

Towards Open-domain Generation of Programs from Natural Language

Graham Neubig
@ UT Austin 10/29/2018



Carnegie Mellon University
Language Technologies Institute

Acknowledgements

Based on work w/

Pengcheng Yin,



Bogdan Vasilescu

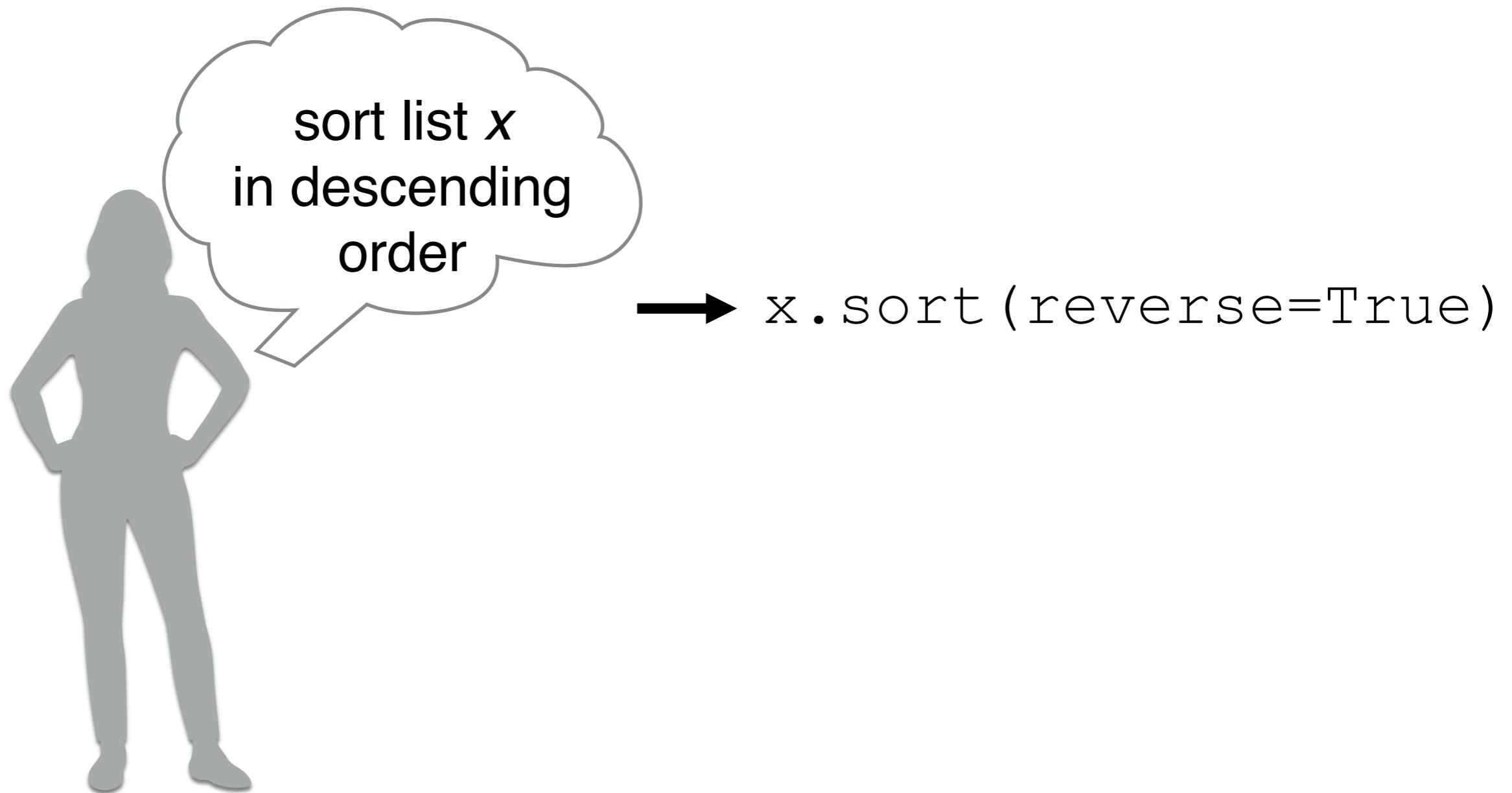


Bowen Deng, Edgar Chen, Junxian He, Chunting Zhou,
Shirley Hayati, Raphaël Olivier, Pravalika Avvaru, Anthony Tomasic

Supported by



Coding = Concept → Implementation



The (Famous) Stack Overflow Cycle

Formulate the Idea

sort my_list in descending order



Search the Web

python sort list in descending order



Browse thru. results

▲ This will give you a sorted version of the array.

167 `sorted(timestamp, reverse=True)`

▼ If you want to sort in-place:

`timestamp.sort(reverse=True)`

share improve this answer

edited Nov 15 '10 at 10:47

answered Nov 15 '10 at 10:42

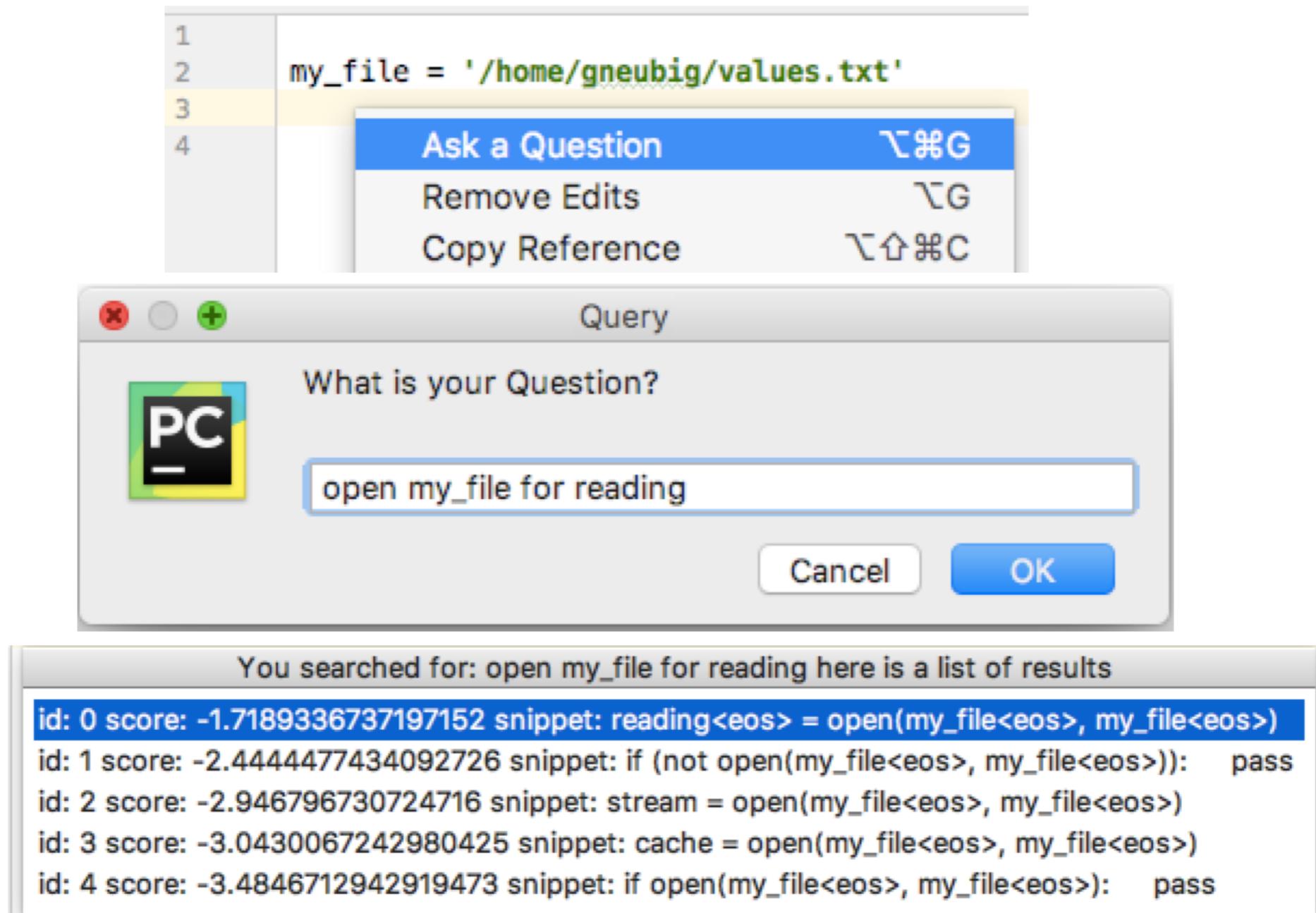
Marcelo Cantos
124k • 23 • 243 • 301



Modify the result

sorted(my_list, reverse=True)

Goal: Assistive Interfaces for Programmers



Interface by William Qian

Today's Agenda: Can Natural Language Help?

- Syntactic models to create code from natural language
- Large-scale mining of open-domain datasets for code generation
- Semi-supervised learning for semantic parsing and code generation
- Retrieval-based Code Generation

Natural Language vs. Programming Language

Natural Language vs. Code

Natural Language

Human interpretable

Ambiguous

Structured, but flexible

Code

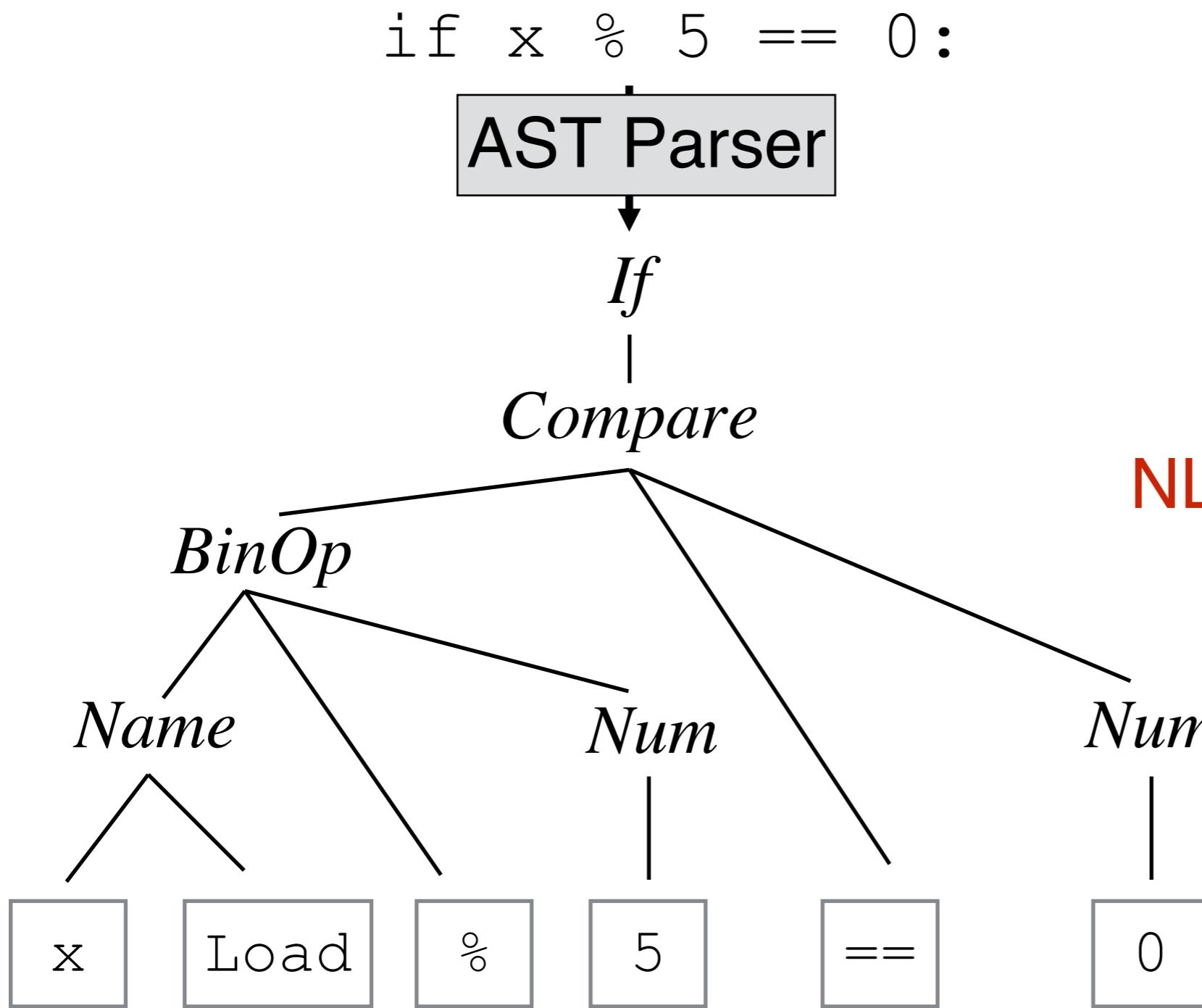
Human and machine interpretable

Precise in interpretation

Structured w/o flexibility

Note: Good summary in Allamanis et al. (2017)

Structure in Code



Can we take
advantage of
this for better
NL-code interfaces?

(used in models of Maddison & Tarlow 2014)

A Syntactic Neural Model for Code Synthesis from Natural Language

(ACL 2017)

Joint Work w/ Pengcheng Yin

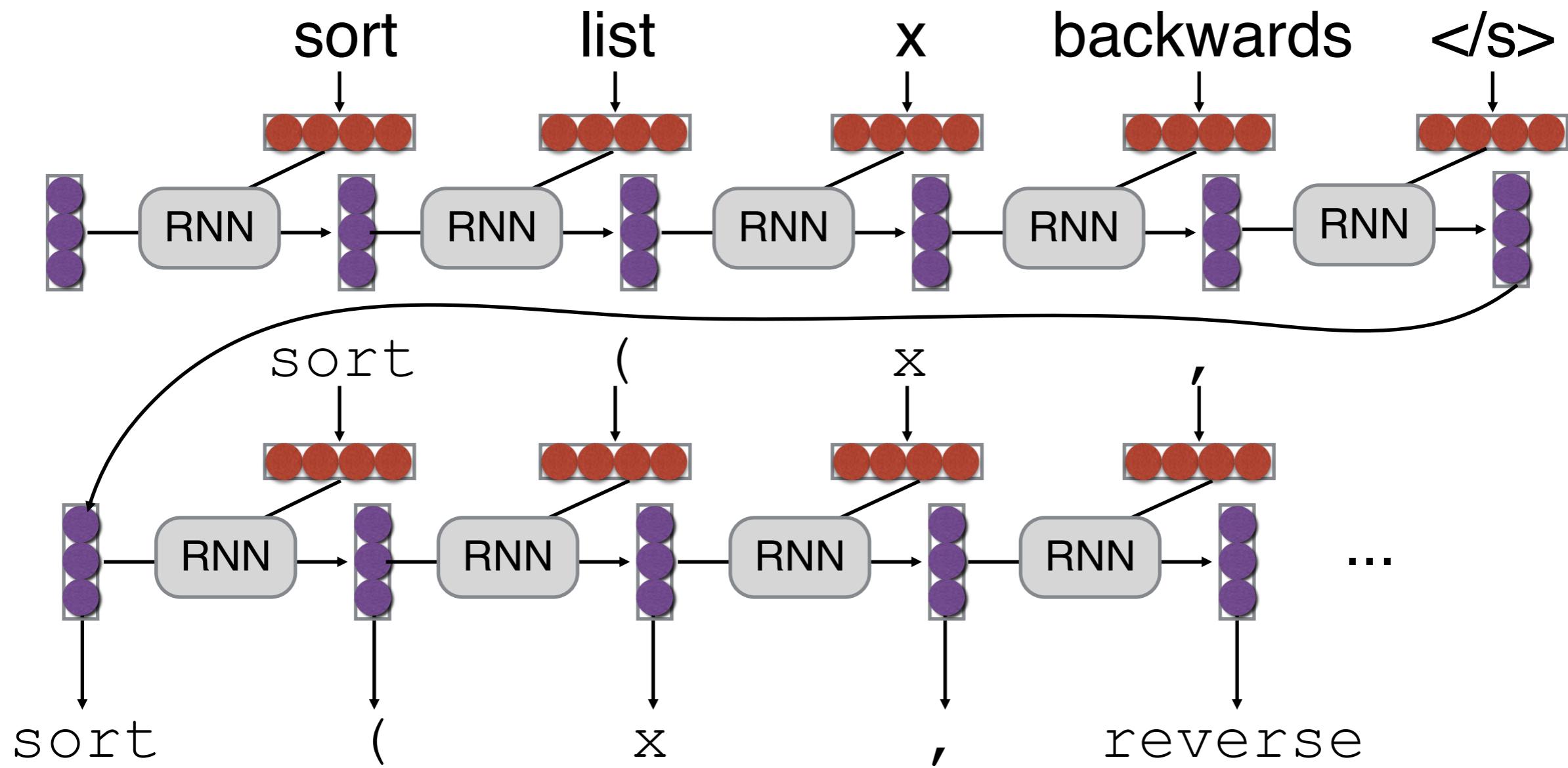
Previous Work

- Lots of work on rule-based methods for natural language programming (e.g. see Balzer 1985)
- Lots of work on semantic parsing w/ grammar-based statistical models (e.g. Wong & Mooney 2007)
- One work on using neural sequence-to-sequence models for code generation in Python (Ling et al. 2016)

