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

September 24, 2018

Collaborators: Jonas Kuhn (advisor, Stuttgart) and Jonathan Berant (work on "polyglot semantic parsing", Tel Aviv)

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.*
	$\rightarrow$
LF:	$\texttt{Type.Article} \sqcap \mathbf{R}[\lambda x.\mathrm{count}(\texttt{AuthorOf}.x)] \leq 2$

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.*

$$\rightarrow$$

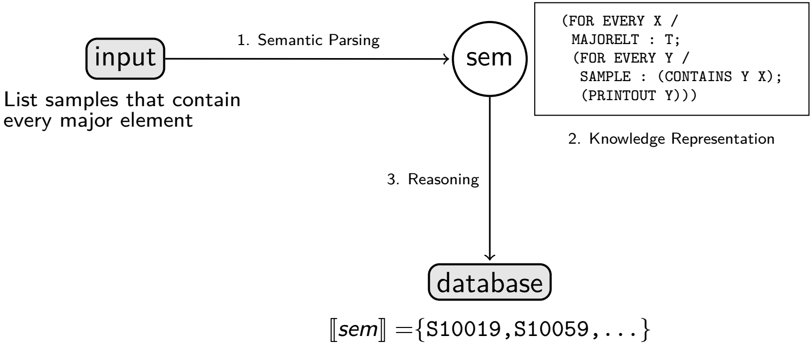
LF: $\texttt{Type.Article} \sqcap \textbf{R}[\lambda x.\text{count}(\texttt{AuthorOf}.x)] \leq 2$

> *"**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**..."*

R.F. Simmons. 1965, *Answering English Question by Computer: A Survey*. Communications of the ACM

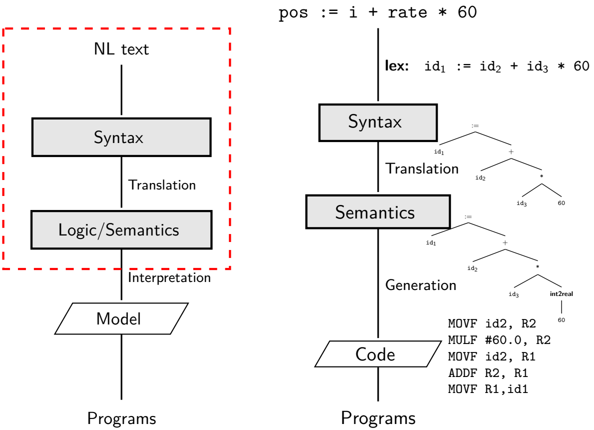
Classical Natural Language Understanding (NLU)

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



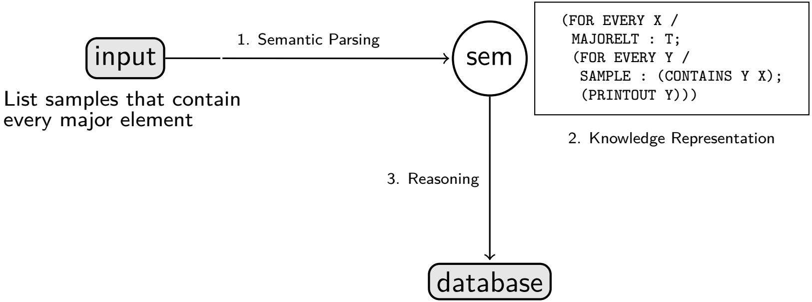
Lunar QA system of Woods (1973)

Why and How? Analogy with Compiler Design



- NLU model is a kind of compiler, involves a **transduction** from NL to a formal (usually logical) language.

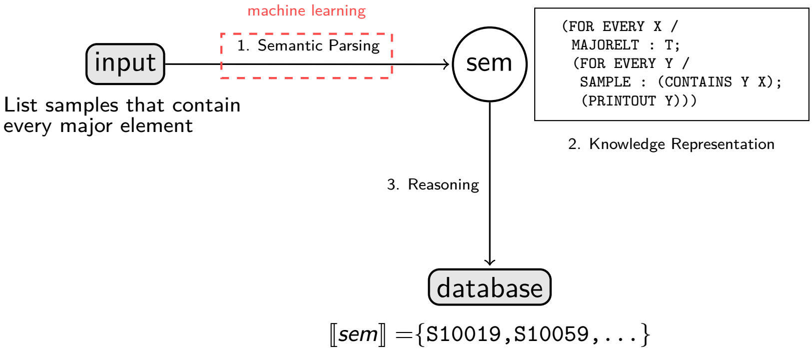
Data-driven Semantic Parsing and NLU



$[\![\textit{sem}]\!] = \{\texttt{S10019},\texttt{S10059},\ldots\}$

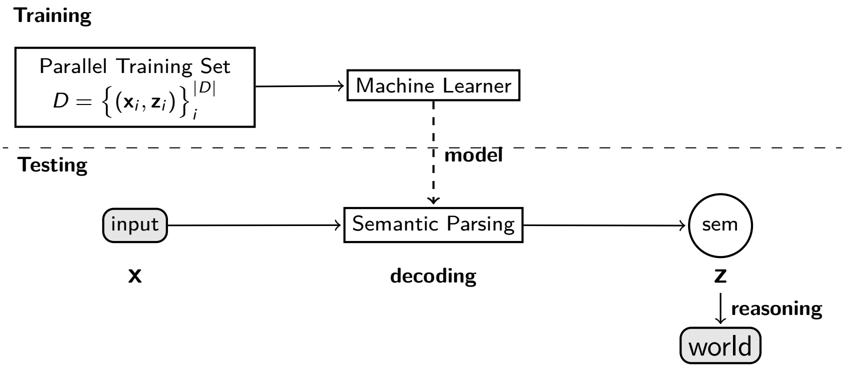
- **Data-driven NLU**: Asks an empirical question: Can we learn NLU models from examples? Building a NL compiler by hand is hard....

Data-driven Semantic Parsing and NLU



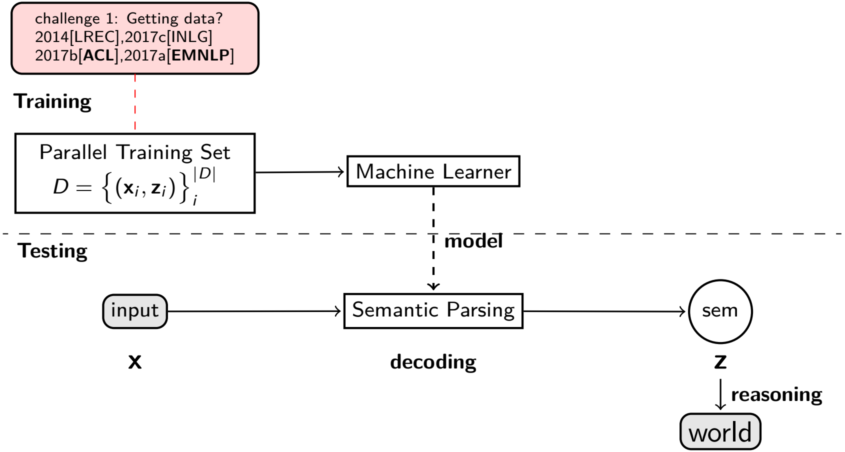
- **Semantic Parser Induction:** Learn semantic parser (weighted transduction) from parallel text/meaning data, constrained SMT task.

Data-driven Semantic Parsing in a Nutshell



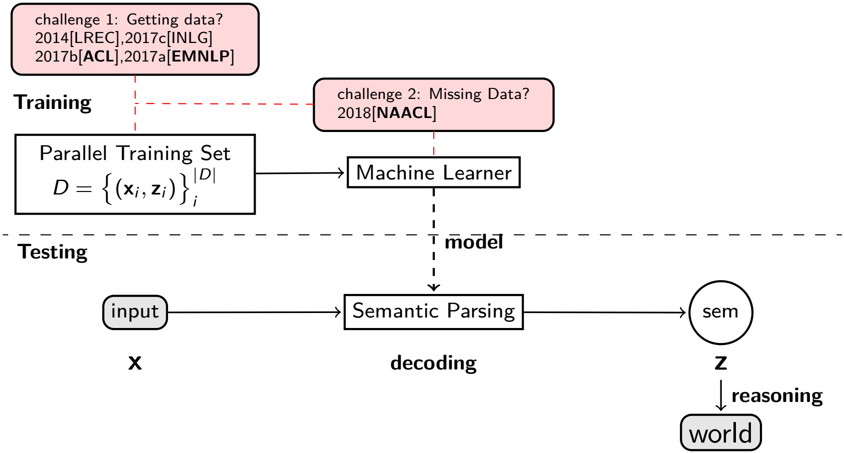
- **Desiderata**: robust and domain agnostic models that require minimal amounts of hand engineering and data supervision.

