

Modelling Natural Language, Programs, and their Intersection

Graham Neubig



Carnegie Mellon University

Language Technologies Institute

Miltos Allamanis



Microsoft

NAACL 2018

Thanks to Collaborators!

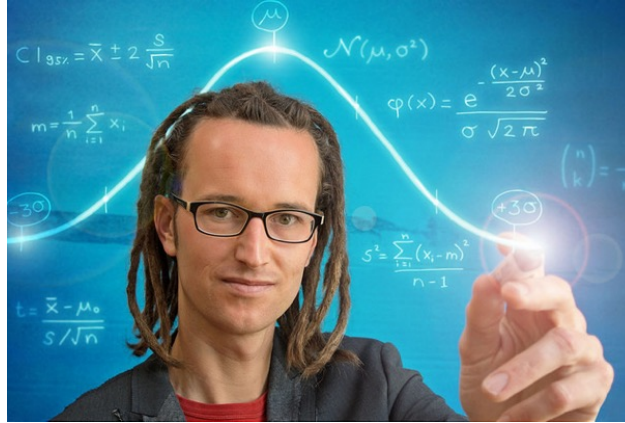
- Graham
 - **Pencheng Yin, Yusuke Oda**, Bowen Deng, Edgar Chen, Hiroyuki Fudaba, Koichi Akabe
 - **Bogdan Vasilescu, Hideaki Hata**, Sakriani Sakti, Tomoki Toda, Satoshi Nakamura
- Miltos
 - Charles Sutton, Marc Brockschmidt, Alex Gaunt

Who Programs?

Programmers



Data Scientists



Chemists, Biologists



Animators



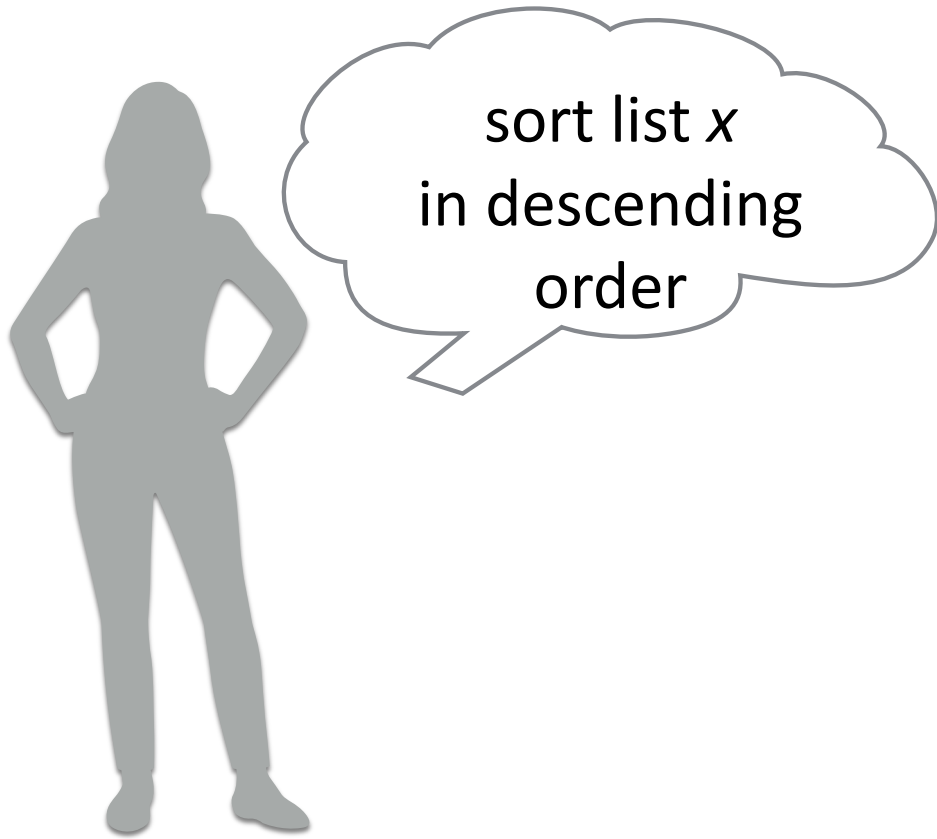
Psychologists



- Most don't want programming to be a large part of their job!

Photo Credits: Joonspoon, Jasper M, DarkoStojanovic, Notre Dame Univ

Coding =
Concept → Implementation



→ `x.sort(reverse=True)`

The Stack Overflow Cycle

Formulate the Idea

sort my_list in descending order



Search the Web

python sort list in descending order



stackoverflow



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

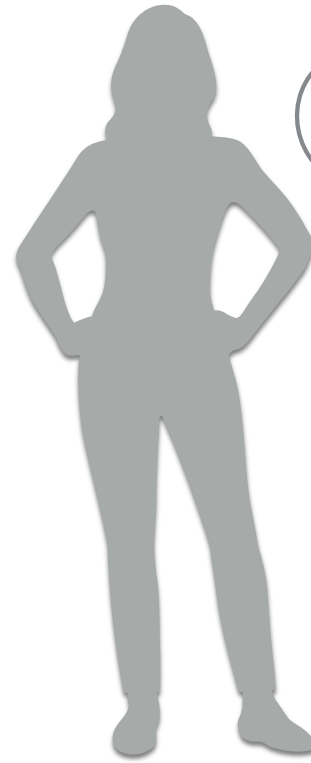


Modify the result

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

Program Understanding =
Implementation → Concept

`x.sort(reverse=True)` →

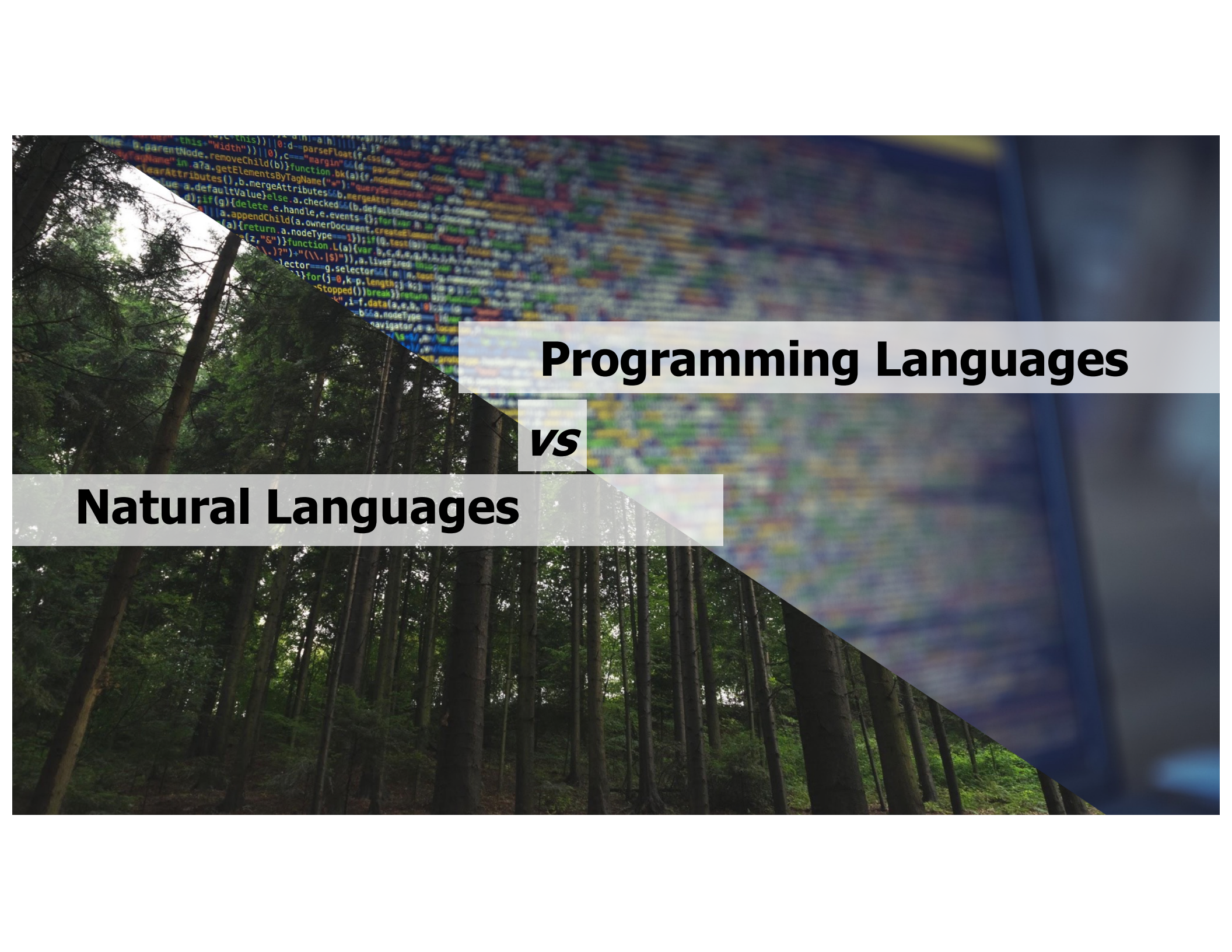


Today's Agenda:

Can Natural Language Help?

- Introduction (Here!)
- Natural language and programming language (15 minutes)
- Curated data sets (10 minutes)
- Methods for mapping from code to natural language (40 minutes)

- Methods for mapping from language to code (45 minutes)
- Modeling natural language aspects of source code (20 minutes)
- Modeling communicative aspects of software projects (10 minutes)
- Conclusion (5 minutes)



Programming Languages

VS

Natural Languages

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: Summary in Allamanis et al. (2017)

Natural Language

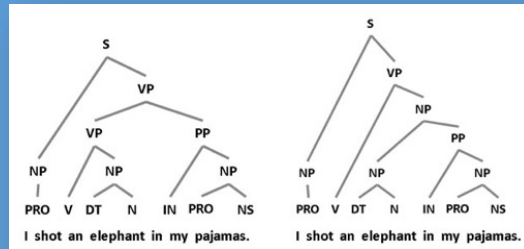
Code

Token

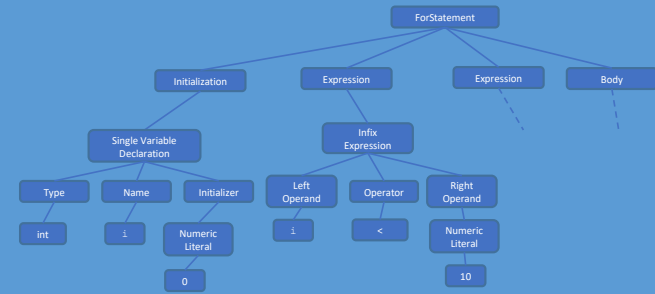
I shot an elephant in my pajamas.

```
for (int i = 0; i < 10; i++){  
    Console.WriteLine(i);  
}
```

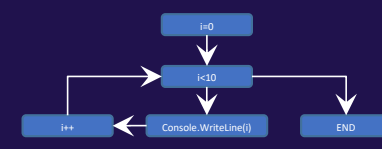
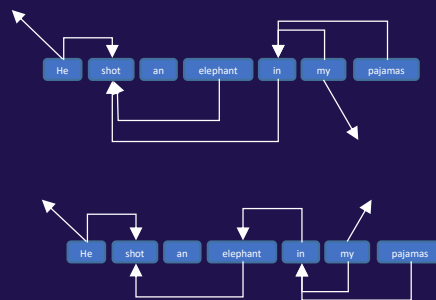
Syntax



(image from Daniel 2015)



Graph



+ Data Flow, Program Dependency Graph, ...



Executability

- Ambiguity
- Translation

Formality

- Reusability (but [Lopes+17] show a large amount of cloning)
- Explicit vs. implicit long-range dependencies
- “Bit rot”

In Code, What is the Unit?

```
class class1:
    def func1(t):
    def func2(t):
        my_list = range(1,t)
        my_val = 0
        for x in my_list:
            my_val += x * x
        return my_val
```

← Classes
[Moreno+ 2013]

} ← Functions/Methods
[Movshovitz-Attias+ 2013], others

← Single lines of code [Oda+ 2015]

← Single variables
[Sridhara+ 2011a, Allamanis+ 2015]

} ← Code blocks
[Sridhara+ 2011b, Wong+ 2013]



Design Implications for Code Models

- Rich, **known**, structure
- Can combine formal methods
 - Look at code as a mathematical object
- Requires explicit definitions of concepts

```
def quick_sort(items):  
    """ Implementation of quick sort """  
    if len(items) > 1:  
        pivot_index = len(items) / 2  
        smaller_items = []  
        larger_items = []  
  
        for i, val in enumerate(items):  
            if i != pivot_index:  
                if val < items[pivot_index]:  
                    smaller_items.append(val)  
                else:  
                    larger_items.append(val)  
            quick_sort(smaller_items)  
            quick_sort(larger_items)  
        items[:] = smaller_items + [items[pivot_index]] + larger_items
```

General-purpose
Language

Which states do not border Texas ?
answer(exclude(state(all), next_to(stateid('tx'))))

Domain-specific
Language

General-purpose Language vs. Domain-specific Language

General-Purpose

- Broad set of operations
- High expressivity
- Huge search space
- Available in large quantity
- Object oriented, procedural, or functional

Domain-Specific

- Limited, domain-tailored operations
- Reduced expressivity
- Smaller, tractable, search space
- Available in moderate quantity
- Usually functional

Where does Language Appear
in Programs/Coding?

In the Code Itself

Functions

Variables

with open(fname) as f:

content = f.readlines()

Comments

you may also want to remove whitespace characters like `\n` at the end of each line

content = [x.strip() for x in content]

- fMRI scans of skilled programmers show that when they reason about code, they use natural language processing parts of the brain!
[Floyd+17]

In the Documentation

`readlines(hint=-1)`

Read and return a list of lines from the stream. *hint* can be specified to control the number of lines read: no more lines will be read if the total size (in bytes/characters) of all lines so far exceeds *hint*.

Note that it's already possible to iterate on file objects using `for line in file: ...` without calling `file.readlines()`.

In Question Answer Forums

Question

In Python, how do I read a file line-by-line into a list?

▲ How do I read every line of a file in Python and store each line as an element in a list?

1630 ▼ I want to read the file line by line and append each line to the end of the list.

python string file readlines

★ share improve this question

406

edited Apr 13 at 13:32

 Knickerless-Noggins
4,294 ● 3 ● 38 ● 51

asked Jul 18 '10 at 22:25

 Julie Raswick
8,193 ● 3 ● 10 ● 3

Answer
Snippet

▲

```
with open(fname) as f:  
    content = f.readlines()  
# you may also want to remove whitespace characters like '\n' at the end of each line  
content = [x.strip() for x in content]
```


 ▼

1570


I'm guessing that you meant `list` and not array.

share improve this answer

edited Jan 11 '17 at 14:24

 holzkohlengrill
334 ● 7 ● 16

answered Jul 18 '10 at 22:28

 SilentGhost
172k ● 43 ● 249 ● 259

Comments

75 Don't use `file.readlines()` in a `for` -loop, a file object itself is enough: `lines = [line.rstrip('\n') for line in file]` – jfs Jan 14 '15 at 10:52

In Developer Discussions

Slicing improvements #1363

 Merged neubig merged 5 commits into master from range-improvements just now


 Conversation 5  Commits 5  Checks 0  Files changed 4





msperber commented on Apr 27

Collaborator +   

- In Python, allows using numpy-like syntax to select ranges and strides across multiple axes, e.g. `my_expr[:,1:4,::2]` will now work. Under the hood, this uses the `strided_select` operator.
- as a special case, when using this to select a range of batch elements, and this range starts with the first batch element, the operation is marked as in-placed. Ideally, this should also work when the range does not start at the first batch element, but that would require some adjusting some pointers so I'll leave it for a future PR.

 msperber added some commits on Apr 26

-  better support for numpy-like slicing @cddccf
-  make `strided_select` in-place if selecting a range of the first n batches ✖ 77dfc2d


 msperber referenced this pull request in `neulab/xnmt` on Apr 27

[WIP] Drop masked decoder states #370

 Open




neubig commented on Apr 27

Member +  ...

Awesome, this will be very useful! Could you add a test for this?

Also maybe check this:

<https://app.codacy.com/app/xunzhang/dynet/pullRequest?prid=1590987>

 msperber added some commits 29 days ago

-  fix issue c8ba14f

Data Sources

Data is Essential!

- We are building data-driven models
- Or we are doing data-driven exploratory research
- We need data with natural language **and** code, and quality and quantity is essential
- How do we create data, and what language is it in?

Natural Language Commands + Implementations

- The most straightforward variety of data, useful in automatic code generation/commenting
- Excellent survey by [Lin+ 2018]

Dataset	PL	# pairs	# words	# tokens	Avg. # w. in nl	Avg. # t. in code	NL collection	Code collection	Semantic alignment	Introduced by
IFTTT	DSL	86,960	–	–	7.0	21.8	scraped	scraped	Noisy	(Quirk et al., 2015)
C#2NL*	C#	66,015	24,857	91,156	12	38				(Iyer et al., 2016)
SQL2NL*	SQL	32,337	10,086	1,287	9	46				(Zhong et al., 2018)
RegexLib	Regex	3,619	13,491	179*	36.4	58.8*			(Ling et al., 2016)	
HeartStone	Python	665	–	–	7	352*	game card description	game card source code	Good*	(Ling et al., 2016)
MTG	Java	13,297	–	–	21	1,080*				
StaQC	Python	147,546	17,635	137,123	9	86	extracted using ML	extracted using ML	Noisy	(Yao et al., 2018)
	SQL	119,519	9,920	21,413	9	60				
NL2RX	Regex	10,000	560	45*†	10.6	26*	synthesized & paraphrased	synthesized	Very Good	(Locascio et al., 2016)
WikiSQL	SQL	80,654	–	–	–	–				(Zhong et al., 2017)
NLMAPS	DSL	2,380	1,014	–	10.9	16.0	synthesized given code	expert written	Very Good	(Haas and Riezler, 2016)
Jobs640*	DSL	640	391	58†	9.8	22.9	user written	expert written given NL		(Tang and Mooney, 2001)
GEO880	DSL	880	284	60†	7.6	19.1				(Zelle and Mooney, 1996)
Freebase917	DSL	917	–	–	–	–				(Cai and Yates, 2013)
ATIS*	DSL	5,410	936	176†	11.1	28.1				(Dahl et al., 1994)
WebQSP	DSL	4,737	–	–	–	–				(Yih et al., 2016)
NL2RX-KB13	Regex	824	715	85*†	7.1	19.0*	search log	expert written given code		(Kushman and Barzilay, 2013)
Django*	Python	18,805	–	–	14.3	–	turker written			(Oda et al., 2015)
NL2Bash	Bash	9,305	7,790	6,234	11.7	7.7				Ours

Datasets: Domain Specific Languages

- NL interfaces to Databases: e.g. GeoQuery [e.g. Zelle+96]

```
answer(count(city(loc_2(countryid(usa))))))
```

How many cities are there in the US?

