>Example Pairs (x, z)</th>
</tr>
</thead>
</table>
| Java    | 7,183  | 4,072    | 82,696 | 3,721  | x: Compares this Calendar to the specified Object.  
z: boolean util.Calendar.equals(Object obj) |
| Ruby    | 6,885  | 3,803    | 67,274 | 5,131  | x: Computes the arc tangent given y and x.  
z: Math.atan2(y,x) → Float |
| PHP<sub>en</sub> | 6,611 | 8,308    | 68,921 | 4,874  | x: Delete an entry in the archive using its name.  
z: bool ZipArchive::deleteName(string $name) |
| Python  | 3,085  | 3,991    | 27,012 | 2,768  | x: Remove the specific filter from this handler.  
z: logging.Filterer.removeFilter(filter) |
| Elisp   | 2,089  | 1,883    | 30,248 | 2,644  | x: Returns the total height of the window.  
z: (window-total-height window round) |
| Geoquery| 880    | 167      | 6,663  | 279    | x: What is the tallest mountain in America?  
z: (highest(mountain(loc_2(countryid usa)))) |

- Documentation for 16 APIs, 10 programming languages, 7 natural languages, from Richardson and Kuhn (2017b).

- **Advantages:** zero annotation, highly multilingual, relatively large.
## Resource 2: Python Projects (Py27)

<table>
<thead>
<tr>
<th>Project</th>
<th># Pairs</th>
<th># Symbols</th>
<th># Words</th>
<th>Vocab.</th>
</tr>
</thead>
<tbody>
<tr>
<td>scapy</td>
<td>757</td>
<td>1,029</td>
<td>7,839</td>
<td>1,576</td>
</tr>
<tr>
<td>zipline</td>
<td>753</td>
<td>1,122</td>
<td>8,184</td>
<td>1,517</td>
</tr>
<tr>
<td>biopython</td>
<td>2,496</td>
<td>2,224</td>
<td>20,532</td>
<td>2,586</td>
</tr>
<tr>
<td>renpy</td>
<td>912</td>
<td>889</td>
<td>10,183</td>
<td>1,540</td>
</tr>
<tr>
<td>pyglet</td>
<td>1,400</td>
<td>1,354</td>
<td>12,218</td>
<td>2,181</td>
</tr>
<tr>
<td>kivy</td>
<td>820</td>
<td>861</td>
<td>7,621</td>
<td>1,456</td>
</tr>
<tr>
<td>pip</td>
<td>1,292</td>
<td>1,359</td>
<td>13,011</td>
<td>2,201</td>
</tr>
<tr>
<td>twisted</td>
<td>5,137</td>
<td>3,129</td>
<td>49,457</td>
<td>4,830</td>
</tr>
<tr>
<td>vispy</td>
<td>1,094</td>
<td>1,026</td>
<td>9,744</td>
<td>1,740</td>
</tr>
<tr>
<td>orange</td>
<td>1,392</td>
<td>1,125</td>
<td>11,596</td>
<td>1,761</td>
</tr>
<tr>
<td>tensorflow</td>
<td>5,724</td>
<td>4,321</td>
<td>45,006</td>
<td>4,672</td>
</tr>
<tr>
<td>pandas</td>
<td>1,969</td>
<td>1,517</td>
<td>17,816</td>
<td>2,371</td>
</tr>
<tr>
<td>sqlalchemy</td>
<td>1,737</td>
<td>1,374</td>
<td>15,606</td>
<td>2,039</td>
</tr>
<tr>
<td>pyspark</td>
<td>1,851</td>
<td>1,276</td>
<td>18,775</td>
<td>2,200</td>
</tr>
<tr>
<td>nupic</td>
<td>1,663</td>
<td>1,533</td>
<td>16,750</td>
<td>2,135</td>
</tr>
<tr>
<td>astropy</td>
<td>2,325</td>
<td>2,054</td>
<td>24,567</td>
<td>3,007</td>
</tr>
<tr>
<td>sympy</td>
<td>5,523</td>
<td>3,201</td>
<td>52,236</td>
<td>4,777</td>
</tr>
<tr>
<td>ipython</td>
<td>1,034</td>
<td>1,115</td>
<td>9,114</td>
<td>1,771</td>
</tr>
<tr>
<td>orator</td>
<td>817</td>
<td>499</td>
<td>6,511</td>
<td>670</td>
</tr>
<tr>
<td>obspy</td>
<td>1,577</td>
<td>1,861</td>
<td>14,847</td>
<td>2,169</td>
</tr>
<tr>
<td>rdkit</td>
<td>1,006</td>
<td>1,380</td>
<td>9,758</td>
<td>1,739</td>
</tr>
<tr>
<td>django</td>
<td>2,790</td>
<td>2,026</td>
<td>31,531</td>
<td>3,484</td>
</tr>
<tr>
<td>ansible</td>
<td>2,124</td>
<td>1,884</td>
<td>20,677</td>
<td>2,593</td>
</tr>
<tr>
<td>statsmodels</td>
<td>2,357</td>
<td>2,352</td>
<td>21,716</td>
<td>2,733</td>
</tr>
<tr>
<td>theano</td>
<td>1,223</td>
<td>1,364</td>
<td>12,018</td>
<td>2,152</td>
</tr>
<tr>
<td>nltk</td>
<td>2,383</td>
<td>2,324</td>
<td>25,823</td>
<td>3,151</td>
</tr>
<tr>
<td>sklearn</td>
<td>1,532</td>
<td>1,519</td>
<td>13,897</td>
<td>2,115</td>
</tr>
<tr>
<td>geoquery</td>
<td>880</td>
<td>167</td>
<td>6,663</td>
<td>279</td>
</tr>
</tbody>
</table>

- 27 English Python projects from Github (Richardson and Kuhn, 2017a).
New Task: Text to Signature Translation

<table>
<thead>
<tr>
<th>text</th>
<th>Returns the greater of two long values</th>
</tr>
</thead>
<tbody>
<tr>
<td>signature</td>
<td>lang.Math long max(long a, long b)</td>
</tr>
</tbody>
</table>

- **Task**: Given text/signatures training pairs, learn a (quasi) semantic parser: text → signature (Richardson and Kuhn, 2017b)

- **Assumption**: predicting within finite signature/translation space.
New Task: Text to Signature Translation

<table>
<thead>
<tr>
<th>text</th>
<th>Returns the greater of two long values</th>
</tr>
</thead>
<tbody>
<tr>
<td>signature</td>
<td>lang.Math long max( long a, long b )</td>
</tr>
</tbody>
</table>

- **Task**: Given text/signatures training pairs, learn a (quasi) *semantic parser*: text → signature (Richardson and Kuhn, 2017b)

- **Assumption**: predicting within finite signature/translation space.

