Microsoft

Understanding Source Code with Deep Learning

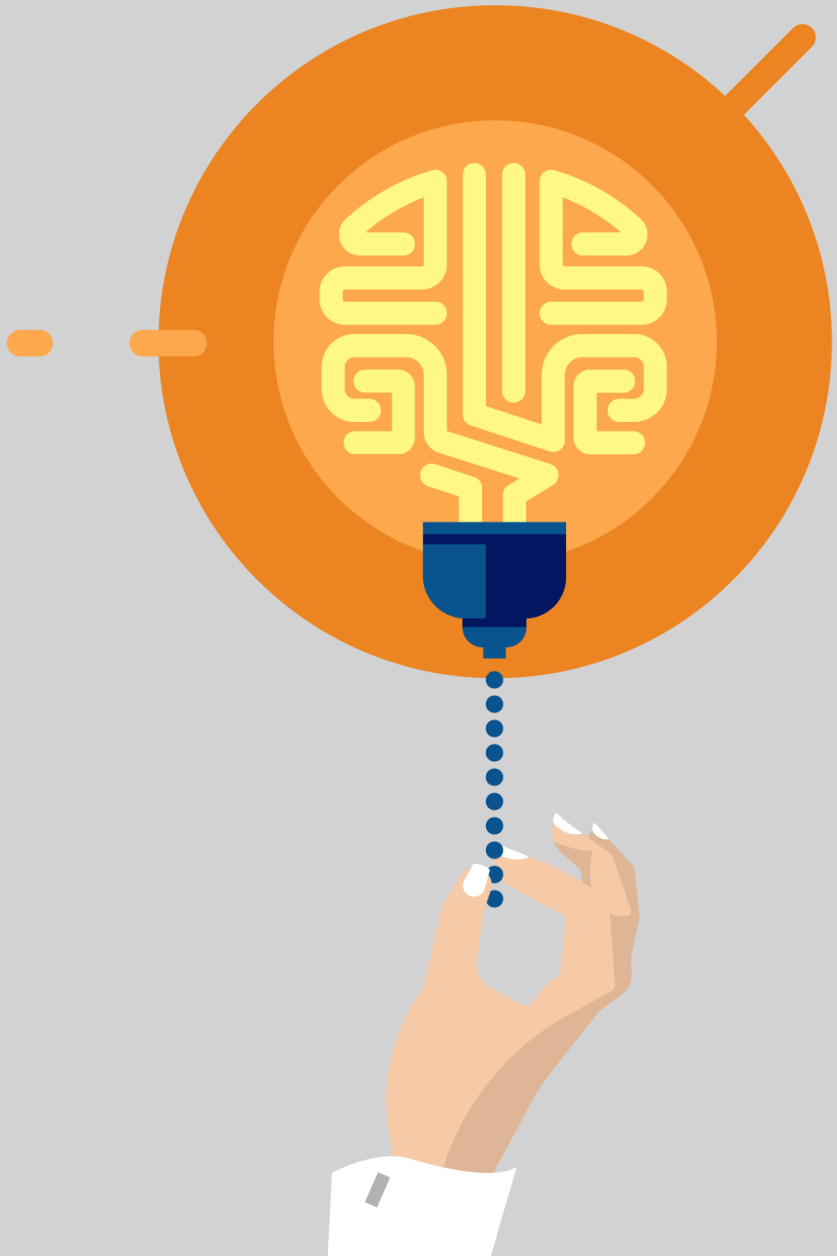
Miltos Allamanis

FOSDEM 2019

miltos1

https://miltos.allamanis.com

Microsoft Research Cambridge



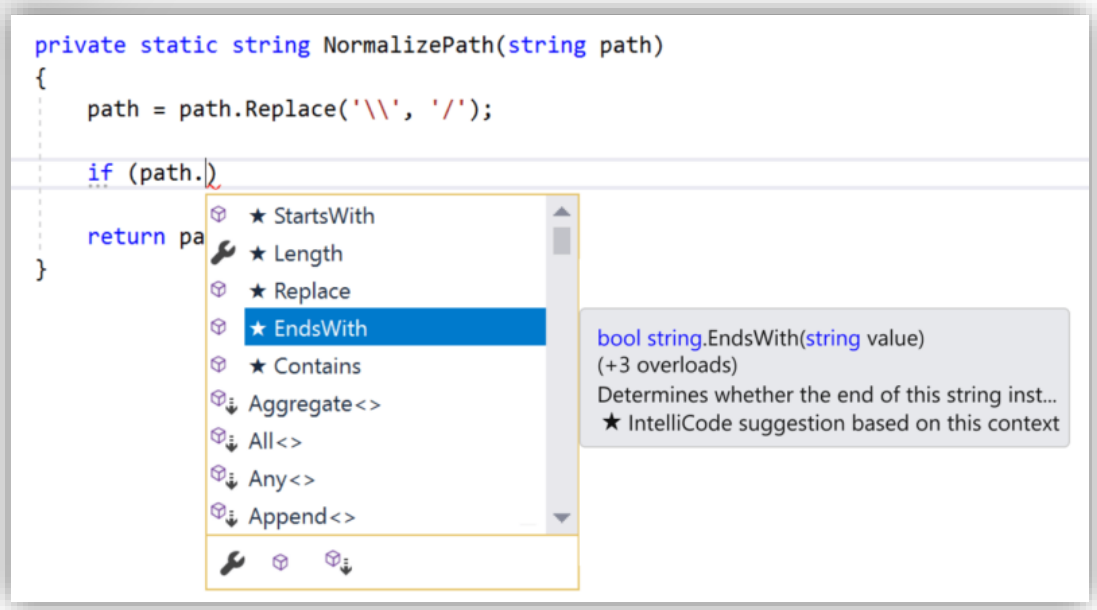
Source Code is Bimodal

Code Autocompletion

```
Text text = new Text(parent, SWT.NONE);
text.
```

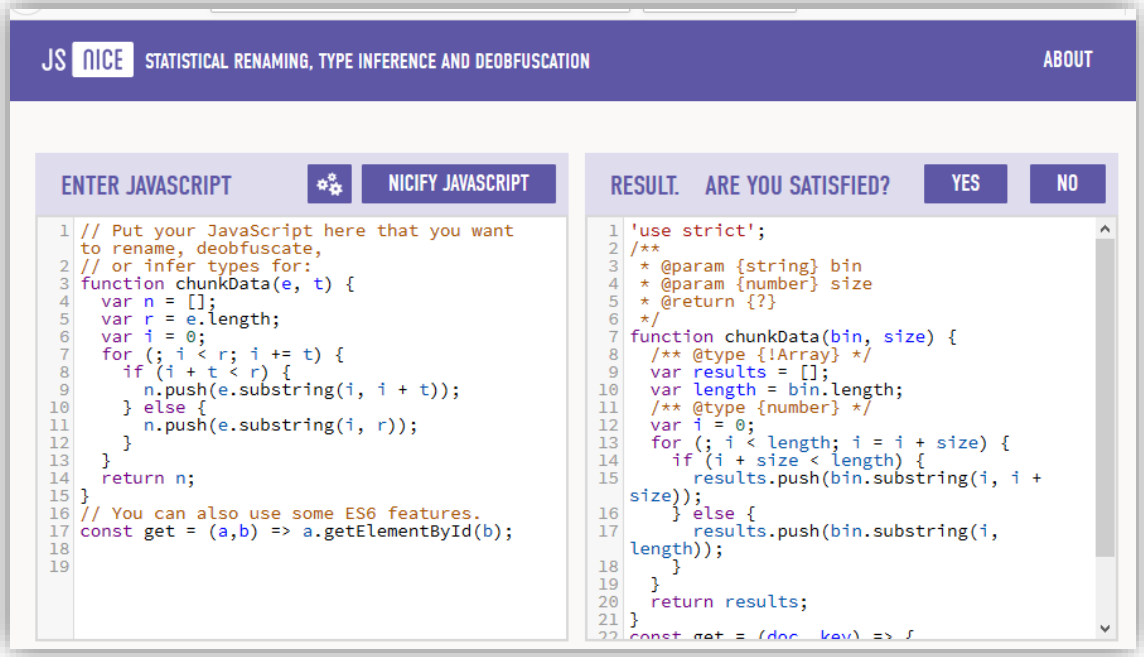
setLayoutData(Object layoutData) : void - Control - 86%
setText(String string) : void - Text - 51%
addModifyListener(ModifyListener listener) : void - Text - 31%
getText() : String - Text - 12%
setEnabled(boolean enabled) : void - Control - 11%
setFont(Font font) : void - Control - 8%
Press '^Space' to show Adaptive Template Proposals (Code Recommenders)