Data-driven Semantic Parsing in a Nutshell



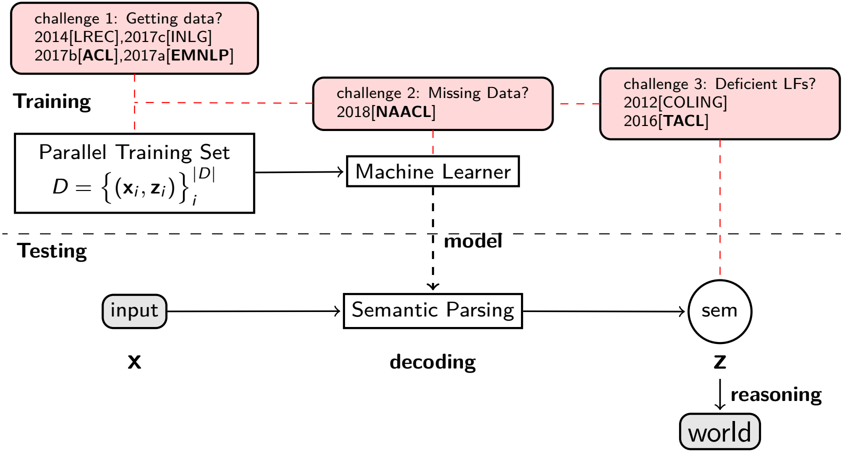
- **Desiderata**: robust and domain agnostic models that require minimal amounts of hand engineering and data supervision.

Data-driven Semantic Parsing in a Nutshell



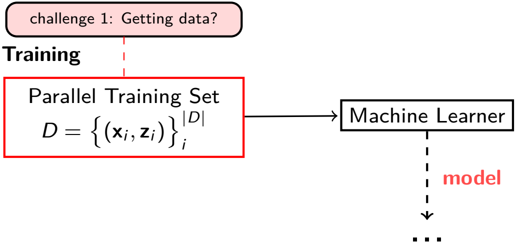
- **Desiderata**: robust and domain agnostic models that require minimal amounts of hand engineering and data supervision.

Data-driven Semantic Parsing in a Nutshell

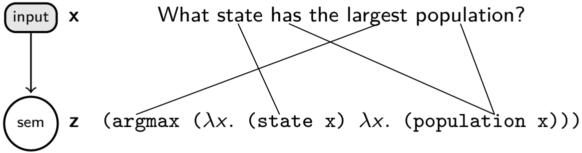


- **Desiderata**: robust and domain agnostic models that require minimal amounts of hand engineering and data supervision.

<Challenge 1>

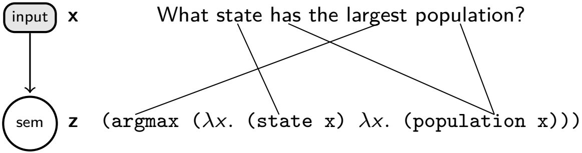

challenge 1: Getting data?
Training
Parallel Training Set
$D = \left\{(\mathbf{x}_i, \mathbf{z}_i)\right\}_i^{|D|}$
Machine Learner
model
...

Semantic Parsing and Parallel Data



- **Learning from LFs**: Pairs of text $\mathbf{x}$ and logical forms $\mathbf{z}$, $D = \{(\mathbf{x}, \mathbf{z})_i\}_i^n$, learn $\texttt{sem} : \mathbf{x} \rightarrow \mathbf{z}$
- **Modularity**: Study the translation independent of other semantic issues.

Semantic Parsing and Parallel Data



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

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.

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.

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)
```

⇓ 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)
```

⇓ 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...)

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

$$\texttt{Signature} ::= \underbrace{\texttt{lang}}_{\texttt{namespace}} \; \underbrace{\texttt{Math}}_{\texttt{class}} \; \underbrace{\texttt{long}}_{\texttt{return}} \; \underbrace{\texttt{max}}_{\texttt{name}} \; (\; \underbrace{\texttt{long a, long b)}}_{\texttt{named/typed arguments}}$$

Resource 1: Standard Library Documentation (Stdlib)

Dataset	#Pairs	#Symbols	#Words	Vocab.	Example Pairs (**x**, **z**)
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)` $\rightarrow$ `Float`
PHP_{en}	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)

Dataset	#Pairs	#Symbols	#Words	Vocab.	Example Pairs (**x**, **z**)
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)` $\rightarrow$ `Float`
PHP_{en}	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)

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

- 27 English Python projects from Github (Richardson and Kuhn, 2017a).

New Task: Text to Signature Translation

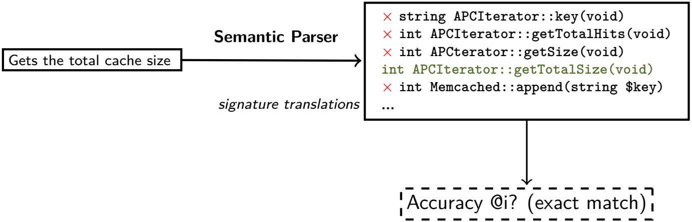
text	*Returns the greater of two long values*
signature	`lang.Math long max( long a, long b )`

- **Task**: Given text/signatures training pairs, learn a (quasi) *semantic parser*: text $\rightarrow$ `signature` (Richardson and Kuhn, 2017b)
  - **Assumption**: predicting within finite signature/translation space.

New Task: Text to Signature Translation

text	*Returns the greater of two long values*
signature	`lang.Math long max( long a, long b )`

- **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):



Text to Signature Translation: How Hard Is It?

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

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

Text to Signature Translation: How Hard Is It?

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

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

  - `lm`: convenient for making strong assumptions about our output language, facilitates **constrained decoding**.

Text to Signature Translation: How Hard Is It?

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

$$\textsc{SemPar}^{nc}(\mathbf{x},\mathbf{z}) = \underbrace{p_\theta(\mathbf{x} \mid \mathbf{z})}_{\text{trans model}} \times \underbrace{p_{\texttt{lm}}(\mathbf{z})}_{\text{valid expression (yes/no)?}}$$

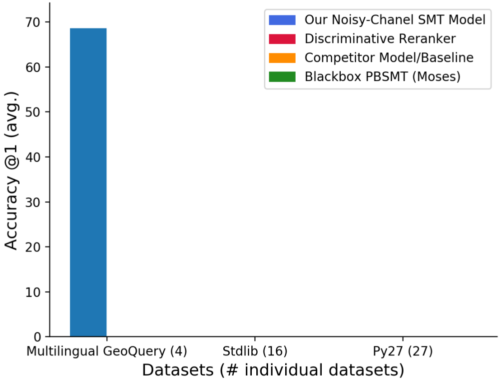
  - `lm`: convenient for making strong assumptions about our output language, facilitates **constrained decoding**.
  - **code case**: make assumptions about what constitutes a *valid function* in a given API.

Text to Signature Translation: How Hard Is It?

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

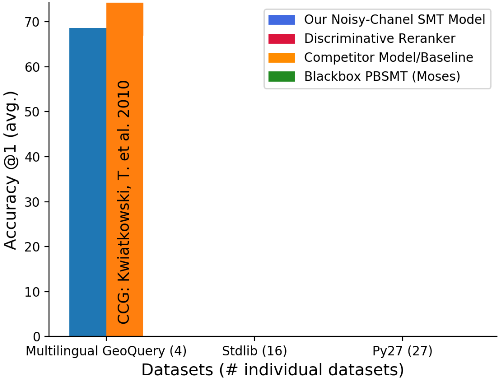
Text to Signature Translation: How Hard Is It?