- **Code Retrieval Analogy**: train/test split, at test time, *retrieve* function signature that matches input *specification* (Deng and Chrupała, 2014):

  ```
  × string APCIterator::key(void)
  × int APCIterator::getTotalHits(void)
  × int APCIterator::getSize(void)
  int APCIterator::getTotalSize(void)
  × int Memcached::append(string $key)
  ...
  ```

  Accuracy @1? (exact match)
Text to Signature Translation: How Hard Is It?

**Initial approach**: noisy-channel (nc) classical translation:

\[
\text{SEMPAR}^{nc}(x, z) = p_{\theta}(x \mid z) \times \underbrace{p_{\text{lm}}(z)}_{\text{valid expression (yes/no)}}
\]
Initial approach: noisy-channel (nc) classical translation:

\[
\text{SEMPAR}^{nc}(x, z) = p_\theta(x \mid z) \times p_{1m}(z)
\]

- \(\text{lm}\): convenient for making strong assumptions about our output language, facilitates constrained decoding.
Initial approach: noisy-channel (nc) classical translation:

\[
\text{SEMPAR}^{nc}(x, z) = p_{\theta}(x \mid z) \times \underbrace{p_{lm}(z)}_{\text{trans model}} \quad \text{valid expression (yes/no)}?
\]

1. \text{lm}: convenient for making strong assumptions about our output language, facilitates constrained decoding.

2. \text{code case}: make assumptions about what constitutes a valid function in a given API.
Text to Signature Translation: How Hard Is It?

- **Our Approach**: Lexical translation model (standard estimation), discriminative reranker, hard constraints on $p(z)$. 
Our Approach: Lexical translation model (standard estimation), discriminative reranker, hard constraints on $p(z)$. 
Text to Signature Translation: How Hard Is It?

- **Our Approach**: Lexical translation model (standard estimation), discriminative reranker, hard constraints on $p(z)$. 

![Graph showing accuracy comparison between different models](chart.png)

- Our Noisy-Channel SMT Model
- Discriminative Reranker
- Competitor Model/Baseline
- Blackbox PBSMT (Moses)
Text to Signature Translation: How Hard Is It?

- **Our Approach**: Lexical translation model (standard estimation), discriminative reranker, hard constraints on $p(z)$.
Text to Signature Translation: How Hard Is It?

- **Our Approach**: Lexical translation model (standard estimation), discriminative reranker, hard constraints on $p(z)$.
Our Approach: Lexical translation model (standard estimation), discriminative reranker, hard constraints on $p(z)$.

Result: achieving high accuracy is not easy, not a trivial problem.
Text to Signature Translation: How Hard Is It?

- **Our Approach**: Lexical translation model (standard estimation), discriminative reranker, hard constraints on $p(z)$. 
Our Approach: Lexical translation model (standard estimation), discriminative reranker, hard constraints on $p(z)$.
Our Approach: Lexical translation model (standard estimation),
discriminative reranker, hard constraints on $p(z)$.

<table>
<thead>
<tr>
<th>text</th>
<th>Returns the index of the first occurrence of char in the string</th>
</tr>
</thead>
<tbody>
<tr>
<td>Moses</td>
<td>(start end occurrence lambda char string string string)</td>
</tr>
</tbody>
</table>
Our Approach: Lexical translation model (standard estimation), discriminative reranker, hard constraints on $p(z)$.

Observation: semantic parsing is not an unconstrained MT problem.
What do these results mean? Code Retrieval Again

<table>
<thead>
<tr>
<th>Dataset (Avg.)</th>
<th>Accuracy @1 (average)</th>
<th>Accuracy @10 (average)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stdlib</td>
<td>31.1</td>
<td>71.0</td>
</tr>
<tr>
<td>Py27</td>
<td>32.3</td>
<td>73.5</td>
</tr>
</tbody>
</table>
so far: Semantic parsing as constrained translation, API as parallel corpus.

</Challenge 1>
Traditional approaches to semantic parsing train individual models for each available parallel dataset.

**Underlying Challenge:** Datasets tend to be small, hard and unlikely to get certain types of parallel data, e.g., (de, Haskell).
Ideally, we want each dataset to have tens of thousands of documented functions.

Most projects have 500 or less documented functions.
Polyglot Models: Training on Multiple Datasets

- Idea: concatenate all datasets into one, build a single-model with shared parameters, capture redundancy (Herzig and Berant, 2017).

- Polyglot Translator: translates from any input language to any output (programming) language.
Polyglot Models: Training on Multiple Datasets

- Idea: concatenate all datasets into one, build a single-model with shared parameters, capture redundancy (Herzig and Berant, 2017).

- Polyglot Translator: translates from any input language to any output (programming) language.

1. Multiple Datasets: Does this help learn better translators?

2. Zero-Short Translation (Johnson et al., 2016): Can we translate between different APIs and unobserved language pairs?
Graph Based Approach

- **Requirements**: must generate well-formed output, be able to translate to target languages on demand.

- **Idea**: Exploit finite-ness of translation space, represent full search space as directed acyclic graph (DAG), add *artifical language tokens.*
**Requirements:** must generate well-formed output, be able to translate to target languages on demand.

**Idea:** Exploit finite-ness of translation space, represent full search space as directed acyclic graph (DAG), add *artifical language tokens*.