http://www.eclipse.org/recommenders/



https://visualstudio.microsoft.com/services/intellicode/

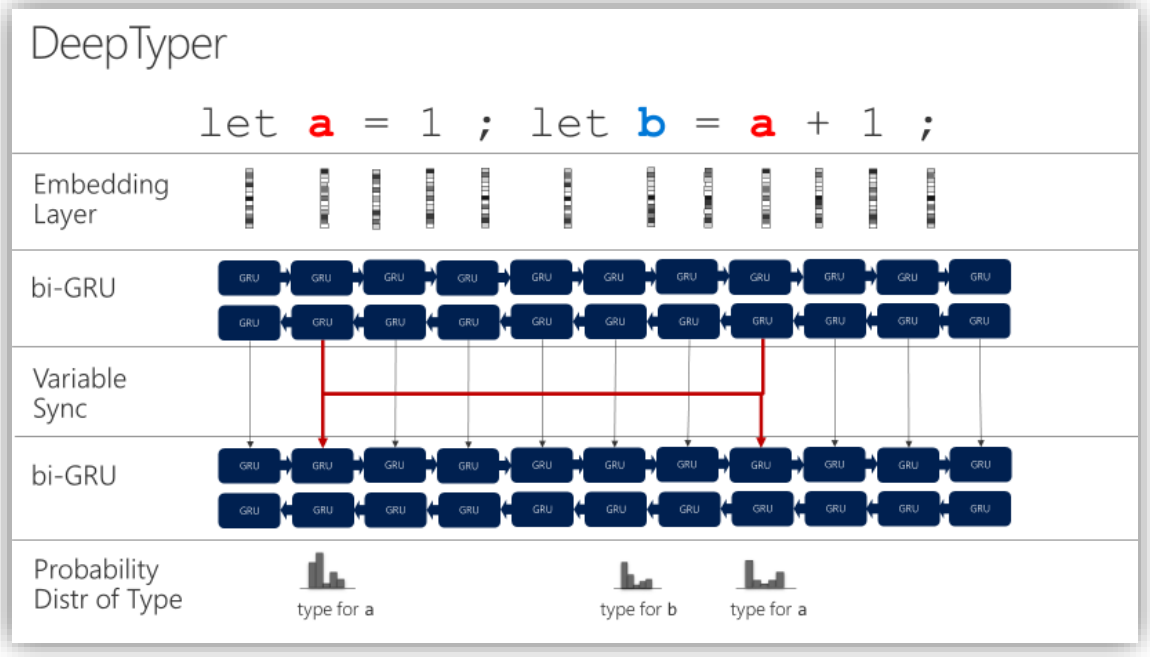
Predicting Types



Predicting Program Properties from Code

V. Raychev, M. Vechev, A. Krause. 2015

http://jsnice.org/



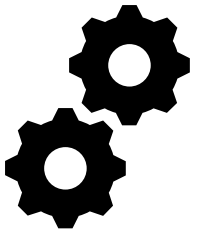
Deep Learning Type Inference

V. Hellendoorn, C. Bird, E.T. Barr, M. Allamanis. 2018

Inferring Type Refinements

Dash *et al.* 2018 “RefiNym: Using Names to Refine Types”

Conceptual Types



Defined Types

“a password” → `string password;`

“a JSON string” → `string data = Json.Load();`

Latent; we don’t observe in the *conceptual* types.

Defined explicitly by the programmer.