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



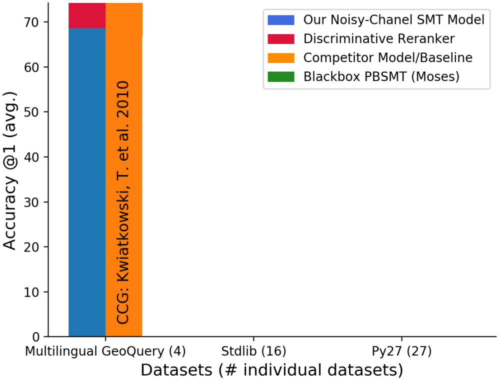
Text to Signature Translation: How Hard Is It?

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



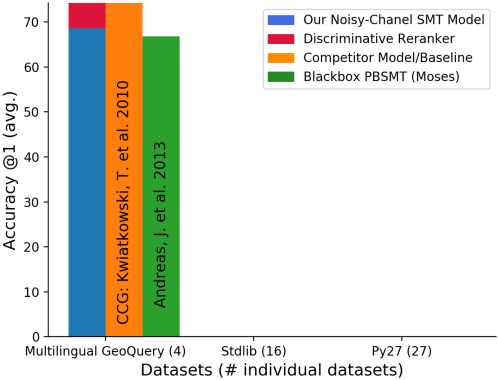
Text to Signature Translation: How Hard Is It?

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



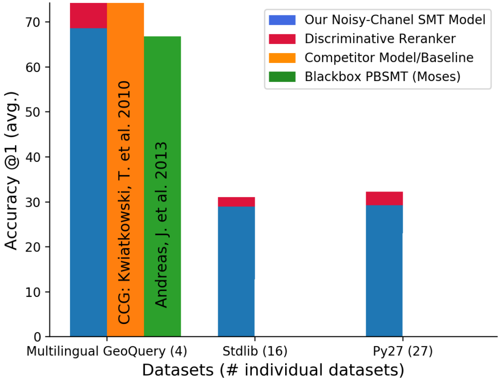
Text to Signature Translation: How Hard Is It?

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



Text to Signature Translation: How Hard Is It?

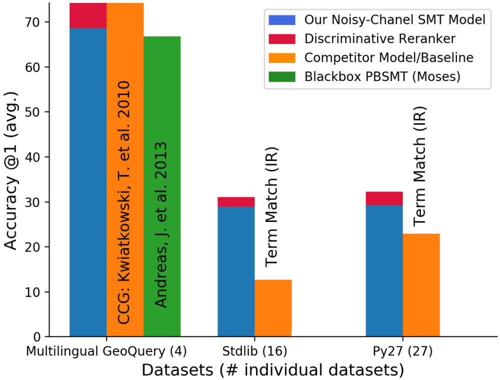
- **Our Approach**: Lexical translation model (standard estimation), discriminative reranker, hard constraints on $p(\mathbf{z})$.



- **Result**: achieving high accuracy is not easy, not a trivial problem.

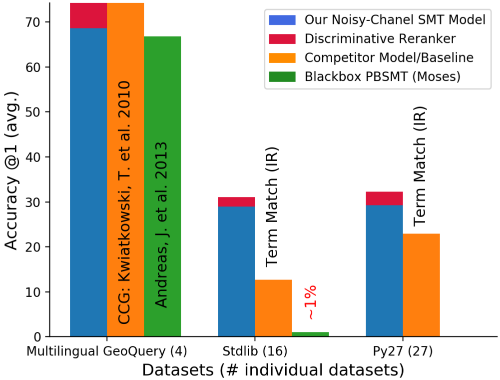
Text to Signature Translation: How Hard Is It?

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



Text to Signature Translation: How Hard Is It?

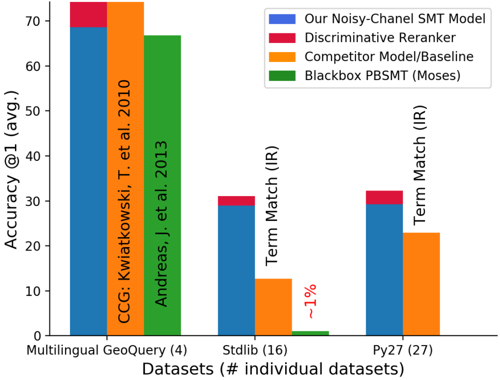
- **Our Approach**: Lexical translation model (standard estimation), discriminative reranker, hard constraints on $p(\mathbf{z})$.



Text to Signature Translation: How Hard Is It?

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

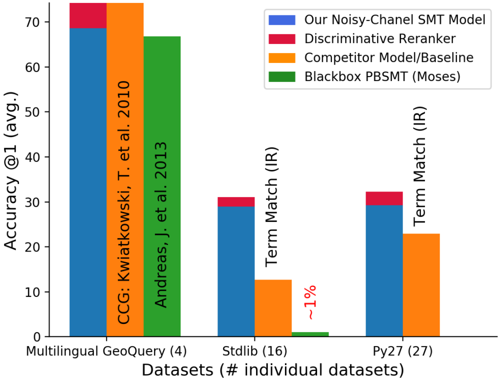
text	*Returns the index of the first occurrrence of char in the string*
Moses	`(start end occurrence lambda char string string string)`



Text to Signature Translation: How Hard Is It?

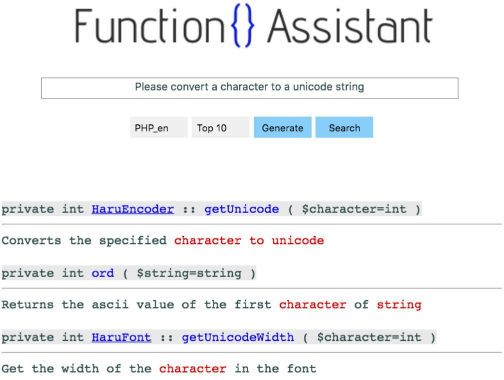
- **Our Approach**: Lexical translation model (standard estimation), discriminative reranker, hard constraints on $p(\mathbf{z})$.

text	*Returns the index of the first occurrrence of char in the string*
Moses	`(start end occurrence lambda char string string string)`



- **Observation**: semantic parsing is not an unconstrained MT problem.

What do these results mean? Code Retrieval Again



Dataset(Avg.)	**Accuracy @1** (average)	**Accuracy @10** (average)
Stdlib	31.1	71.0
Py27	32.3	73.5

so far: Semantic parsing as constrained translation, API as parallel corpus.

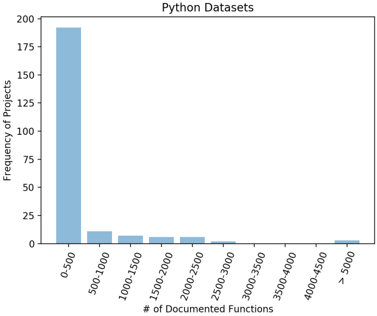
</Challenge 1>

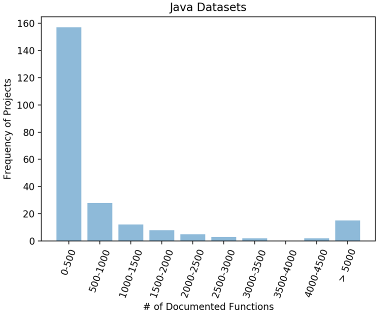
Challenge 2: Insufficient and Missing Data

(en,PHP)	$\longrightarrow$	$\theta_{\texttt{en}} \to$ `PHP`
(en,`Lisp`)	$\longrightarrow$	$\theta_{\texttt{en}} \to$ `Lisp`
(de, `PHP`)	$\longrightarrow$	$\theta_{\texttt{de}} \to$ `PHP`
(ja, `Python`)	$\longrightarrow$	$\theta_{\texttt{ja}} \to$ `Python`
(en, `Haskell`)	$\longrightarrow$	$\theta_{\texttt{en}} \to$ `Haskell`
...		

- 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`).