- Regular Expressions [Kushman+13]

Text Description	Regular Expression
three letter word starting with 'X'	<code>\bX[A-Za-z]{2}\b</code>

- If This Then That [Quirk+15]



<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

Datasets: General Language, Specific Domain

- Django (Python)
[Oda+15]

Intent *call the function `_generator`, join the result into a string, return the result*

Target `return ''.join(_generator())`

- HearthStone (Python),
Magic (Java) [Ling+16]



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()
```

Datasets: General Domain

- NL2Bash (Bash) [Lin+18]

Natural Language	Bash Command(s)
<i>find .java files in the current directory tree that contain the pattern 'TODO' and print their names</i>	<pre>grep -l "TODO" *.java find . -name "*.java" -exec grep -il "TODO" {} \; find . -name "*.java" xargs -I {} grep -l "TODO" {}</pre>
<i>display the 5 largest files in the current directory and its sub-directories</i>	<pre>find . -type f sort -nk 5,5 tail -5 du -a . sort -rh head -n5 find . -type f -printf '%s %p\n' sort -rn head -n5</pre>
<i>search for all jpg images on the system and archive them to tar ball "images.tar"</i>	<pre>tar -cvf images.tar \$(find / -type f -name *.jpg) tar -rvf images.tar \$(find / -type f -name *.jpg) find / -type f -name "*.jpg" -exec tar -cvf images.tar {} \;</pre>

- Conala (Python) [Yin+18]

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))` ✓

Automatic Mining [Yin+18, Yao+18]

- **Problem:** Stack Overflow is an attractive source of data, but very noisy
- **Solution:** Train a classifier to automatically identify which data is good
 - Hand-crafted features [Wong+13]
 - A neural model that calculates probability of code given NL, vice-versa [Yin+18]

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 it removes 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
```

Question

▲
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]  
>>> t  
[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

Answers

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

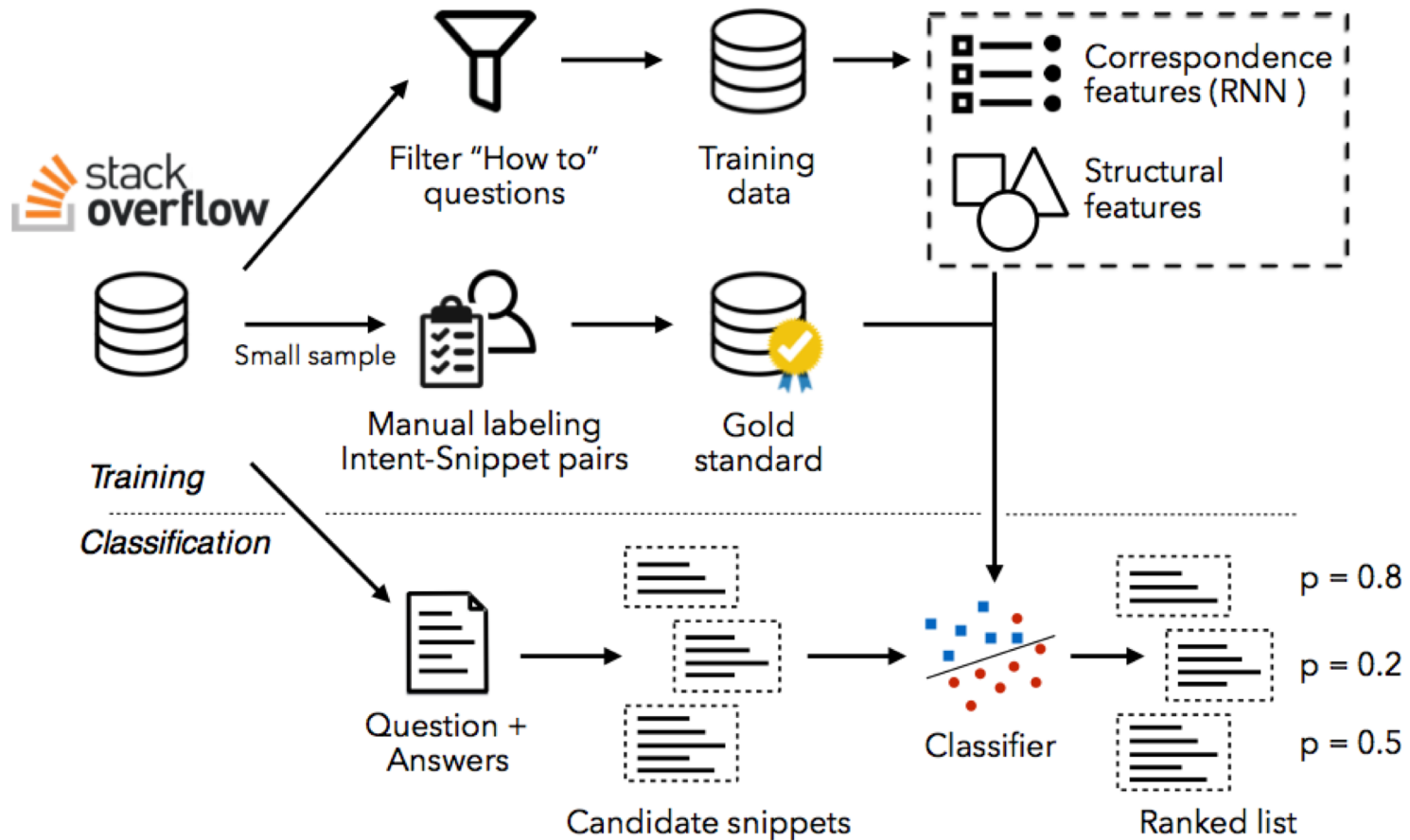
FWIW, the new (v2.7) Python way for removing duplicates from an iterable while keeping it in the original order is:

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

← Context 2

← Snippet 2

Mining Method



CoNaLa: The Code/Natural Language Challenge

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

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')",

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",

question_id: 22240602

Other Types of Data: Doc Strings

[Movshovitz-Attias+13, Richardson+17, Miceli Barone+17]

- Gives information about what functions do

```
1. Java 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)
```

- Compared to QA sites, much more abstract, less tied to implementation
- But give hints about how to use APIs if we want!

Other Types of Data: Comments

[Wong+15]

- Inline comments in code can also be informative

```
1 | try {
2 |     if (new File(jarEntryURL.toURI()).canWrite()) {
3 |         connection.setUseCaches(false);
4 |     }
5 | } catch (URISyntaxException ex) {
6 |     // Wrap the exception and re-throw
7 |     IOException ex2 = new IOException();
8 |     ex2.initCause(ex);
9 |     throw ex2;
10| }
```

- **Problem:** comments often don't describe *what* is being done, but rather *why*

Other Types of Data: Diff Messages

[Loyola+17, Jiang+17]

- Version control systems keep track of changes and textual descriptions
- Possible source of data to learn how to describe changes made to code

Diff:

```
--- a/core/.../CursorToBulkCursorAdaptor.java
+++ b/core/.../CursorToBulkCursorAdaptor.java
@@ -143,8 +143,7 @@ public final class
CursorToBulkCursorAdaptor ...
    public void close() {
        maybeUnregisterObserverProxy();
-       mCursor.deactivate();
-
+       mCursor.close();
    }
    public int requery(IContentObserver observer, ...
```

Reference Message:

“Call close () instead of deactivate () in
CursorToBulkCursorAdaptor . close () ”

Program Understanding: Mapping from Code to Natural Language

From Code to Natural Language

```
if (DEBUG) assert n >= 0;
int r = 0;
while (n >= MIN_MERGE) {
    r |= (n & 1);
    n >>= 1;
}
return n + r;
```



Some natural language
description/summary

- Oda et al. "*Learning to generate pseudo-code from source code using statistical machine translation*" 2015
- Allamanis et al. "*A convolutional attention network for extreme summarization of source code*" 2016
- Iyer et al. "*Summarizing source code using a neural attention model*" 2016
- Barone et al. "*A parallel corpus of Python functions and documentation strings for automated code documentation and code generation*" 2017
- And many more...

Applications

 Explaining Code

 Code Search

 Accessibility

 Documentation

 Linking Code to NL Artifacts (Traceability)

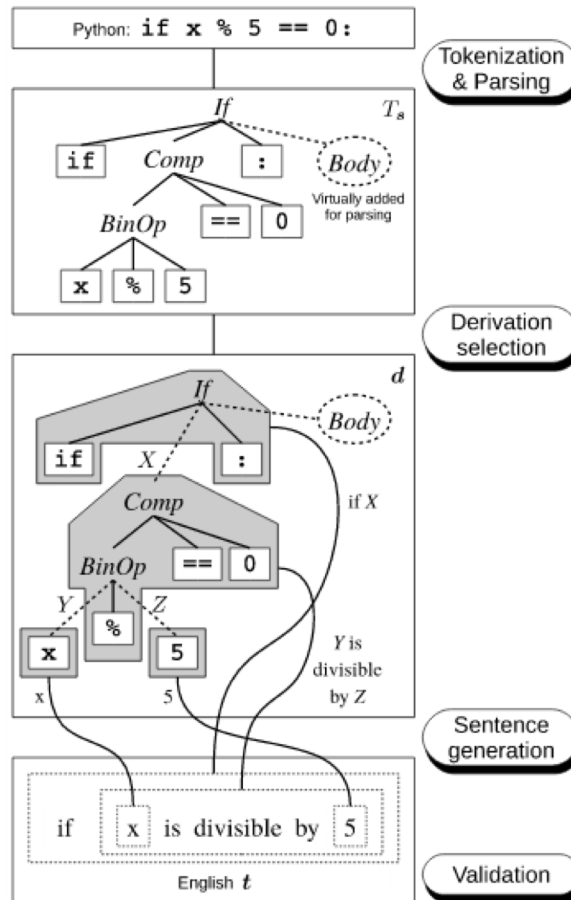
“Translating” Code

```
def fizzbuzz(n):                                # define the function fizzbuzz with an argument n.
    if not isinstance(n, int):                  # if n is not an integer value,
        raise TypeError('n is not an integer') # throw a TypeError exception with a message ...
    if n % 3 == 0:                              # if n is divisible by 3,
        return 'fizzbuzz' if n % 5 == 0 else 'fizz' # return 'fizzbuzz' if n is divisible by 5, or 'fizz' if not.
    elif n % 5 == 0:                            # if not, and n is divisible by 5,
        return 'buzz'                          # return the string 'buzz'.
    else:                                       # otherwise,
        return str(n)                          # return the string representation of n.
```

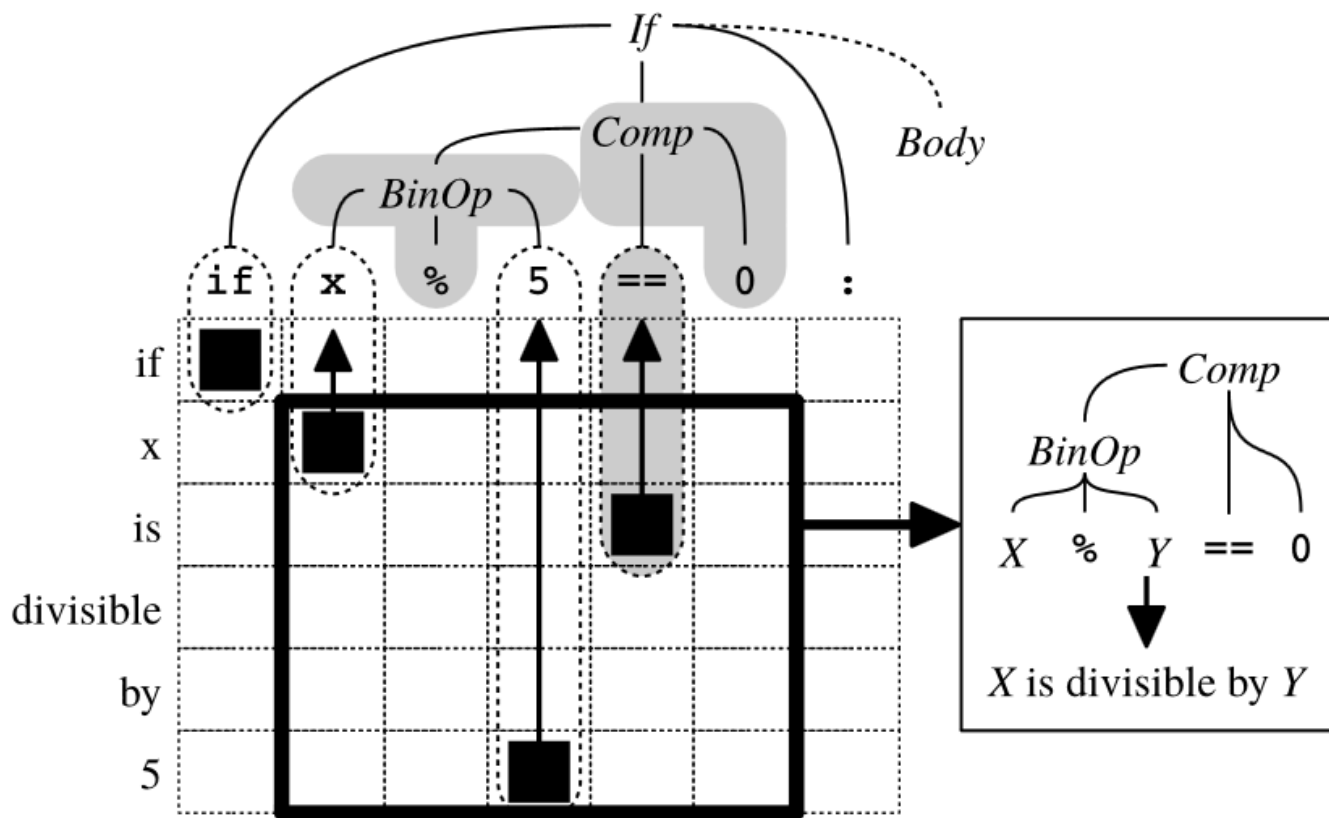
Source code (Python)

Pseudo-code (English)

Oda *et al.* “Learning to Generate Pseudo-code from Source Code using Statistical Machine Translation” ASE 2015



“Translating” Code



Python	<code>for node in graph.leaf_nodes (app_name) :</code>
PBMT	for node in graph.leaf_nodes with an argument app_name,
Raw-T2SMT	for every node in, return value is the return value of the graph.leaf_nodes app_name,
Head-T2SMT	for every node in graph.leaf_nodes app_name,
Reduced-T2SMT	for every node in return value of the graph.leaf_nodes with an argument app_name,
Python	<code>if self._isdst (dt) :</code>
PBMT	if self.call the method _isdst with 2 arguments dt, if it evaluates to true,
Raw-T2SMT	self._isdst with an argument dt, if it evaluates to true,
Head-T2SMT	if self._isdst with an argument dt, return the result.
Reduced-T2SMT	call the method self._isdst with an argument dt, if it evaluates to true,

Code Summarization to Natural Language

1. Source Code (C#):

```
public int TextWidth(string text) {  
    TextBlock t = new TextBlock();  
    t.Text = text;  
    return  
        (int)Math.Ceiling(t.ActualWidth);  
}
```

Descriptions:

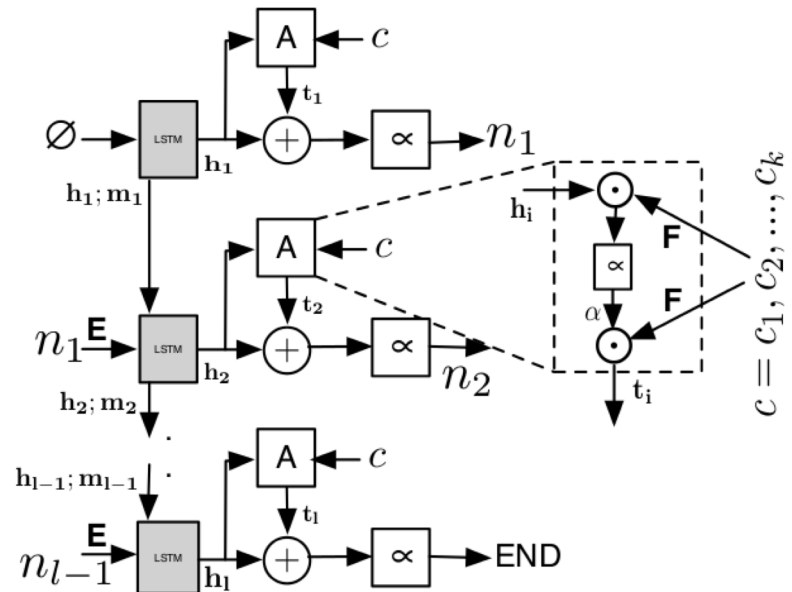
- Get rendered width of string rounded up to the nearest integer
- Compute the actual textwidth inside a textblock