Argument Swapping

Type	Parameter	Original argument	Correct argument
Duration	responseTTLDuration	frequencyCapDuration	responseTTLDuration
Duration	frequencyCapDuration	responseTTLDuration	frequencyCapDuration
List<A>	slotResponse	slotResponse	slotResponse

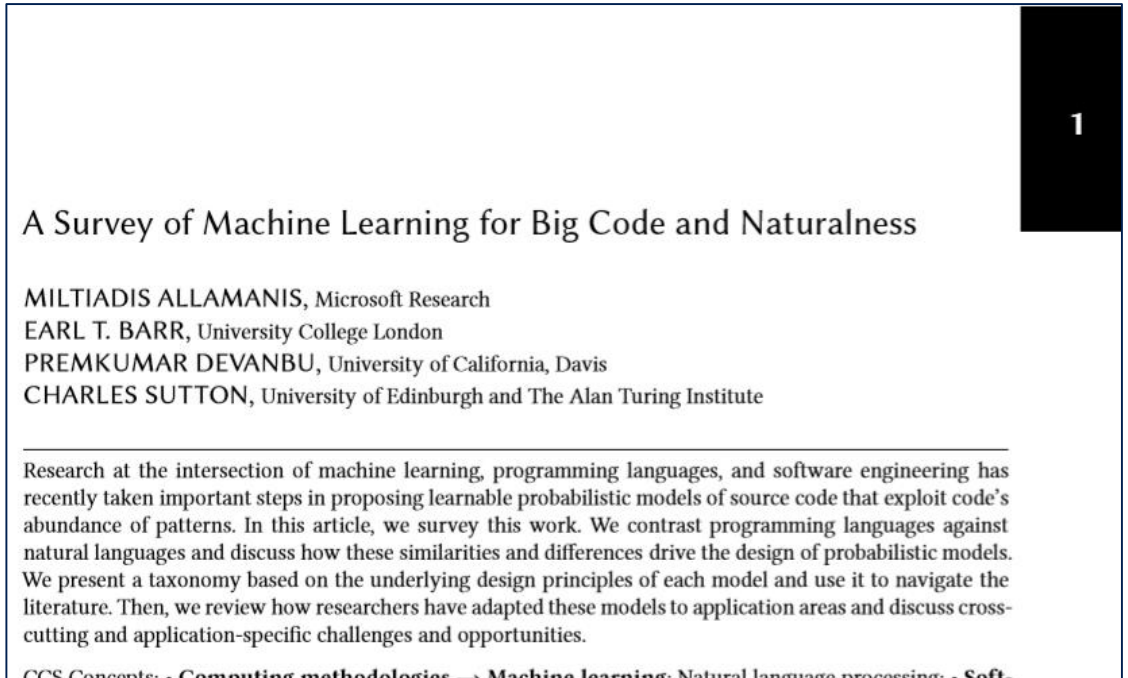
Builder	builder	builder	builder
boolean	isTransposed	isTransposed	isTransposed
int	startColumnIndex	a.getStartColumnIndex()	0
int	endColumnIndex	a.getEndColumnIndex()	rows.size()
int	startRow	0	a.getStartColumnIndex()
int	endRow	rows.size()	a.getEndColumnIndex()

Rice *et al.* 2017 “Detecting Argument Selection Defects”