Code Domain: Projects often Lack Documentation





- 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

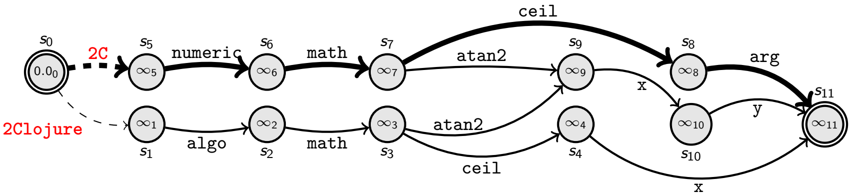
(en,`PHP`)
(en,`Lisp`)
(de, `PHP`)
(ja, `Python`)
(en, `Haskell`)
...

$\longrightarrow \theta_{\texttt{polyglot}}$

- **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?

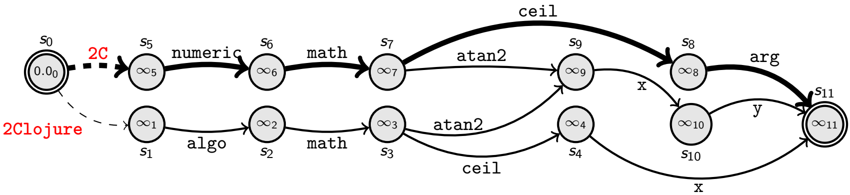
Graph Based Approach



- **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*.

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

  x : 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:

$$\underbrace{d[v]}_{\substack{\uparrow \\ \text{node score}}} = \min_{(u,v,z)\in E} \Big\{ \underbrace{d[u]}_{\substack{\uparrow \\ \text{incoming node score}}} + \underbrace{w(u,v,z)}_{\substack{\uparrow \\ \text{edge score}}} \Big\}$$

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:

$$\underbrace{d[v]}_{\substack{\uparrow \\ \text{node score}}} = \min_{(u,v,z)\in E} \left\{ \underbrace{d[u]}_{\substack{\uparrow \\ \text{incoming node score}}} + \underbrace{\textsc{trans}(\mathbf{x}, z)}_{\substack{\uparrow \\ \text{translation}}} \right\}$$

  - 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:

$$\underbrace{d[v]}_{\substack{\uparrow \\ \text{node score}}} = \min_{(u,v,z) \in E} \left\{ \underbrace{d[u]}_{\substack{\uparrow \\ \text{incoming node score}}} + \underbrace{\textcolor{red}{\text{TRANS}}(\mathbf{x}, z)}_{\substack{\uparrow \\ \textcolor{red}{\text{translation}}}} \right\}$$

  - 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):

$$
\begin{aligned}
p(\mathbf{z} \mid \mathbf{x}) &= \textsc{ConditionalRNNLM}(\mathbf{z}) \\
&= \prod_{i}^{|\mathbf{z}|} p_{\Theta}(z_i \mid z_{<i}, \mathbf{x})
\end{aligned}
$$

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):

$$
\begin{aligned}
p(\mathbf{z} \mid \mathbf{x}) &= \textsc{ConditionalRNNLM}(\mathbf{z}) \\
&= \prod_{i}^{|\mathbf{z}|} p_{\Theta}(z_i \mid z_{<i}, \mathbf{x})
\end{aligned}
$$

- **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):

$$p(\mathbf{z} \mid \mathbf{x}) = \textsc{ConditionalRNNLM}(\mathbf{z})$$
$$= \prod_{i}^{|\mathbf{z}|} p_{\Theta}(z_i \mid z_{<i}, \mathbf{x})$$

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

```
d[b] ← 0.0
for node v ∈ 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):

$$p(\mathbf{z} \mid \mathbf{x}) = \textsc{ConditionalRNNLM}(\mathbf{z})$$
$$= \prod_{i}^{|\mathbf{z}|} p_{\Theta}(z_i \mid z_{<i}, \mathbf{x})$$

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

0: $d[b] \leftarrow 0.0$
1: **for** node $v \in V$ in topologically sorted order
2: **do** $d(v) = \min_{(u,v,z_j) \in E} \Big\{ d(u) + - \log p_{\Theta}(z_j \mid z_{<j}, \mathbf{x}) \Big\}$

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):

$$p(\mathbf{z} \mid \mathbf{x}) = \textsc{ConditionalRNNLM}(\mathbf{z})$$
$$= \prod_{i}^{|\mathbf{z}|} p_{\Theta}(z_i \mid z_{<i}, \mathbf{x})$$

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

0: $d[b] \leftarrow 0.0$
1: **for** node $v \in V$ in topologically sorted order
2: **do** $d(v) = \min_{(u,v,z_j) \in E} \left\{ d(u) + - \log p_{\Theta}(z_j \mid z_{<j}, \mathbf{x}) \right\}$
3: $s[v] \leftarrow$ `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):

$$p(\mathbf{z} \mid \mathbf{x}) = \textsc{ConditionalRNNLM}(\mathbf{z})$$
$$= \prod_{i}^{|\mathbf{z}|} p_{\Theta}(z_i \mid z_{<i}, \mathbf{x})$$

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

0: $d[b] \leftarrow 0.0$
1: **for** node $v \in V$ in topologically sorted order
2: **do** $d(v) = \min_{(u,v,z_j) \in E} \left\{ d(u) + -\log p_{\Theta}(z_j \mid z_{<j}, \mathbf{x}) \right\}$
3: $s[v] \leftarrow$ `RNN` state for min edge and z_j
4: **return** $\min_{v \in V} \left\{ d(v) \right\}$

Training on Multiple Datasets: Does this help?

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

		Method	Acc@1 (averaged)
Multilingual Geoquery	mono.	UBL Kwiatkowski et al. (2010)	74.2
		TreeTrans Jones et al. (2012)	76.8
		Lexical SMT SSSP	68.6
		Best Seq2Seq SSSP	78.0
	poly.	Lexical SMT SSSP	67.3
		Best Seq2Seq SSSP	**79.6**

Training on Multiple Datasets: Does this help?

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

		Method	Acc@1 (averaged)
Multilingual Geoquery	mono.	UBL Kwiatkowski et al. (2010)	74.2
		TreeTrans Jones et al. (2012)	76.8
		Lexical SMT SSSP	68.6
		Best Seq2Seq SSSP	78.0
	poly.	Lexical SMT SSSP	67.3
		Best Seq2Seq SSSP	**79.6**

		Method	Acc@1	Acc@10	MRR
stdlib	**mono.**	Best Monolingual Model	29.9	69.2	43.1
	poly.	Lexical SMT SSSP	**33.2**	70.7	45.9
		Best Seq2Seq SSSP	13.9	36.5	21.5
py27	**mono.**	Best Monolingual Model	32.4	73.5	46.5
	poly.	Lexical SMT SSSP	**41.3**	77.7	54.1
		Best Seq2Seq SSSP	9.0	26.9	15.1

Training on Multiple Datasets: Does this help?

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

		Method	Acc@1 (averaged)
Multilingual Geoquery	mono.	UBL Kwiatkowski et al. (2010)	74.2
		TreeTrans Jones et al. (2012)	76.8
		Lexical SMT SSSP	68.6
		Best Seq2Seq SSSP	78.0
	poly.	Lexical SMT SSSP	67.3
		Best Seq2Seq SSSP	**79.6**

		Method	Acc@1	Acc@10	MRR
stdlib	mono.	Best Monolingual Model	29.9	69.2	43.1
	poly.	Lexical SMT SSSP	**33.2**	70.7	45.9
		Best Seq2Seq SSSP	13.9	36.5	21.5
py27	mono.	Best Monolingual Model	32.4	73.5	46.5
	poly.	Lexical SMT SSSP	**41.3**	77.7	54.1
		Best Seq2Seq SSSP	9.0	26.9	15.1

- **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.

1.	Source API (stdlib): (***es*, PHP)**	**Input**: 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)**	**Input**: 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)**	**Input:** Compute the Moore-Penrose pseudo-inverse of a matrix.
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 )`

New Tasks: Any/Mixed Language Decoding

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

1.	Source API (stdlib): (*es*, **PHP**)	**Input**: 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**)	**Input**: 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**)	**Input:** Compute the Moore-Penrose pseudo-inverse of a matrix.
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.

1.	Source API (stdlib): (***es*, PHP**)	**Input**: 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**)	**Input**: 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**)	**Input:** Compute the Moore-Penrose pseudo-inverse of a matrix.
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

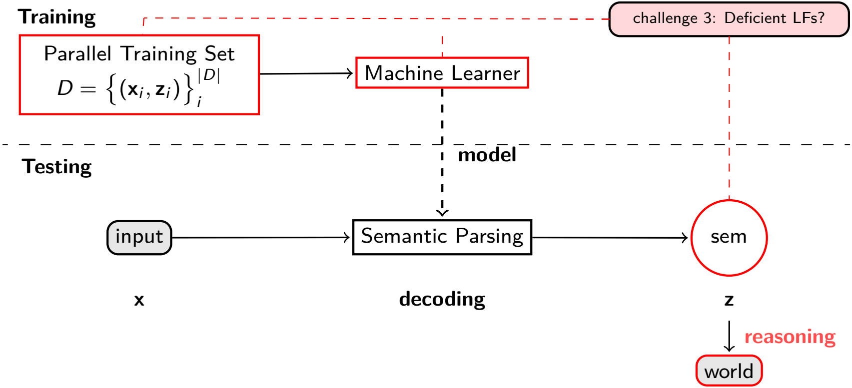
- **Mixed Language Decoding**: translating from input with NPs from multiple languages, introduced a new mixed GeoQuery test set.

Mixed Lang.	**Input**: Wie hoch liegt der höchstgelegene punkt in Αλαμπάμα?
LF: `answer(elevation_1(highest(place(loc_2(stateid('alabama'))))))`	

	Method	Accuracy (averaged)
Mixed	Best Monolingual Seq2Seq	4.2
	Polyglot Seq2Seq	**75.2**

</Challenge 2>

<Challenge 3>


Training
challenge 3: Deficient LFs?
Parallel Training Set
$D = \left\{ (\mathbf{x}_i, \mathbf{z}_i) \right\}_i^{|D|}$
Machine Learner
model
Testing
input
Semantic Parsing
sem
x
decoding
z
reasoning
world

Semantic Parsing and Entailment

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

input $\xrightarrow{\text{Semantic Parsing}}$ sem