2. Source Code (C#):

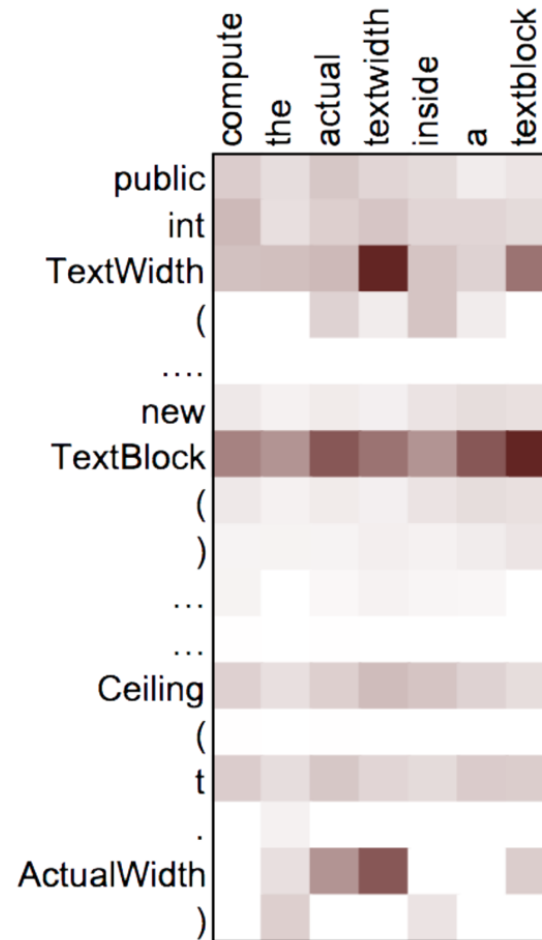
```
var input = "Hello";  
var regex = new Regex("World");  
return !regex.IsMatch(input);
```

Descriptions:

- Return if the input doesn't contain a particular word in it
- Lookup a substring in a string using regex



Code Summarization to Natural Language

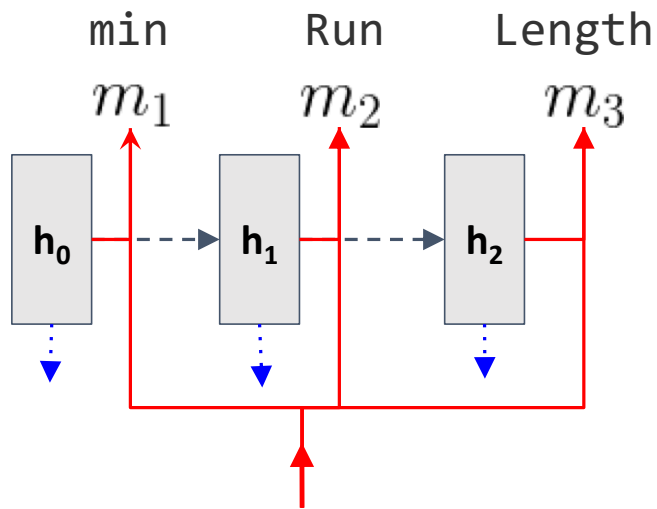


Iyer *et al.* "Summarizing Source Code using a Neural Attention Model" 2016

Predicting Method Names (\approx Summarization)

```
1 private void ██████████ () {  
2     String vertexShader = "literal_1";  
3     String fragmentShader = "literal_2";  
4     shader = new ShaderProgram(vertexShader,  
5         fragmentShader);  
6     if(shader.isCompiled() == false)  
7         throw new IllegalArgumentException(  
8             "literal_3" + shader.getLog());  
9 }
```

(Subtoken) Summary



```
if (DEBUG) assert n >= 0;  
int r = 0;  
while (n >= MIN_MERGE) {  
    r |= (n & 1);  
    n >>= 1;  
}  
return n + r;
```

Code

An RNN to predict summary subtokens

$$P(m_i | m_0 \dots m_{i-1}, \mathbf{code})$$

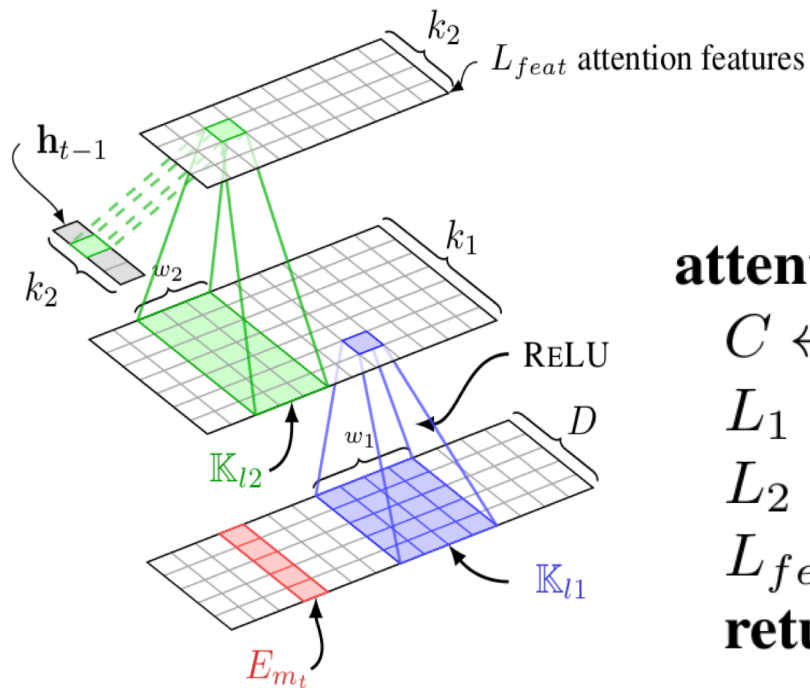
Context-Dependent
Convolutional Attention Features

Allamanis *et al.* "A Convolutional Attention Network for Extreme Summarization of Source Code" ICML 2016

Convolutional Neural Attention Models for Code Summaries

👁 **Attention Mechanisms**

- › Weight token embeddings
- › Direct copy of code token to the summary
 - Similar to pointer networks [*Vinyals et al, 2015*]
- › Choosing between mechanisms



attention_features (code tokens \mathbf{c} , context \mathbf{h}_{t-1})

$$C \leftarrow \text{LOOKUPANDPAD}(\mathbf{c}, E)$$

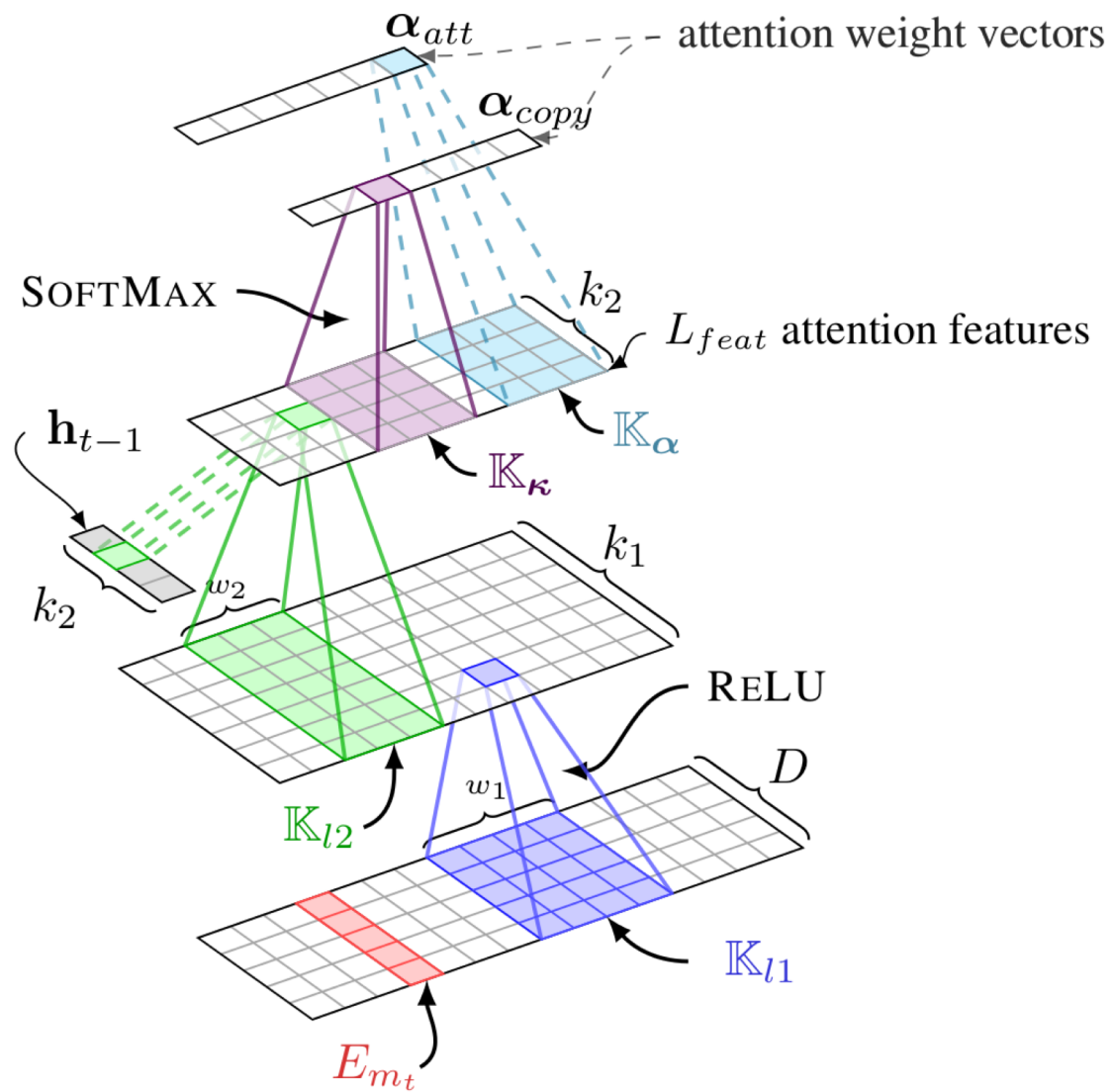
$$L_1 \leftarrow \text{RELU}(\text{CONV1D}(C, \mathbb{K}_{l_1}))$$

$$L_2 \leftarrow \text{CONV1D}(L_1, \mathbb{K}_{l_2}) \odot \mathbf{h}_{t-1}$$

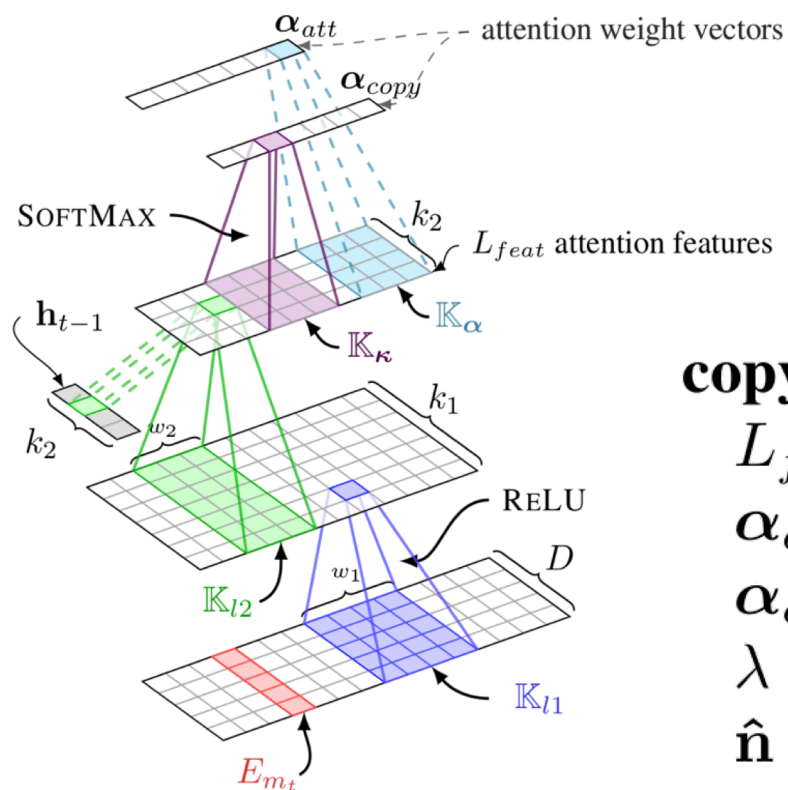
$$L_{feat} \leftarrow L_2 / \|L_2\|_2$$

return L_{feat}

**Extracting Attention
Features**



Computing Multiple Attention Weights



copy_attention (code \mathbf{c} , previous state \mathbf{h}_{t-1})

$$L_{feat} \leftarrow \mathbf{attention_features}(\mathbf{c}, \mathbf{h}_{t-1})$$

$$\alpha_{att} \leftarrow \mathbf{attention_weights}(L_{feat}, \mathbb{K}_{att})$$

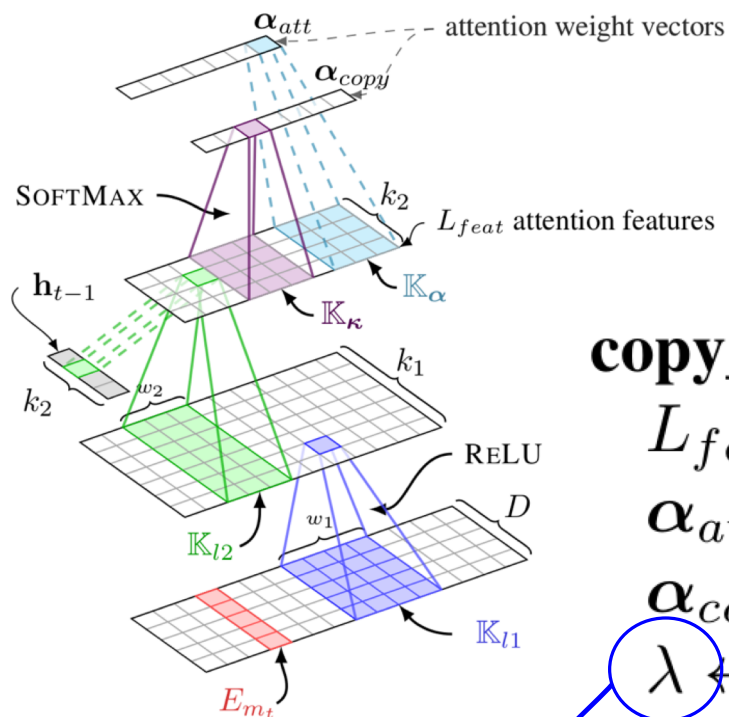
$$\alpha_{copy} \leftarrow \mathbf{attention_weights}(L_{feat}, \mathbb{K}_{copy})$$

$$\lambda \leftarrow \max(\sigma(\mathbf{CONV1D}(L_{feat}, \mathbb{K}_{\lambda})))$$

$$\hat{\mathbf{n}} \leftarrow \sum_i \alpha_i E_{c_i}$$

$$\mathbf{n} \leftarrow \mathbf{SOFTMAX}(E \hat{\mathbf{n}}^{\top} + \mathbf{b})$$

$$\mathbf{return} \lambda \mathbf{POS2VOC}(\boldsymbol{\kappa}, \mathbf{c}) + (1 - \lambda) \mathbf{TOMAP}(\mathbf{n}, V)$$



copy_attention (code \mathbf{c} , previous state \mathbf{h}_{t-1})

$L_{feat} \leftarrow \mathbf{attention_features}(\mathbf{c}, \mathbf{h}_{t-1})$

$\alpha_{att} \leftarrow \mathbf{attention_weights}(L_{feat}, \mathbb{K}_{att})$

$\alpha_{copy} \leftarrow \mathbf{attention_weights}(L_{feat}, \mathbb{K}_{copy})$

$\lambda \leftarrow \max(\sigma(\mathbf{CONV1D}(L_{feat}, \mathbb{K}_{\lambda})))$

$\hat{\mathbf{n}} \leftarrow \sum_i \alpha_i E_{c_i}$

$\mathbf{n} \leftarrow \mathbf{SOFTMAX}(E \hat{\mathbf{n}}^T + \mathbf{b})$

return $\lambda \mathbf{POS2VOC}(\kappa, \mathbf{c}) + (1 - \lambda) \mathbf{TOMAP}(\mathbf{n}, V)$

meta-attention
mechanism

Target Name

set m_1 use m_2 browser m_3 cache m_4

Target	Attention Vectors	λ
m_1 set	$\alpha_{att} =$ <code><s> { this . use Browser Cache = use <u>Browser</u> <u>Cache</u> ; } </s></code> $\alpha_{copy} =$ <code><s> { this . use Browser Cache = use <u>Browser</u> <u>Cache</u> ; } </s></code>	0.012
m_2 use	$\alpha_{att} =$ <code><s> { this . use <u>Browser</u> <u>Cache</u> = use <u>Browser</u> <u>Cache</u> ; } </s></code> $\alpha_{copy} =$ <code><s> { this . use <u>Browser</u> <u>Cache</u> = use <u>Browser</u> <u>Cache</u> ; } </s></code>	0.974
m_3 browser	$\alpha_{att} =$ <code><s> { this . use <u>Browser</u> <u>Cache</u> = use <u>Browser</u> <u>Cache</u> ; } </s></code> $\alpha_{copy} =$ <code><s> { this . use Browser <u>Cache</u> = use <u>Browser</u> <u>Cache</u> ; } </s></code>	0.969
m_4 cache	$\alpha_{att} =$ <code><s> { this . use <u>Browser</u> <u>Cache</u> = use <u>Browser</u> <u>Cache</u> ; } </s></code> $\alpha_{copy} =$ <code><s> { this . use <u>Browser</u> Cache = use <u>Browser</u> <u>Cache</u> ; } </s></code>	0.583
m_5 END	$\alpha_{att} =$ <code><s> { this . use <u>Browser</u> <u>Cache</u> = use <u>Browser</u> <u>Cache</u> ; } </s></code> $\alpha_{copy} =$ <code><s> { this . use <u>Browser</u> Cache = use <u>Browser</u> <u>Cache</u> ; } </s></code>	0.066

Attention Visualization

Data and Visualizations:

<http://groups.inf.ed.ac.uk/cup/codeattention/>

```
void reverseRange(Object[] a, int lo, int hi)
```

```
hi--;  
while (lo < hi) {  
    Object t = a[lo];  
    a[lo++] = a[hi];  
    a[hi--] = t;  
}
```