Sequence-to-sequence Models

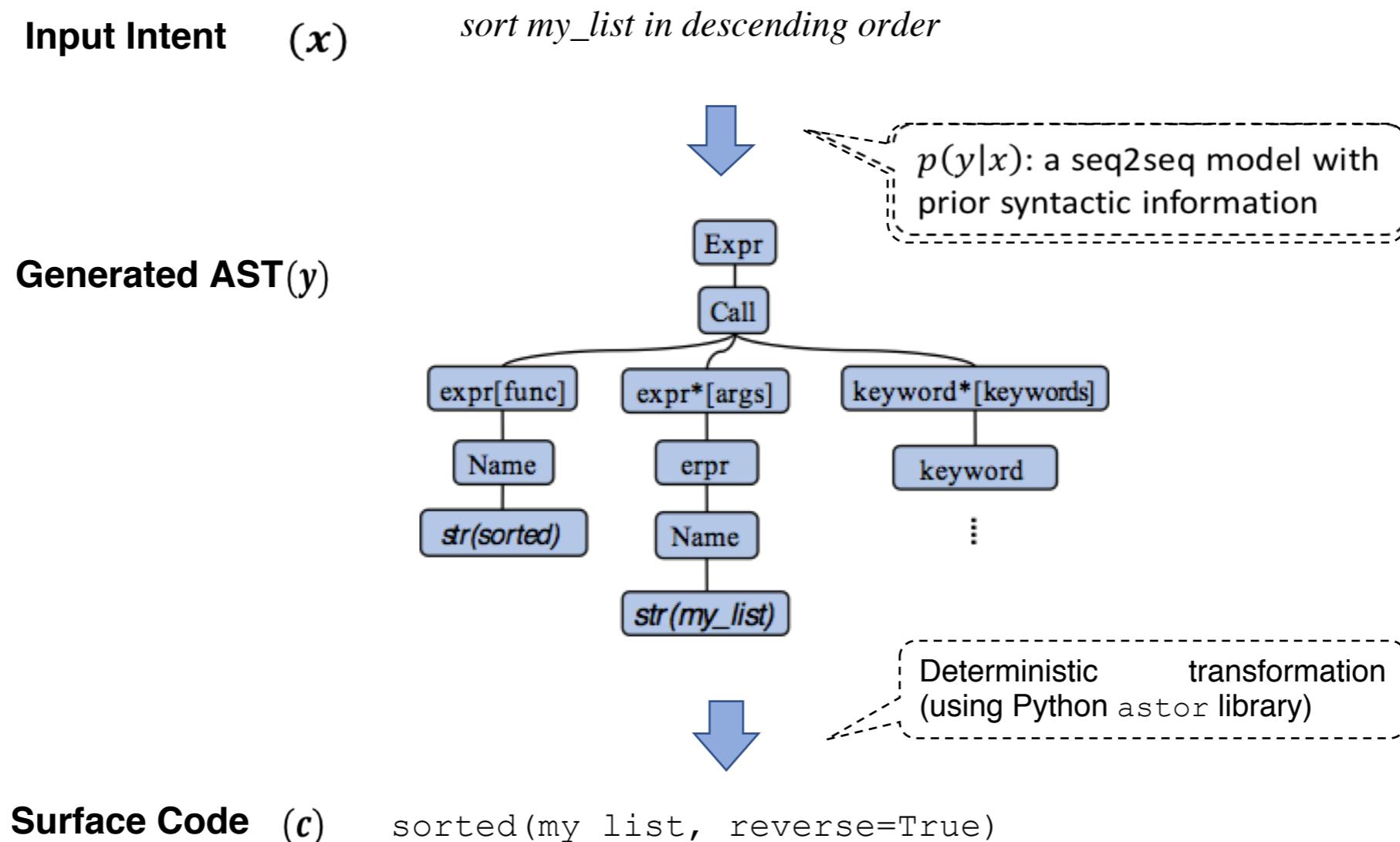
(Sutskever et al. 2014, Bahadanau et al. 2015)

- Neural network models for transducing sequences



Proposed Method: Syntactic Neural Models for Code Synthesis

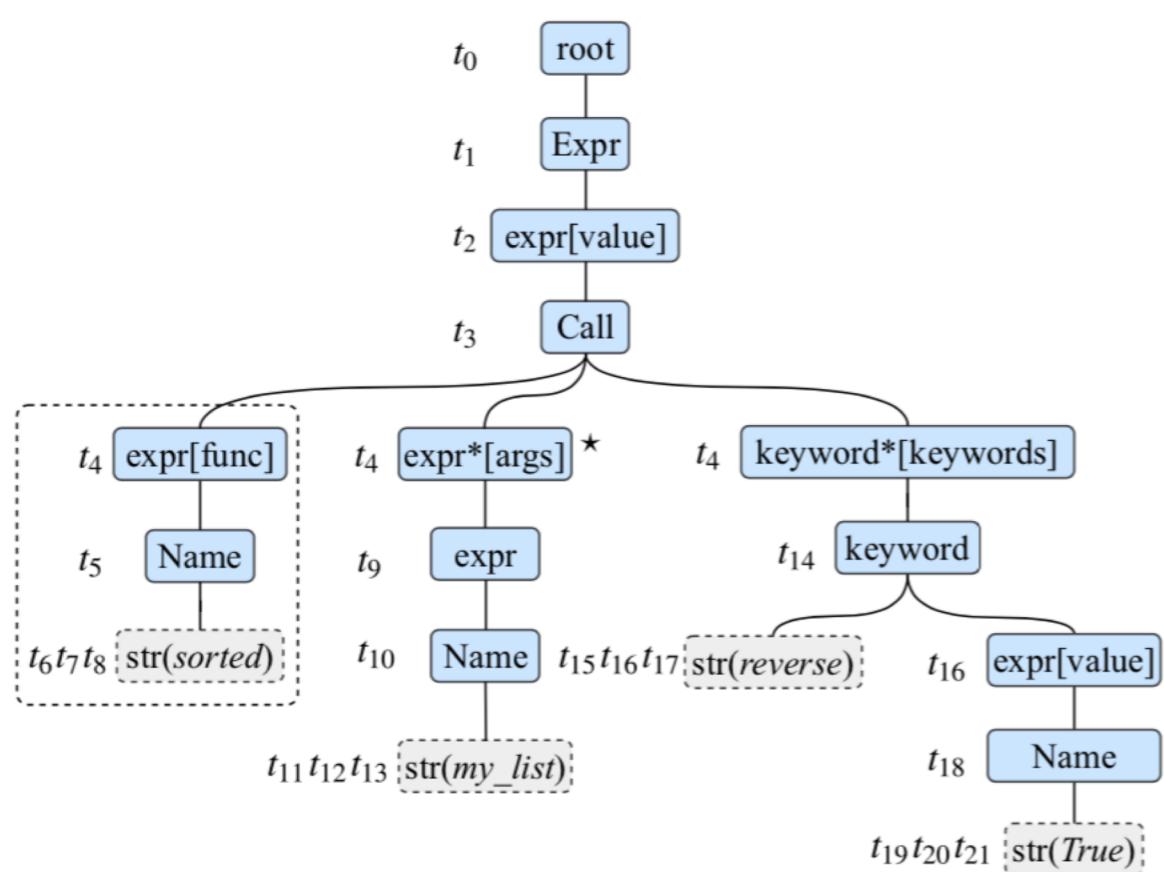
- **Key idea:** use the grammar of the programming language (Python) as prior knowledge in a neural model



NOTE: very nice contemporaneous work by Rabinovich et al. (2017)

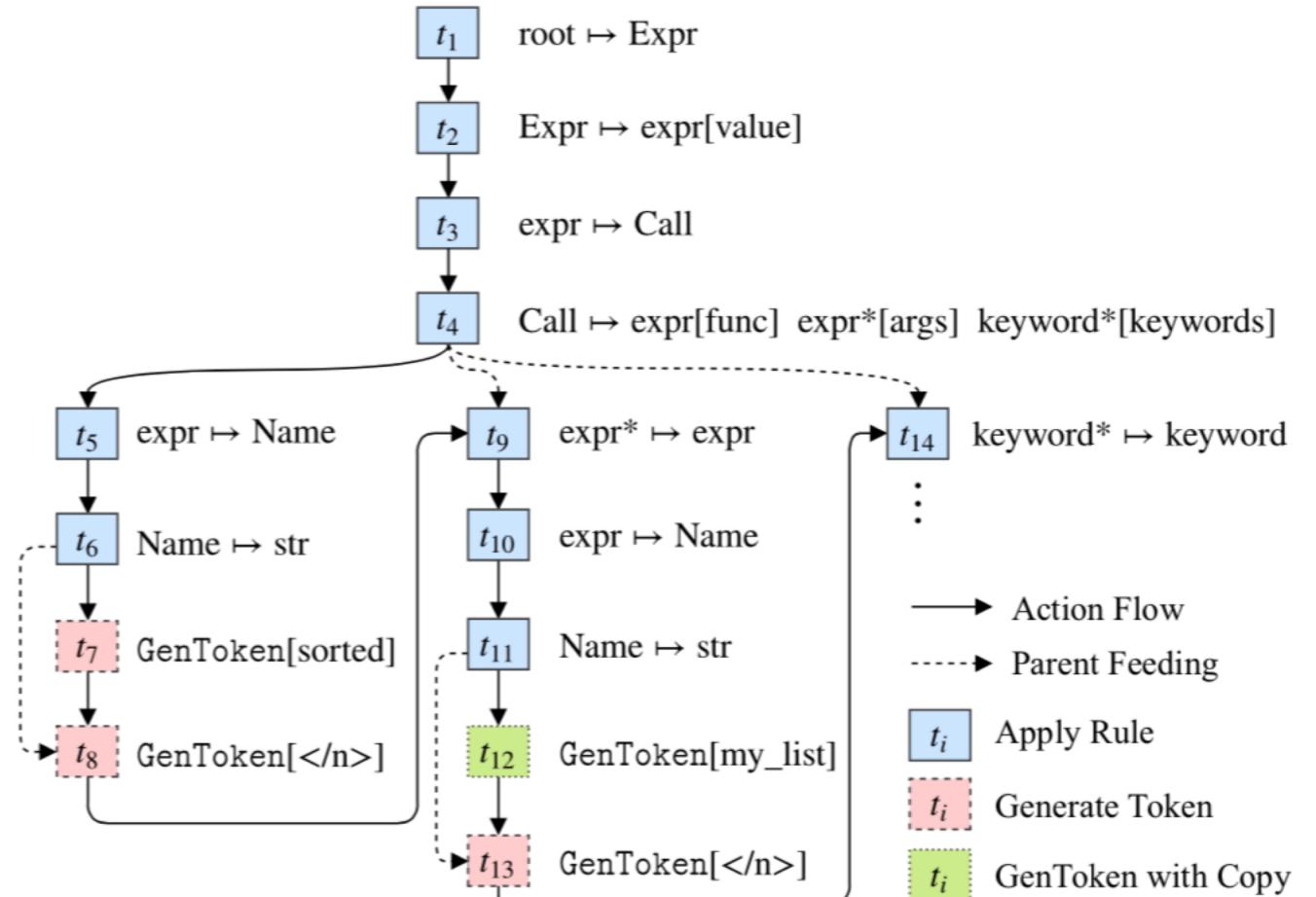
Generation Process

- Factorize the AST into actions:
 - `ApplyRule`: generate an internal node in the AST
 - `GenToken`: generate (part of) a token



(a)

Input: sort my_list in descending order

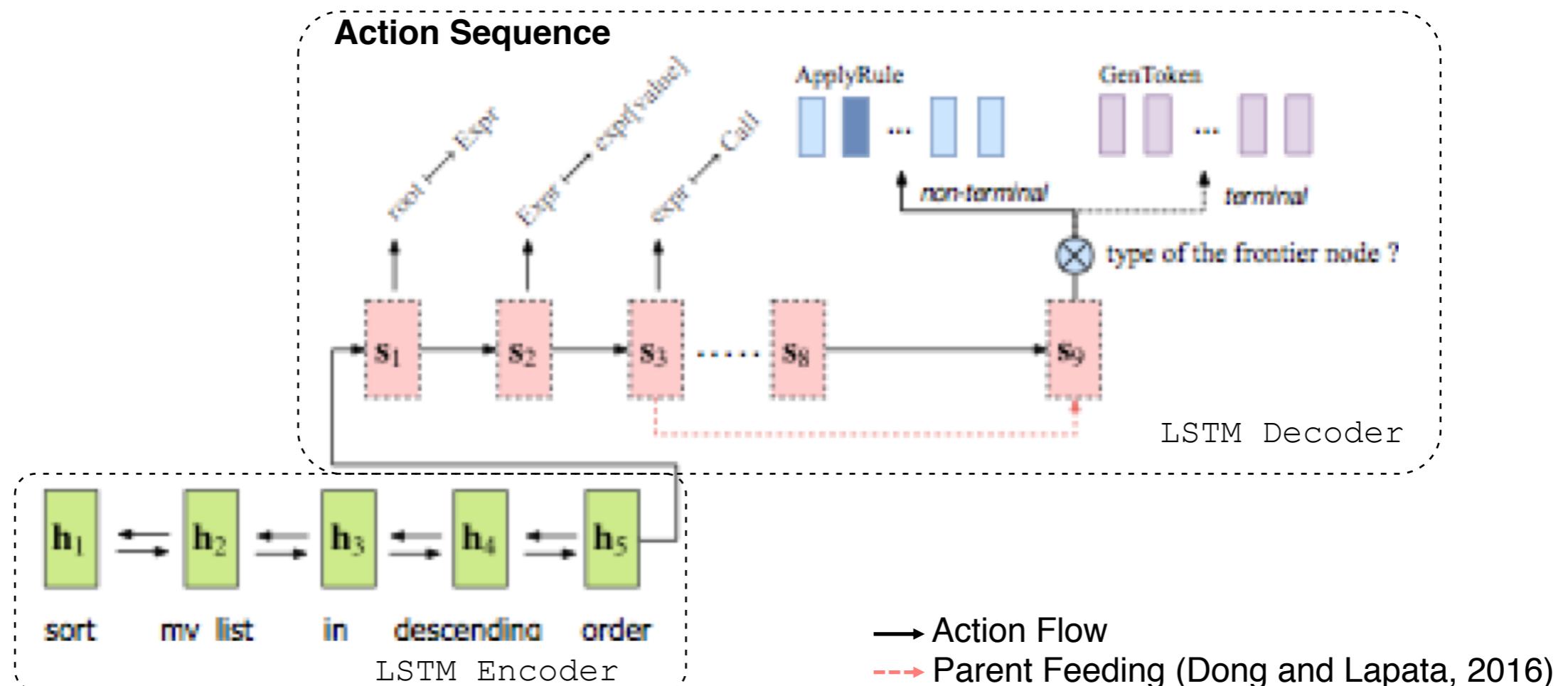


(b)