**Decoding** (test time): Reduces to finding a path given an input \( x \): 

\[ x : \text{The ceiling of a number} \]

We formulate search in terms of single source shortest shortest-path (SSSP) search (Cormen et al., 2009) on DAGs.
Shortest Path Decoding in a Nutshell

- **Standard SSSP**: Traverse labeled edges $E$ (label $z$) in order (e.g., sorted or best-first order), and solve for each node $v$ the following recurrence:

$$d[v] = \min_{(u,v,z) \in E} \left\{ d[u] + w(u, v, z) \right\}$$

- Use trained translation model to dynamically weight edges, general framework for directly comparing models (Richardson et al., 2018).
- Constrained decoding: ensure that output is well-formed, related efforts: Krishnamurthy et al. (2017); Yin and Neubig (2017).
Shortest Path Decoding in a Nutshell

- **Standard SSSP**: Traverse labeled edges $E$ (label $z$) in order (e.g., sorted or best-first order), and solve for each node $v$ the following recurrence:

  $$d[v] = \min_{(u,v,z) \in E} \left\{ d[u] + \text{TRANS}(x, z) \right\}$$

  - node score
  - incoming node score
  - translation

- Use trained **translation model** to dynamically weight edges, general framework for directly comparing models (Richardson et al., 2018).
Shortest Path Decoding in a Nutshell

- **Standard SSSP**: Traverse labeled edges $E$ (label $z$) in order (e.g., sorted or best-first order), and solve for each node $v$ the following recurrence:

$$d[v] = \min_{(u,v,z) \in E} \left\{ d[u] + \text{TRANS}(x, z) \right\}$$

- **Use trained translation model** to dynamically weight edges, general framework for directly comparing models (Richardson et al., 2018).

- **Constrained decoding**: ensure that output is well-formed, related efforts: Krishnamurthy et al. (2017); Yin and Neubig (2017).
DAG Decoding for Neural Semantic Parsing (Example)

- **Seq2Seq**: popular in semantic parsing (Dong and Lapata, 2016), variants of (Bahdanau et al., 2014), direct decoder model (unconstrained):

\[
p(z \mid x) = \text{CONDITIONALRNNLM}(z)
\]

\[
= \prod_{i} p_{\Theta}(z_i \mid z_{<i}, x)
\]
DAG Decoding for Neural Semantic Parsing (Example)

- **Seq2Seq**: popular in semantic parsing (Dong and Lapata, 2016), variants of (Bahdanau et al., 2014), direct decoder model (unconstrained):

\[
p(z \mid x) = \text{CONDITIONALRNNLM}(z)
\]

\[
= \prod_{i} p_{\Theta}(z_i \mid z_{<i}, x)
\]

- **DAGs** \( \mathcal{G} = (V, E) \), **numerically sorted nodes** (acyclic), trained decoder.
DAG Decoding for Neural Semantic Parsing (Example)

- **Seq2Seq**: popular in semantic parsing (Dong and Lapata, 2016), variants of (Bahdanau et al., 2014), direct decoder model (unconstrained):

\[
p(z \mid x) = \text{CONDITIONAL RNNLM}(z) = \prod_{i} p_{\Theta}(z_i \mid z_{<i}, x)
\]

- **DAGs** \( G = (V, E) \), **numerically sorted nodes** (acyclic), trained decoder.

0: \( d[b] \leftarrow 0.0 \)
1: for node \( v \in V \) in topologically sorted order
DAG Decoding for Neural Semantic Parsing (Example)

- **Seq2Seq**: popular in semantic parsing (Dong and Lapata, 2016), variants of (Bahdanau et al., 2014), direct decoder model (unconstrained):

\[ p(z \mid x) = \text{CONDITIONALRNNLM}(z) \]
\[ = \prod_{i} p_{\Theta}(z_{i} \mid z_{<i}, x) \]

- **DAGs** \( G = (V, E) \), **numerically sorted nodes** (acyclic), trained decoder.

0: \( d[b] \leftarrow 0.0 \)
1: for node \( v \in V \) in topologically sorted order
2: do \( d(v) = \min_{(u,v,z_{j}) \in E} \left\{ d(u) + -\log p_{\Theta}(z_{j} \mid z_{<j}, x) \right\} \)
DAG Decoding for Neural Semantic Parsing (Example)

- **Seq2Seq**: popular in semantic parsing (Dong and Lapata, 2016), variants of (Bahdanau et al., 2014), direct decoder model (unconstrained):

\[
p(z \mid x) = \text{CONDITIONAL RNNLM}(z) = \prod_i p_\Theta(z_i \mid z_{<i}, x)
\]

- **DAGs** \( G = (V, E) \), numerically sorted nodes (acyclic), trained decoder.

0: \( d[b] \leftarrow 0.0 \)
1: **for** node \( v \in V \) in topologically sorted order
2: **do** \( d(v) = \min_{(u,v,z_j) \in E} \{ d(u) + - \log p_\Theta(z_j \mid z_{<j}, x) \} \)
3: \( s[v] \leftarrow \text{RNN state for min edge and } z_j \)
DAG Decoding for Neural Semantic Parsing (Example)

- **Seq2Seq**: popular in semantic parsing (Dong and Lapata, 2016), variants of (Bahdanau et al., 2014), direct decoder model (unconstrained):

\[
p(z \mid x) = \text{CONDITIONALRNNLM}(z) = \prod_{i} p_{\Theta}(z_i \mid z_{<i}, x)
\]

- **DAGs** \( G = (V, E) \), **numerically sorted nodes** (acyclic), trained decoder.

0: \( d[b] \leftarrow 0.0 \)
1: for node \( v \in V \) in topologically sorted order
2: \( \text{do } d(v) = \min_{(u,v,z_j) \in E} \left\{ d(u) + -\log p_{\Theta}(z_j \mid z_{<j}, x) \right\} \)
3: \( s[v] \leftarrow \text{RNN state for min edge and } z_j \)
4: \( \text{return } \min_{v \in V} \{ d(v) \} \)
Training on Multiple Datasets: Does this help?

- **Strategy**: train models on multiple datasets (polyglot models), decoding to target languages and check for improvement.

<table>
<thead>
<tr>
<th>Multilingual Geoquery</th>
<th>Method</th>
<th>Acc@1 (averaged)</th>
</tr>
</thead>
<tbody>
<tr>
<td>mono.</td>
<td>UBL Kwiatkowski et al. (2010)</td>
<td>74.2</td>
</tr>
<tr>
<td></td>
<td>TreeTrans Jones et al. (2012)</td>
<td>76.8</td>
</tr>
<tr>
<td></td>
<td>Lexical SMT SSSP</td>
<td>68.6</td>
</tr>
<tr>
<td></td>
<td>Best Seq2Seq SSSP</td>
<td>78.0</td>
</tr>
<tr>
<td>poly.</td>
<td>Lexical SMT SSSP</td>
<td>67.3</td>
</tr>
<tr>
<td></td>
<td>Best Seq2Seq SSSP</td>
<td>79.6</td>
</tr>
</tbody>
</table>
Training on Multiple Datasets: Does this help?