```
(FOR EVERY X /
 MAJORELT : T;
 (FOR EVERY Y /
  SAMPLE : (CONTAINS Y X);
  (PRINTOUT Y)))
```

Knowledge Representation

All samples that contain
a major element

$\rightarrow$

Some sample that contains
a major element

sem $\xrightarrow{\text{Reasoning}}$ database

$[\![sem]\!] = \{\texttt{S10019},\texttt{S10059},\ldots\} \supseteq \{S10019\}$

Semantic Parsing and Entailment

- **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.*

	sentence	**analysis**
t	*All samples that contain a major element*	$\texttt{LF}_{\mathbf{t}}$
h	*Some sample that contains a major element*	$\texttt{LF}_{\mathbf{h}}$
	inference $\texttt{t} \rightarrow \texttt{h}$	`Entailment` (RTE[1])
	inference $\texttt{h} \rightarrow \texttt{t}$	`Unknown` (RTE)

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

	sentence	analysis
t	*Pink 3 passes to Pink 7*	`pass(pink3,pink7)`
h	*Pink 3 quickly kicks to Pink 7*	`pass(pink3,pink7)`
inference (human) $\texttt{t} \rightarrow \texttt{h}$		`Unknown` (RTE)
inference (`LF` match) $\texttt{t} \rightarrow \texttt{h}$		`Entail` (RTE)

Semantic Parsing and Entailment

- **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).

	sentence	analysis
t	*The pink goalie passes to pink 7*	`pass(pink1,pink7)`
h	*Pink 1 kicks the ball*	`kick(pink1)`
inference (human) `t` → `h`		`Entail` (RTE)
inference (LF match) `t` → `h`		`Contradict` (RTE)

Semantic Parsing and Entailment

- **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).

Inference Model	Accuracy
Majority Baseline	33.1%
RTE Classifier	52.4%
LF Matching	59.6%

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

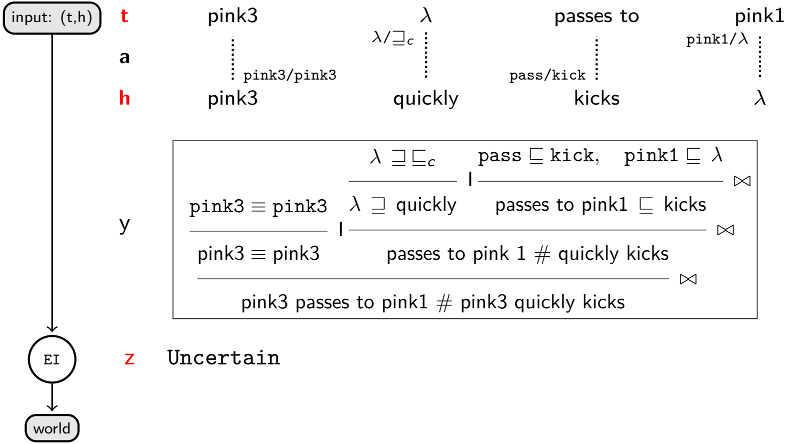
- **Underlying Challenge:** Semantic representations are underspecified, fail to capture entailments, background knowledge missing.
- **Goal:** Capture the missing knowledge and inferential properties of text, **incorporate entailment information into learning.**

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

Paradigm and Supervision	Dataset $D =$	Learning Goal
Learning from LFs	$\{(input_i, LF_i)\}_i^N$	$input \xrightarrow{Trans.} LF$
Learning from Entailment	$\{(input_t, input'_h)_i, \texttt{EI}_i)\}_i^N$	$(input_t, input'_h) \xrightarrow{Proof} \texttt{EI}$

Learning from Entailment: Illustration

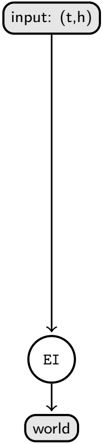
- Entailments are used to reason about target symbols and find holes in the analyses.



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.



t	pink3	λ	passes to	pink1
a	`pink3/pink3`	$\lambda/\sqsupseteq_c$	`pass/kick`	`pink1/`λ
h	pink3	quickly	kicks	λ

y

$$\dfrac{\dfrac{\mathtt{pink3} \equiv \mathtt{pink3}}{\text{pink3} \equiv \text{pink3}}\,\mathrm{I} \qquad \dfrac{\dfrac{\lambda \sqsupseteq \sqsubseteq_c}{\lambda \sqsupseteq \text{quickly}}\,\mathrm{I} \quad \dfrac{\mathtt{pass} \sqsubseteq \mathtt{kick},\quad \mathtt{pink1} \sqsubseteq \lambda}{\text{passes to pink1} \sqsubseteq \text{kicks}}\bowtie}{\text{passes to pink 1} \# \text{quickly kicks}}\bowtie}{\text{pink3 passes to pink1} \# \text{pink3 quickly kicks}}\bowtie$$

z `Uncertain`

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.

input: (t,h) → EI → world

t	pink3	λ	passes to	pink1
a	`pink3/pink3`	$\lambda/\sqsupseteq_c$	`pass/kick`	`pink1/`λ
h	pink3	quickly	kicks	λ

y

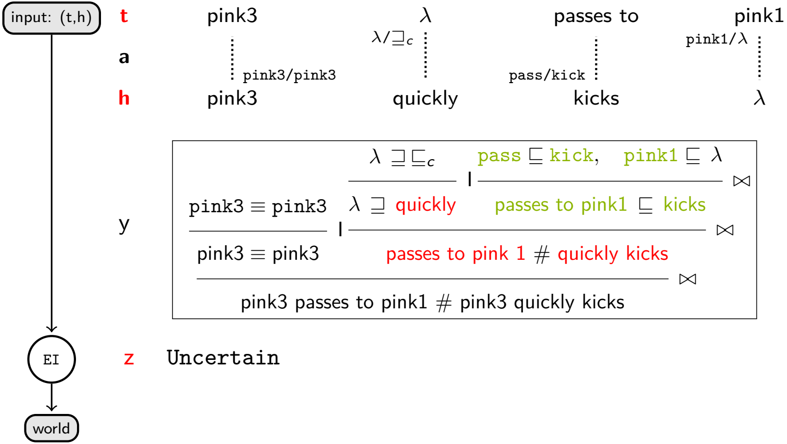
$$\dfrac{\dfrac{\texttt{pink3} \equiv \texttt{pink3}}{\text{pink3} \equiv \text{pink3}} \mid \dfrac{\dfrac{\lambda \sqsupseteq \sqsubseteq_c}{\lambda \sqsupseteq \text{quickly}} \mid \dfrac{\texttt{pass} \sqsubseteq \texttt{kick}, \quad \texttt{pink1} \sqsubseteq \lambda}{\text{passes to pink1} \sqsubseteq \text{kicks}} \bowtie}{\text{passes to pink 1} \# \text{quickly kicks}} \bowtie}{\text{pink3 passes to pink1} \# \text{pink3 quickly kicks}} \bowtie$$

z `Uncertain`

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

Learning from Entailment: Illustration

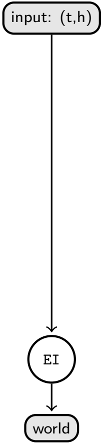
- Entailments are used to reason about target symbols and find holes in the analyses.