Code: `sorted(my_list, reverse=True)`

Formulation as a Neural Model

- **Encoder:** summarize the semantics of the NL intent
- **Decoder:**
 - Hidden state keeps track of the generation process of the AST
 - Based on the current state, predict an action to grow the AST



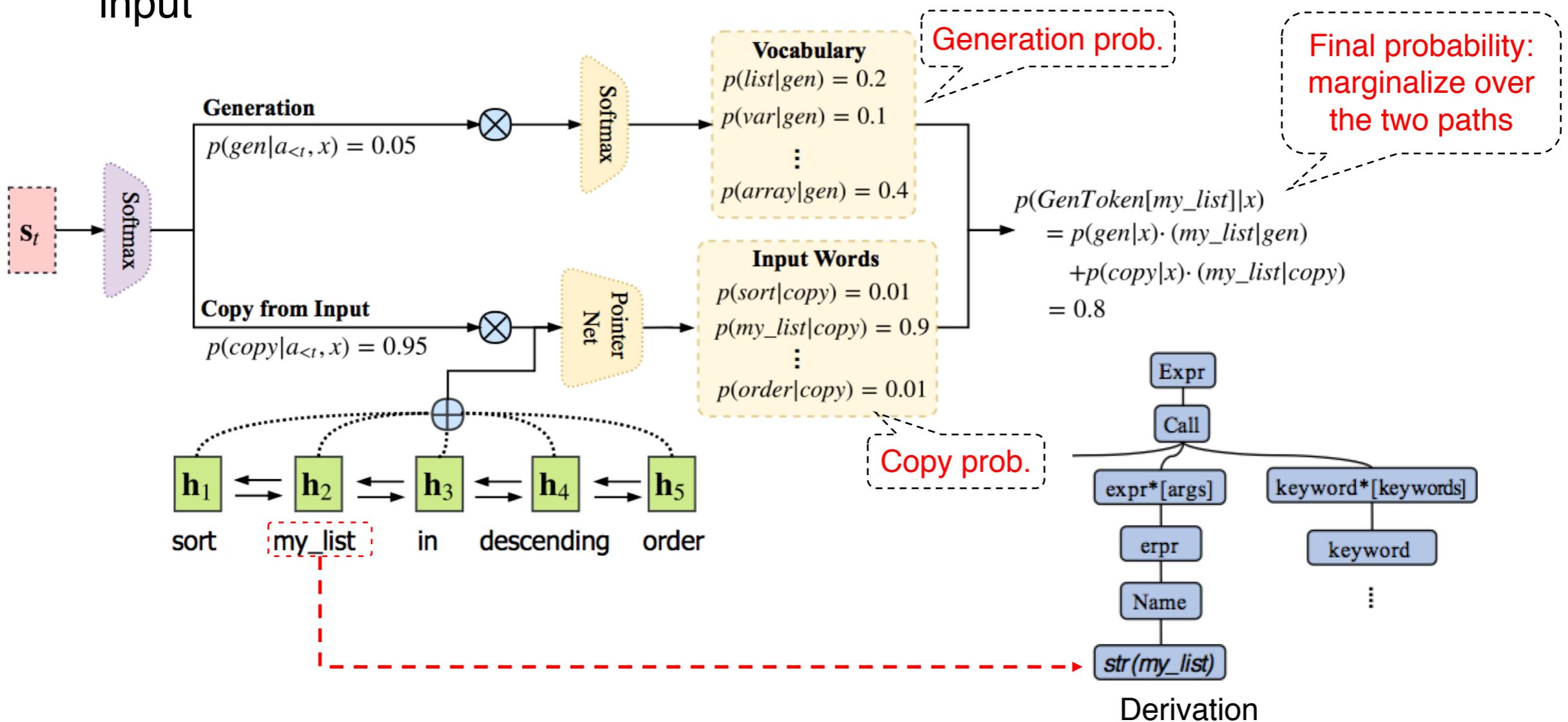
Computing Action Probabilities

- `ApplyRule[r]` : apply a production rule r to the current derivation

$$p(a_t = \text{APPLYRULE}[r]|x, a_{<t}) = \text{softmax}(\mathbf{W} \cdot g(\mathbf{s}_t))$$

- `GenToken[v]` : append a token v to the current terminal node

- Deal with OOV: learning to generate a token or directly copy it from the input



Experiments

- **Natural Language \mapsto Python code:**
 - *HearthStone* (*Ling et al., 2016*): card game implementation
 - *Django* (*Oda et al., 2015*): web framework
- **Natural Language \mapsto Domain Specific Language (Semantic Parsing)**
 - *IFTTT* (*Quirk et al., 2015*): personal task automation APP

Django Dataset

- **Description:** manually annotated descriptions for 18K lines of code
- **Target code:** one liners
- Covers a wide range of real-world use cases like I/O operation, string manipulation and exception handling

Intent	<i>call the function _generator, join the result into a string, return the result</i>
Target	<code>return '' .join(_generator())</code>

HearthStone Dataset

- **Description:** properties/fields of an HS card
- **Target code:** implementation as a Python class from HearthBreaker



Intent (Card Property)

*<name> Divine Favor </name> <cost> 3 </cost> <desc>
Draw cards until you have as many in hand as your
opponent </desc>*

Target (Python class, extracted from HearthBreaker)

```
class DivineFavor(SpellCard):  
    def __init__(self):  
        super().__init__("Divine Favor", 3, CHARACTER_CLASS.PALADIN,  
                        CARD_RARITY.RARE)  
    def use(self, player, game):  
        super().use(player, game)  
        difference = len(game.other_player.hand) - len(player.hand)  
        for i in range(0, difference):  
            player.draw()
```

[Ling *et al.*, 2016]

IFTTT Dataset

- Over 70K user-generated task completion snippets crawled from ifttt.com
- Wide variety of topics: home automation, productivity, etc.
- Domain-Specific Language (DSL): IF-THIS-THEN-THAT structure, much simpler grammar



<https://ifttt.com/applets/1p-autosave-your-instagram-photos-to-dropbox>

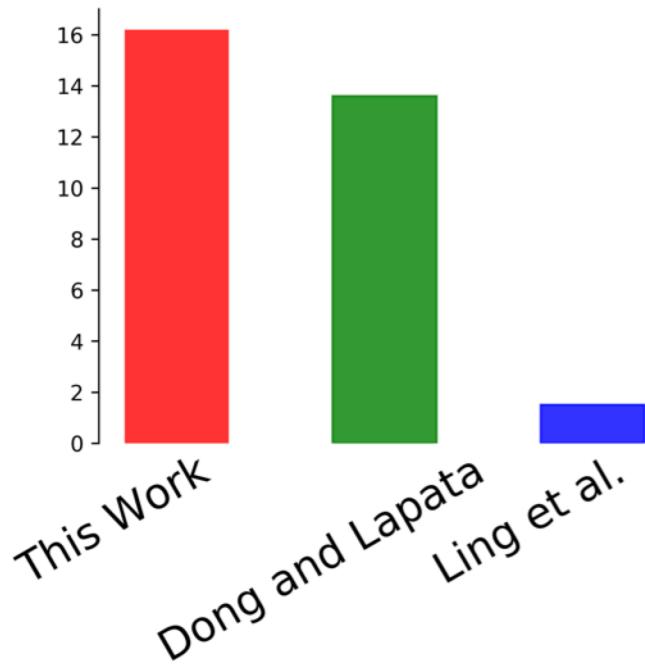
Intent *Autosave your Instagram photos to Dropbox*

Target **IF** Instagram.AnyNewPhotoByYou
THEN Dropbox.AddFileFromURL

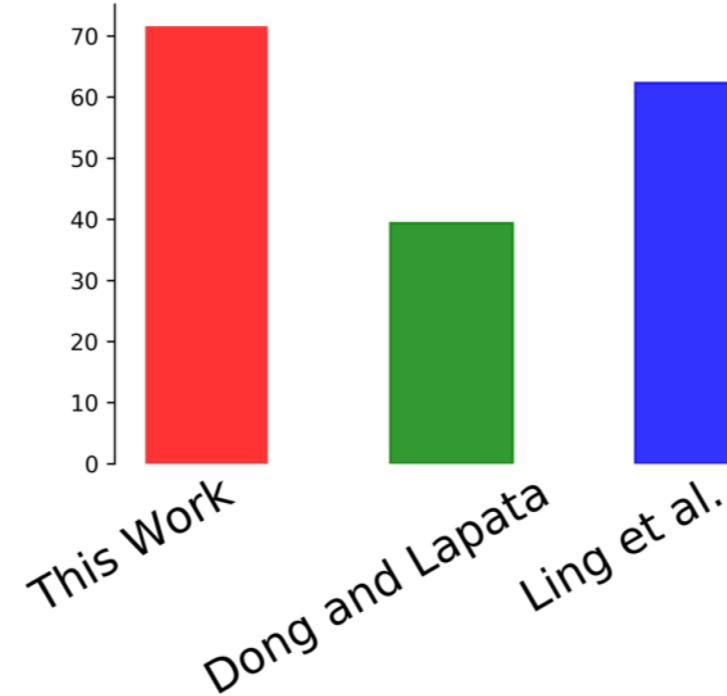
[Quirk *et al.*, 2015]

Results

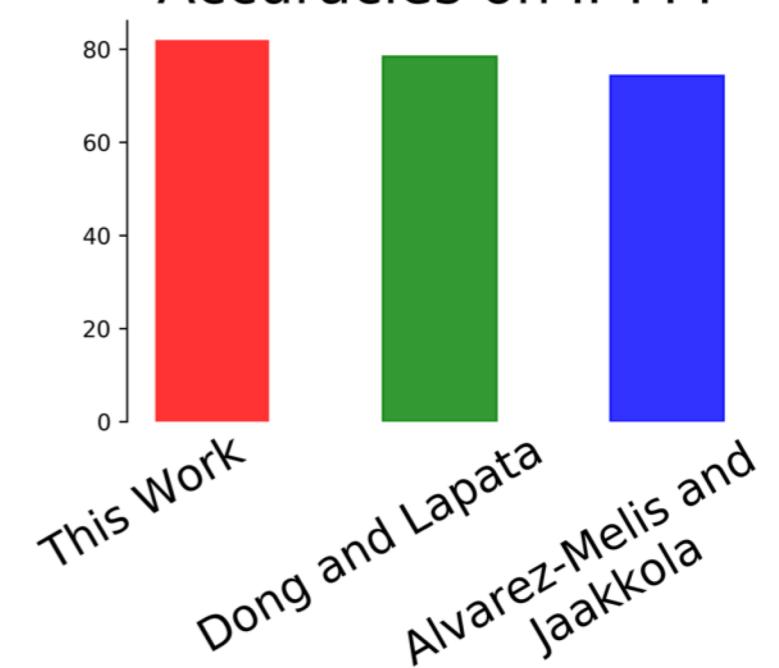
Accuracies on HearthStone



Accuracies on Django



Accuracies on IFTTT



- Baseline systems (*do not model syntax a priori*):
 - Latent Predictor Network [Ling *et al.*, 2016]
 - Seq2Tree [Dong and Lapata., 2016]
 - Doubly recurrent RNN [Alvarez-Melis and Jaakkola., 2017]
- **Take Home Msg:**
 - Modeling syntax helps for code generation and semantic parsing

Examples

Intent *join app_config.path and string 'locale' into a file path, substitute it for localedir.*

Pred. `localedir = os.path.join(app_config.path, 'locale')` ✓

Intent *self.plural is an lambda function with an argument n, which returns result of boolean expression n not equal to integer 1*

Pred. self.plural = lambda n: len(n) ✗

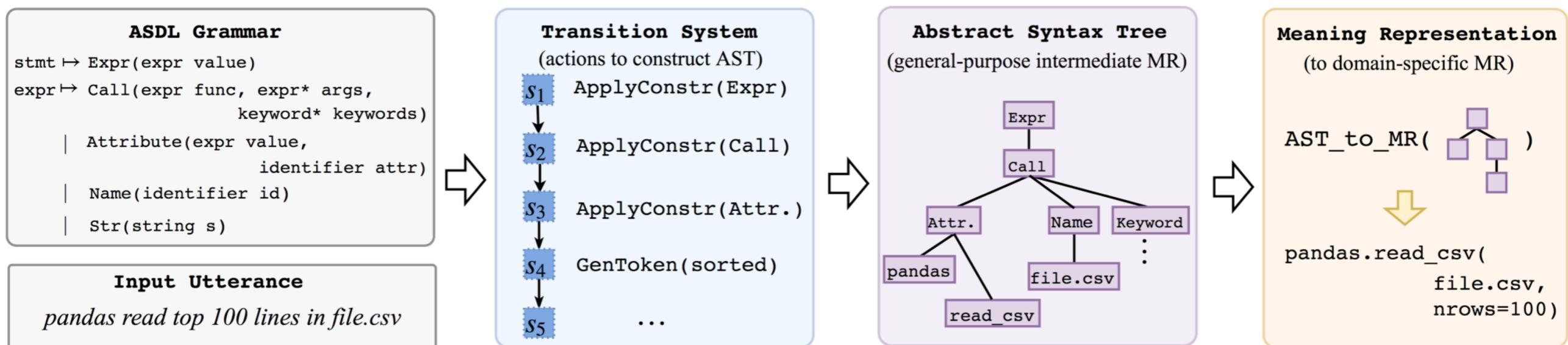
Ref. self.plural = lambda n: int(n!=1)

Intent <*name*> Burly Rockjaw Trogg </*name*> <*cost*> 5 </*cost*> <*attack*> 3 </*attack*> <*defense*> 5 </*defense*> <*desc*> Whenever your opponent casts a spell, gain 2 Attack. </*desc*> <*rarity*> Common </*rarity*> ...

Ref. `class BurlyRockjawTrogg(MinionCard):
 def __init__(self):
 super().__init__('Burly Rockjaw Trogg', 4, CHARACTER_CLASS.ALL, CARD_RARITY.COMMON)
 def create_minion(self, player):
 return Minion(3, 5, effects=[Effect(SpellCast(player=EnemyPlayer()),
 ActionTag(Give(ChangeAttack(2)), SelfSelector()))])` ✓

TranX Parser [Yin+18]

- Transition-based AST parser based on “abstract syntax description language”
- Can define language flexibly for various types of semantic parsing
- Good results out-of-the-box!



<https://github.com/pcyin/tranX>

Learning to Mine NL/Code Pairs from Stack Overflow

(MSR 2018)

Joint Work w/
Pengcheng Yin, Bowen Deng, Edgar Chen, Bogdan Vasilescu

Datasets are Important!

- Our previous work used Django, HearthStone, IFTTT, manually curated datasets
- It couldn't have been done without these
- But these are **extremely specific, and small**

StackOverflow is Promising!

- StackOverflow promises a large data source for code synthesis
- But code snippets don't necessarily reflect the answer to the original question

Removing duplicates in lists  Intent

▲ 406

Pretty much I need to write a program to check if a list has any duplicates and if it does remove them and returns a new list with the items that weren't duplicated/removed. This is what I have but to be honest I do not know what to do.

▼

★ 138

```
def remove_duplicates():
    t = ['a', 'b', 'c', 'd']
    t2 = ['a', 'c', 'd']
    for t in t2:
        t.append(t.remove())
    return t
```

▲ 780

▼

✓

The common approach to get a unique collection of items is to use a `set`. Sets are *unordered* collections of *distinct* objects. To create a set from any iterable, you can simply pass it to the built-in `set()` function. If you later need a real list again, you can similarly pass the set to the `list()` function.

The following example should cover whatever you are trying to do:

```
>>> t = [1, 2, 3, 1, 2, 5, 6, 7, 8]
>>> l
[1, 2, 3, 1, 2, 5, 6, 7, 8]
>>> list(set(t))
[1, 2, 3, 5, 6, 7, 8]
>>> s = [1, 2, 3]
>>> list(set(t) - set(s))
[8, 5, 6, 7]
```

 Context 1 Snippet 1

As you can see from the example result, the original order is not maintained. As mentioned above, sets themselves are unordered collections, so the order is lost. When converting a set back to a list, an arbitrary order is created.