Predictions

- reverse, range (22.2%)
- reverse (13.0%)
- reverse, lo (4.1%)
- reverse, hi (3.2%)
- merge, range (2.0%)

float getAspectRatio()

```
return (height == 0) ?  
    Float.NaN : width / height;
```

Predictions

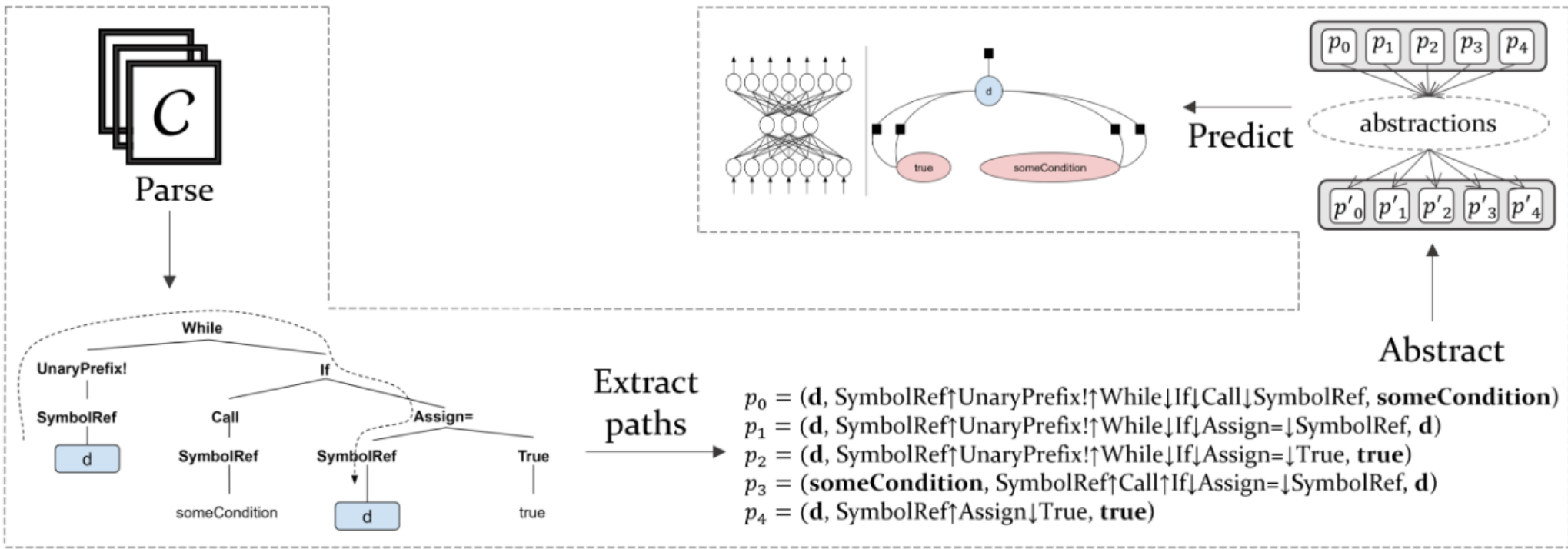
- get, UNK (9%)
- get, height (8.7%)
- get, width (6.5%)
- get (5.7%)
- get, size (4.2%)

`boolean shouldRender()`

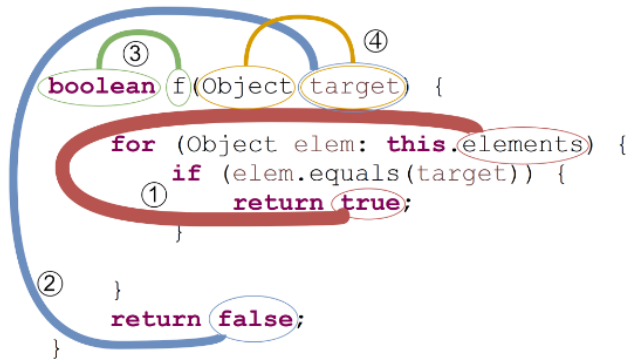
```
try {  
    return renderRequested || isContinuous;  
} finally {  
    renderRequested = false;  
}
```

Predictions

- `is, render` (27%)
- `is, continuous` (11%)
- `is, requested` (8%)
- `render, continuous` (7%)



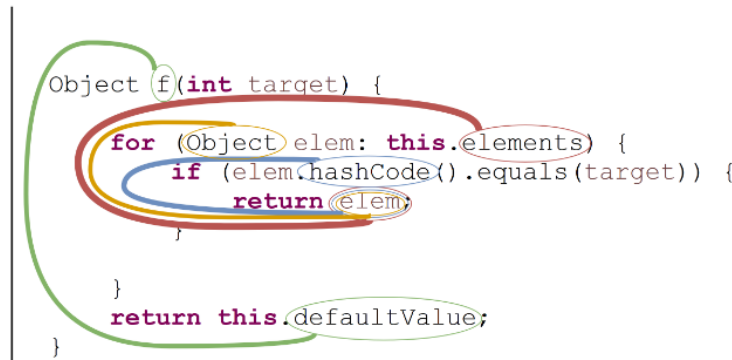
Alon *et al.* "A General Path-Based Representation for Predicting Program Properties" 2018



(a)

Predictions:

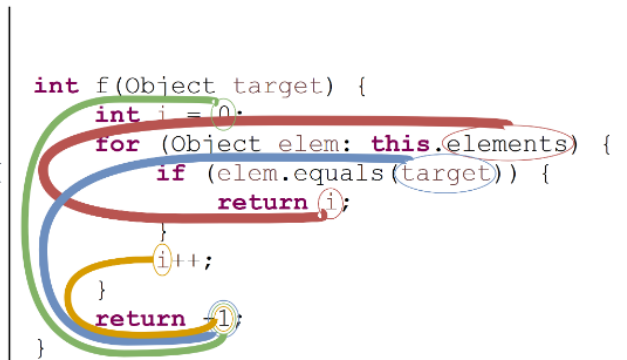
contains		90.93%
matches		3.54%
canHandle		1.15%
equals		0.87%
containsExact		0.77%



(b)

Predictions

get		31.09%
getProperty		20.25%
getValue		14.34%
getElement		14.00%
getObject		6.05%



(c)

Predictions

indexOf		96.65%
getIndex		2.24%
findIndex		0.33%
indexOfNull		0.20%
getInstructionIndex		0.13%

Diff:

```
--- a/core/.../CursorToBulkCursorAdaptor.java
+++ b/core/.../CursorToBulkCursorAdaptor.java
@@ -143,8 +143,7 @@ public final class
CursorToBulkCursorAdaptor ...
    public void close() {
        maybeUnregisterObserverProxy();
- mCursor.deactivate();
-
+ mCursor.close();
    }
    public int requery(IContentObserver observer, ...
```

Generated Message:

“CursorToBulkCursorAdapter . Close must call
mCursor . Close instead of mCursor . Deactivate . ”

Reference Message:

“Call close () instead of deactivate () in
CursorToBulkCursorAdaptor . close () ”

Program Generation: Mapping from Natural Language to Code

Machine Translation and Code Generation

- **Machine translation:** natural language to natural language

if the store is open tomorrow



もし お店 が 明日 空いている なら

- **Code generation:** natural language to programming language

if x is divisible by 5



if x % 5 == 0:

Features of Program Generation

if x is divisible by 5



if x % 5 == 0:

- Strong syntax for the target code
- Precise checking of the semantics of the target code
- Weaker connection between command and code
- But much potential for copying words

A Long History in Natural Language Programming

- Early methods: parse natural language specifications, then use rule-based transformations to derive program [e.g. Balzer+78]
- This, obviously, is hard because natural language is nuanced
 - Some even called it “foolish” [Dijkstra79]
- Similarly to machine translation: data driven methods help resolve this ambiguity and move closer to reality
 - Grammar-based models, mostly for DSLs [e.g. Wong+06]
 - Neural models [e.g. Ling+16]

A Few Distinctions

- Natural language programming vs. programming by demonstration

if x is divisible by 5



if $x \% 5 == 0$:

$x=3 \rightarrow \text{false}$ $x=15 \rightarrow \text{true}$



if $x \% 5 == 0$:

- Code generation vs. code search

Generate entirely new code

Retrieve existing code

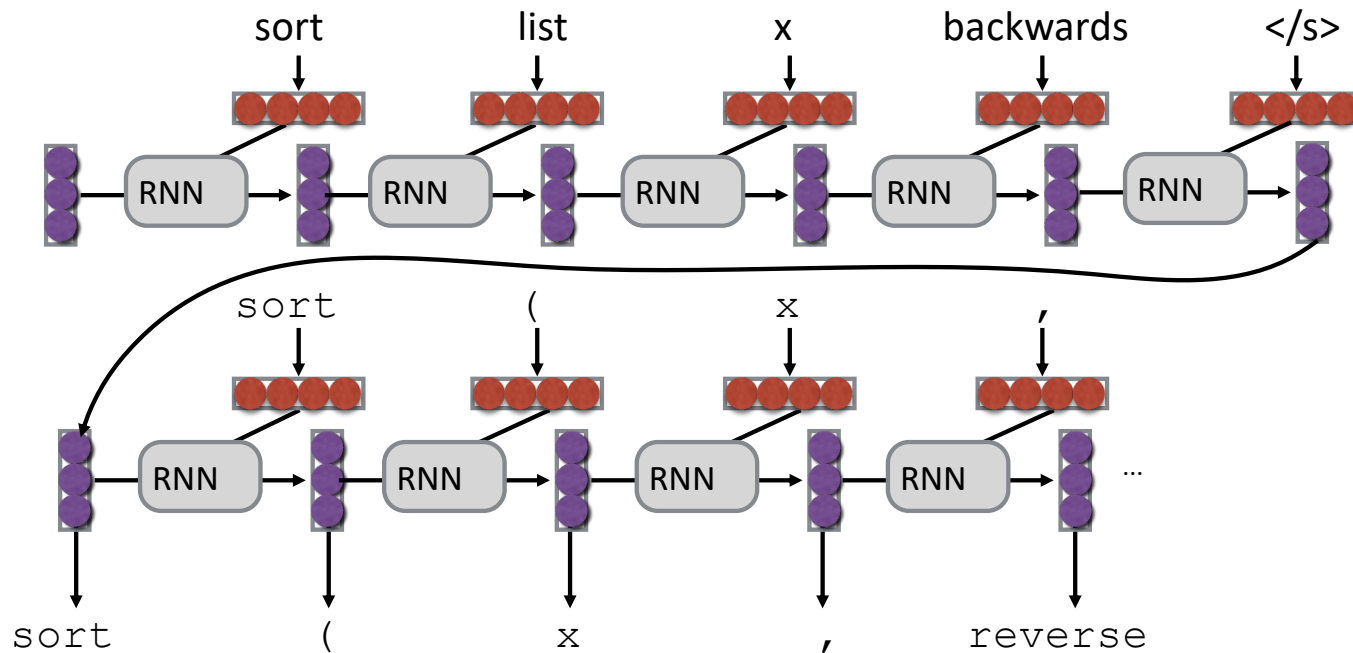
- Code generation vs. semantic parsing

NL \rightarrow code

NL \rightarrow a structured meaning representation,
could be code, could be other

A Naïve Neural Attempt

- Run a sequence-to-sequence model and generate code



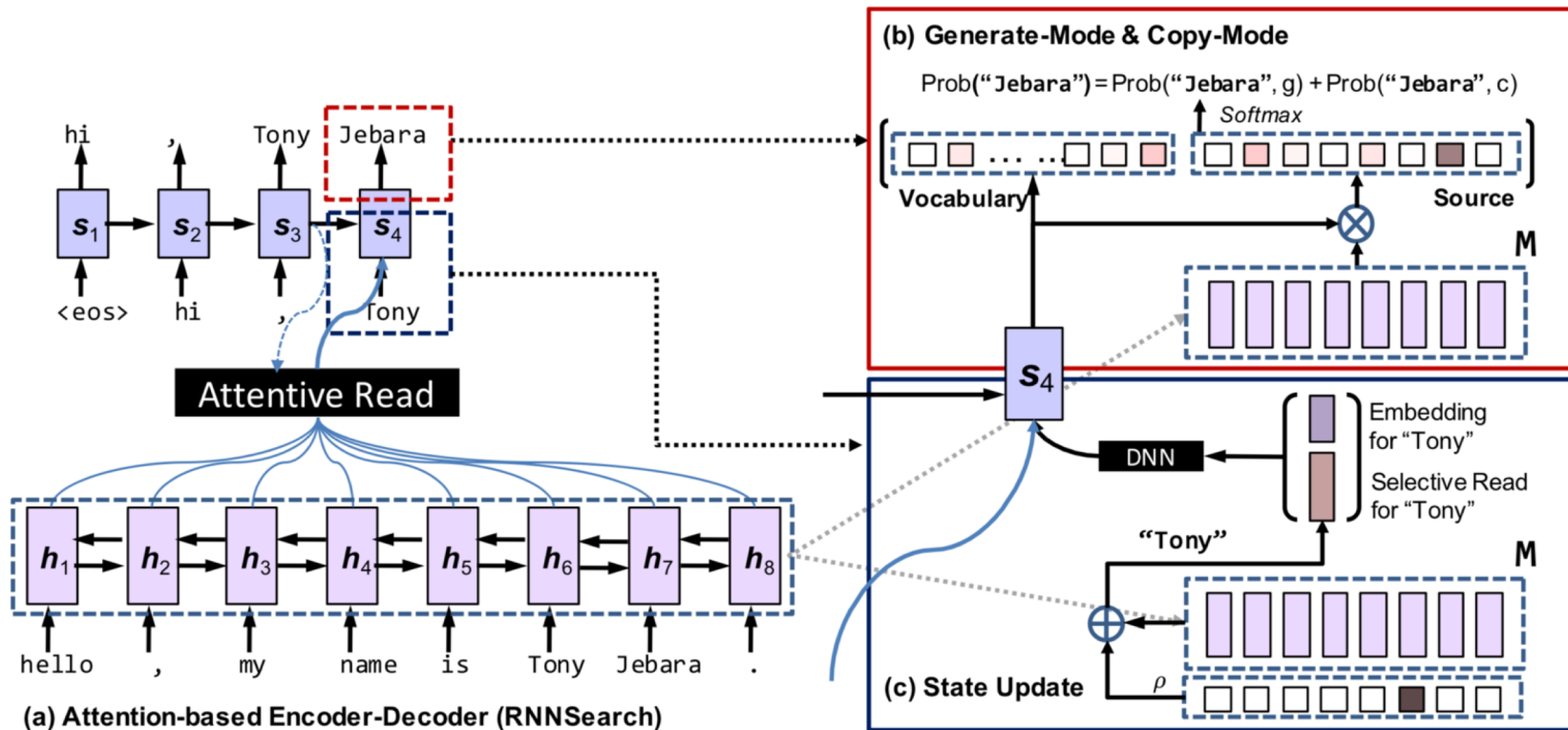
- Works somewhat, e.g. for regexes [Locascio+16]
- For more complex tasks, we need to do e.g. data augmentation to be competitive [Jia+16]

Taking Advantages of Features of Code

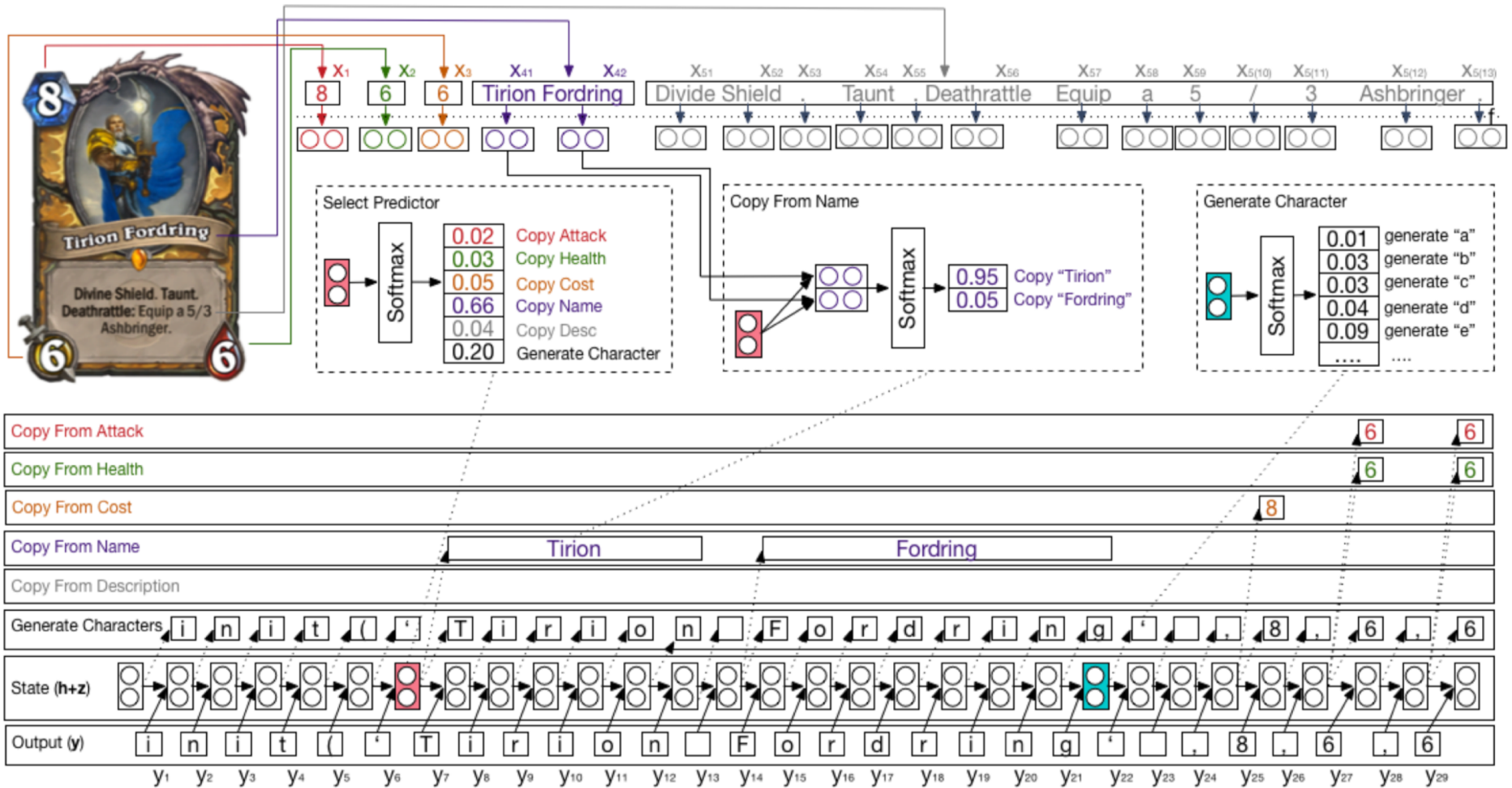
- **Copy** variables names
- Use the **program grammar**
- Use the fact that **code is executable**