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

Learning from Entailment: Illustration

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



t	pink3	λ	passes to	pink1
a	pink3/pink3	$\lambda/\sqsupseteq_c$	pass/kick	pink1/λ
h	pink3	quickly	kicks	λ

y

$$\dfrac{\dfrac{\texttt{pink3} \equiv \texttt{pink3}}{\text{pink3} \equiv \text{pink3}} \;\Big|\; \dfrac{\dfrac{\lambda \sqsupseteq \sqsubseteq_c}{\lambda \sqsupseteq \text{quickly}} \;\Big|\; \dfrac{\texttt{pass} \sqsubseteq \texttt{kick},\quad \texttt{pink1} \sqsubseteq \lambda}{\text{passes to pink1} \sqsubseteq \text{kicks}} \bowtie}{\text{passes to pink 1} \# \text{quickly kicks}} \bowtie}{\text{pink3 passes to pink1} \# \text{pink3 quickly kicks}} \bowtie$$

z `Uncertain`

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))

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

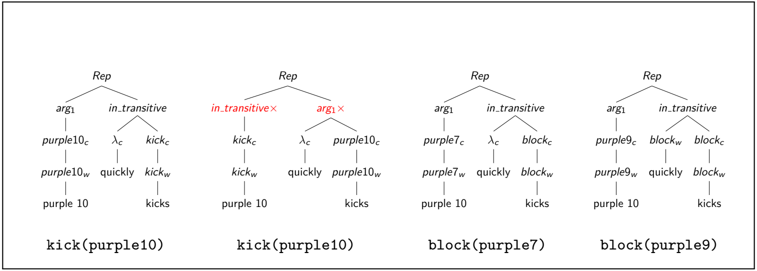
$\downarrow$ (rule extraction)

Grammar Approach: Sentences to Logical Form

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

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

$\downarrow$ (rule extraction)

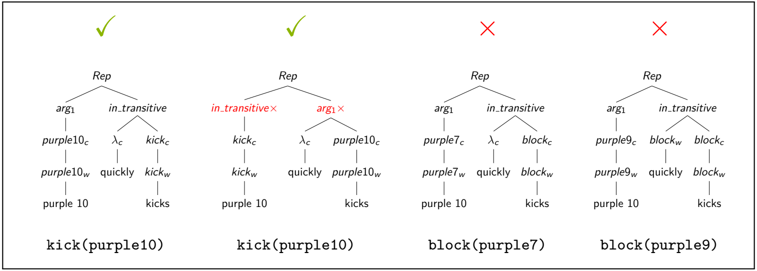


Grammar Approach: Sentences to Logical Form

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

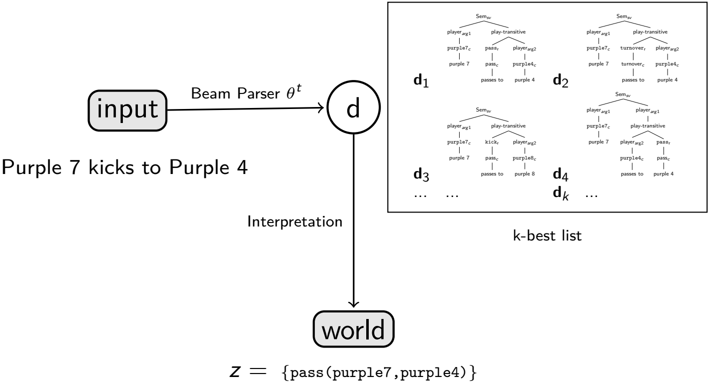
$$(x : \text{purple 10 quickly kicks}, z : \{\texttt{kick(purple10)}, \text{block(purple7)}, \ldots\})$$

$\downarrow$ (rule extraction)



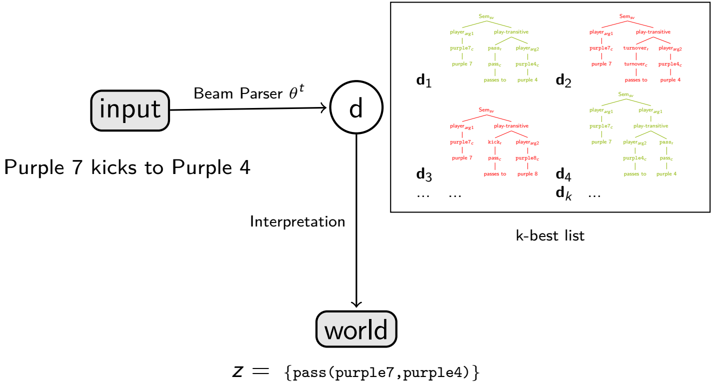
Semantic Parsing as Grammatical Inference

- 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.



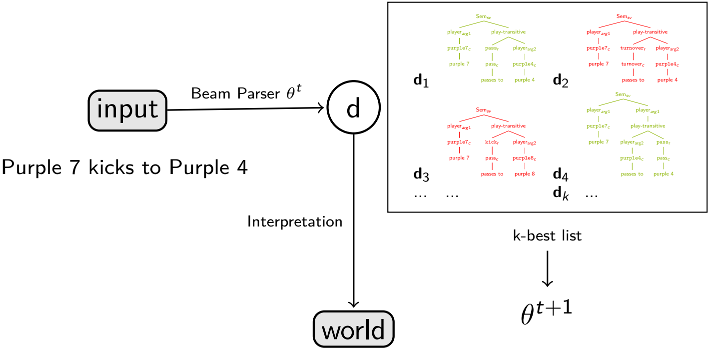
Semantic Parsing as Grammatical Inference

- 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.



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.



$z =$ {pass(purple7,purple4)}

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).

$$\mathbf{x} = (\mathtt{t}, \mathtt{h}),\ \mathbf{z} \in \{\mathtt{Entail}, \mathtt{Contradict}, \mathtt{Unknown}\}$$

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).

$$\mathbf{x} = (\mathtt{t}, \mathtt{h}),\ \mathbf{z} \in \{\mathtt{Entail}, \mathtt{Contradict}, \mathtt{Unknown}\}$$

$$p(\mathbf{z} \mid \mathbf{x}) = \sum_{\underbrace{y \in \mathcal{Y}_{\mathbf{x}}}_{\text{proofs}}} \underbrace{p(\mathbf{z} \mid y)}_{\substack{\uparrow \\ \text{valid inference?}}} \times \underbrace{p_\theta(y \mid \mathbf{x})}_{\substack{\uparrow \\ \text{proof score}}}$$

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).

$$\mathbf{x} = (\mathtt{t}, \mathtt{h}),\ \mathbf{z} \in \{\mathtt{Entail}, \mathtt{Contradict}, \mathtt{Unknown}\}$$

$$p(\mathbf{z} \mid \mathbf{x}) = \underbrace{\sum_{y \in \mathcal{Y}_{\mathbf{x}}}}_{\text{proofs}} \underbrace{p(\mathbf{z} \mid y)}_{\substack{\uparrow \\ \text{valid inference?}}} \times \underbrace{p_\theta(y \mid \mathbf{x})}_{\substack{\uparrow \\ \text{proof score}}}$$

  - $p(\mathbf{z} \mid y)$: 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).

$$\mathbf{x} = (\mathtt{t}, \mathtt{h}),\ \mathbf{z} \in \{\mathtt{Entail}, \mathtt{Contradict}, \mathtt{Unknown}\}$$

$$p(\mathbf{z} \mid \mathbf{x}) = \sum_{\underbrace{y \in \mathcal{Y}_{\mathbf{x}}}_{\text{proofs}}} \underbrace{p(\mathbf{z} \mid y)}_{\substack{\uparrow \\ \text{valid inference?}}} \times \underbrace{p_\theta(y \mid \mathbf{x})}_{\substack{\uparrow \\ \text{proof score}}}$$

  - $p(\mathbf{z} \mid y)$: 1 if proof derives correct entailment, 0 otherwise
  - $p_\theta(y \mid \mathbf{x})$: 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).

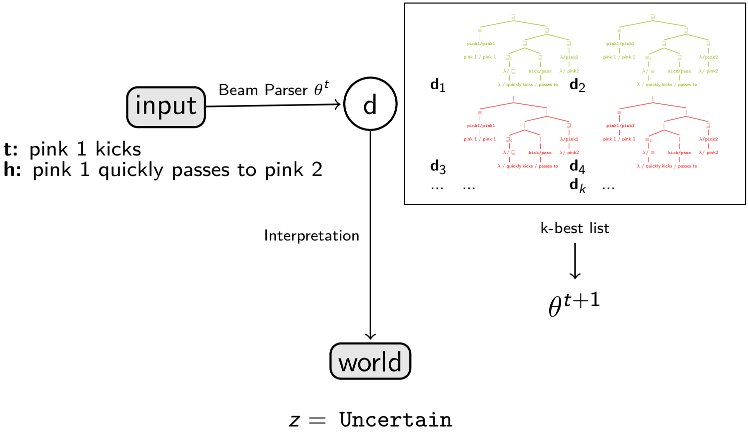
$$\mathbf{x} = (\mathtt{t}, \mathtt{h}),\ \mathbf{z} \in \{\mathtt{Entail}, \mathtt{Contradict}, \mathtt{Unknown}\}$$

$$p(\mathbf{z} \mid \mathbf{x}) = \underbrace{\sum_{y \in \mathcal{Y}_{\mathbf{x}}}}_{\text{proofs}} \underbrace{p(\mathbf{z} \mid y)}_{\substack{\uparrow \\ \text{valid inference?}}} \times \underbrace{p_\theta(y \mid \mathbf{x})}_{\substack{\uparrow \\ \text{proof score}}}$$

  - $p(\mathbf{z} \mid y)$: 1 if proof derives correct entailment, 0 otherwise