**Strategy**: train models on multiple datasets (polyglot models), decoding to target languages and check for improvement.

<table>
<thead>
<tr>
<th>Multilingual Geoquery</th>
<th>Method</th>
<th>Acc@1 (averaged)</th>
</tr>
</thead>
<tbody>
<tr>
<td>stdlib</td>
<td>UBL Kwiatkowski et al. (2010)</td>
<td>74.2</td>
</tr>
<tr>
<td></td>
<td>TreeTrans Jones et al. (2012)</td>
<td>76.8</td>
</tr>
<tr>
<td></td>
<td>Lexical SMT SSSP</td>
<td>68.6</td>
</tr>
<tr>
<td></td>
<td>Best Seq2Seq SSSP</td>
<td>78.0</td>
</tr>
<tr>
<td>poly.</td>
<td>Lexical SMT SSSP</td>
<td>67.3</td>
</tr>
<tr>
<td></td>
<td>Best Seq2Seq SSSP</td>
<td>79.6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Acc@1</th>
<th>Acc@10</th>
<th>MRR</th>
</tr>
</thead>
<tbody>
<tr>
<td>stdlib</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mono.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Best Monolingual Model</td>
<td>29.9</td>
<td>69.2</td>
<td>43.1</td>
</tr>
<tr>
<td>Lexical SMT SSSP</td>
<td>33.2</td>
<td>70.7</td>
<td>45.9</td>
</tr>
<tr>
<td>Best Seq2Seq SSSP</td>
<td>13.9</td>
<td>36.5</td>
<td>21.5</td>
</tr>
<tr>
<td>poly.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Best Monolingual Model</td>
<td>32.4</td>
<td>73.5</td>
<td>46.5</td>
</tr>
<tr>
<td>Lexical SMT SSSP</td>
<td>41.3</td>
<td>77.7</td>
<td>54.1</td>
</tr>
<tr>
<td>Best Seq2Seq SSSP</td>
<td>9.0</td>
<td>26.9</td>
<td>15.1</td>
</tr>
</tbody>
</table>
Training on Multiple Datasets: Does this help?

**Strategy**: train models on multiple datasets (polyglot models), decoding to target languages and check for improvement.

<table>
<thead>
<tr>
<th>Multilingual Geouery</th>
<th>Method</th>
<th>Acc@1 (averaged)</th>
</tr>
</thead>
<tbody>
<tr>
<td>mono.</td>
<td>UBL Kwiatkowski et al. (2010)</td>
<td>74.2</td>
</tr>
<tr>
<td></td>
<td>TreeTrans Jones et al. (2012)</td>
<td>76.8</td>
</tr>
<tr>
<td></td>
<td>Lexical SMT SSSP</td>
<td>68.6</td>
</tr>
<tr>
<td>poly.</td>
<td>Best Seq2Seq SSSP</td>
<td>78.0</td>
</tr>
<tr>
<td></td>
<td>Lexical SMT SSSP</td>
<td>67.3</td>
</tr>
<tr>
<td></td>
<td>Best Seq2Seq SSSP</td>
<td>79.6</td>
</tr>
</tbody>
</table>

**Findings**: Polyglot modeling can help improve accuracy depending on the model used, Seq2Seq models did not perform well on code datasets.
**New Tasks: Any/Mixed Language Decoding**

▶ **Any Language Decoding**: translating between multiple APIs, letting the decoder decide output language.

<table>
<thead>
<tr>
<th></th>
<th>Source API (stdlib): (<strong>es</strong>, <strong>PHP</strong>)</th>
<th>Input: Devuelve el mensaje asociado al objeto lanzado.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output</td>
<td>Language: <strong>PHP</strong></td>
<td>Translation: public string Throwable::getMessage ( void )</td>
</tr>
<tr>
<td></td>
<td>Language: <strong>Java</strong></td>
<td>Translation: public String lang.getMessage( void )</td>
</tr>
<tr>
<td></td>
<td>Language: <strong>Clojure</strong></td>
<td>Translation: (tools.logging.fatal throwable message &amp; more)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Source API (stdlib): (<strong>ru</strong>, <strong>PHP</strong>)</th>
<th>Input: конвертирует строку из формата UTF-32 в формат UTF-16.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output</td>
<td>Language: <strong>PHP</strong></td>
<td>Translation: string PDF=utf32.to_utf16 ( ... )</td>
</tr>
<tr>
<td></td>
<td>Language: <strong>Ruby</strong></td>
<td>Translation: String#utf16 =&gt; string</td>
</tr>
<tr>
<td></td>
<td>Language: <strong>Haskell</strong></td>
<td>Translation: Encoding.encodeUtf16LE :: Text -&gt; ByteString</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Source API (py): (<strong>en</strong>, <strong>stats</strong>)</th>
<th>Input: Compute the Moore-Penrose pseudo-inverse of a matrix.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output</td>
<td>Project: <strong>sympy</strong></td>
<td>Translation: matrices.matrix.base.pinv_solve( B, ... )</td>
</tr>
<tr>
<td></td>
<td>Project: <strong>sklearn</strong></td>
<td>Translation: utils.pinvh( a, cond=None,rcond=None,... )</td>
</tr>
<tr>
<td></td>
<td>Project: <strong>stats</strong></td>
<td>Translation: tools.pinv2( a,cond=None,rcond=None )</td>
</tr>
</tbody>
</table>
New Tasks: Any/Mixed Language Decoding

- **Any Language Decoding**: translating between multiple APIs, letting the decoder decide output language.