Research in ML+Code

- Infer latent intent
- Ambiguous information
- Learned heuristics

https://ml4code.github.io



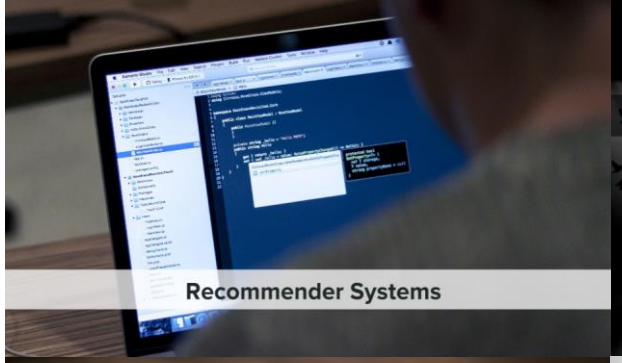
1

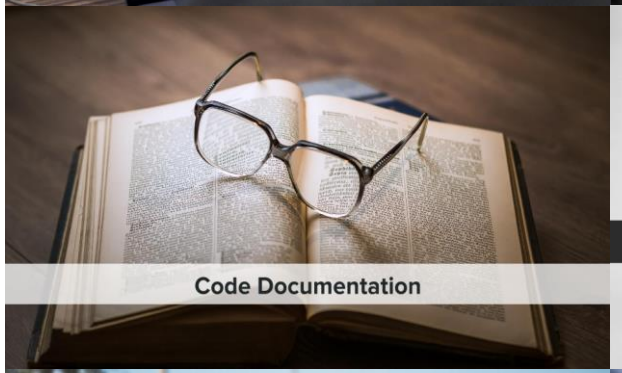
A Survey of Machine Learning for Big Code and Naturalness

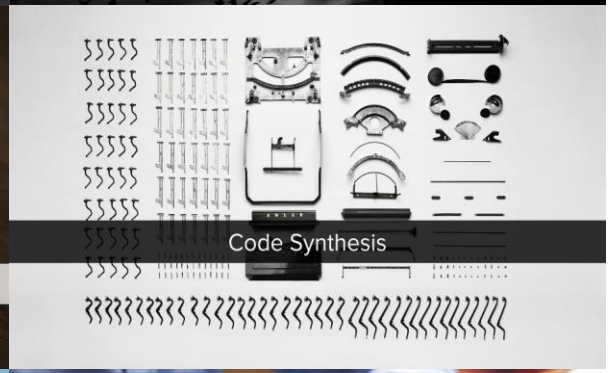
MILTIADIS ALLAMANIS, Microsoft Research
EARL T. BARR, University College London
PREMKUMAR DEVANBU, University of California, Davis
CHARLES SUTTON, University of Edinburgh and The Alan Turing Institute

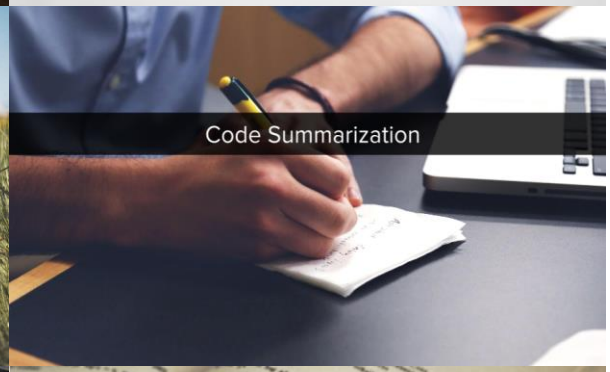
Research at the intersection of machine learning, programming languages, and software engineering has recently taken important steps in proposing learnable probabilistic models of source code that exploit code's abundance of patterns. In this article, we survey this work. We contrast programming languages 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.

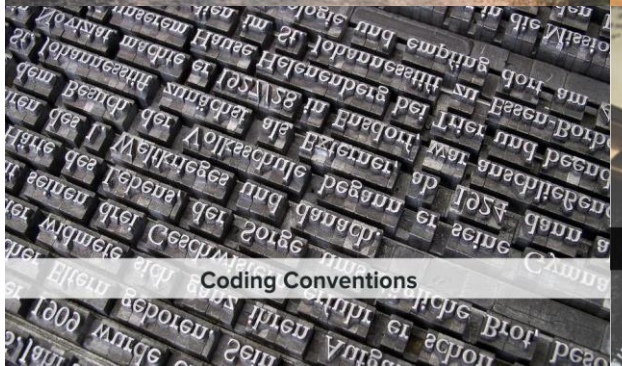
CCS Concepts: • Computing methodologies → Machine learning; Natural language processing; • Soft-