  - $p_\theta(y \mid \mathbf{x})$: 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.

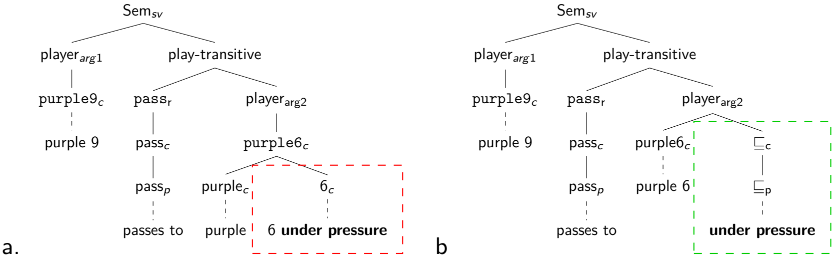
Learning Entailment Rules

- 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 ($\mathcal{Y}_{\mathbf{x}} \approx \textsc{kbest}(\mathbf{x})$).



Reasoning about Entailment

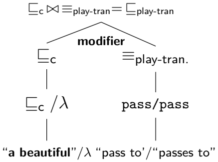
- Improving the internal representations (before, a, after, b).

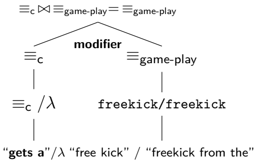


Reasoning about Entailment

- Learned modifiers from example proofs trees.

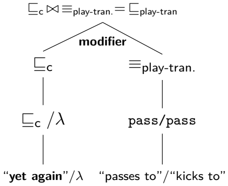
(t, h): (a **beautiful** pass to,passes to) (**gets a** free kick,freekick from the)

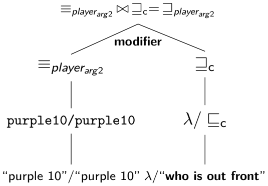




analysis:
generalization: $\texttt{beautiful}(X) \sqsubseteq X$ $\texttt{get}(X) \equiv X$

(t, h): (**yet again** passes to,kicks to) (purple 10,purple 10 **who is out front**)





analysis:
generalization: $\texttt{yet-again}(X) \sqsubseteq X$ $X \sqsupseteq \texttt{out\_front}(X)$

Reasoning about Entailment

- Learned lexical relations from example proof trees

(t, h): (**pink team** is offsides,**purple 9** passes)

$|_{team_{arg1}}$

substitute

`pink team/purple9`

“pink team”/“purple 9”

analysis:
relation: `pink team | purple9`

(t, h): (**bad pass..**, **loses the ball to**)

$\sqsubseteq$play-tran

substitute

`bad pass/turnover`

“bad pass .. picked off by”/“loses the ball to”

analysis:
relation: `bad pass` $\sqsubseteq$ `turnover`

(t, h): (**free kick for**, **steals the ball from**)

|game-play

substitute

`free kick/steal`

“free kick for”/“steals the ball from”

analysis:
relation: `free kick| steal`

(t, h): (purple 6 **kicks to**,purple 6 **kicks**)

$\sqsubseteq$play-tran.

substitute

`pass/kick`

“kicks to”/“kicks’

analysis:
relation: `pass` $\sqsubseteq$ `kick`

Learning from Entailment: Summary

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

Inference Model	Accuracy
Majority Baseline	33.1%
RTE Classifier	52.4%
LF Matching	59.6%
Logical Inference Model	73.4%

Learning from Entailment: Summary

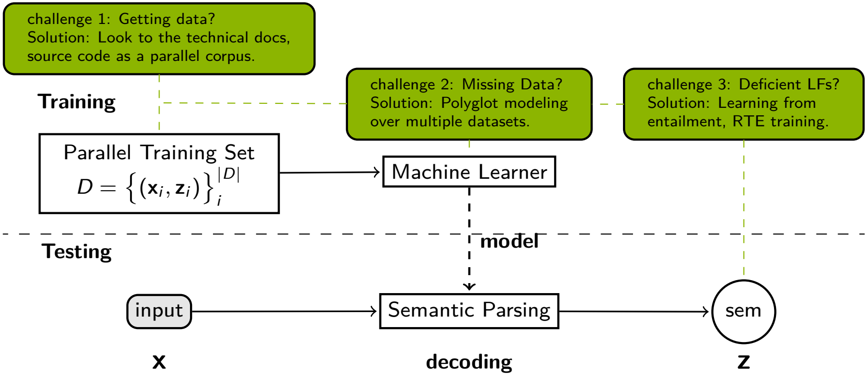
- **New Evaluation**: Evaluating semantic parsers on recognizing textual entailment, check if we are learning the missing information.

Inference Model	**Accuracy**
Majority Baseline	33.1%
RTE Classifier	52.4%
LF Matching	59.6%
Logical Inference Model	73.4%

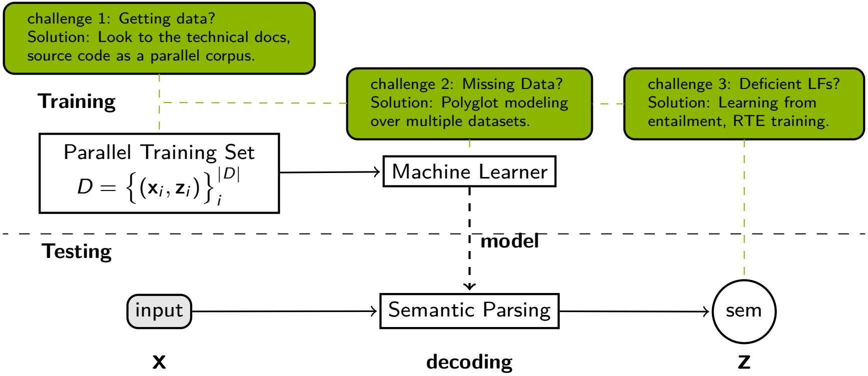
- Entailments prove to be a good learning signal for learning improved representations (joint models also achieve SOTA on original semantic parsing task).

</Challenge 3>

Conclusions and Looking Ahead

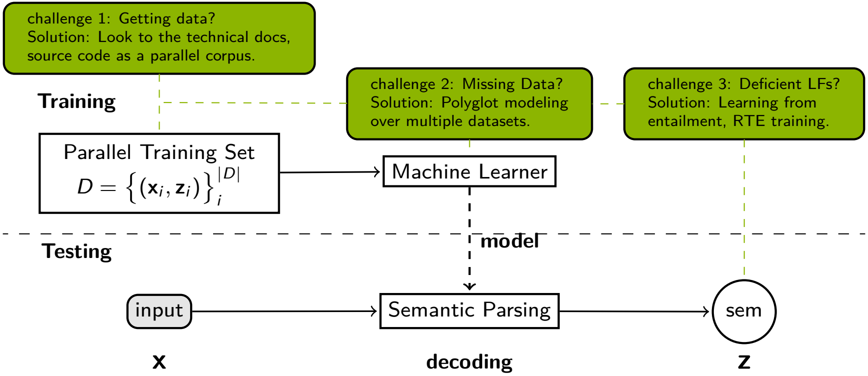


Conclusions and Looking Ahead



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

Conclusions and Looking Ahead



- **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

$\downarrow$

Signature (informal)	`lang Math long max(long a,long b)`
Normalized	`java lang Math::max(long:a,long:b) -> long`
Expansion to Logic	$[\![\texttt{java lang Math::max(long:a,long:b) -> long}]\!]$ $\Updownarrow$ $\lambda x_1 \lambda x_2 \exists v \exists f \exists n \exists c\ \texttt{eq}(v, \texttt{max}(x_1, x_2)) \wedge \texttt{fun}(f, \texttt{max}) \wedge \texttt{type}(v, \texttt{long}) \wedge \texttt{lang}(f, \texttt{java}) \wedge \texttt{var}(x_1, \texttt{a}) \wedge \texttt{param}(x_1, f, 1) \wedge \texttt{type}(x_1, \texttt{long}) \wedge \texttt{var}(x_2, \texttt{b}) \wedge \texttt{param}(x_2, f, 2) \wedge \texttt{type}(x_2, \texttt{long}) \wedge \texttt{namespace}(\texttt{n}, \texttt{lang}) \wedge \texttt{in\_namespace}(f, n) \wedge \texttt{class}(\texttt{c}, \texttt{Math}) \wedge \texttt{in\_class}(f, c)$

- **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 I

Allamanis, M., Barr, E. T., Devanbu, P., and Sutton, C. (2018). A Survey of Machine Learning for Big Code and Naturalness. *ACM Computing Surveys (CSUR)*, 51(4):81.

Allamanis, M., Tarlow, D., Gordon, A., and Wei, Y. (2015). Bimodal modelling of source code and natural language. In *International Conference on Machine Learning*, pages 2123–2132.