<table>
<thead>
<tr>
<th>Source API (stdlib): (es, PHP)</th>
<th><strong>Input</strong>: Devuelve el mensaje asociado al objeto lanzado.</th>
</tr>
</thead>
</table>
| **Output**                    | Language: **PHP**  Translation: public string Throwable::getMessage ( void )  
|                               | Language: **Java** Translation: public String lang.getMessage( void )  
|                               | Language: **Clojure** Translation: (tools.logging.fatal throwable message & more)  |

<table>
<thead>
<tr>
<th>Source API (stdlib): (ru, PHP)</th>
<th><strong>Input</strong>: конвертирует строку из формата UTF-32 в формат UTF-16.</th>
</tr>
</thead>
</table>
| **Output**                    | Language: **PHP**  Translation: string PDF_utf32_to_utf16 ( ... )  
|                               | Language: **Ruby**  Translation: String#toutf16 => string  
|                               | Language: **Haskell** Translation: Encoding.encodeUtf16LE :: Text -> ByteString  |

<table>
<thead>
<tr>
<th>Source API (py): (en, stats)</th>
<th><strong>Input</strong>: Compute the Moore-Penrose pseudo-inverse of a matrix.</th>
</tr>
</thead>
</table>
| **Output**                   | Project: **sympy**  Translation: matrices.matrix.base.pinv_solve( B, ... )  
|                              | Project: **sklearn** Translation: utils.pinvh( a, cond=None,rcond=None,... )  
|                              | Project: **stats**  Translation: tools.pinv2( a,cond=None,rcond=None )  |

**Challenge 2**: Can be used for finding missing data, data augmentation.
New Tasks: Any/Mixed Language Decoding

- **Any Language Decoding**: translating between multiple APIs, letting the decoder decide output language.

<table>
<thead>
<tr>
<th>Source API (stdlib):</th>
<th>Input:</th>
<th>Output:</th>
</tr>
</thead>
</table>
| 1. Source API (stdlib): (es, PHP) | Devuelve el mensaje asociado al objeto lanzado. | **Output Language:** PHP  
Translation: public string Throwable::getMessage ( void )  
Language: Java  
Translation: public String lang.getMessage( void )  
Language: Clojure  
Translation: (tools.logging.fatal throwable message & more) |
| 2. Source API (stdlib): (ru, PHP) | конвертирует строку из формата UTF-32 в формат UTF-16. | **Output Language:** PHP  
Translation: string PDF_utf32_to_utf16 ( ... )  
Language: Ruby  
Translation: String#toutf16 => string  
Language: Haskell  
Translation: Encoding.encodeUtf16LE :: Text -> ByteString |
| 3. Source API (py): (en, stats) | Compute the Moore-Penrose pseudo-inverse of a matrix. | **Project:** sympy  
Translation: matrices.matrix.base.pinv_solve( B, ... )  
Project: sklearn  
Translation: utils.pinvh( a, cond=None,rcond=None,... )  
Project: stats  
Translation: tools.pinv2( a,cond=None,rcond=None ) |

**Challenge 2**: Can be used for finding missing data, data augmentation.
New Tasks: Any/Mixed Language Decoding

- **Mixed Language Decoding**: translating from input with NPs from multiple languages, introduced a new mixed GeoQuery test set.

<table>
<thead>
<tr>
<th>Mixed Lang.</th>
<th>Input: Wie hoch liegt der höchstgelegene punkt in Αλαμπάμα?</th>
</tr>
</thead>
<tbody>
<tr>
<td>LF: answer(elevation_1(highest(place(loc_2(stateid('alabama'))))))</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy (averaged)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mixed Best Monolingual Seq2Seq</td>
<td>4.2</td>
</tr>
<tr>
<td>Polyglot Seq2Seq</td>
<td>75.2</td>
</tr>
</tbody>
</table>
</Challenge 2>
Challenge 3: Missing Data?

**Training**

Parallel Training Set

\[ D = \{(x_i, z_i)\}_{i}^{|D|} \]

- Machine Learner

**Testing**

- Input: \( x \)
- Semantic Parsing: \( \text{decoding} \)
- Reasoning: \( \text{world} \)
- Model: \( \text{sem} \)
- \( z \)
Semantic Parsing and Entailment

▶ **Entailment**: One of the *basic aims* of semantics (Montague, 1970).
▶ Representations should be *grounded* in judgements about entailment.

![Diagram showing semantic parsing and entailment](image)
Entailment: One of the basic aims of semantics (Montague, 1970).

Representations should be grounded in judgements about entailment.

Entailment as a Unit Test: For a set of target sentences, check that our semantic model (via some analysis for each sentence, e.g., an LF) accounts for particular entailment patterns observed between pairs of sentences; modify our model when such tests fail.

<table>
<thead>
<tr>
<th>sentence</th>
<th>analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>t <em>All samples that contain a major element</em></td>
<td>LFₜ</td>
</tr>
<tr>
<td>h <em>Some sample that contains a major element</em></td>
<td>LFₜ</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>inference</th>
<th>Entailment (RTE¹)</th>
</tr>
</thead>
<tbody>
<tr>
<td>t → h</td>
<td></td>
</tr>
<tr>
<td>h → t</td>
<td>Unknown (RTE)</td>
</tr>
</tbody>
</table>

¹Would a person reading t ordinarily infer h? (Dagan et al., 2005)
Semantic Parsing and Entailment

▶ **Question**: What happens if we *unit test* our semantic parsers?

▶ **Sportscaster**: \(\approx 1,800\) Robocup soccer descriptions paired with logical forms (LFs) (Chen and Mooney, 2008).

<table>
<thead>
<tr>
<th>sentence</th>
<th>analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>(t)  \textit{Pink 3 passes to Pink 7}</td>
<td>\textit{pass(pink3,pink7)}</td>
</tr>
<tr>
<td>(h)  \textit{Pink 3 quickly kicks to Pink 7}</td>
<td>\textit{pass(pink3,pink7)}</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>inference</th>
<th>t (\rightarrow) h</th>
<th>analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>(human)</td>
<td></td>
<td>\textit{Unknown (RTE)}</td>
</tr>
<tr>
<td>(LF match)</td>
<td></td>
<td>\textit{Entail (RTE)}</td>
</tr>
</tbody>
</table>
Question: What happens if we *unit test* our semantic parsers?

Sportscaster: ≈1,800 Robocup soccer descriptions paired with logical forms (LFs) (Chen and Mooney, 2008).