Copying Variables

- A simple way to copy variables in neural models: have a “copy” mechanism that can choose to generate from input sentence [Gu+16]



Character-based Generation + Copying [Ling+16]



Incorporating Grammar: Pre-neural Synchronous Grammar-based Methods [e.g. Wong+06]

- Idea: we have a grammar that parses input sentence, generates code

if <X1> -> if <X1>:

<X1> is divisible by <X2>

if x is divisible by five

-> <X1> % <X2> == 0

if x % 5 == 0:

x -> x

5 -> 5

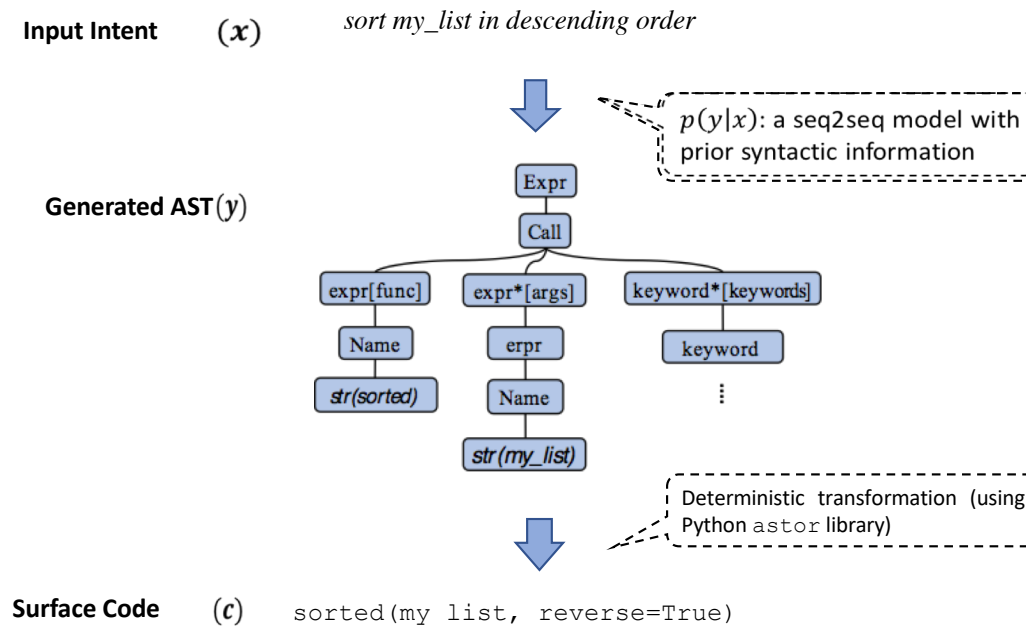
- Grammar rules are extracted from alignments and scored
- **Advantage:** good at modeling compositionality
- **Disadvantage:** don't work well when NL/code connections are tenuous

Neural Models w/ Grammar

- Neural models are better at handling indirect relationships between input and output, can be easily globally optimized
- How do we incorporate grammar?
 - As **constraints** on the output space
 - As a way to **model information flow** in the network

Syntactic Methods

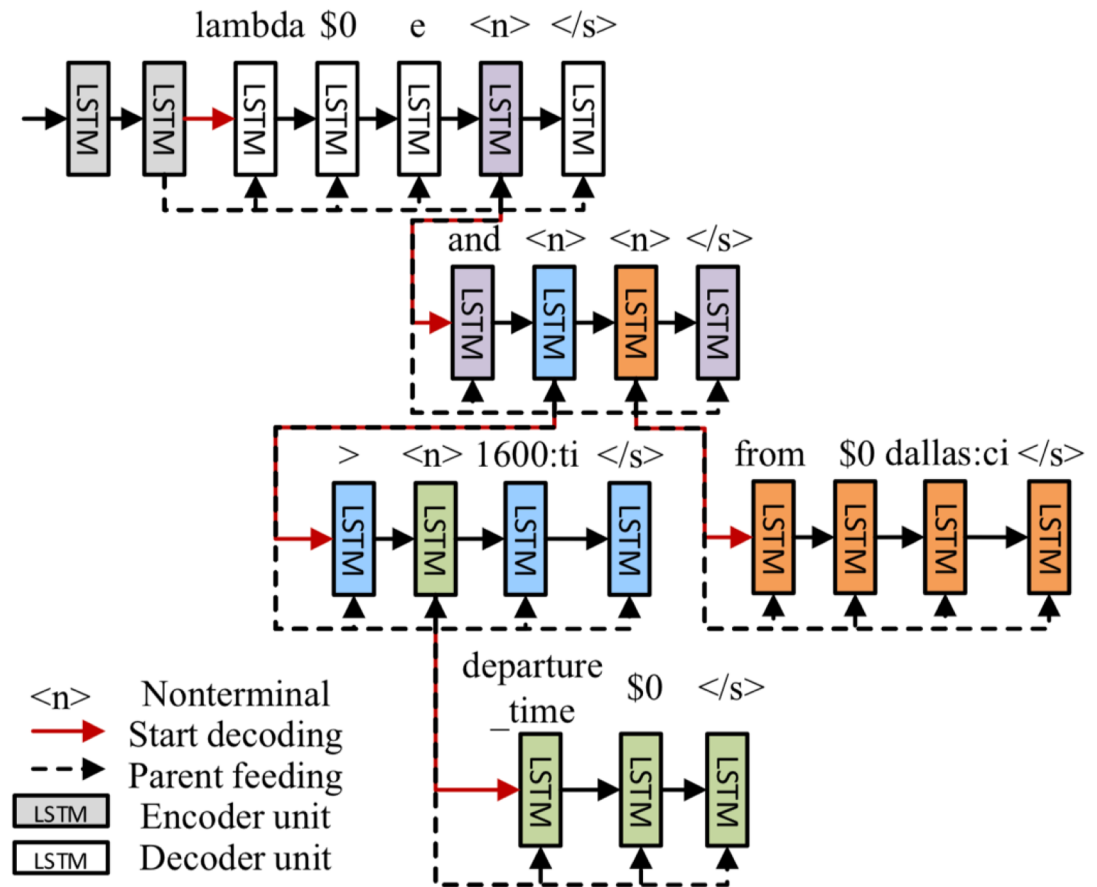
- **Key idea:** use the grammar of the programming language (Python) as prior knowledge in a neural model



Level-by-level Generation of Tree Structures

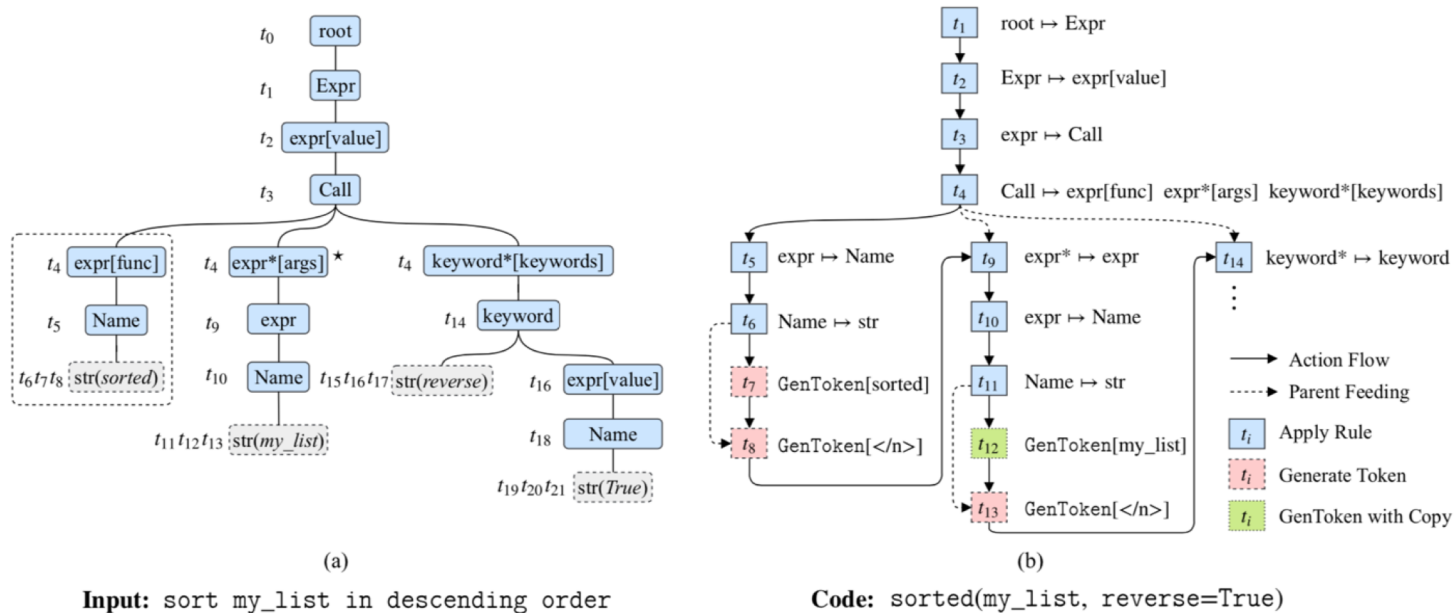
[Dong+16]

- Sequence-to-tree model for generation of tree-structured outputs
- Pass information from top to bottom
- No explicit idea of grammar or explicit constraints



Top-down Generation of CFG Rules [Yin+17]

- Generate AST using CFG rules gathered from parsed corpus
- Factorize the AST into actions:
 - `ApplyRule`: generate an internal node in the AST
 - `GenToken`: generate (part of) a token



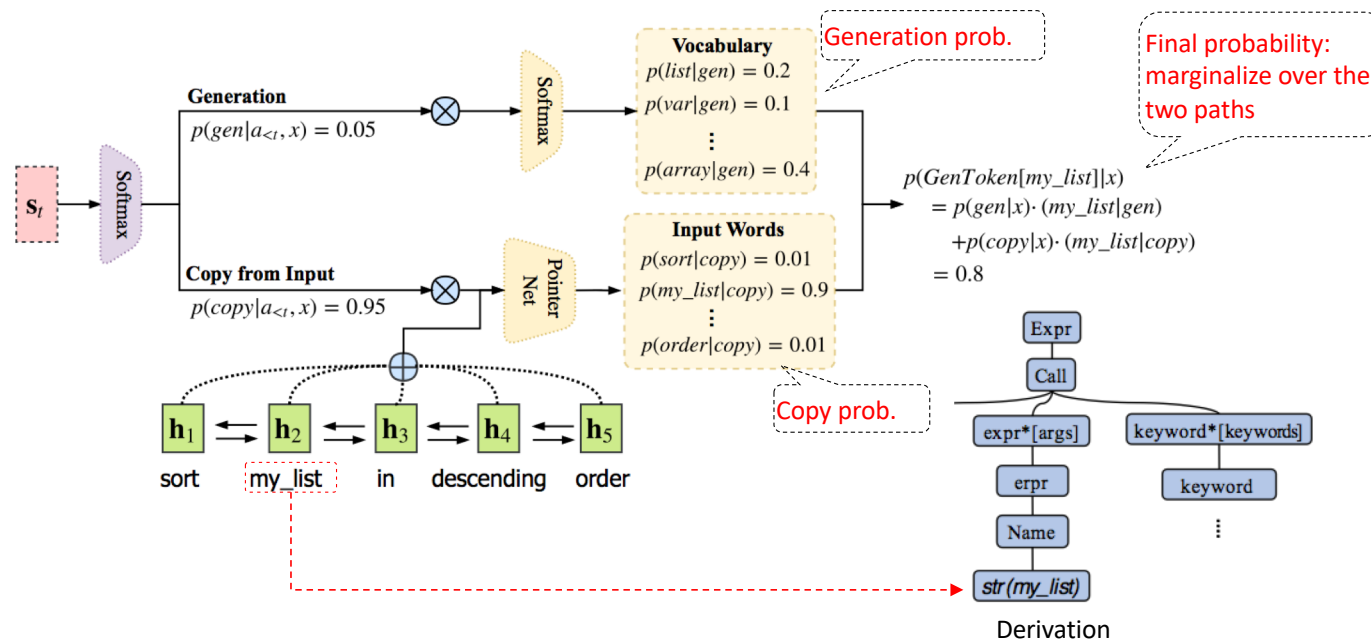
Computing Action Probabilities

$\text{ApplyRule}[r]$: apply a production rule to a non-terminal node

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

$\text{GenToken}[r]$: append a token to the current terminal node

dealing with OOV: make it possible to copy, and also generate with subwords



Using Abstract Syntax Description Language [Rabinovich+17]

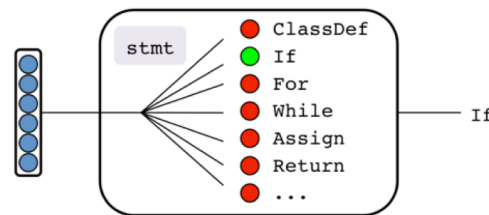
- Every programming language has a specification
- Create a number of “modules” that generate parts of the tree based on this

```

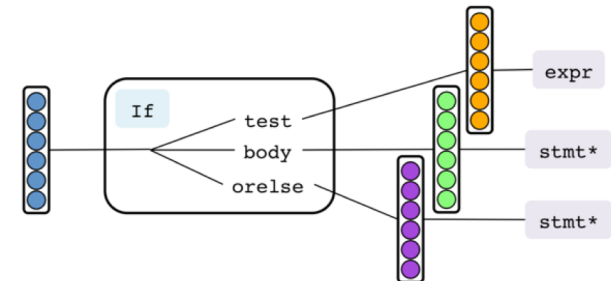
primitive types: identifier, object, ...

stmt
= FunctionDef(
  identifier name, arg* args, stmt* body)
| ClassDef(
  identifier name, expr* bases, stmt* body)
| Return(expr? value)
| ...

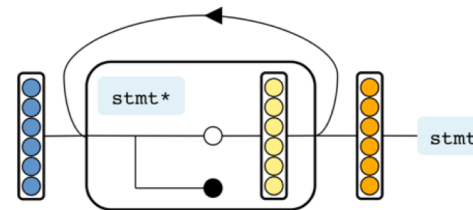
expr
= BinOp(expr left, operator op, expr right)
| Call(expr func, expr* args)
| Str(string s)
| Name(identifier id, expr_context ctx)
| ...
  
```



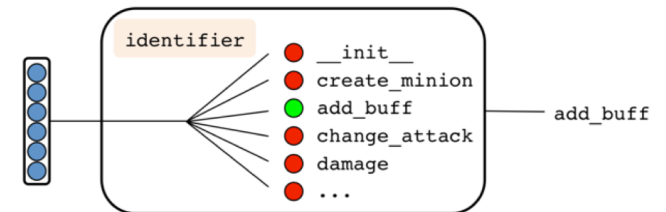
(a) A composite type module choosing a constructor for the corresponding type.



(b) A constructor module computing updated vertical LSTM states.



(c) A constructor field module (sequential cardinality) generating children to populate the field. At each step, the module decides whether to generate a child and continue (white circle) or stop (black circle).



(d) A primitive type module choosing a value from a closed list.

Coarse-to-fine Learning

[Dong+18]

- Idea, there is a limited number of “sketches” of programs that people want to generate.

Dataset	Length	Example
GEO	7.6	<i>x</i> : which state has the most rivers running through it?
	13.7	<i>y</i> : (argmax \$0 (state:t \$0) (count \$1 (and (river:t \$1) (loc:t \$1 \$0))))
	6.9	<i>a</i> : (argmax#1 state:t@1 (count#1 (and river:t@1 loc:t@2)))
ATIS	11.1	<i>x</i> : all flights from dallas before 10am
	21.1	<i>y</i> : (lambda \$0 e (and (flight \$0) (from \$0 dallas:ci) (< (departure_time \$0) 1000:ti)))
	9.2	<i>a</i> : (lambda#2 (and flight@1 from@2 (< departure_time@1 ?)))
DJANGO	14.4	<i>x</i> : if length of bits is lesser than integer 3 or second element of bits is not equal to string 'as' ,
	8.7	<i>y</i> : if len(bits) < 3 or bits[1] != 'as':
	8.0	<i>a</i> : if len (NAME) < NUMBER or NAME [NUMBER] != STRING :
WIKISQL	17.9	Table schema: Pianist Conductor Record Company Year of Recording Format
	13.3	<i>x</i> : What record company did conductor Mikhail Snitko record for after 1996?
	13.0	<i>y</i> : SELECT Record Company WHERE (Year of Recording > 1996) AND (Conductor = Mikhail Snitko)
	2.7	<i>a</i> : WHERE > AND =

- First predict the sketch, then predict the variables, etc.

Using Execution Results

- Another advantage of programs: we can execute the program and see the results!

Type	Training Time	Test Input	Test Output
Programming by Demonstration / Inductive Program Synthesis	Input/Output + Program	Input/Output	Program
Weakly Supervised Semantic Parsing	Natural Language + Input(?)/Output	Natural Language	Program
Programming by Demonstration and Language	Natural Language + Input/Output + Program	Natural Language + Input/Output	Program

Programming by Demonstration/ Inductive Program Synthesis

Miltos Allamanis	→ M. Allamanis
Graham Neubig	→ G. Neubig
Big Bird	→ ???