▲ 222

▼

FWIW, the new (v2.7) Python way for removing duplicates from an iterable while keeping it in the original order is:

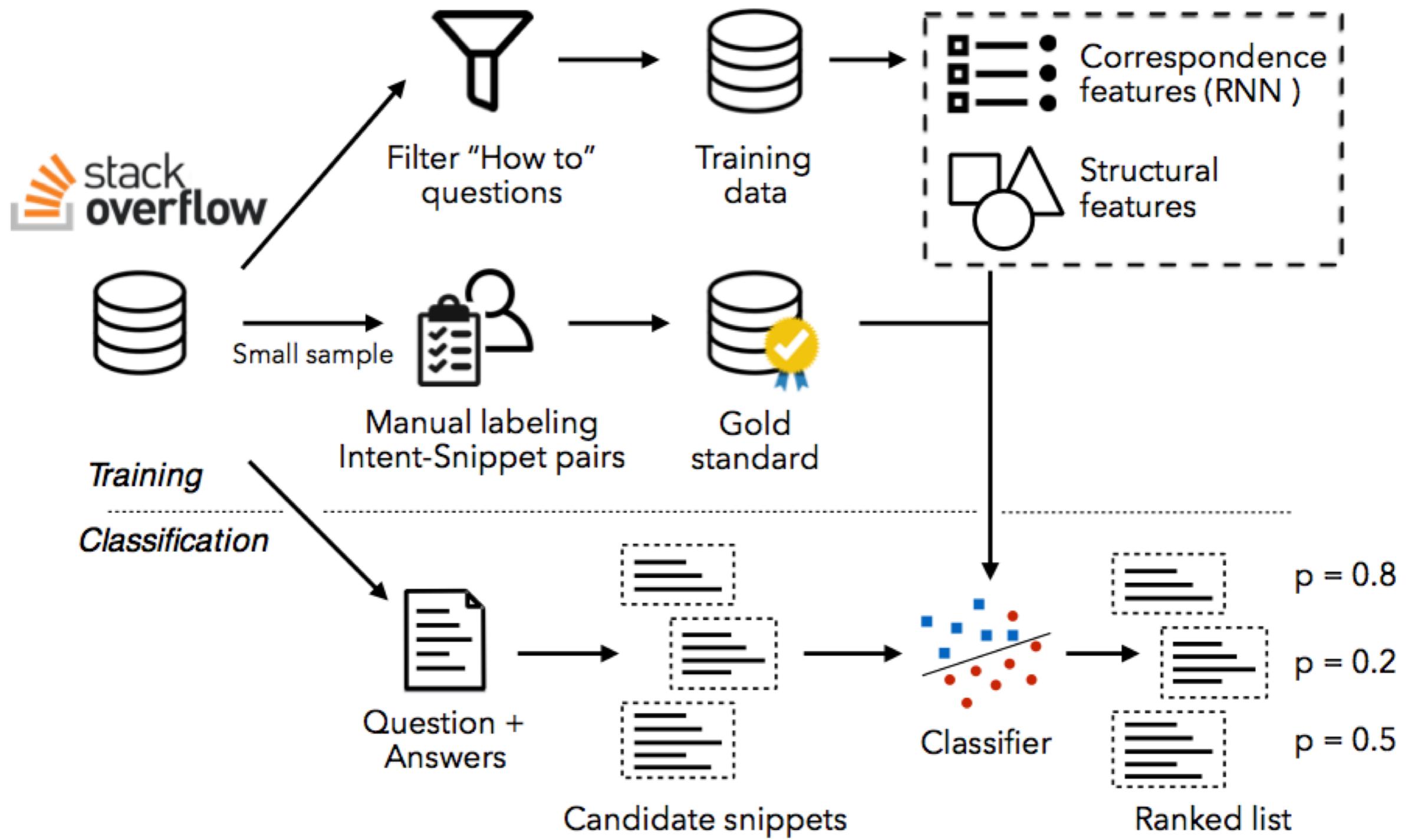
```
>>> from collections import OrderedDict
>>> list(OrderedDict.fromkeys('abracadabra'))
['a', 'b', 'r', 'c', 'd']
```

 Context 2 Snippet 2

Question

Answers

Mining Method



Annotation

You can also use `os.path.isfile`

2437 Return `True` if path is an existing regular file. This follows symbolic links, so both `islink()` and `.isfile()` can be true for the same path.

```
import os.path
os.path.isfile(fname)
```

if you need to be sure it's a file.

Starting with Python 3.4, the `pathlib` module offers an object-oriented approach:

```
from pathlib import Path
my_file = Path("/path/to/file")
if my_file.is_file():
    # file exists
```

answered Sep 17 '08 at 12:57

55 It's fair to point out though, that if you're talking about script running locally, it should be no problem. Aug 5 '11 at 19:38

37 It's also worthwhile to mention that the "potential security vulnerabilities" are characteristic for any concurrent system and the canonical way to deal with race conditions is to obtain a lock on the file before calling `'isfile'`. Oct 27 '12 at 9:21

0 How could I return a Boolean value if the file is found? Jun 29 '14 at 23:00

3 @Vladimir Putin, this method returns `'true'` if the file exists or `'false'` if the file does not exist. You can get to a path by using `'cd /path/to/file'` in the terminal and then executing the python shell by `'python'` or `'python2'` depending on your system. Then just enter the file you want that is in the directory you just navigated to in the terminal. Jul 11 '14 at 2:08

4 What about for device files, like `'/dev/null'`? Won't this return `'False'` in that case? Nov 27 '14 at 14:35

0 I am using `'.isfile("a.txt")'` and `'.exists("a.txt")'` on my Mac, it seems like it will create `'a.txt'` file and return `'False'` if it does not already exist. How can I prevent it from creating the file? Jun 29 '15 at 22:30

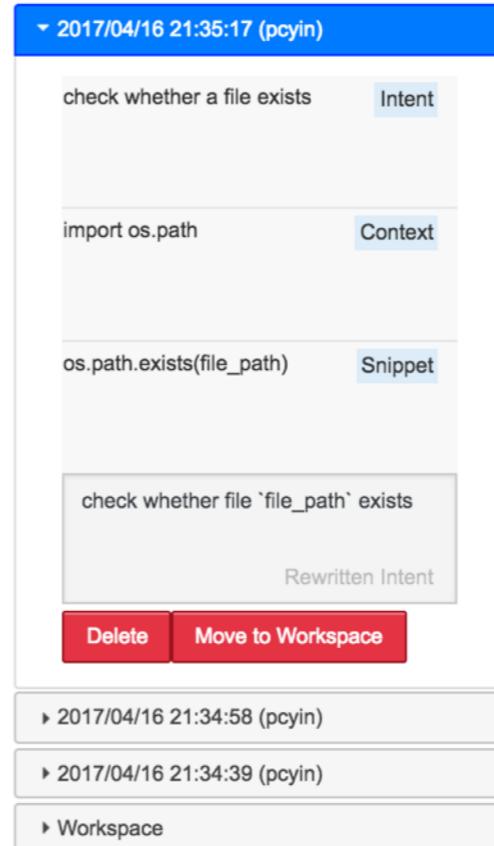
3 `path.exists()` seems much more obvious to check for existence of a file than `path.isfile()`. Why do double the number of voters think `isfile()` is more useful than `exists()`? Sep 4 '15 at 7:06

6 @Ash, as PierreBdR said, "`"os.path.exists"` ... returns `'True'` for both files and directories." The OP asked specifically how to check for existence of a file. Sometimes it makes a difference but you are welcome to use whatever suits your use. Oct 26 '15 at 18:38

0 @Lucretiel - If it is still relevant: `"os.path.isfile("/dev/null)"` returns `"False"`. Nov 13 '16 at 17:03

1 `pathlib` is also backported to 2.7. Jan 20 '17 at 18:59

1 The official backport is [pathlib2](<https://pypi.python.org/pypi/pathlib2/>), `'pathlib'` module isn't maintained anymore. Feb 27 '17 at 21:45



- ~100 posts for Python/Java

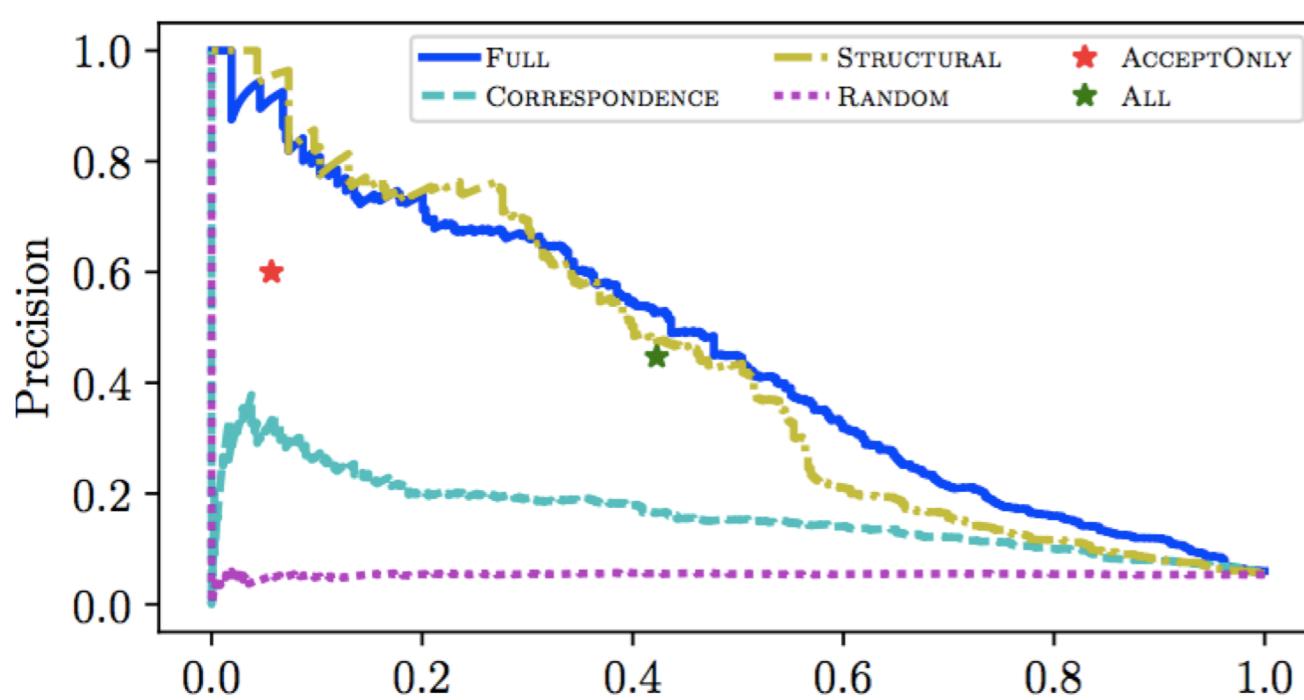
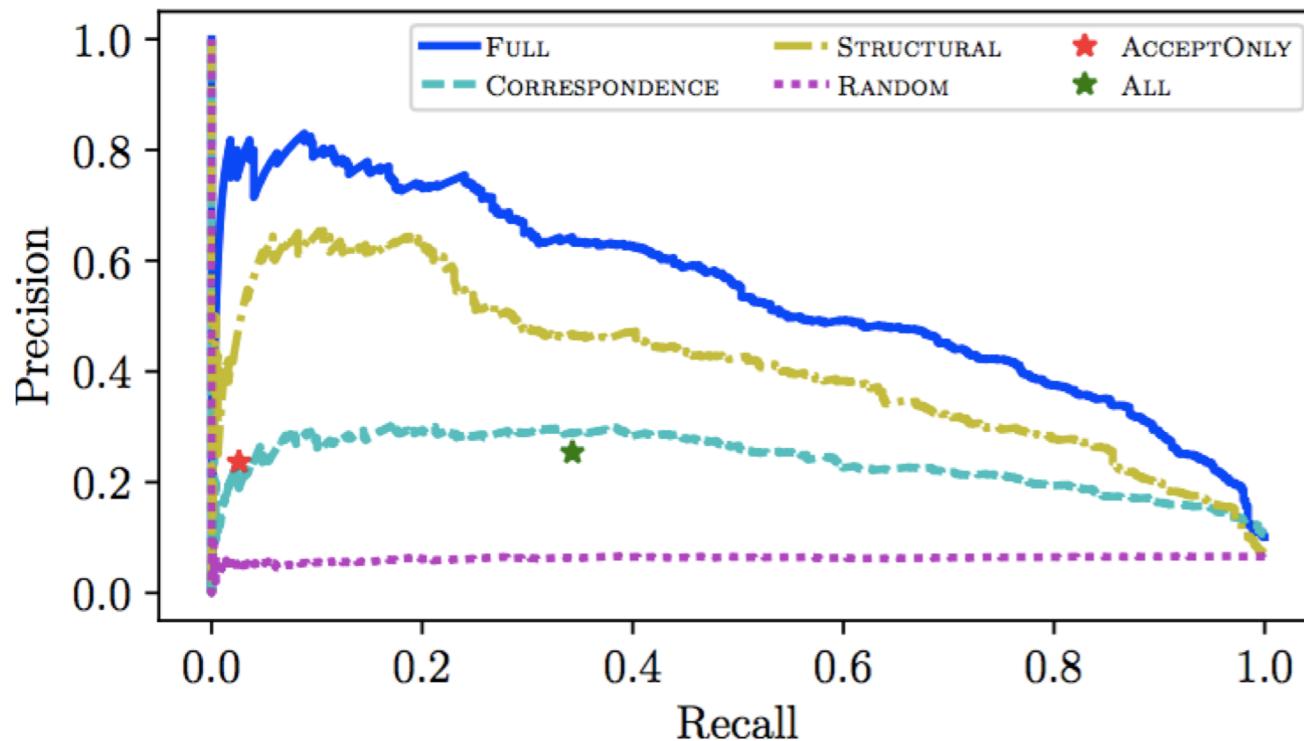
Features (1): Structural Features

- "does this look like a valid snippet?"
- Position:** Is the snippet a full block? The start/end of a block?
The only block in an answer?
- Code Features:** Contains import? Starts w/ assignment? Is value?
- Answer Quality:** Answer is accepted? Answer is rank 1, 2, 3?
- Length:** What is the number of lines?

Features (2): Correspondence Features

- "do the intent and snippet look like they match?"
 - Train an RNN to predict $P(\text{intent} \mid \text{snippet})$ and $P(\text{snippet} \mid \text{intent})$ given heuristically extracted noisy data
 - Use log probabilities and normalized by z score over post, etc.