<table>
<thead>
<tr>
<th>sentence</th>
<th>analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>t The pink goalie passes to pink 7</td>
<td>pass(pink1,pink7)</td>
</tr>
<tr>
<td>h Pink 1 kicks the ball</td>
<td>kick(pink1)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>inference</th>
<th>t → h</th>
<th>analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>(human)</td>
<td>Entail (RTE)</td>
<td></td>
</tr>
<tr>
<td>(LF match)</td>
<td>Contradict (RTE)</td>
<td></td>
</tr>
</tbody>
</table>
Question: What happens if we unit test our semantic parsers?

Sportscaster: \( \approx 1,800 \) Robocup soccer descriptions paired with logical forms (LFs) (Chen and Mooney, 2008).

<table>
<thead>
<tr>
<th>Inference Model</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Majority Baseline</td>
<td>33.1%</td>
</tr>
<tr>
<td>RTE Classifier</td>
<td>52.4%</td>
</tr>
<tr>
<td>LF Matching</td>
<td>59.6%</td>
</tr>
</tbody>
</table>
Challenge 3: Deficient LFs, Missing Knowledge

- **Underlying Challenge:** Semantic representations are underspecified, fail to capture entailments, background knowledge missing.

- **Goal:** Capture the missing knowledge and inferential properties of text, incorporate entailment information into learning.
Challenge 3: Deficient LFs, Missing Knowledge

▶ **Underlying Challenge:** Semantic representations are underspecified, fail to capture entailments, background knowledge missing.

▶ **Goal:** Capture the missing knowledge and inferential properties of text, incorporate entailment information into learning.

▶ **Solution:** Use entailment information (EI) and logical inference as weak signal to train parser, jointly optimize model to reason about entailment.

<table>
<thead>
<tr>
<th>Paradigm and Supervision</th>
<th>Dataset $D =$</th>
<th>Learning Goal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learning from LFs</td>
<td>${(input_i, LF_i)}_{i}^{N}$</td>
<td>$\text{input} \xrightarrow{\text{Trans.}} \text{LF}$</td>
</tr>
<tr>
<td>Learning from Entailment</td>
<td>${(input_t, input_t', EI_i)}_{i}^{N}$</td>
<td>$(input_t, input_t') \xrightarrow{\text{Proof}} \text{EI}$</td>
</tr>
</tbody>
</table>
Learning from Entailment: Illustration

- Entailments are used to reason about target symbols and find holes in the analyses.

Data: \( D = \{((t, h)_i, z_i)\}_{i=1}^N \), generic logical calculus. Task: learn (latent) proof \( y \)
Learning from Entailment: Illustration

Entailments are used to reason about target symbols and find holes in the analyses.

Data: $D = \{(t, h)_i, z_i\}_{i=1}^N$, generic logical calculus. Task: learn (latent) proof $y$
Entailments are used to reason about target symbols and find holes in the analyses.

Data: \( D = \{(t, h)_i, z_i\}_{i=1}^N \). generic logical calculus. Task: learn (latent) proof y
Learning from Entailment: Illustration

- Entailments are used to reason about target symbols and find holes in the analyses.

Data: $D = \{(t, h), z_i\}_{i=1}^N$, generic logical calculus. Task: learn (latent) proof y
Learning from Entailment: Illustration

Entailments are used to reason about target symbols and find holes in the analyses.

Data: $D = \{(t, h_i, z_i)\}_{i=1}^{N}$, generic logical calculus. Task: learn (latent) proof $y$
Grammar Approach: Sentences to Logical Form

- Use a semantic CFG, rules constructed from target representations using small set of templates (Börschinger et al. (2011))

\[(x: \text{purple } 10 \text{ quickly kicks}, z: \{\text{kick(purple10)}, \text{block(purple7)}, \ldots\})\]

↓ (rule extraction)
Grammar Approach: Sentences to Logical Form

- Use a semantic CFG, rules constructed from target representations using small set of templates (Börschinger et al. (2011))

\((x : \text{purple 10 quickly kicks, } z : \{\text{kick(purple10), block(purple7),...}\})\)

\[\downarrow \text{(rule extraction)}\]

```
kick(purple10) kick(purple10) block(purple7) block(purple9)
```
Grammar Approach: Sentences to Logical Form

- Use a semantic CFG, rules constructed from target representations using small set of templates (Börschinger et al. (2011))

\[(x : \text{purple 10 quickly kicks}, z : \{\text{kick(purple10)}, \text{block(purple7)}, \ldots\})\]

\[
\downarrow \text{(rule extraction)}
\]

\[
\begin{array}{cccc}
\checkmark & \checkmark & \times & \times \\
\text{Rep} & \text{Rep} & \text{Rep} & \text{Rep} \\
\text{arg}_1 & \text{arg}_1 & \text{arg}_1 & \\
\text{in\_transitive} & \text{in\_transitive} & \text{in\_transitive} & \\
\text{kick}_c & \text{kick}_c & \text{block}_c & \\
\text{purple10}_c & \text{purple10}_c & \text{purple9}_c & \\
\text{purple10}_w & \text{purple10}_w & \text{purple9}_w & \\
\text{rapid10} & \text{kick}_w & \text{kick}_w & \\
\text{quickly} & \text{quickly} & \text{quickly} & \\
\text{kicks} & \text{kicks} & \text{kicks} & \\
\end{array}
\]

- kick(purple10)  kick(purple10)  block(purple7)  block(purple9)
Semantic Parsing as Grammatical Inference

- Rules used to define a PCFG $\mathcal{G}_\theta$, learn correct derivations.
- **Learning**: EM bootstrapping approach (Angeli et al., 2012), maximum (marginal) likelihood with beam search.

Purple 7 kicks to Purple 4

$Z = \{\text{pass(purple7,purple4)}\}$
Semantic Parsing as Grammatical Inference

- Rules used to define a PCFG $G_{\theta}$, learn correct derivations.

- **Learning**: EM bootstrapping approach (Angeli et al., 2012), maximum (marginal) likelihood with beam search.

$$Z = \{\text{pass(purple7,purple4)}\}$$
Semantic Parsing as Grammatical Inference

- Rules used to define a PCFG $G_\theta$, learn correct derivations.
- **Learning**: EM bootstrapping approach (Angeli et al., 2012), maximum (marginal) likelihood with beam search.