- This is a whole other tutorial, [Gaunt+16] give a nice overview
- Many methods including:
 - **Satisfiability modulo theory solvers** [Summers+86], **sketches** [Solar-Lezama+08]
 - **Neural methods:** encode input/output examples, generate program [Gaunt+16]
- Harder than learning from NL because of fewer hints, but easier because it's verifiable

Semantic Parsing from Question/Answer Pairs

[Clarke+10]

what state has the largest capital → arizona

what city hosts Carnegie Mellon University → Pittsburgh

- In a DSL for database queries, try to generate several possible queries, then update towards the one that is correct

what state has the largest capital

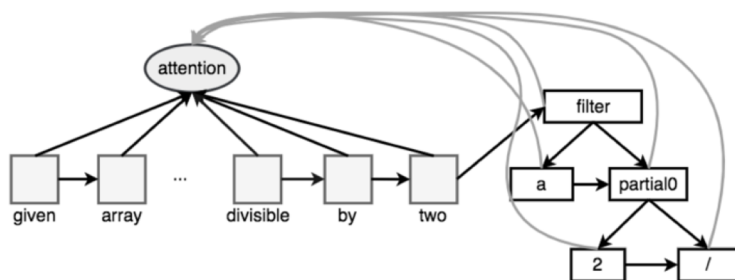
↗	<code>answer(A,largest(A,(state(A),loc(A,B),capital(B))))</code>	Wrong!
↘	<code>answer(A,(state(A),loc(B,A),largest(B,capital(B))))</code>	Correct!

- Motivation: this “weak supervision” often easier to create question/answer pairs
- Similar methods can be used for code as well, e.g. when generating SQL queries [e.g. Zhong+17]

Code Synthesis with Natural Language Guidance [e.g. Polosukhin+18]

Abbreviate the first name + Milto​s Allamanis → M. Allamanis
+ Graham Neubig → G. Neubig
Big Bird → ???

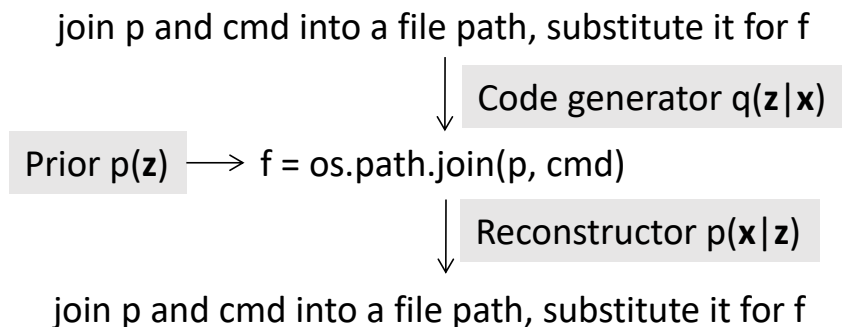
- **Problem:** Code synthesis methods based on exact search (e.g. sketching) only work for quite simple problems
- **Idea:** use standard code synthesis machinery, but additionally use a natural language query to guide search



When full tree found, confirm it passes unit tests!

Reconstruction Loss: Supervision w/o Execution [Yin+18]

- **Motivation:** we have lots of unlabeled user inputs to learn from
- **Method:** after generating code, try to reconstruct the user input
- This makes sure that information in the input is preserved in code
- Specifically, use VAE formulation, which also makes it possible to design prior over code (e.g. using large datasets)



NL	<i>join p and cmd into a file path, substitute it for f</i>	
z_1^s	<code>f = os.path.join(p, cmd)</code> ✓	
	$\log q(z x) = -1.00$	$\log p(x z) = -2.00$
	$\log p(z) = -24.33$	$l(x, z) = 9.14$
z_2^s	<code>p = path.join(p, cmd)</code> ✗	
	$\log q(z x) = -8.12$	$\log p(x z) = -20.96$
	$\log p(z) = -27.89$	$l(x, z) = -9.47$
NL	<i>append i-th element of existing to child_loggers</i>	
z_1^s	<code>child_loggers.append(existing[i])</code> ✓	
	$\log q(z x) = -2.38$	$\log p(x z) = -9.66$
	$\log p(z) = -13.52$	$l(x, z) = 1.32$
z_2^s	<code>child_loggers.append(existing[existing])</code> ✗	
	$\log q(z x) = -1.83$	$\log p(x z) = -16.11$
	$\log p(z) = -12.43$	$l(x, z) = -5.08$

A Final Alternative: Code Search

[e.g. Zhang+16]

- Assume that we have the code we want somewhere on the web
- **Idea:** query a search engine (e.g. Bing, Google) with the natural language, find the top N pages, and return the code snippets

[dictionary - How to reverse order of keys in python dict? - Stack ...](#)

<https://stackoverflow.com/questions/5455606/how-to-reverse-order-of-keys-in-python-dict>

This is my code : a = {0:'000000',1:'11111',3:'333333',4:'444444'} for i in a: print i it shows: 0 1 3 4
but I want it to show: 4 3 1 0 so, what can I do?

6 answers

✓ **Top answer**
26 votes

The order keys are iterated in is arbitrary. It was only a coincidence that they were in...

Answer 2 of 6
12 votes

Dictionaries are unordered so you cannot reverse them. The order of the current output is...

Answer 3 of 6
8 votes

Try: for i in sorted(a, key=reverse=True): print i

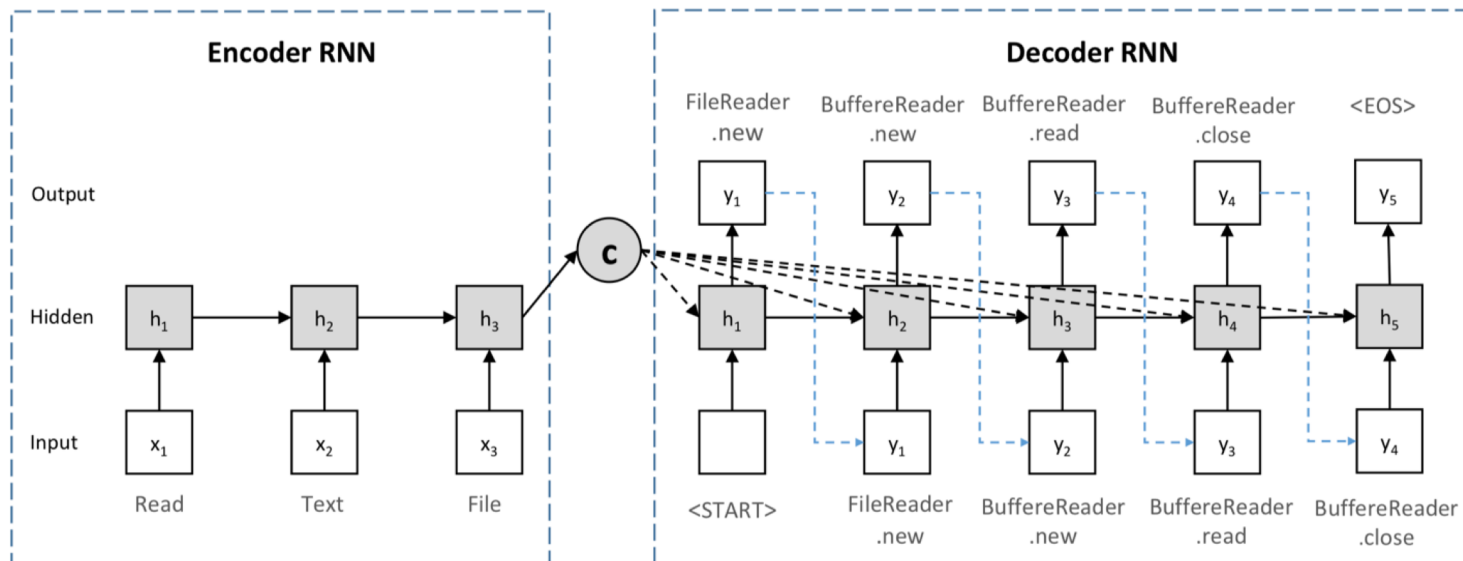
[python dictionary sorting in descending order based on values - Stack ...](#)

<https://stackoverflow.com/questions/20577840/python-dictionary-sorting-in-descending-order-...>

I want to sort this dictionary d based on value of sub key key3 in descending order. See below: d = { '123': { 'key1': 3, 'key2': 11, 'key3': 3 }, '124': { 'key1': 6, 'key2': 56, 'key3': 6 }, '125': { 'key1': 7, 'key2': ... [MORE](#) ▾

API (Sequence) Search [Gu+2016]

- Many intents can be realized by a sequence of API calls
- Train encoder-decoder that outputs API call sequence over full language



Modeling Natural Language in Code

Modelling Natural Language Aspects of Code



Variable Names



Type Inference



Program Analysis (via code's NL aspects)

Variable Naming Task

```
int SumEven(int[] arr, int lim) {  
    int ████ = 0;  
    for (int i = 0; i < lim; i++)  
        if (arr[i] % 2 == 0)  
            ████ += arr[i];  
  
    return ████;  
}
```

Usage
context

Sum

Of

Even

Allamanis *et al.* 2014, 2015, 2018 Raychev *et al.* 2015, Vasilescu *et al.* 2017,
Bavishi *et al.* 2018, Alon *et al.* 2018

Predicting Variable Names

```
int SumEven(int[] arr, int lim) {  
    int ██████ = 0;  
    for (int i = 0; i < lim; i++)  
        if (arr[i] % 2 == 0)  
            ██████ += arr[i];  
  
    return ██████;  
}
```

Encode Usage Context

- Use language model
- Build Discriminative Model

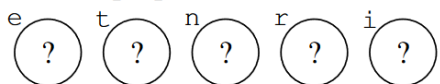
```
function chunkData(e, t) {
  var n = [];
  var r = e.length;
  var i = 0;
  for (; i < r; i += t) {
    if (i + t < r) {
      n.push(e.substring(i, i + t));
    } else {
      n.push(e.substring(i, r));
    }
  }
  return n;
}
```

(a) JavaScript program with minified identifier names

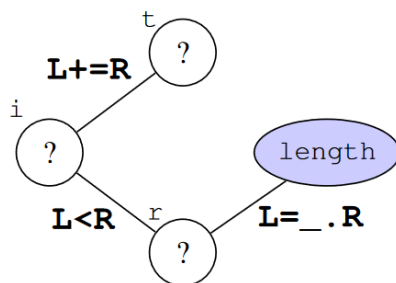
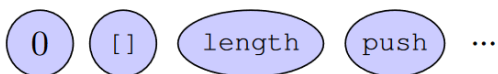
```
/* str: string, step: number, return: Array */
function chunkData(str, step) {
  var colNames = []; /* colNames: Array */
  var len = str.length;
  var i = 0; /* i: number */
  for (; i < len; i += step) {
    if (i + step < len) {
      colNames.push(str.substring(i, i + step));
    } else {
      colNames.push(str.substring(i, len));
    }
  }
  return colNames;
}
```

(e) JavaScript program with new identifier names and types

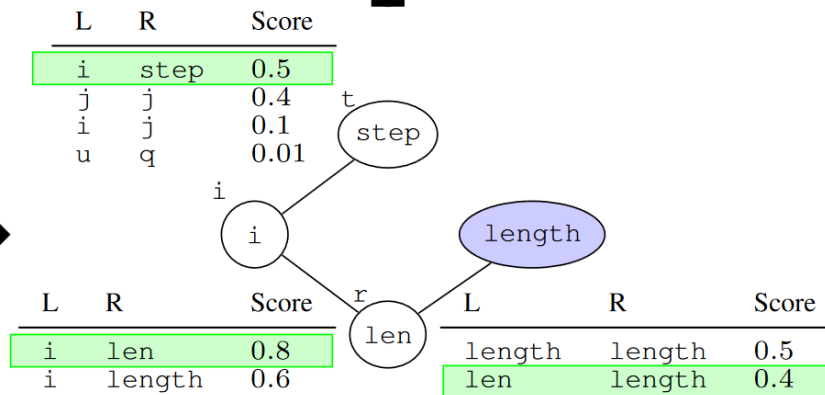
Unknown properties (variable names):



Known properties (constants, APIs):



(c) Dependency network



(d) Result of MAP inference

(b) Known and unknown name properties

Type Inference using Natural Language

```
1: def gzip(f, *args, **kwargs):  
2:     resp = f(*args, **kwargs) → type: Response  
3:     url = resp.url  
4:     mthd = resp.method → type: string  
5:     data = compress(resp)  
6:     ...  
7:     result = resp  
8:     return result
```

Modelling NL Aspects of Code for Program Analysis

Declaration: `string Substring(int startIndex, int offset)`

Uses:

- `str1.Substring(startIdx, offset)`
- `str1.Substring(off, start)`

Rice *et al.* "Detecting Argument Selection Defects" 2017

Pradel and Sen "Deep Learning to Find Bugs" 2017

Modelling NL Aspects of Code for Program Analysis

Type	Parameter	Original argument	Correct argument
Duration	responseTTLDuration	frequencyCapDuration	responseTTLDuration
Duration	frequencyCapDuration	responseTTLDuration	frequencyCapDuration
List<A>	slotResponse	slotResponse	slotResponse
long	communityId	a.toDataObject().getId()	a.toDataObject().getId()
long	senderId	e.getSenderId()	e.getSenderId()
long	recipientId	e.getRecipientId()	e.getRecipientId()
long	subject	subject	subject
String	textContent	htmlContent	textContent
String	htmlContent	textContent	htmlContent

Rice *et al.* "Detecting Argument Selection Defects" 2017

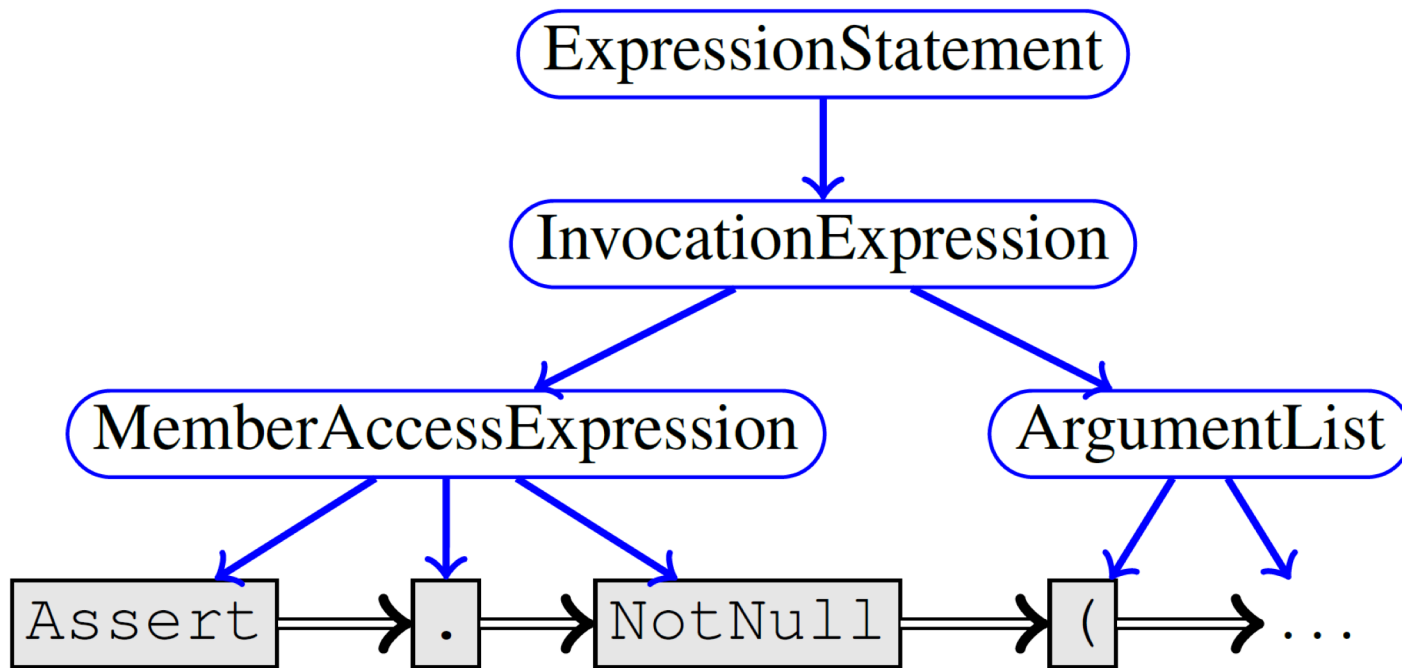
Pradel and Sen "Deep Learning to Find Bugs" 2017

Modelling NL Aspects of Code for Program Analysis

```
var clazz=classTypes["Root"].Single() as JsonCodeGenerator.ClassType;  
Assert.NotNull(clazz);  
  
var first=classTypes["RecClass"].Single() as JsonCodeGenerator.ClassType;  
Assert.NotNull(██████████);  
  
Assert.Equal("string", first.Properties["Name"].Name);  
Assert.False(clazz.Properties["Name"].IsArray);
```

Possible type-correct options: `clazz`, `first`

Representing Program Structure as a Graph



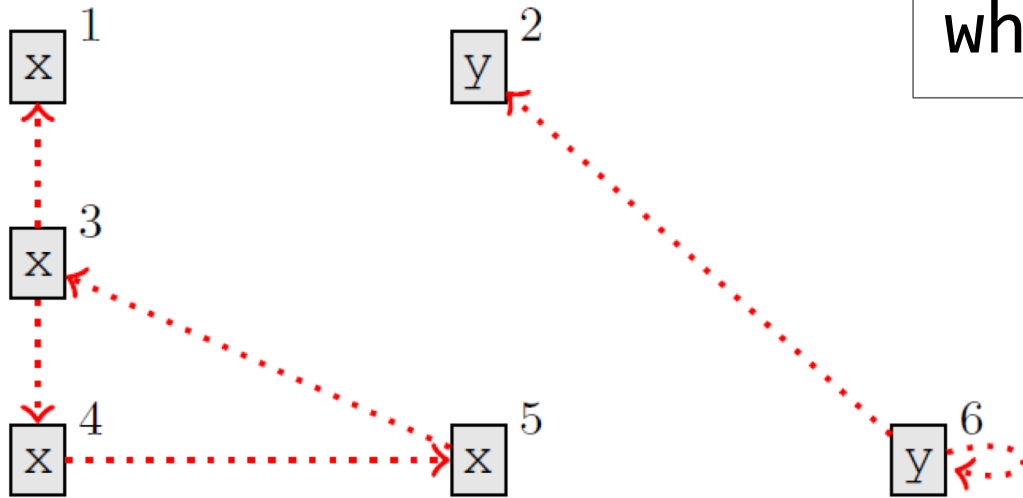
```
var clazz=classTypes["Root"].Single() as JsonCodeGenerator.ClassType;
Assert.NotNull(clazz);

var first=classTypes["RecClass"].Single() as JsonCodeGenerator.ClassType;
Assert.NotNull(clazz);

Assert.Equal("string", first.Properties["Name"].Name);
Assert.False(clazz.Properties["Name"].IsArray);
```