Main Results

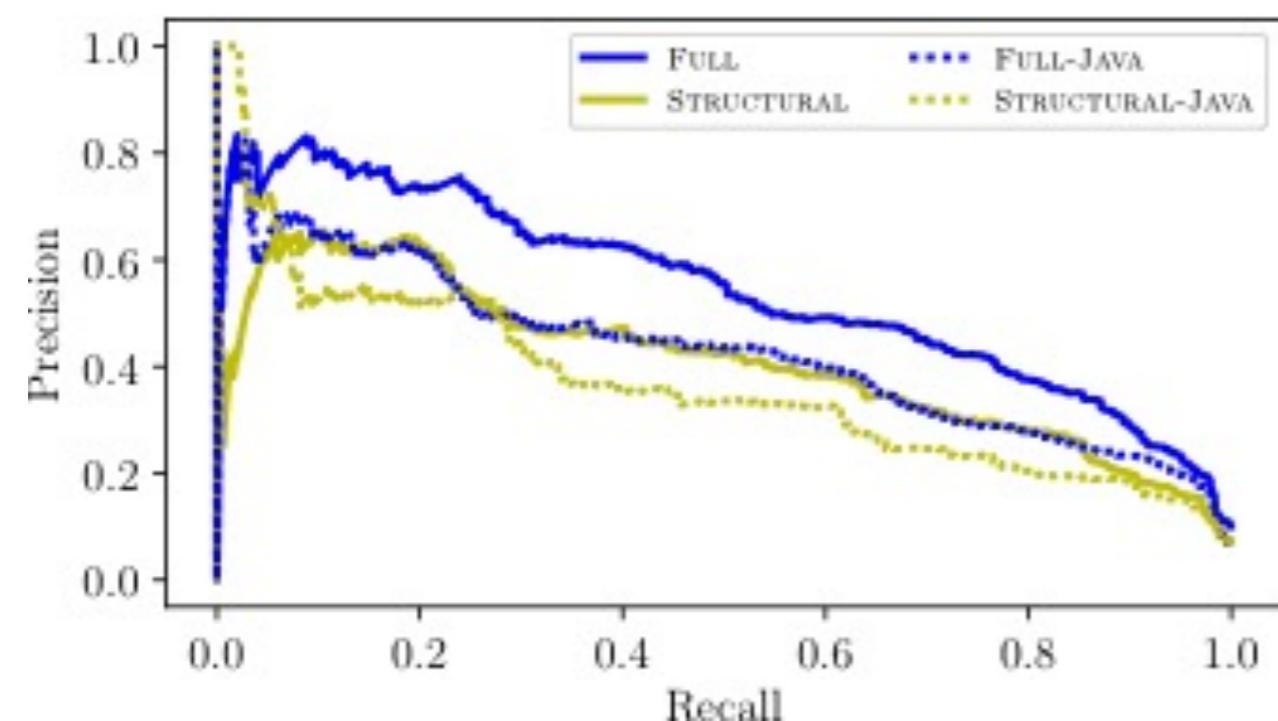


- On both Python and Java, better results than heuristic strategies
- Both structural and correspondence features were necessary

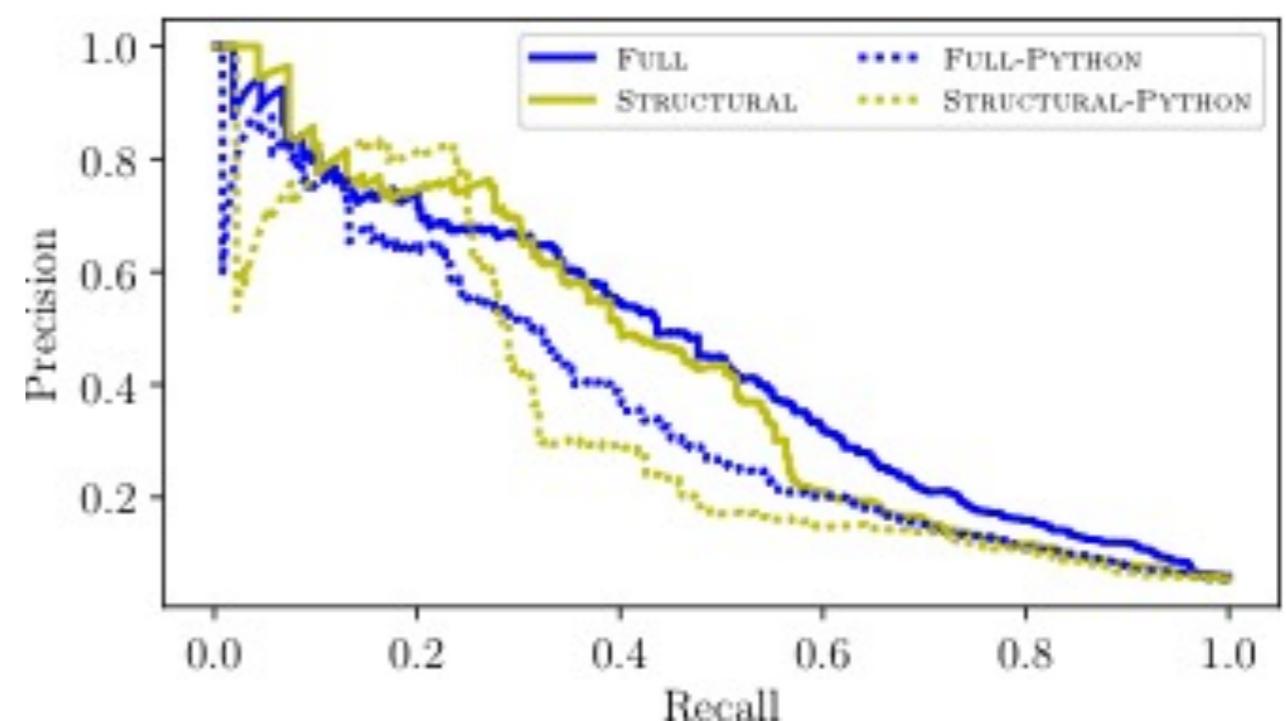
Transfer Learning

- Can we perform classification w/ no labeled data for that language?

Python



Java



Examples

I₁: Remove specific characters from a string in python

URL: <https://stackoverflow.com/questions/3939361/>

Top Predictions:

S₁ string.replace('1', '') ✓

S₂ line = line.translate(None, '!@#\$') ✓

S₃ line = re.sub('![#@\\$]', '', line) ✓

I₂: Get Last Day of the Month in Python

URL: <https://stackoverflow.com/questions/42950/>

Top Predictions:

S₁ calendar.monthrange(year, month)[1] ✓

S₂ calendar.monthrange(2100, 2) ✓

S₃ (datetime.date(2000, 2, 1) - datetime.timedelta(days=1))✓

I₃: Delete a dictionary item if the key exists

URL: <https://stackoverflow.com/questions/15411107/>

Top Predictions:

S₁ mydict.pop('key', None) ✓

S₂ del mydict[key] ✓

S₃ new_dict = {k: mydict[k] for k in keys_to_keep}✗

I₄: Python:take the content of a list and append it to another list

URL: <https://stackoverflow.com/questions/8177079/>

Top Predictions:

S₁ list2.append(list1) ✗

S₂ list2.extend(list1) ✓

S₃ list1.extend(mylog) ✓



CoNaLa: Code Natural-language Challenge

- ~2500 mined and manually verified examples
- ~600k automatically mined examples

```
{  
  "question_id": 36875258,  
  "intent": "copying one file's contents to another in python",  
  "rewritten_intent": "copy the content of file 'file.txt' to file 'file2.txt'",  
  "snippet": "shutil.copy('file.txt', 'file2.txt')"  
}  
  
{  
  "question_id": 22240602,  
  "intent": "How do I check if all elements in a list are the same?",  
  "rewritten_intent": "check if all elements in list `mylist` are the same",  
  "snippet": "len(set(mylist)) == 1"  
}
```

<http://conala-corpus.github.io>

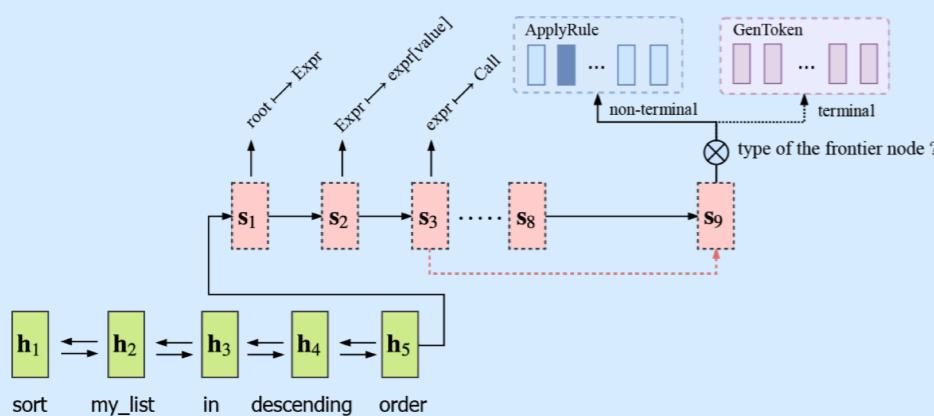
StructVAE: Semi-supervised Learning for Semantic Parsing

(ACL 2018)

Joint Work w/
Pengcheng Yin, Junxian He, Chunting Zhou

Motivation

Neural Models are Data Hungry



Purely supervised neural semantic parsing models require large amounts of training data



Data Collection is Costly

Copy the content of file 'file.txt' to file 'file2.txt'
`shutil.copy('file.txt', 'file2.txt')`

Get a list of words `words` of a file 'myfile'
`words = open('myfile').read().split()`

Check if all elements in list `mylist` are the same
`len(set(mylist)) == 1`

Collecting parallel
training data costs
and

[Yin et al., 2018] 1700 USD for 3K Python code generation examples
[Berant et al., 2013] 3000 USD for 5.7K question-to-logical form examples

Existing Solutions

Weakly supervised Learning

*Q: Which college did Obama go to?
(and (Type University)
(Education BarackObama))*

A: Occidental College, Columbia Univ.

Clarke et al. (2010)

Liang et al. (2011)

Berant et al. (2013)

Berant and Liang (2014)

Yih et al. (2015)

Data Augmentation

*What states border **texas**?*

`is_state(x) and
border(x, texas)`

*What states border **ohio**?*

`is_state(x) and
border(x, ohio)`

Jia and Liang, (2016)

Wang et al. (2015)

Zero-Shot Learning and Domain Adaptation

Fan et al. (2017)
Su and Yan, (2017)
Herzig and Berant,
(2018)

Semi-supervised Semantic Parsing

Limited Amount of Labeled Data

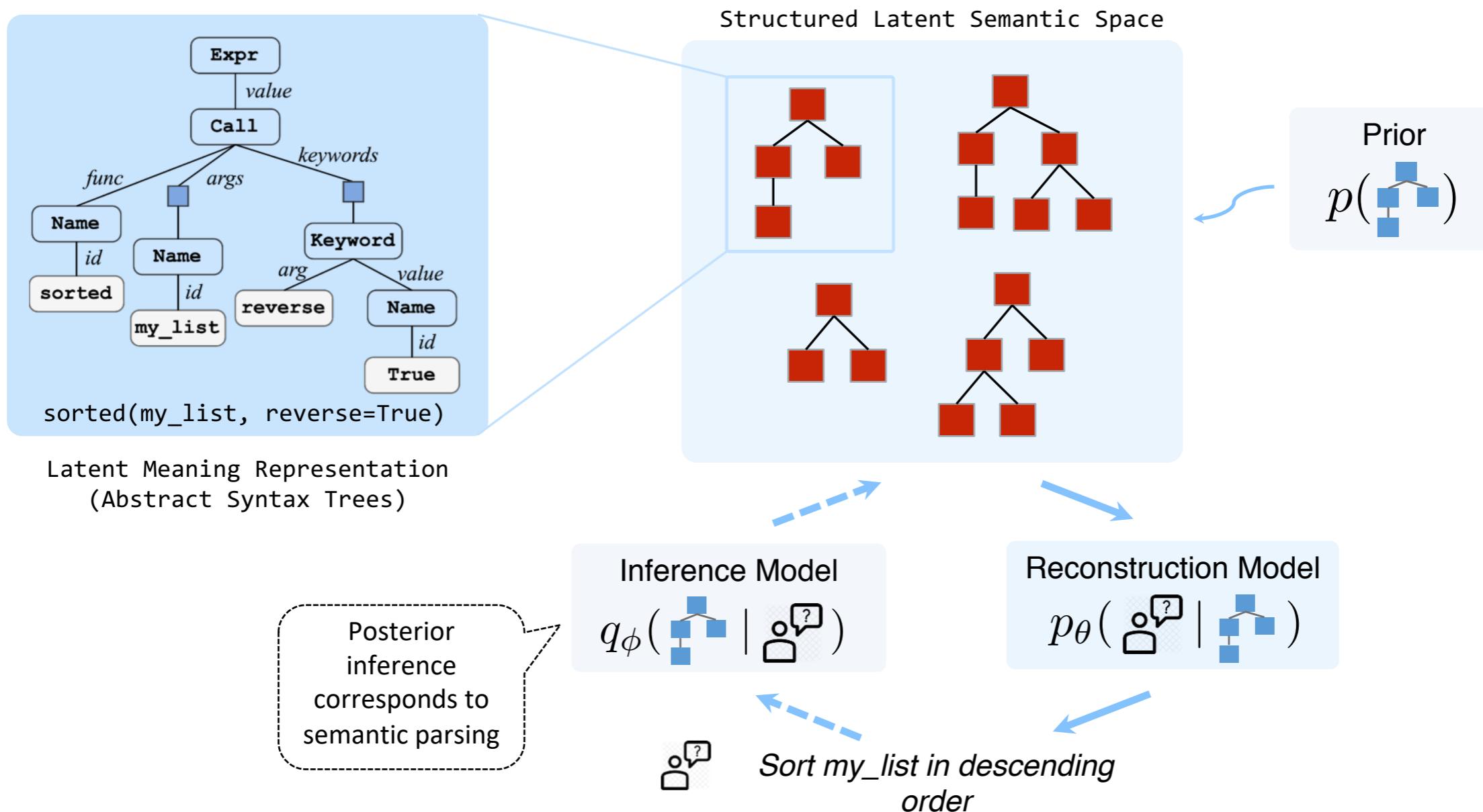
- 👤❓ Sort my_list in descending order
🤖 sorted(my_list, reverse=True)
- 👤❓ Copy the content of file 'file.txt' to file 'file2.txt'
🤖 shutil.copy('file.txt', 'file2.txt')
- 👤❓ Check if all elements in list `mylist` are the same
🤖 len(set(mylist)) == 1



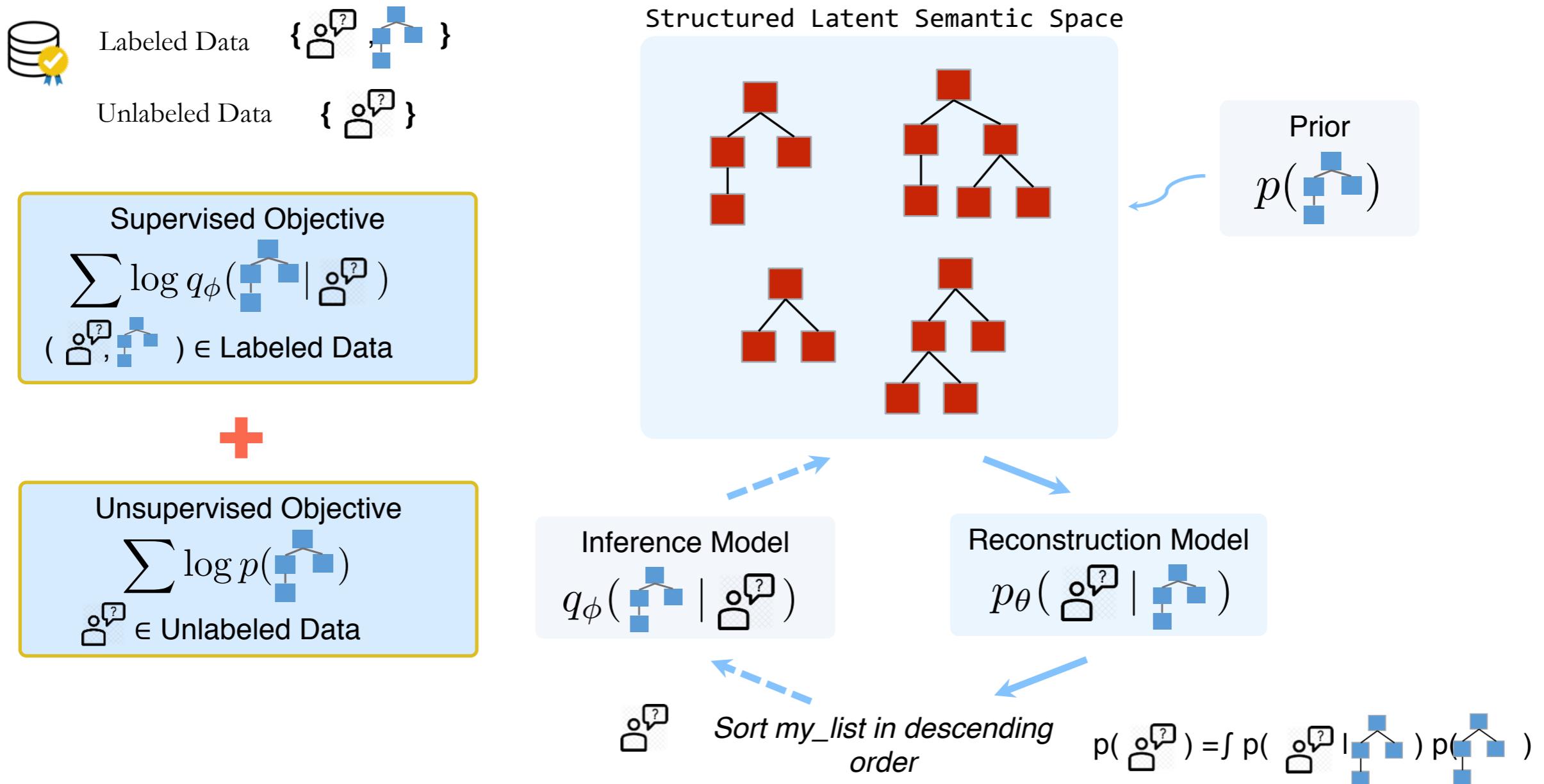
Extra Unlabeled Utterances

- 👤❓ Get a list of words `words` of a file 'myfile'
- 👤❓ Convert a list of integers into a single integer
- 👤❓ Format a datetime object `when` to extract date only
- 👤❓ Swap values in a tuple/list in list `mylist`
- 👤❓ BeautifulSoup search string 'Elsie' inside tag 'a'
- 👤❓ Convert string to lowercase