Purple 7 kicks to Purple 4

```
Z = \{\text{pass(purple7,purple4)}\}
```
Weakly-supervised semantic parsing (Liang et al., 2013; Berant et al., 2013), treat as partially-observed random process (Guu et al., 2017).

\[ x = (t, h), \ z \in \{\text{Entail, Contradict, Unknown}\} \]
Joint Entailment Modeling and Reasoning

- Weakly-supervised semantic parsing (Liang et al., 2013; Berant et al., 2013), treat as partially-observed random process (Guu et al., 2017).

\[ x = (t, h), \; z \in \{ \text{Entail, Contradict, Unknown} \} \]

\[ p(z | x) = \sum_{y \in \mathcal{Y}_x} p(z | y) \times \left\{ \begin{array}{c} \text{valid inference?} \\ \text{proofs} \end{array} \right\} \times \left\{ \begin{array}{c} \text{proof score} \\ p_\theta(y | x) \end{array} \right\} \]
Joint Entailment Modeling and Reasoning

- Weakly-supervised semantic parsing (Liang et al., 2013; Berant et al., 2013), treat as partially-observed random process (Guu et al., 2017).

\[ x = (t, h), \quad z \in \{\text{Entail, Contradict, Unknown}\} \]

\[ p(z \mid x) = \sum_{y \in \mathcal{Y}_x} \underbrace{p(z \mid y)}_{\text{proofs}} \times \underbrace{p_\theta(y \mid x)}_{\text{proof score}} \]

- \( p(z \mid y) \): 1 if proof derives correct entailment, 0 otherwise
Joint Entailment Modeling and Reasoning

- Weakly-supervised semantic parsing (Liang et al., 2013; Berant et al., 2013), treat as partially-observed random process (Guu et al., 2017).

\[ x = (t, h), \ z \in \{\text{Entail}, \text{Contradict}, \text{Unknown}\} \]

\[ p(z \mid x) = \sum_{y \in \mathcal{Y}_x} p(z \mid y) \times \frac{1}{\text{proof score}} p(\theta(y \mid x)) \]

- \( p(z \mid y) \): 1 if proof derives correct entailment, 0 otherwise
- \( p_\theta(y \mid x)\): Model proof structures and rules as PCFG, use variant of natural logic calculus (MacCartney and Manning, 2009).
Joint Entailment Modeling and Reasoning

- Weakly-supervised semantic parsing (Liang et al., 2013; Berant et al., 2013), treat as partially-observed random process (Guu et al., 2017).

\[ x = (t, h), \quad z \in \{\text{Entail}, \text{Contradict}, \text{Unknown}\} \]

\[ p(z \mid x) = \sum_{y \in \mathcal{Y}_x} \begin{cases} \begin{aligned} p(z \mid y) \times p(\theta(y \mid x)) \end{cases} \end{aligned} \]

- \( p(z \mid y) \): 1 if proof derives correct entailment, 0 otherwise
- \( p(\theta(y \mid x)) \): Model proof structures and rules as PCFG, use variant of natural logic calculus (MacCartney and Manning, 2009).

- Results in an interesting probabilistic logic, efficient proof search via reduction to (P)CFG search.
**Learning Entailment Rules**

- Integrates a symbolic reasoner directly into the semantic parser, allows for joint training using a single generative model.
- **Learning**: Grammatical inference problem as before, maximum (marginal) likelihood with beam search ($\mathcal{V}_x \approx k\text{BEST}(x)$).

\[ \text{input} \xrightarrow{\text{Beam Parser } \theta^t} \text{d} \]

**t**: pink 1 kicks

**h**: pink 1 quickly passes to pink 2

Interpretation

\[ \text{world} \]

\[ z = \text{Uncertain} \]
Reasoning about Entailment

- Improving the internal representations (before, a, after, b).

a.

b.
Reasoning about Entailment

Learned modifiers from example proofs trees.

\((t,h)\):

\(\text{(a beautiful pass to, passes to)}\)

\[
\begin{align*}
\subseteq_c &\equiv \text{play-tran} = \subseteq_c \text{play-tran} \\
\subseteq_c &\equiv \text{play-tran.} \\
\subseteq_c &\equiv \text{pass/pass} \\
\end{align*}
\]

\begin{align*}
\text{"a beautiful"/} &\lambda \text{"pass to"/"passes to"} \\
\end{align*}

\text{generalization:}

\[
\text{beautiful}(X) \subseteq X
\]

\(\text{(gets a free kick, freekick from the)}\)

\[
\begin{align*}
\subseteq_c &\equiv \text{game-play} = \subseteq_c \text{game-play} \\
\subseteq_c &\equiv \text{game-play} \\
\subseteq_c &\equiv \text{freekick/freekick} \\
\end{align*}
\]

\begin{align*}
\text{"gets a"/} &\lambda \text{"free kick" / "freekick from the"} \\
\end{align*}

\text{generalization:}

\[
\text{get}(X) \equiv X
\]

\(\text{(t, h)}\):

\(\text{(yet again passes to, kicks to)}\)

\[
\begin{align*}
\subseteq_c &\equiv \text{play-tran} = \subseteq_c \text{play-tran} \\
\subseteq_c &\equiv \text{play-tran.} \\
\subseteq_c &\equiv \text{pass/pass} \\
\end{align*}
\]

\begin{align*}
\text{"yet again"/} &\lambda \text{"passes to"/"kicks to"} \\
\end{align*}

\text{generalization:}

\[
\text{yet-again}(X) \subseteq X
\]

\(\text{(purple 10, purple 10 who is out front)}\)

\[
\begin{align*}
\subseteq_c &\equiv \text{player}_{\text{arg}2} \exists \subseteq_c \equiv \text{player}_{\text{arg}2} \\
\subseteq_c &\equiv \text{player}_{\text{arg}2} \\
\subseteq_c &\equiv \text{purple10/purple10} \\
\end{align*}
\]

\begin{align*}
\text{"purple 10"/} &\lambda \text{"purple 10" / "who is out front"} \\
\end{align*}

\text{generalization:}

\[
X \subseteq \text{out_front}(X)
\]
Reasoning about Entailment

Learned lexical relations from example proof trees

\[(t, h)\]: \((\text{pink team is offsides}, \text{purple 9 passes})\) \quad \((\text{bad pass...}, \text{loses the ball to})\)