Representing Program Structure as a Graph

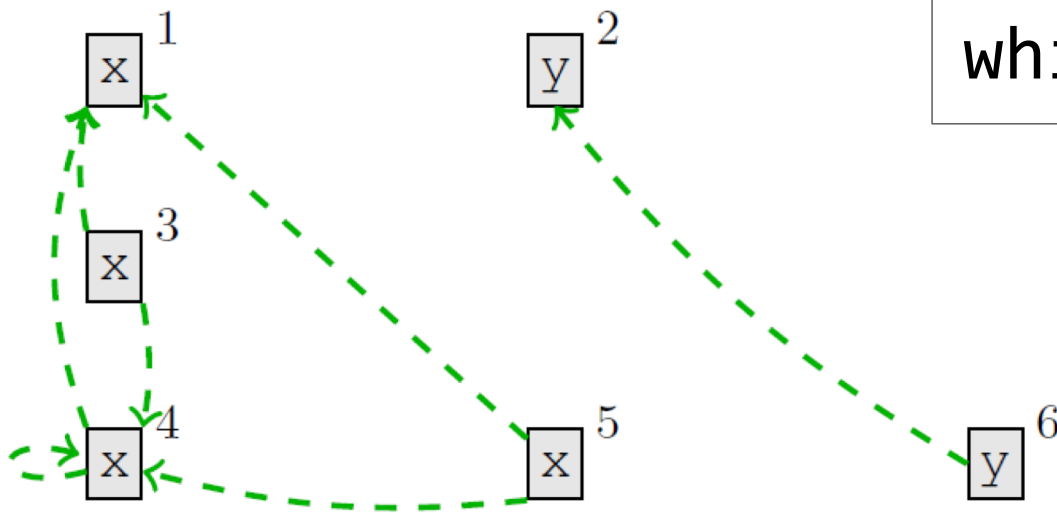
```
(x1, y2) = Foo();  
while (x3 > 0) x4 = x5 + y6
```



LastUse

Representing Program Structure as a Graph

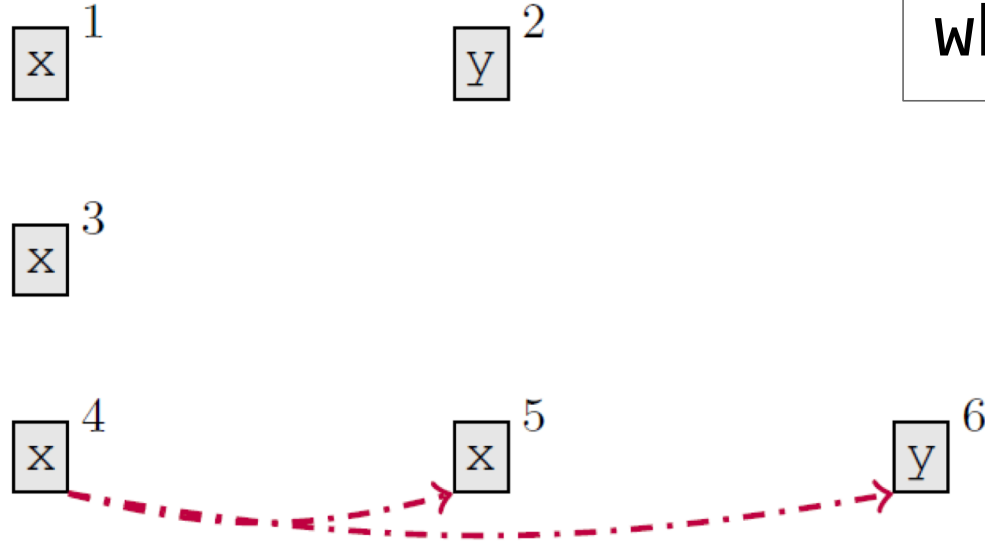
```
(x1, y2) = Foo();  
while (x3 > 0) x4 = x5 + y6
```



LastWrite

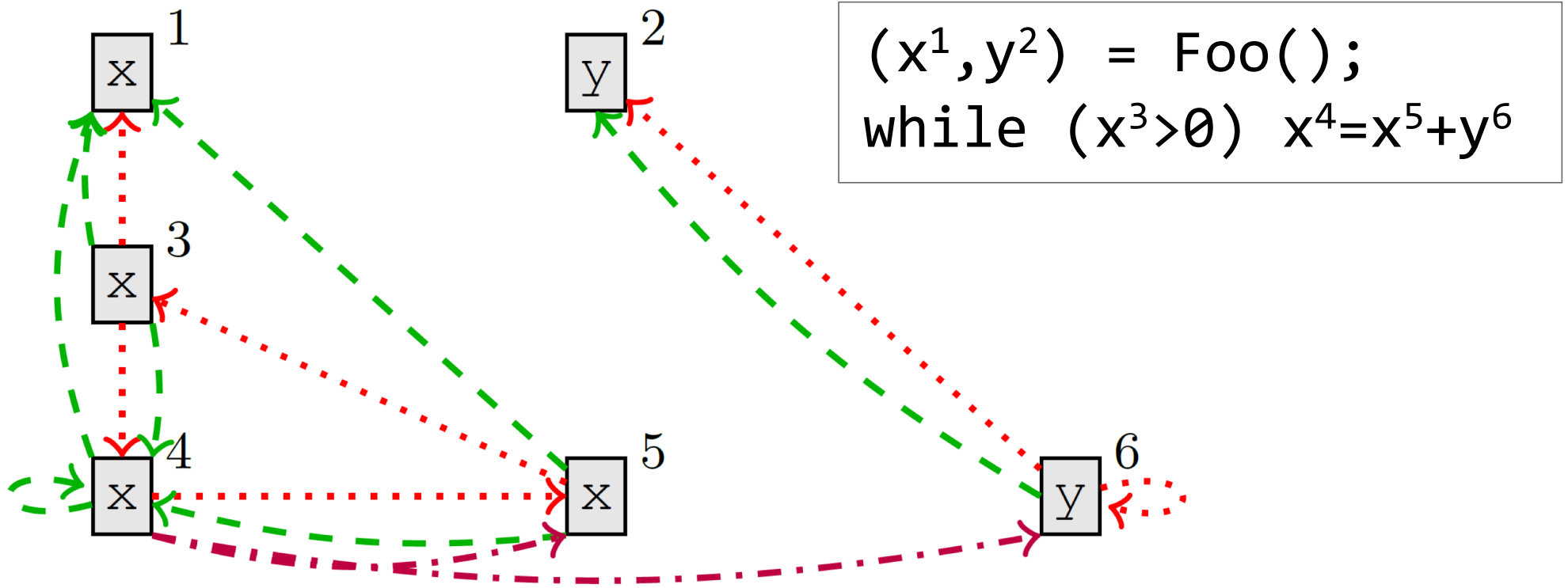
Representing Program Structure as a Graph

```
(x1, y2) = Foo();  
while (x3 > 0) x4 = x5 + y6
```



ComputedFrom

Representing Program Structure as a Graph



Representing Program Structure as a Graph

Additional Edge Types:

- ReturnsTo
- FormalArgName

```
b = foo(result);
```

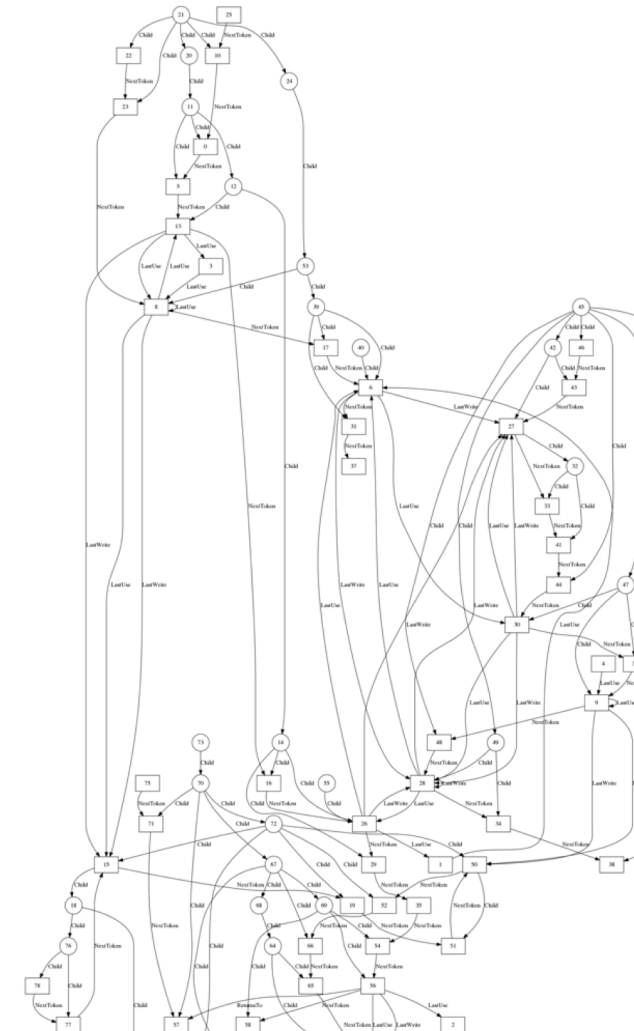


```
sum
```

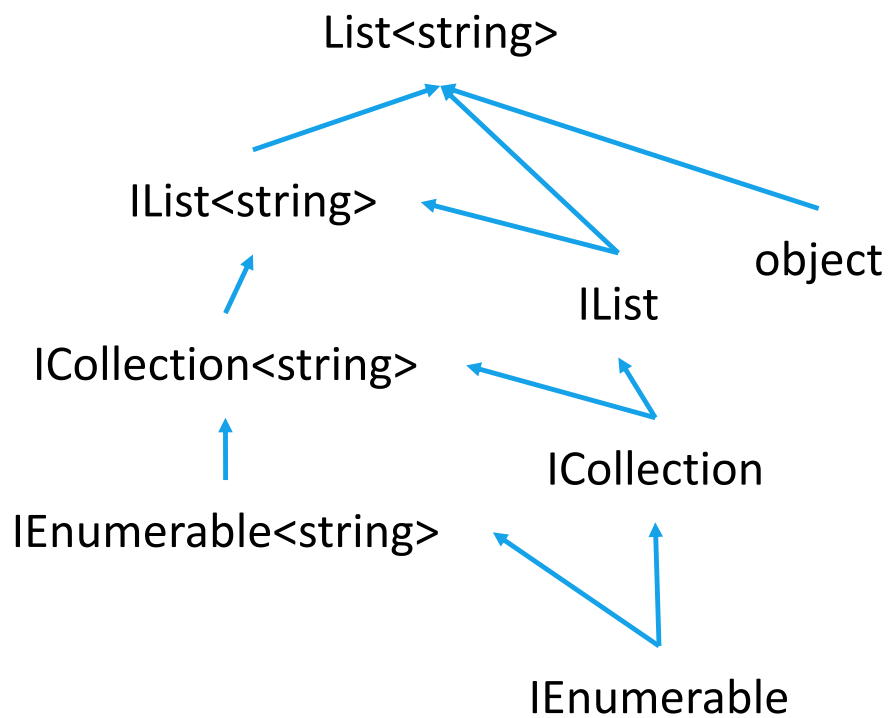
```
void foo(int sum) { ... }
```

Modelling NL Aspects of Code for Program Analysis

```
int SumPositive(int[] arr, int lim) {  
    int sum = 0;  
    for (int i=0; i < lim; i++)  
        if (arr[i] > 0)  
            += arr[i];  
    return sum;  
}
```



Representing Variable Type Information



$$\tau^*(v) = \{\tau_{List<string>}, \tau_{IList}, \tau_{object}, \dots\}$$

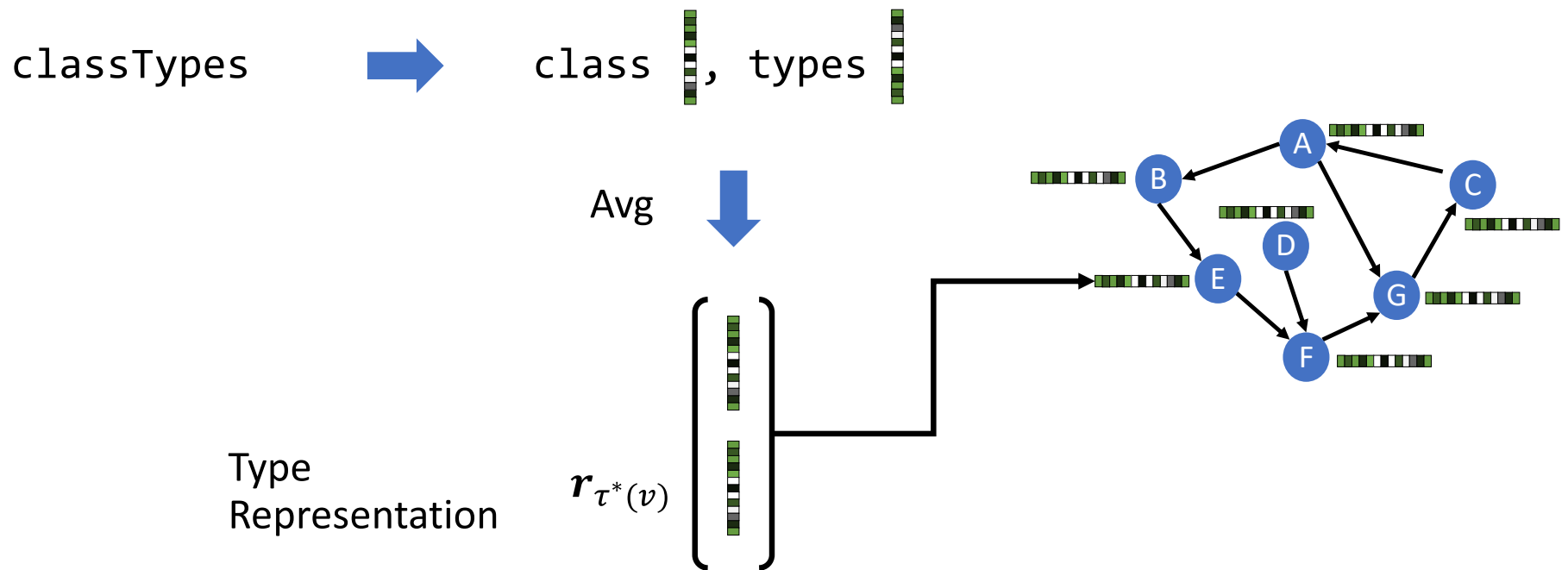
\downarrow \downarrow \downarrow

$r_{List<string>}$ r_{IList} r_{object}

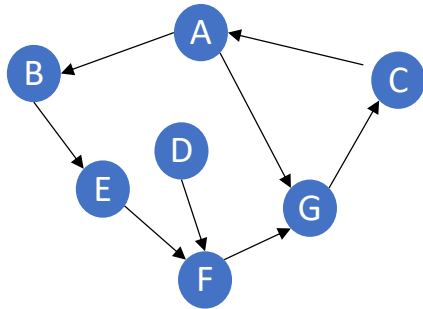
↓ Elementwise Max

$r_{\tau^*(v)}$

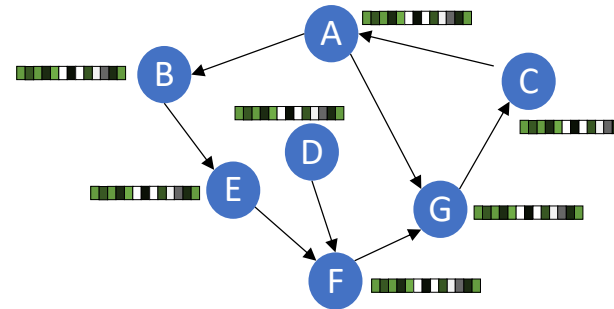
Representing Nodes



Graph Neural Networks



Graph Representation
of Problem

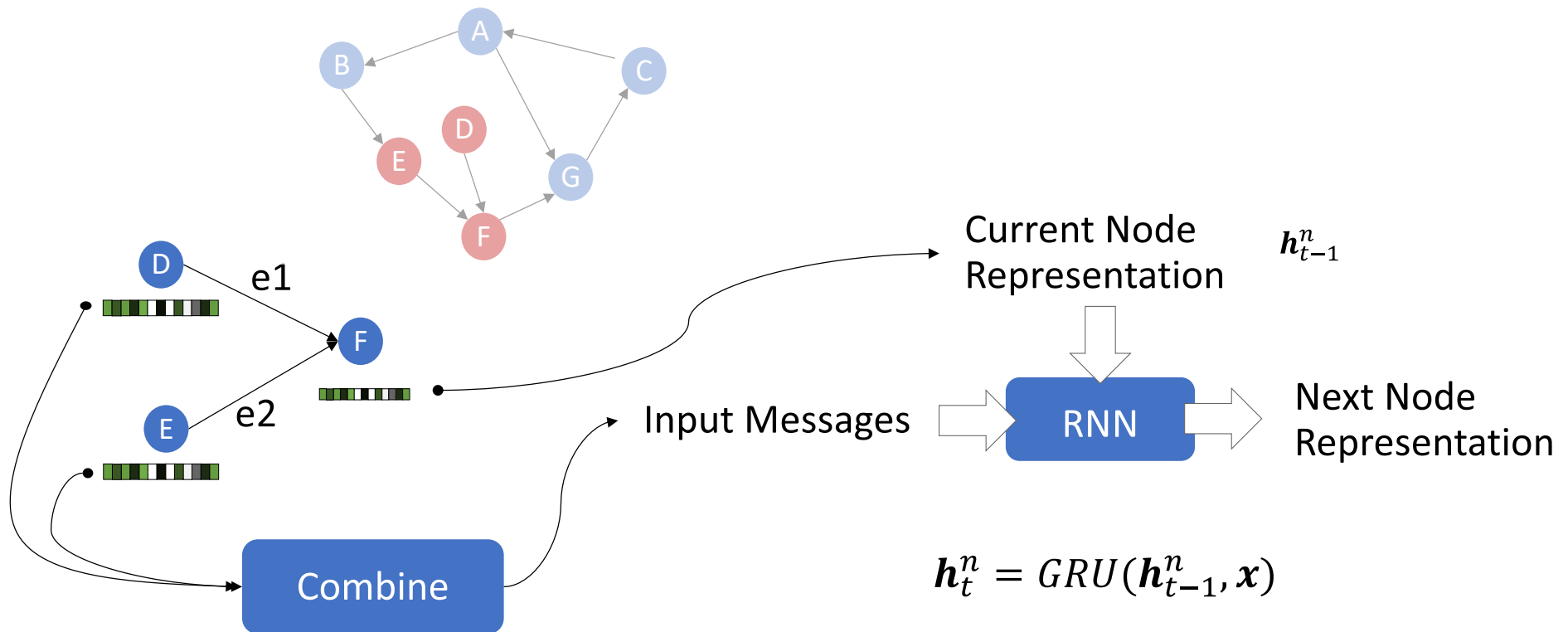


Initial Representation
of each node

Li et al (2015). Gated Graph Sequence Neural Networks.