Tree-structured Latent Variables



Semi-supervised Learning w/ StructVAE



StructVAE: VAEs with Structured Latent Variables

Inference Model
 $q_\phi(\text{Tree} \mid \text{Text})$

Neural semantic parser

Reconstruction Model
 $p_\theta(\text{Text} \mid \text{Tree})$

Neural sequence-to-sequence model

Prior
 $p(\text{Tree})$

Neural Language Model

(use linearized trees as inputs)

Unsupervised Objective
 $\sum_{\text{Text} \in \text{Unlabeled Data}} \log p(\text{Tree} \mid \text{Text})$

Variational approximation of the marginal likelihood

$$\log p(\text{Text}) \geq \sum_{\text{Tree}' \sim q_\phi(\text{Tree} \mid \text{Text})} \log p_\theta(\text{Text} \mid \text{Tree}') - \text{KL-Divergence}[q_\phi(\text{Tree} \mid \text{Text}) \parallel p(\text{Tree})]$$

[Miao and Blunsom,
2016]

How Does Unsupervised Data Help?

Supervised Objective
 $\sum_{(x, \hat{x}) \in \text{Labeled Data}} \log q_\phi(\hat{x} | x)$

$$\nabla = \sum_{\text{Training Examples}} \frac{\partial \log q_\phi(\hat{x} | x)}{\partial \phi}$$

How Does Unsupervised Data Help?

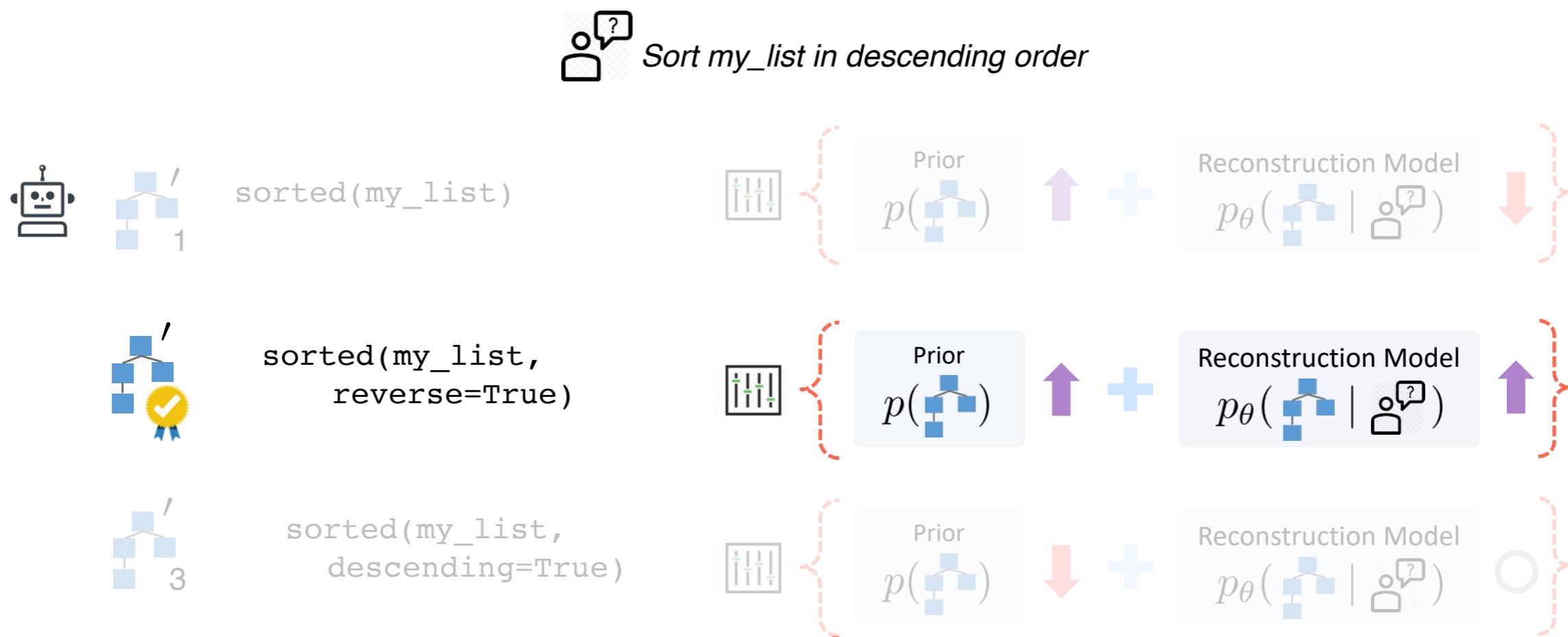
Unsupervised Objective
 $\sum_{\text{Unlabeled Data}} \log p(\text{latent} | \text{obs})$

$$\nabla \propto \sum_{\text{Sampled}} , \quad \text{latent} \times \frac{\partial q_\phi(\text{latent}' | \text{obs})}{\partial \phi}$$

The learning signal  $\approx \{$ Prior $p(\text{latent})$ + Reconstruction Model $p_\theta(\text{latent} | \text{obs})$ $\}$

Learning signal acts as the tuning weights of gradients received by different sampled latent meaning representations from the inference model

How Does Unsupervised Data Help?

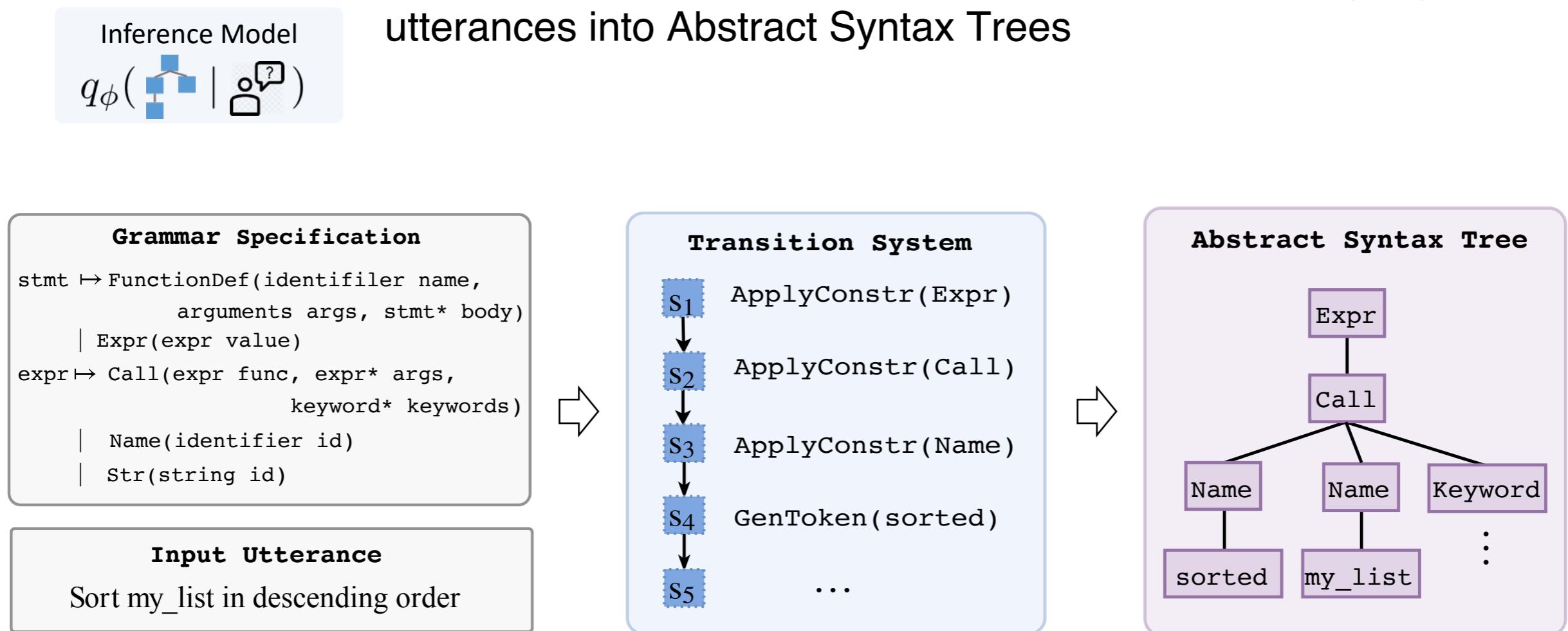


Learning fevers sampled latent meaning representations that are both:

- Faithfully encode the semantics of the utterance -> high reconstruction score
- Succinct and natural -> high prior probability

The Inference Model: AST-based Parser

A transition-based parser that transduces natural language utterances into Abstract Syntax Trees

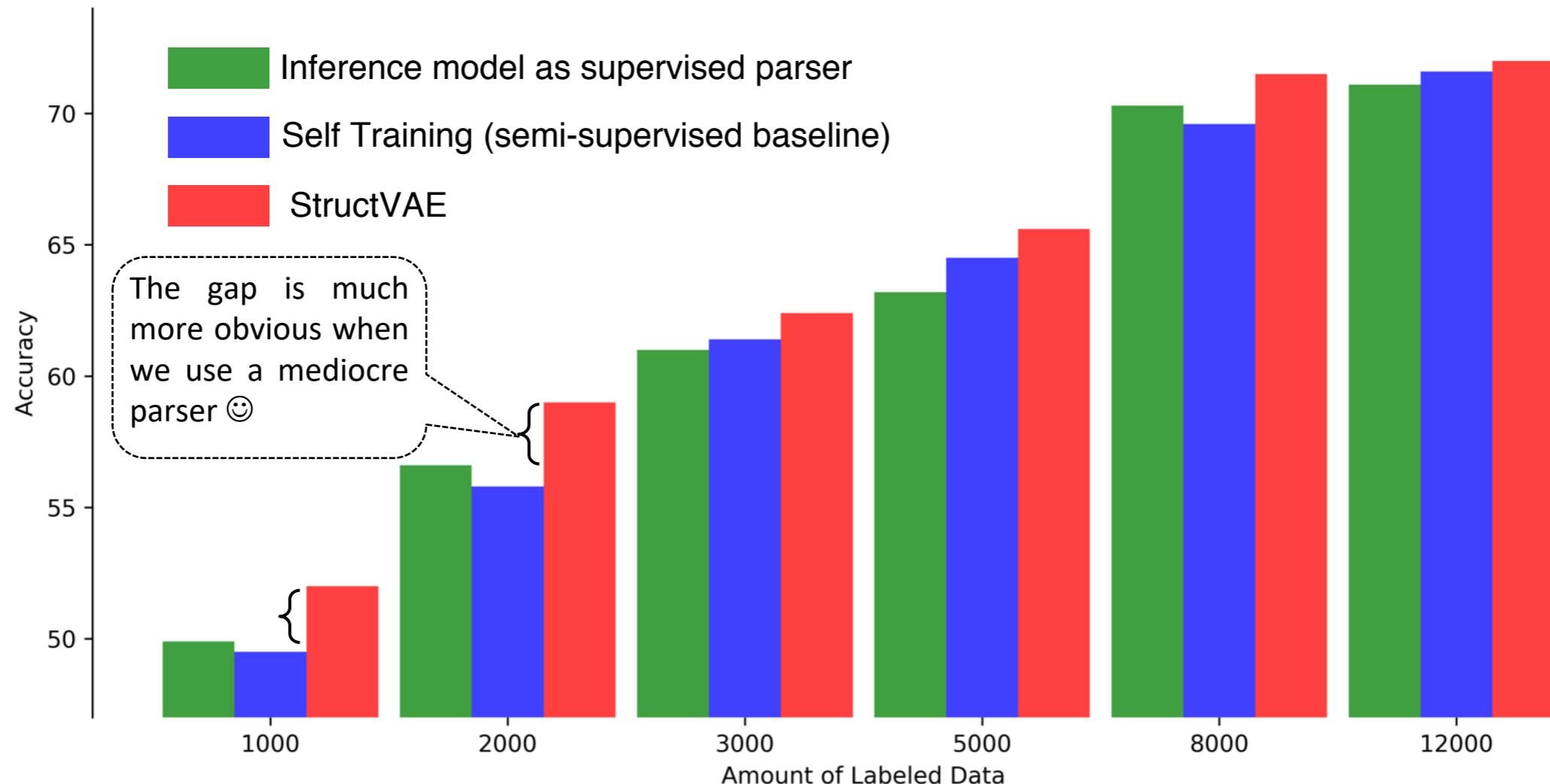


[Yin and Neubig, 2017;
Rabinovich et al. 2017]

Research Questions

- **RQ1** Does StructVAE outperforms purely supervised semantic parsers with extra unlabeled data?
- **RQ2** Can we get some empirical evidence about why StructVAE works?