Angeli, G., Manning, C. D., and Jurafsky, D. (2012). Parsing time: Learning to interpret time expressions. In *Proceedings of NAACL-2012*, pages 446–455.

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv preprint arXiv:1409.0473*.

Berant, J., Chou, A., Frostig, R., and Liang, P. (2013). Semantic Parsing on Freebase from Question-Answer Pairs. In *in Proceedings of EMNLP-2013*, pages 1533–1544.

Börschinger, B., Jones, B. K., and Johnson, M. (2011). Reducing grounded learning tasks to grammatical inference. In *Proceedings of EMNLP-2011*, pages 1416–1425.

Chen, D. L. and Mooney, R. J. (2008). Learning to sportscast: A test of grounded language acquisition. In *Proceedings of ICML-2008*, pages 128–135.

Cormen, T., Leiserson, C., Rivest, R., and Stein, C. (2009). *Introduction to Algorithms*. MIT Press.

Dagan, I., Glickman, O., and Magnini, B. (2005). The pascal recognizing textual entailment challenge. In *Proceedings of the PASCAL Challenges Workshop on Recognizing Textual Entailment*.

References II

Deng, H. and Chrupała, G. (2014). Semantic approaches to software component retrieval with English queries. In *Proceedings of LREC-14*, pages 441–450.

Dong, L. and Lapata, M. (2016). Language to Logical Form with Neural Attention. *arXiv preprint arXiv:1601.01280*.

Gu, X., Zhang, H., Zhang, D., and Kim, S. (2016). Deep api learning. In *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, pages 631–642. ACM.

Guu, K., Pasupat, P., Liu, E., and Liang, P. (2017). From Language to Programs: Bridging Reinforcement Learning and Maximum Marginal Likelihood. In *Proceedings of ACL*.

Herzig, J. and Berant, J. (2017). Neural semantic parsing over multiple knowledge-bases. In *Proceedings of ACL*.

Iyer, S., Konstas, I., Cheung, A., and Zettlemoyer, L. (2016). Summarizing Source Code using a Neural Attention Model. In *Proceedings of ACL*, volume 1.

Johnson, M., Schuster, M., Le, Q. V., Krikun, M., Wu, Y., Chen, Z., Thorat, N., Viégas, F., Wattenberg, M., Corrado, G., et al. (2016). Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation. *arXiv preprint arXiv:1611.04558*.

Jones, B. K., Johnson, M., and Goldwater, S. (2012). Semantic parsing with Bayesian Tree Transducers. In *Proceedings of ACL-2012*, pages 488–496.

References III

Krishnamurthy, J., Dasigi, P., and Gardner, M. (2017). Neural Semantic Parsing with Type Constraints for Semi-Structured Tables. In *Proceedings of EMNLP*.

Kwiatkowski, T., Zettlemoyer, L., Goldwater, S., and Steedman, M. (2010). Inducing probabilistic CCG grammars from logical form with higher-order unification. In *Proceedings of EMNLP-2010*, pages 1223–1233.

Liang, P., Jordan, M. I., and Klein, D. (2013). Learning dependency-based compositional semantics. *Computational Linguistics*, 39(2):389–446.

MacCartney, B. and Manning, C. D. (2009). An extended model of natural logic. In *Proceedings of the eighth International Conference on Computational Semantics*, pages 140–156.

Montague, R. (1970). Universal grammar. *Theoria*, 36(3):373–398.

Richardson, K. (2018). A Language for Function Signature Representations. *arXiv preprint arXiv:1804.00987*.

Richardson, K., Berant, J., and Kuhn, J. (2018). Polyglot Semantic Parsing in APIs. In *Proceedings of NAACL*.

Richardson, K. and Kuhn, J. (2017a). Function Assistant: A Tool for NL Querying of APIs. In *Proceedings of EMNLP-17*.

Richardson, K. and Kuhn, J. (2017b). Learning Semantic Correspondences in Technical Documentation. In *Proceedings of ACL-17*.

Wang, Y., Berant, J., and Liang, P. (2015). Building a semantic parser overnight. In *Proceedings of ACL*, volume 1, pages 1332–1342.

References IV

Woods, W. A. (1973). Progress in natural language understanding: an application to lunar geology. In *Proceedings of the June 4-8, 1973, National Computer Conference and Exposition*, pages 441–450.

Yin, P. and Neubig, G. (2017). A Syntactic Neural Model for General-Purpose Code Generation. *arXiv preprint arXiv:1704.01696*.