<table>
<thead>
<tr>
<th>(\text{team}_{\text{arg1}})</th>
<th>(\text{substitute})</th>
<th>(\text{pink team}/\text{purple9})</th>
<th>(\subseteq \text{play-tran})</th>
<th>(\text{substitute})</th>
<th>(\text{bad pass/turnover})</th>
</tr>
</thead>
<tbody>
<tr>
<td>\text{relation:}</td>
<td>\text{pink team}</td>
<td>| \text{purple9}</td>
<td>| \text{bad pass _ picked off by}</td>
<td>| \text{loses the ball to}</td>
<td></td>
</tr>
</tbody>
</table>

\[(t, h)\]: \((\text{free kick for, steals the ball from})\) \quad \((\text{purple 6 kicks to, purple 6 kicks})\)

<table>
<thead>
<tr>
<th>(\text{game-play})</th>
<th>(\text{substitute})</th>
<th>(\text{free kick/steal})</th>
<th>(\subseteq \text{play-tran.})</th>
<th>(\text{substitute})</th>
<th>(\text{pass/kick})</th>
</tr>
</thead>
<tbody>
<tr>
<td>\text{analysis:}</td>
<td>\text{&quot;free kick for&quot;/&quot;steals the ball from&quot;}</td>
<td>| \text{&quot;kicks to&quot;/&quot;kicks’}</td>
<td>| \text{pass}</td>
<td>| \subseteq \text{kick}</td>
<td></td>
</tr>
</tbody>
</table>

\text{relation:} \quad \text{free kick} \| \text{steal}
Learning from Entailment: Summary

- **New Evaluation**: Evaluating semantic parsers on recognizing textual entailment, check if we are learning the missing information.

<table>
<thead>
<tr>
<th>Inference Model</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Majority Baseline</td>
<td>33.1%</td>
</tr>
<tr>
<td>RTE Classifier</td>
<td>52.4%</td>
</tr>
<tr>
<td>LF Matching</td>
<td>59.6%</td>
</tr>
<tr>
<td><strong>Logical Inference Model</strong></td>
<td><strong>73.4%</strong></td>
</tr>
</tbody>
</table>
Learning from Entailment: Summary

- **New Evaluation**: Evaluating semantic parsers on recognizing textual entailment, check if we are learning the missing information.

<table>
<thead>
<tr>
<th>Inference Model</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Majority Baseline</td>
<td>33.1%</td>
</tr>
<tr>
<td>RTE Classifier</td>
<td>52.4%</td>
</tr>
<tr>
<td>LF Matching</td>
<td>59.6%</td>
</tr>
<tr>
<td><strong>Logical Inference Model</strong></td>
<td><strong>73.4%</strong></td>
</tr>
</tbody>
</table>

- Entailments prove to be a good learning signal for learning improved representations (joint models also achieve SOTA on original semantic parsing task).
</Challenge 3>
Conclusions and Looking Ahead

**Challenge 1:** Getting data?  
Solution: Look to the technical docs, source code as a parallel corpus.

**Training**

Parallel Training Set
\[ D = \{ (x_i, z_i) \}_{i}^{\mid D \mid} \]

**Challenge 2:** Missing Data?  
Solution: Polyglot modeling over multiple datasets.

**Challenge 3:** Deficient LFs?  
Solution: Learning from entailment, RTE training.

**Testing**

- Input: \( X \)
- Semantic Parsing: \( X \rightarrow \text{decoding} \rightarrow \text{sem} \)
- Machine Learner

- Parallel Training Set: \( D = \{ (x_i, z_i) \}_{i}^{\mid D \mid} \)

- Technical topics: graph-based constrained decoding, (probabilistic) logics for joint semantic parsing and reasoning.

- Looking ahead: more work on end-to-end NLU, neural learning from entailment, structured decoding frameworks, code retrieval.
Conclusions and Looking Ahead

**Challenge 1: Getting Data?**
Solution: Look to the technical docs, source code as a parallel corpus.

**Challenge 2: Missing Data?**
Solution: Polyglot modeling over multiple datasets.

**Challenge 3: Deficient LFs?**
Solution: Learning from entailment, RTE training.

**Parallel Training Set**
\[ D = \{(x_i, z_i)\}_{i=1}^{D} \]

**Machine Learner**

**Testing**

**Input**

**Semantic Parsing**

**Output**

**Technical topics**: graph-based constrained decoding, (probabilistic) logics for joint semantic parsing and reasoning.
Conclusions and Looking Ahead

- **Challenge 1: Getting data?**
  Solution: Look to the technical docs, source code as a parallel corpus.

- **Challenge 2: Missing Data?**
  Solution: Polyglot modeling over multiple datasets.

- **Challenge 3: Deficient LFs?**
  Solution: Learning from entailment, RTE training.

**Parallel Training Set**

\[ D = \{(x_i, z_i)\}_{i} \]

**Technical topics:** graph-based constrained decoding, (probabilistic) logics for joint semantic parsing and reasoning.

**Looking ahead:** more work on end-to-end NLU, neural learning from entailment, structured decoding frameworks, code retrieval.
Source Code and NLU: Beyond Text-to-Code Translation

Returns the greater of two long values

<table>
<thead>
<tr>
<th>Signature (informal)</th>
<th>lang Math long max(long a, long b)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normalized</td>
<td>java lang Math::max(long:a, long:b) -&gt; long</td>
</tr>
</tbody>
</table>

Expansion to Logic

\[
\lambda x_1 \lambda x_2 \exists v \exists f \exists n \exists c \ eq(v, max(x_1, x_2)) \land fun(f, max) \land type(v, long) \\
\land lang(f, java) \\
\land var(x_1, a) \land param(x_1, f, 1) \land type(x_1, long) \\
\land var(x_2, b) \land param(x_2, f, 2) \land type(x_2, long) \\
\land namespace(n, lang) \land in\_namespace(f, n) \\
\land class(c, Math) \land in\_class(f, c)
\]

What do signature actually mean? Signatures can be given a formal semantics (Richardson, 2018).

Might prove to a good resource for investigating end-to-end NLU and symbolic reasoning, APIs contain loads of declarative knowledge.

see Neubig and Allamnis NAACL18 tutorial Modeling NL, Programs and their Intersection. and Allamanis et al. (2018).
Thank You
References


References III