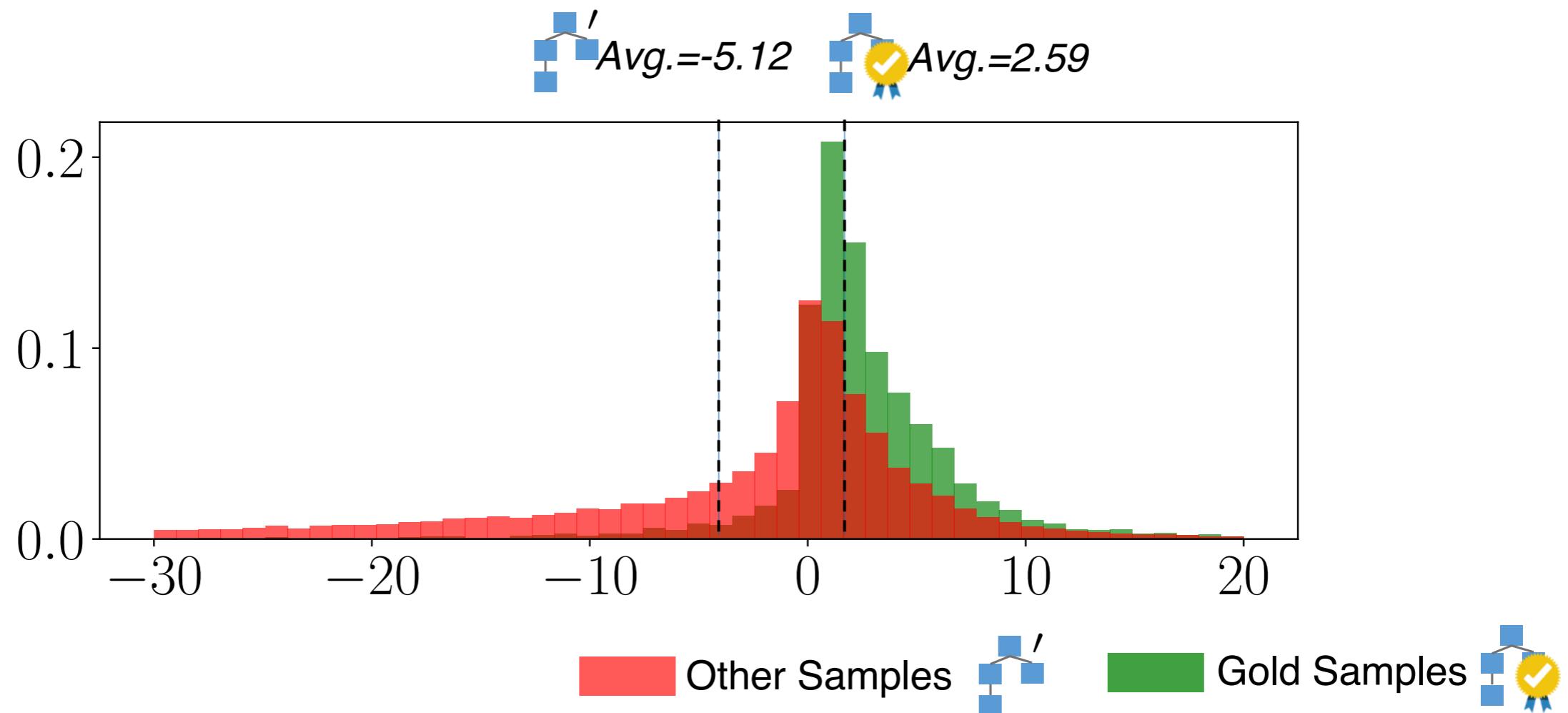
StructVAE vs. Baselines



all available training utterances as unlabeled data

Why does StructVAE Work?

- For each unlabeled utterance , compute the learning signal  for gold samples and other (imperfect) samples



Learning Signal

 Join p and cmd into a file path, substitute it for f

	Parser Score $q_\phi(\text{join} \mid \text{path})$	Prior $p(\text{join})$	Reconstruction Score $p_\theta(\text{join} \mid \text{path})$	Learning Signal
 <code>f = os.path.join(p, cmd)</code>	-1.00	-24.33	-2.00	9.14
 <code>p = path.join(p, cmd)</code>	-8.12	-27.89	-20.96	-9.47

 Split string pks by ',', substitute the result for `primary_keys`

	Parser Score $q_\phi(\text{split} \mid \text{pk})$	Prior $p(\text{split})$	Reconstruction Score $p_\theta(\text{split} \mid \text{pk})$	Learning Signal
 <code>primary_keys = pk.split(',')</code>	-2.38	-10.24	-11.39	2.05
 <code>primary_keys = pk.split + ','</code>	-1.83	-20.41	-14.87	-2.60

Retrieval-based Neural Code Generation

(EMNLP 2018)

Joint Work w/
Shirley Hayati, Raphaël Olivier, Pravalika Avvaru,
Pengcheng Yin, Anthony Tomasic

The Stack Overflow Cycle

Formulate the Idea

sort my_list in descending order



Search the
Web

python sort list in descending order



Browse thru.
results

This will give you a sorted version of the array.

167

`sorted(timestamp, reverse=True)`

If you want to sort in-place:

`timestamp.sort(reverse=True)`

share improve this answer

edited Nov 15 '10 at 10:47

answered Nov 15 '10 at 10:42

Marcelo Cantos
124k ● 23 ● 243 ● 301



Modify the
result

`sorted(my_list, reverse=True)`

Can we do the same thing in code generation models?!

Reminder: Syntax-based Generation

Input: params is an empty list

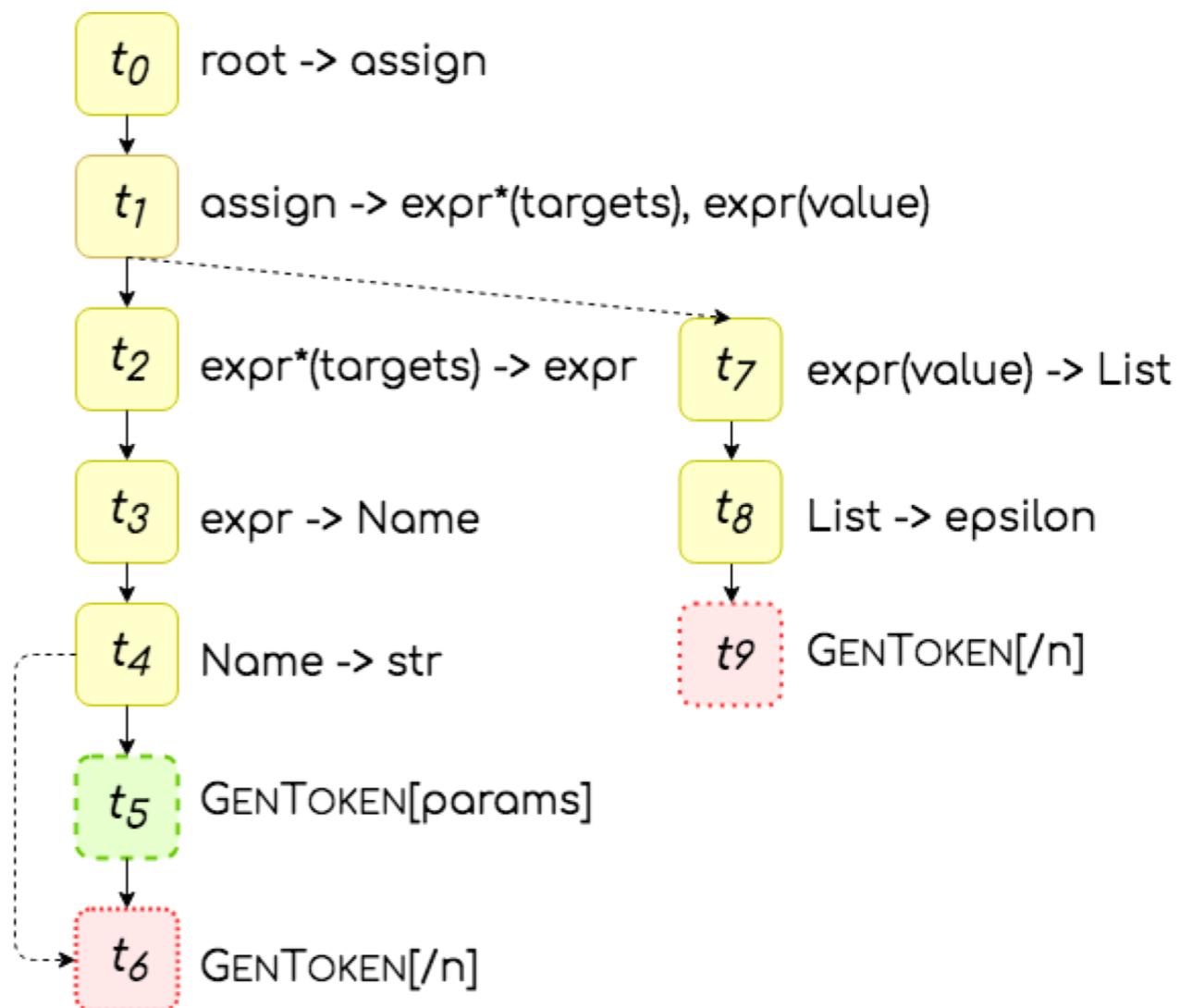
Output: params = []

Neural Model: bidirectional Encoder-Decoder with Action Embedding, Context Vector, Parent Feeding, Copying Mechanism

Actions:

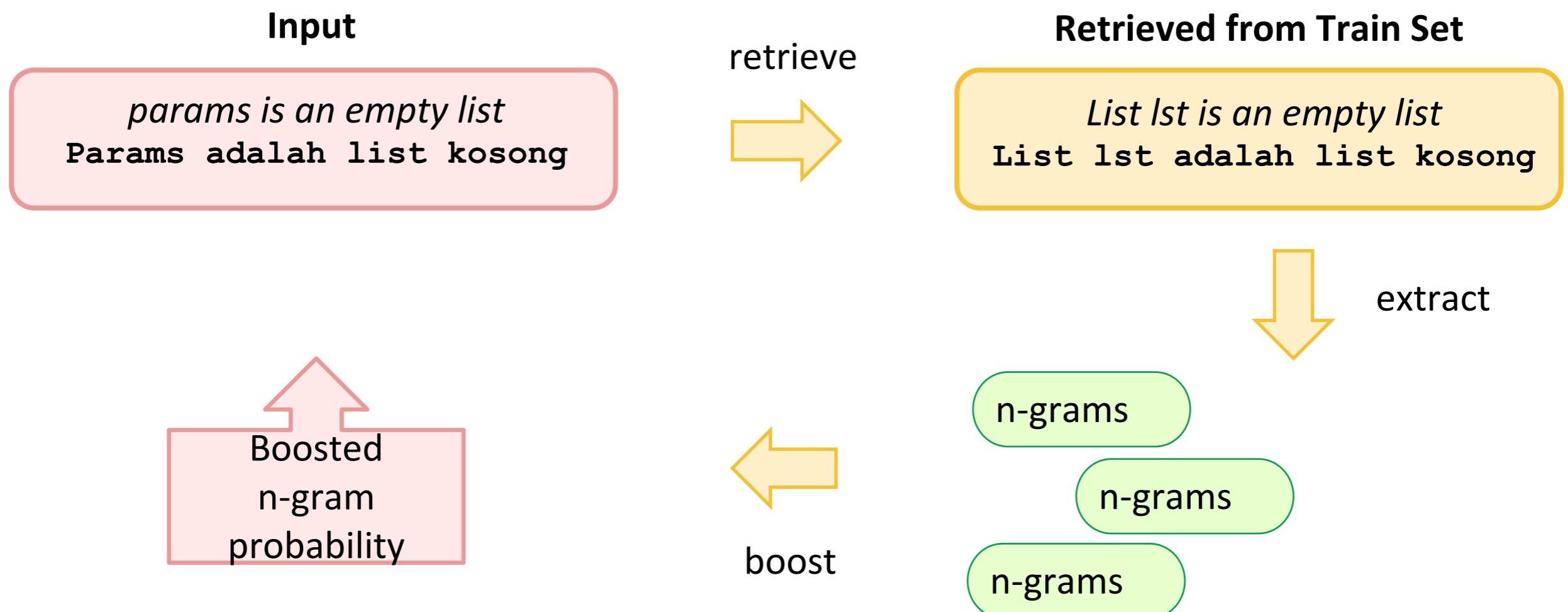
- Apply Rule
- Generate Token
- Generate Token with Copy

Action Tree:

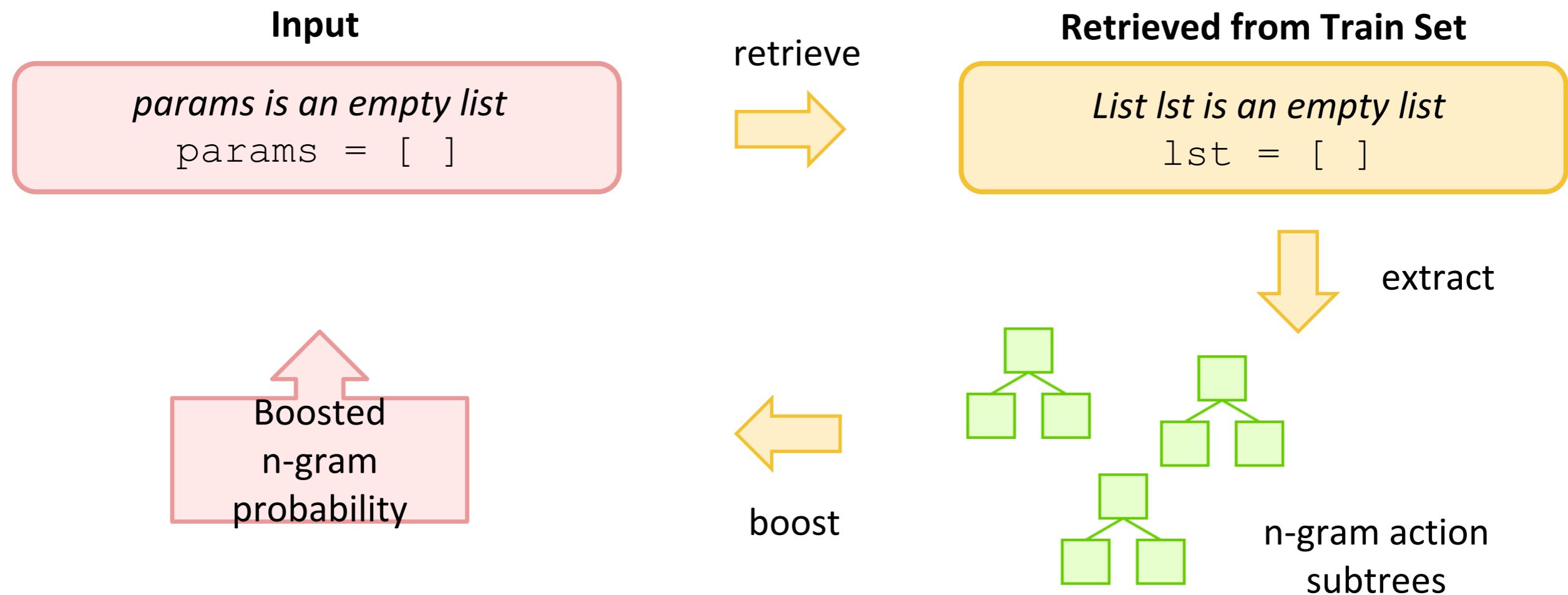


Neural Machine Translation + Retrieval

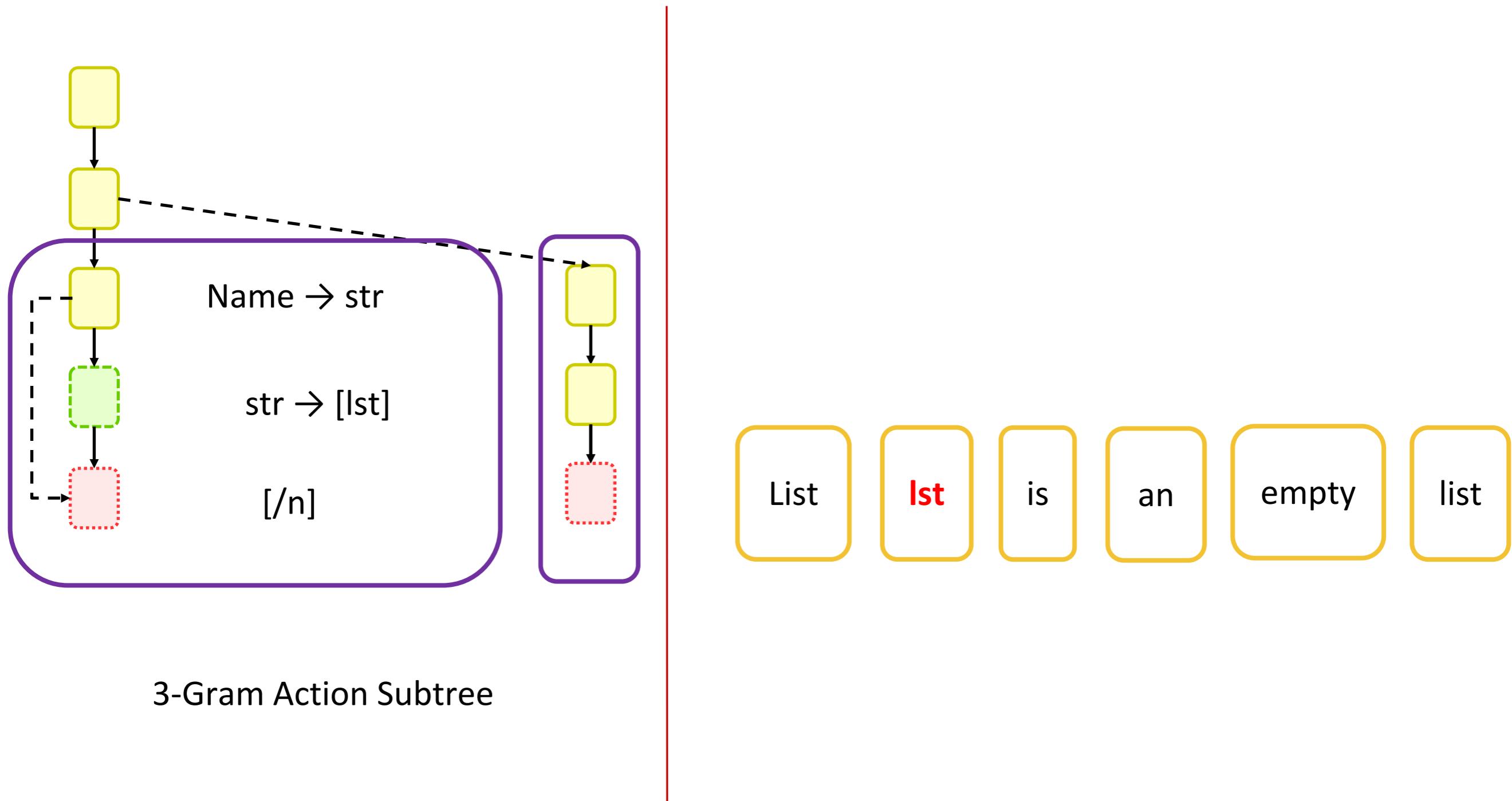
[Gu+2018, Zhang+2018]