Gilmer et al (2017). Neural Message Passing for Quantum Chemistry.

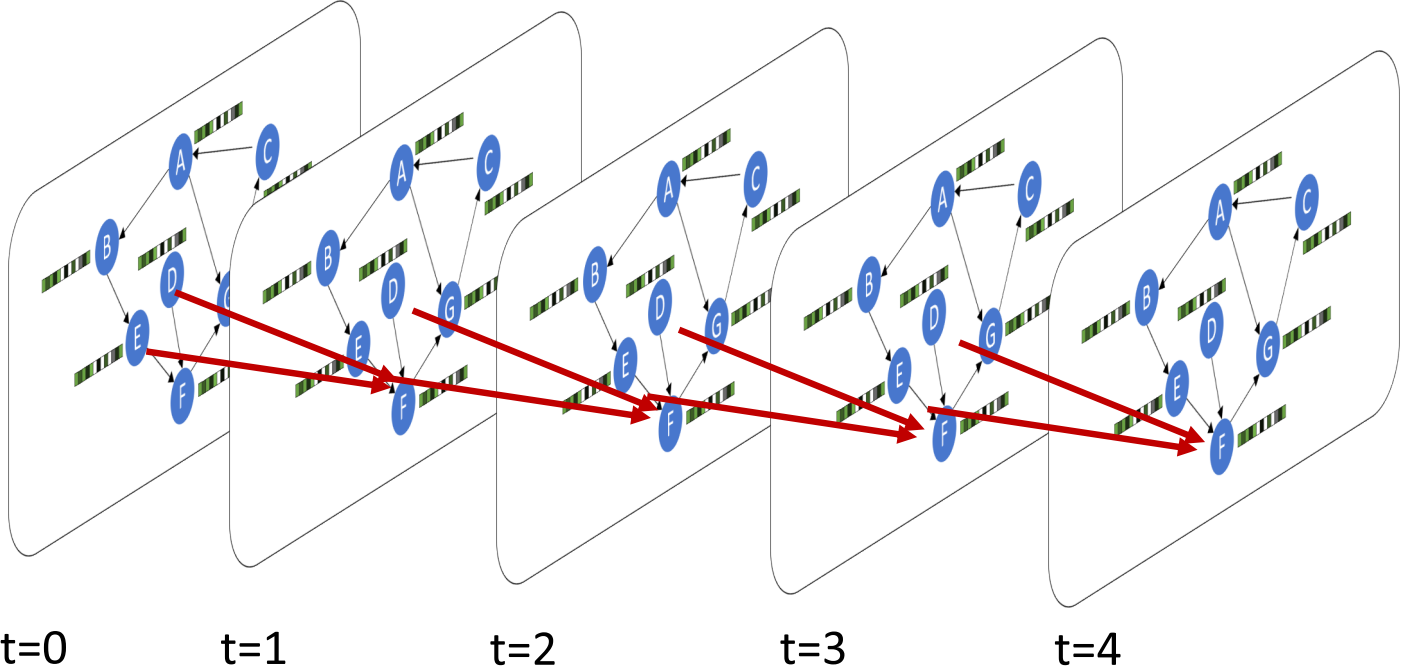
Graph Neural Networks: Message Propagation



$$x = \sum_{n' \in \text{neig}(n)} E_{\tau(n' \rightarrow n)} h_{t-1}^{n'} + b_{\tau(n' \rightarrow n)}$$

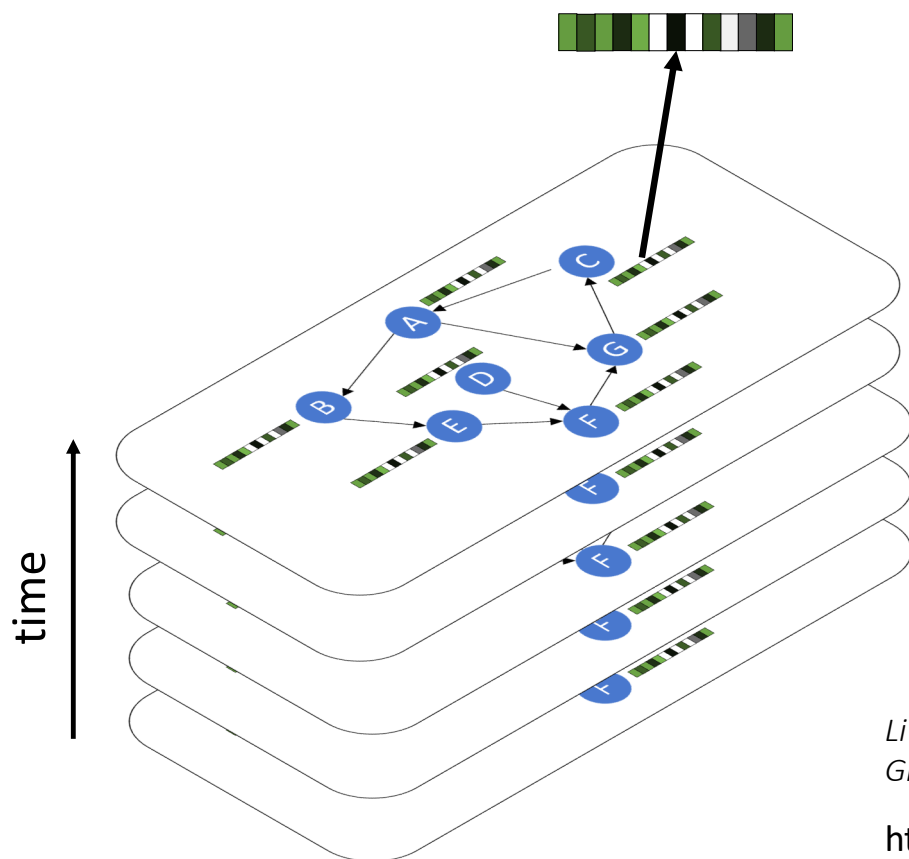
Li et al (2015). Gated graph sequence neural networks.

Graph Neural Networks: Unrolling



Li et al (2015). Gated graph sequence neural networks.

Graph Neural Networks: Unrolling

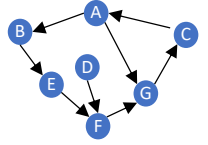


- node selection
- node classification
- graph classification

Li et al (2015). Gated Graph Sequence Neural Networks.

Gilmer et al (2017). Neural Message Passing for Quantum Chemistry.

<https://github.com/Microsoft/gated-graph-neural-network-samples>



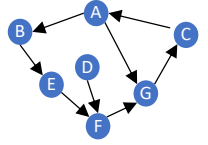
Graph Representation for Variable Misuse

```
var clazz=classTypes["Root"].Single() as JsonCodeGenerator.ClassType;
Assert.NotNull(clazz);

var first=classTypes["RecClass"].Single() as JsonCodeGenerator.ClassType;
Assert.NotNull(██████████);

Assert.Equal("string", first.Properties["Name"].Name);
Assert.False(clazz.Properties["Name"].IsArray);
```

Possible type-correct options: `clazz`, `first`



Graph Representation for Variable Misuse

```
var clazz=classTypes["Root"].Single() as JsonCodeGenerator.ClassType;  
Assert.NotNull(clazz);  
  
var first=classTypes["RecClass"].Single() as JsonCodeGenerator.ClassType;  
Assert.NotNull(SLOT; first clazz  
  
Assert.Equal("string", first.Properties["Name"].Name);  
Assert.False(clazz.Properties["Name"].isArray);
```

Goal: make the representation of SLOT as close as possible to the representation of the correct candidate node

$$(\mathbf{h}_T^{SLOT})^T \mathbf{h}_T^{first} \gg (\mathbf{h}_T^{SLOT})^T \mathbf{h}_T^{clazz}$$

Modeling Communicative Aspects of Software Development

Software Ecosystems are a Rich Discussion Ground

neulab / xnmt

Unwatch 20

Unstar 90

Fork 25

Code

Issues 25

Pull requests 4

Projects 0

Wiki

Filters is:issue is:open

Labels Milestones

25 Open 77 Closed Author Labels Projects Milestones

Running into a NaN problem **major bug**

#399 opened 16 days ago by pierregodard

Multi-task learning and checkpoint saving

#395 opened 19 days ago by mbollmann

Integrating sacrebleu **enhancement**

#393 opened 20 days ago by pmichel31415

Implementing character based embeddings

#392 opened 20 days ago by pmichel31415

pmichel31415 commented 20 days ago

Member + 🗨️

I'm trying to reproduce a model from [Synthetic and Natural Noise Both Break Neural Machine Translation](#) where the word embeddings are the average of the character embeddings.

How would I go about implementing that in xnmt?

To be clear I don't want a character based model, I want a model that processes the input as a sequence of words, but instead of having one embedding per word, has one embedding per character and represents each word as the average of its characters.

neubig commented 20 days ago

Member + 🗨️

Probably you need to create a version of the input class that doesn't convert words to integer IDs, but instead saves their strings. Then it would be as simple as implementing a new version of the word embedder that reads in the characters and does lookup and sum appropriately.

Graham

neubig commented 5 days ago

Member + 🗨️

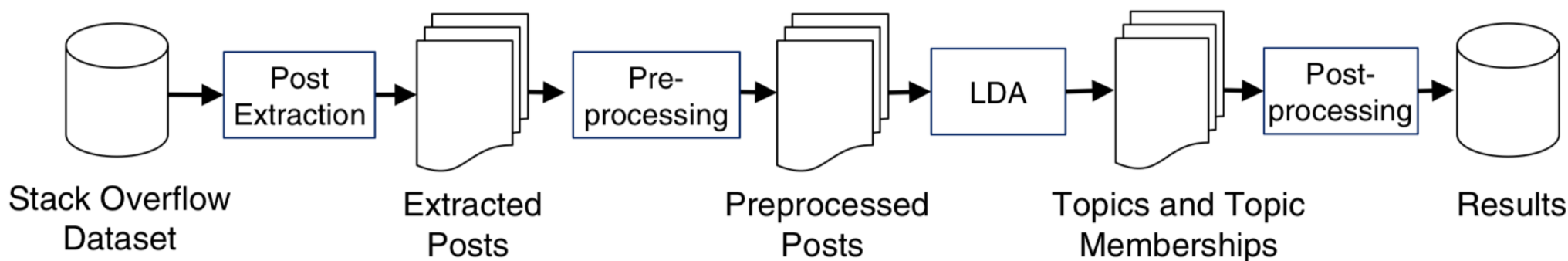
Just remembered that [@philip30](#) already has something like this in his segmenting transducer branch. He might be able to give pointers.

philip30 commented 5 days ago

Member + 🗨️

Modeling Discussion Topics [Barua+14]

- **Research Question:** What are developers talking about?
- **Methodology:** Topic modeling to extract topics, manual inspection



- **Results:** Discover topics about particular development languages (C++, Python, web development platforms), but also job advice, how to learn, etc.

Modeling Language Complexity [Kavaler+17]

- Use language models, global and project specific to answer research questions
- Do people conform to project norms in posting issues?
As people are on the project longer, their entropy drops → Yes
- Does conforming to norms reduce issue resolution time?
Lower language model entropy is associated with faster response times
→ Yes?

Sentiment Analysis for Software [e.g. Lin+18]

- Sentiment analysis has led to many insights in software engineering:
 - More distributed teams have higher sentiment
 - Positive sentiment in issue descriptions correlated with faster fix time
 - Negative sentiment correlated with failing of integration tests
- Challenges in adapting to the SE context

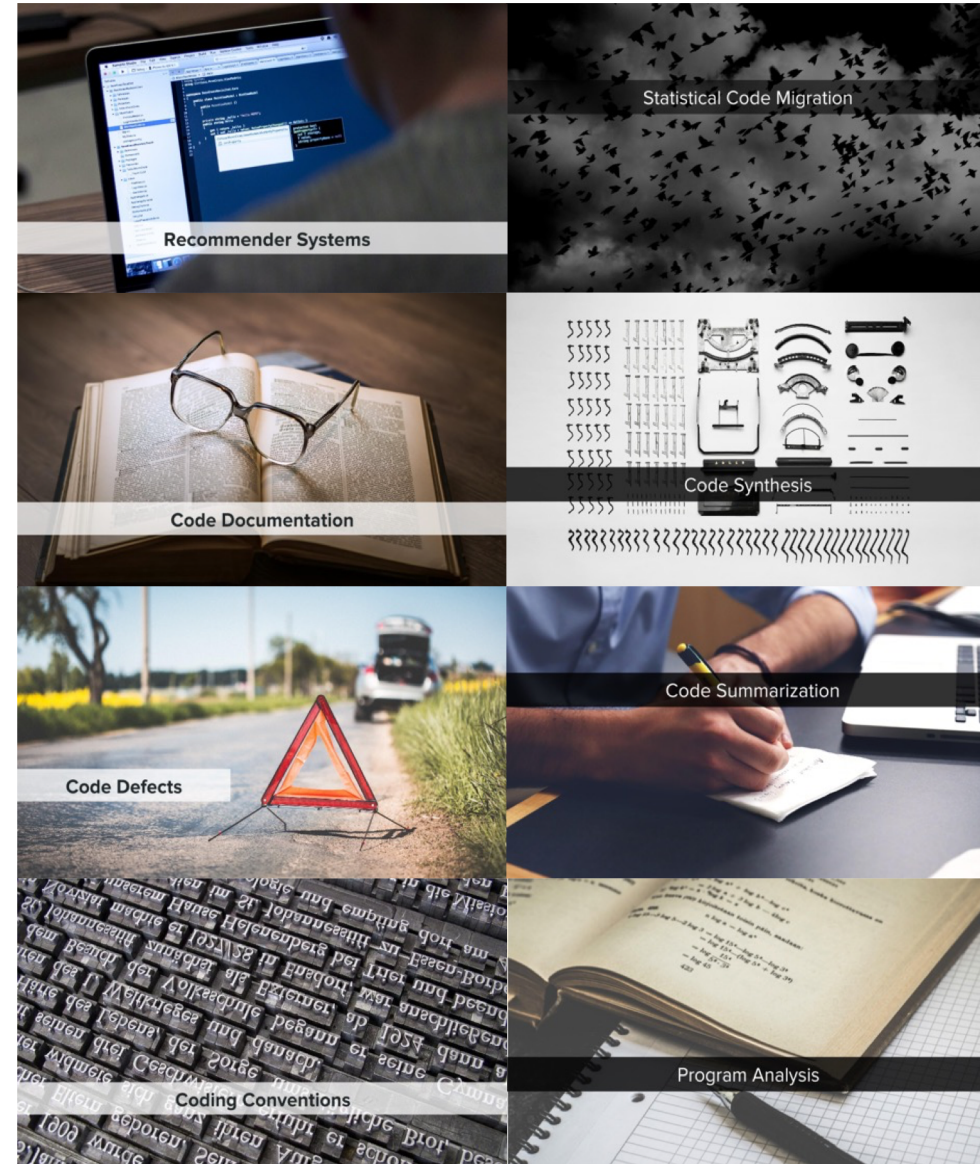
Why Software Language?

- Large, open data of questions/answers, discussions
- Grounded in code
- Task-driven interaction

Conclusion

Research in ML+Code

<https://ml4code.github.io>



1

A Survey of Machine Learning for Big Code and Naturalness

MILTADIS ALLAMANIS, Microsoft Research

EARL T. BARR, University College London

PREMKUMAR DEVANBU, University of California, Davis

CHARLES SUTTON, University of Edinburgh and The Alan Turing Institute

Research at the intersection of machine learning, programming languages, and software engineering has recently taken important steps in proposing learnable probabilistic models of source code that exploit code's abundance of patterns. In this article, we survey this work. We contrast programming languages against natural languages and discuss how these similarities and differences drive the design of probabilistic models. We present a taxonomy based on the underlying design principles of each model and use it to navigate the literature. Then, we review how researchers have adapted these models to application areas and discuss cross-cutting and application-specific challenges and opportunities.

CCS Concepts: Computing methodologies → Machine learning; Natural language processing → Soft

Conclusion

- Lots of interesting problems!
 - Code -> Text
 - Text -> Code
 - Modeling Natural Language in Code
 - Modeling Communication in Software
- Lots of datasets!
 - Curated datasets for code->text tasks
 - Large uncurated resources for you to be creative with
- Lots of potential!
 - There is an increasing technical divide, how can we use technology to close it?

Questions?!