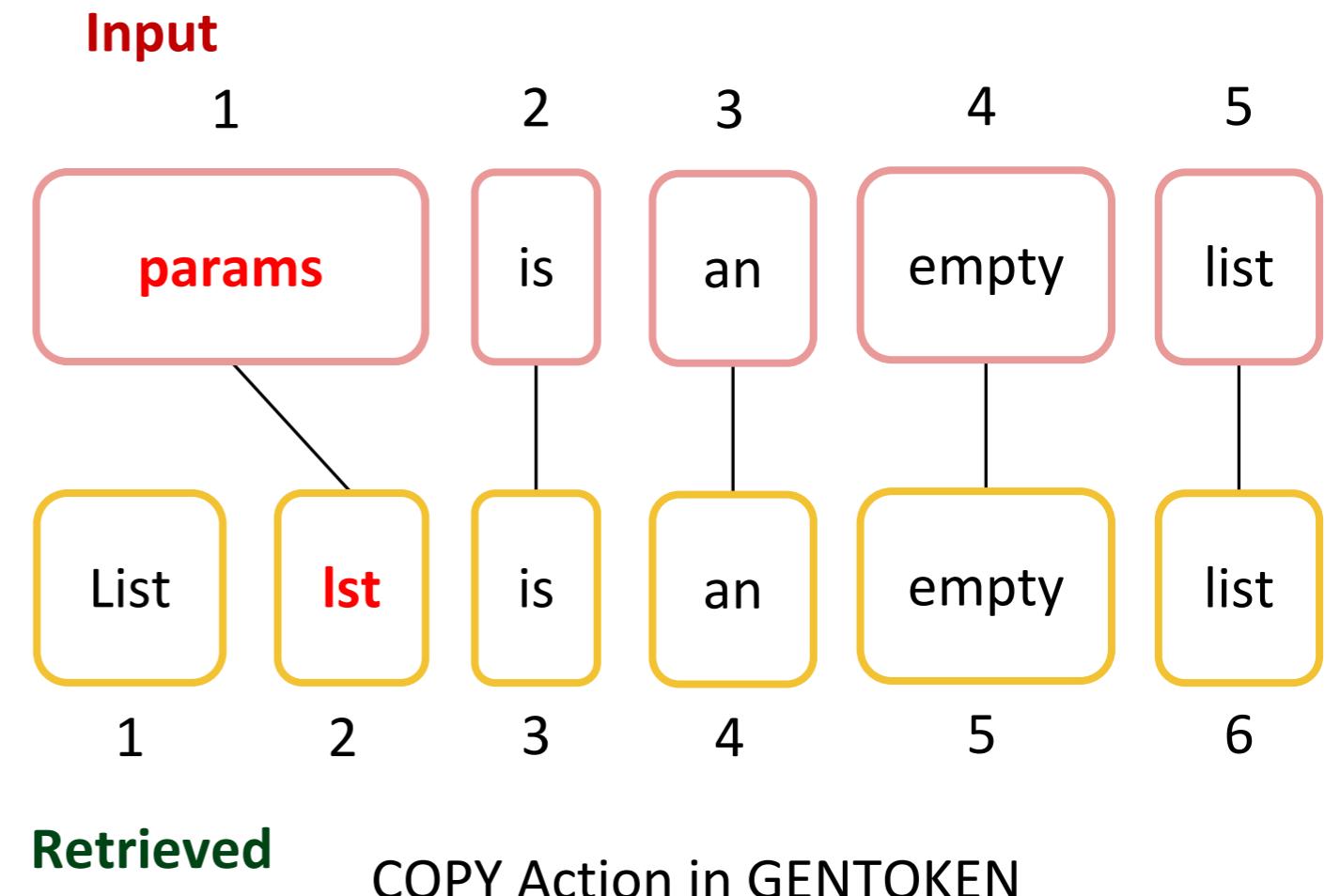
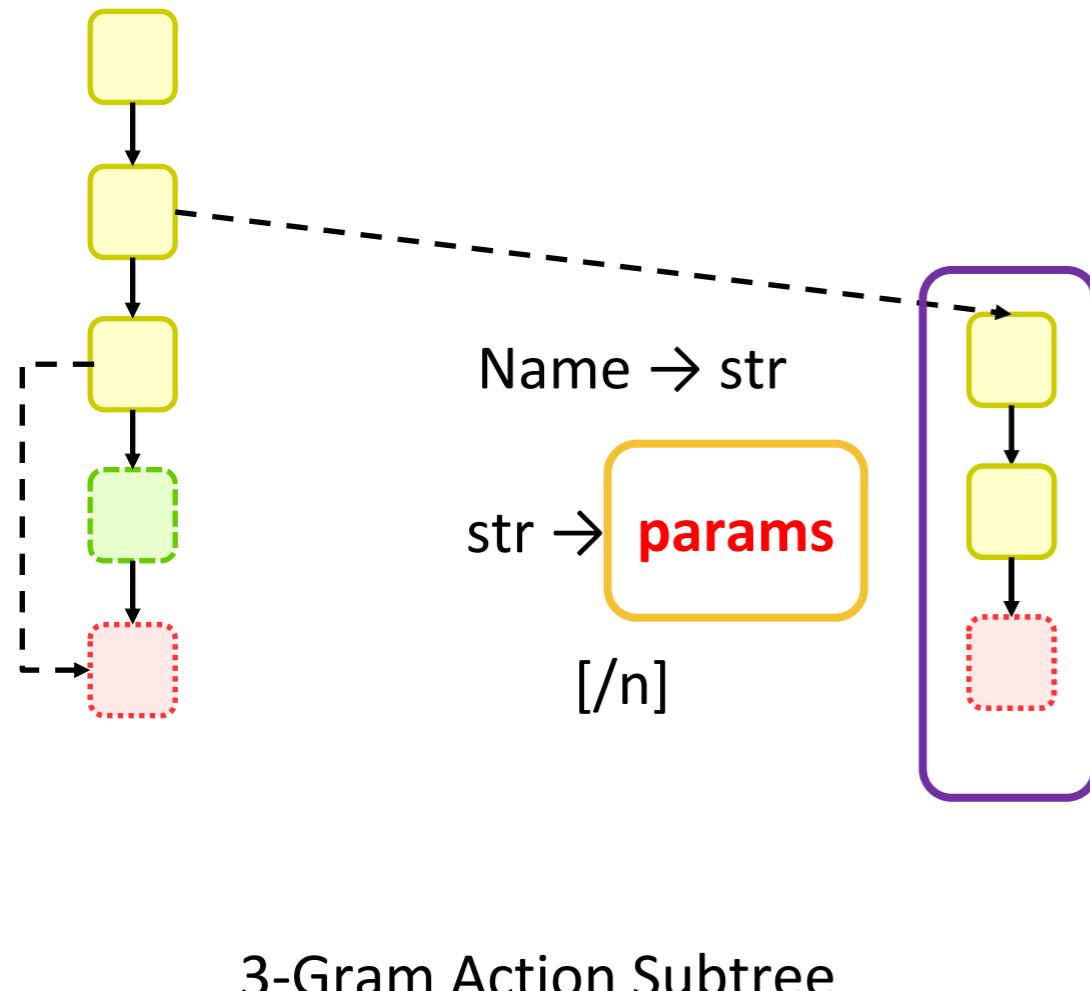
ReCode: Neural Code Retrieval + Generation



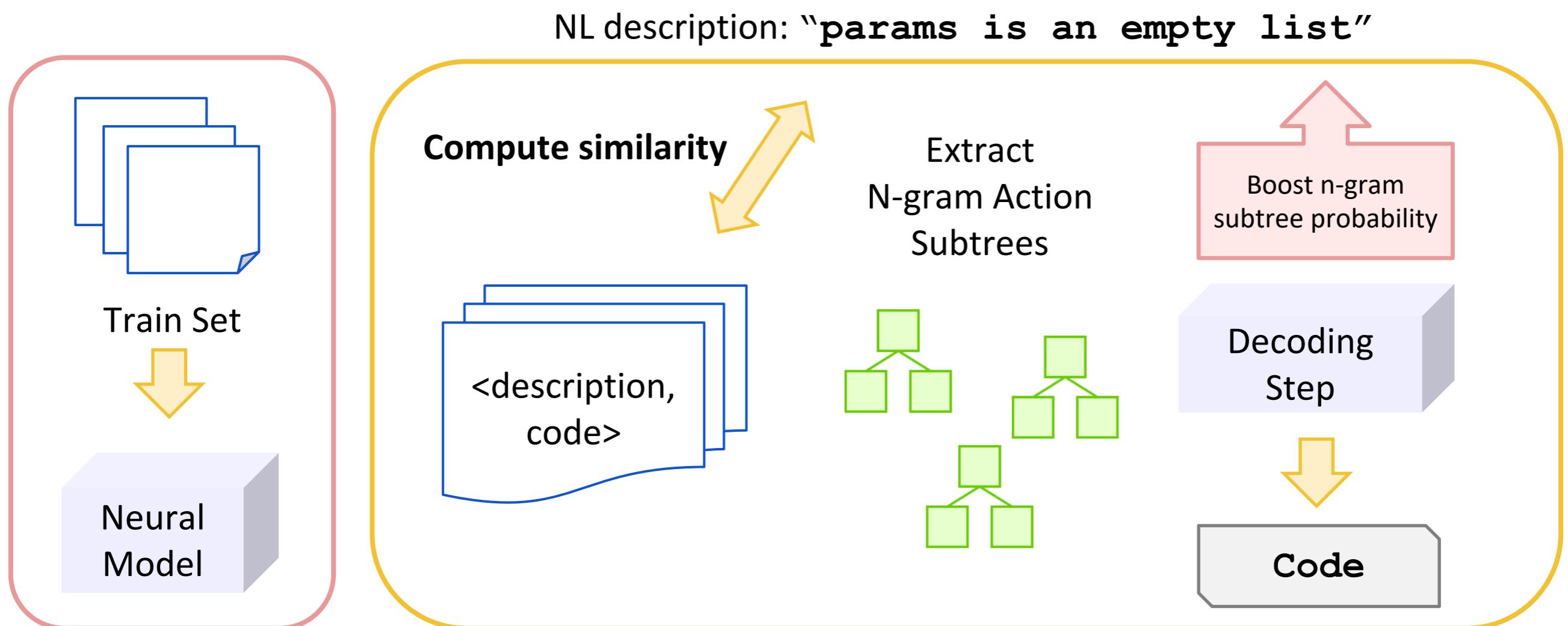
N-gram Action Subtrees



N-gram Action Subtrees w/ Copying

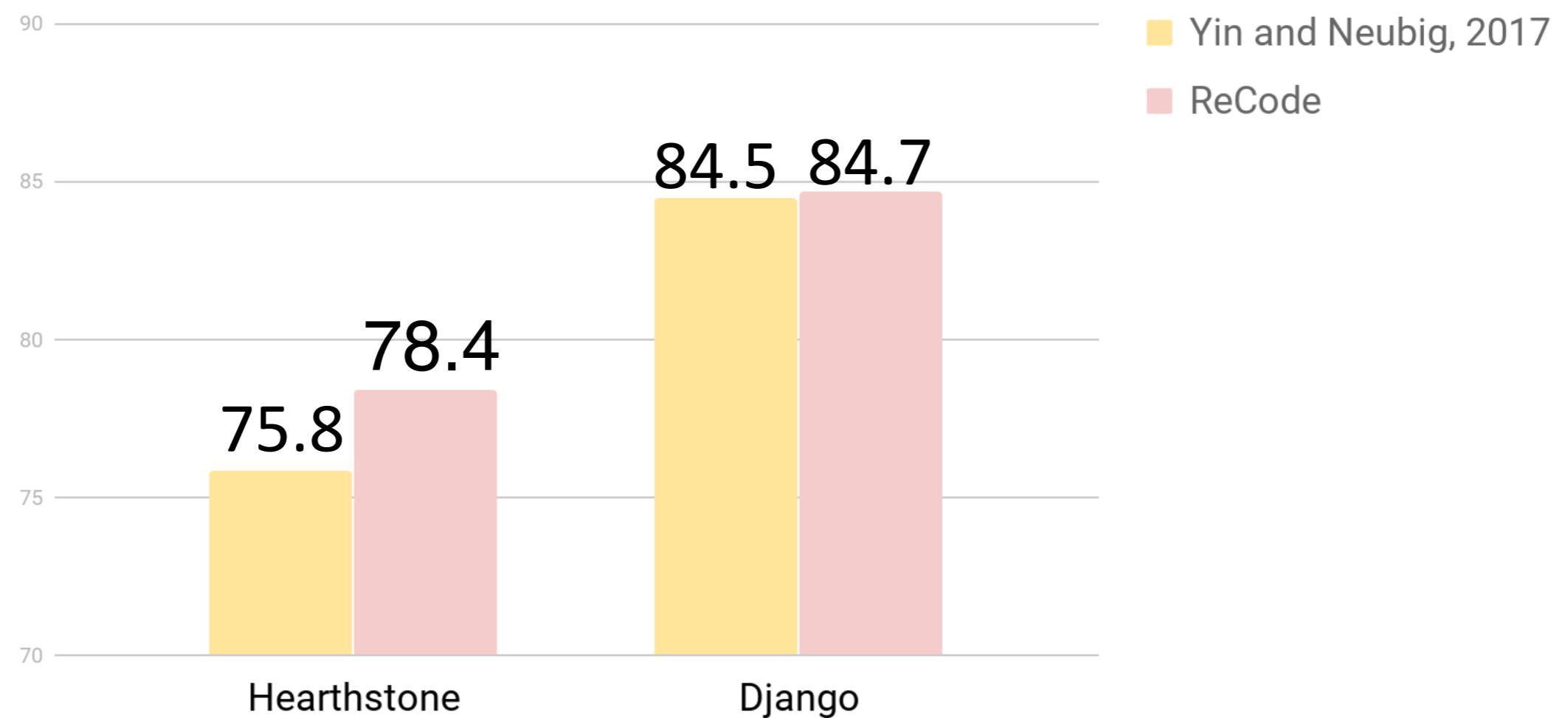


ReCode Pipeline



Results

All improvements are statistically significant with $p < 0.001$



Conclusion

Conclusion

- Data-driven language → code within reach!
- Modeling structure of the PL is important and helpful
- Data is difficult, but we're making progress through mining
- Semi-supervised learning and retrieval to take advantage of large datasets

Questions?