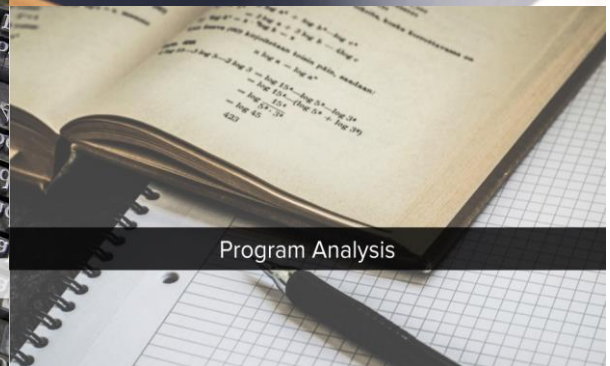












Microsoft

Detecting Variable Misuse Bugs

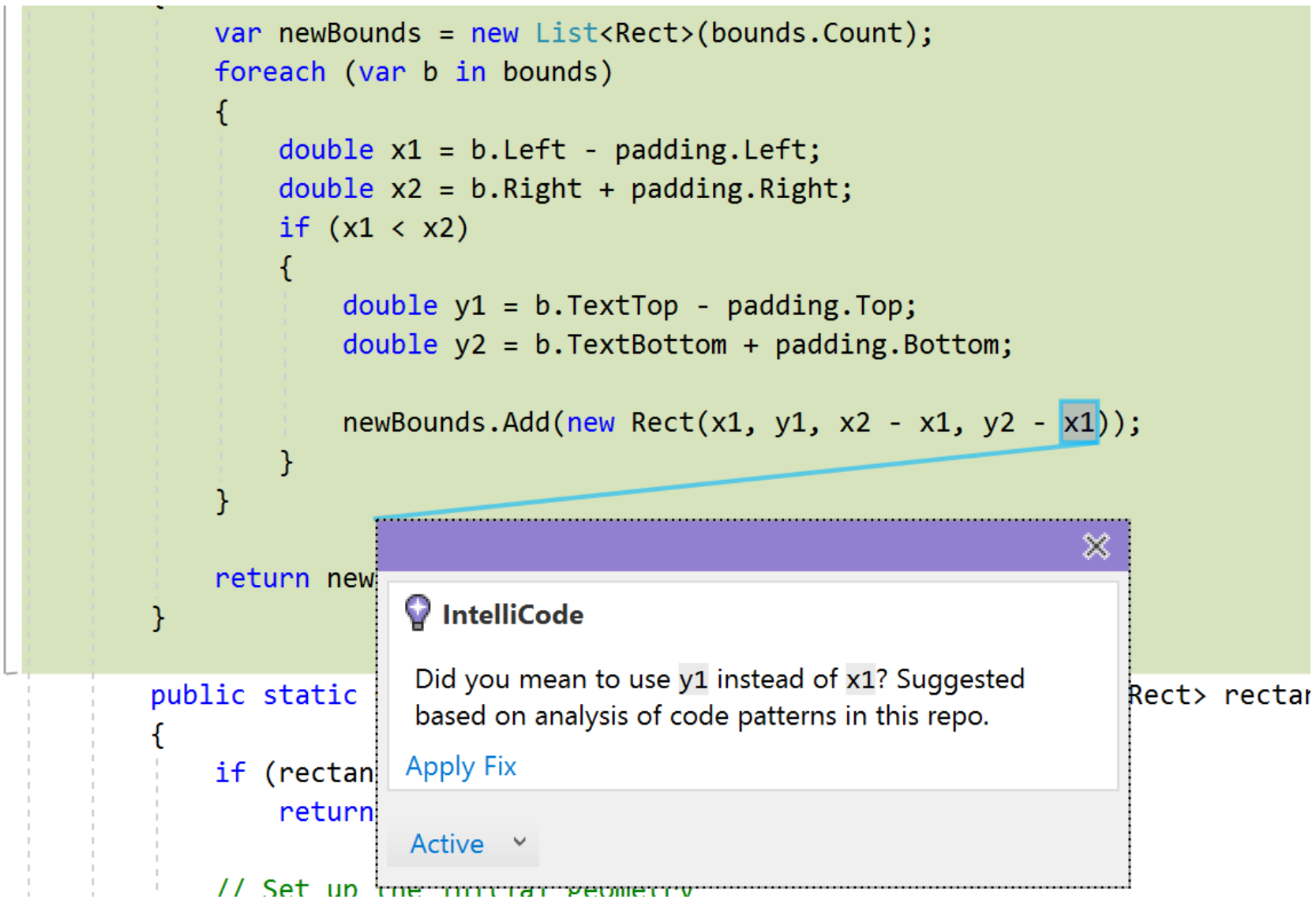
By representing source code as graphs

Target Task

```
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`

Not easy to catch with static analysis tools.

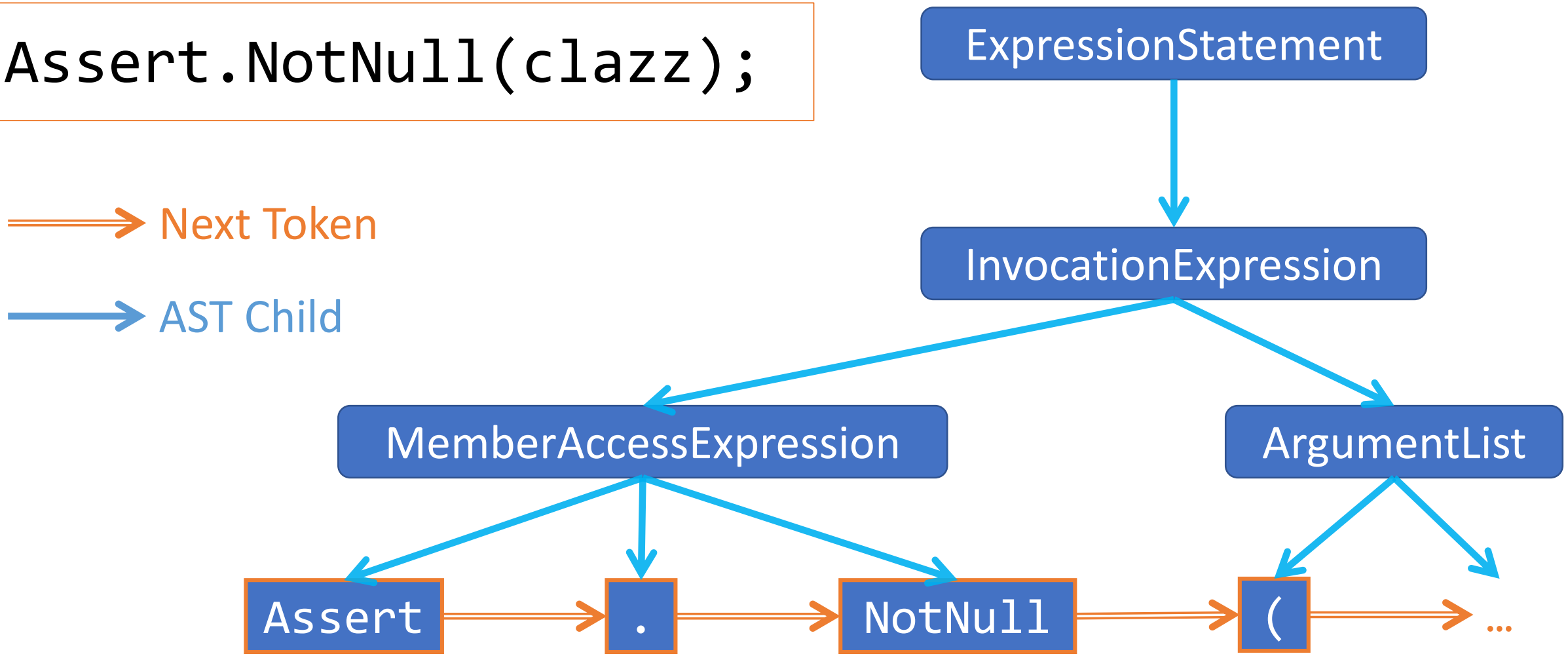

var newBounds = new List<Rect>(bounds.Count);
foreach (var b in bounds)
{
double x1 = b.Left - padding.Left;
double x2 = b.Right + padding.Right;
if (x1 < x2)
{
double y1 = b.TextTop - padding.Top;
double y2 = b.TextBottom + padding.Bottom;
newBounds.Add(new Rect(x1, y1, x2 - x1, y2 - x1));
}
}
return new
}
public static
Rect> rectar
{
if (rectan
return
// Set up
IntelliCode
Did you mean to use y1 instead of x1? Suggested based on analysis of code patterns in this repo.
Apply Fix
Active

Programs as Graphs: Key Idea

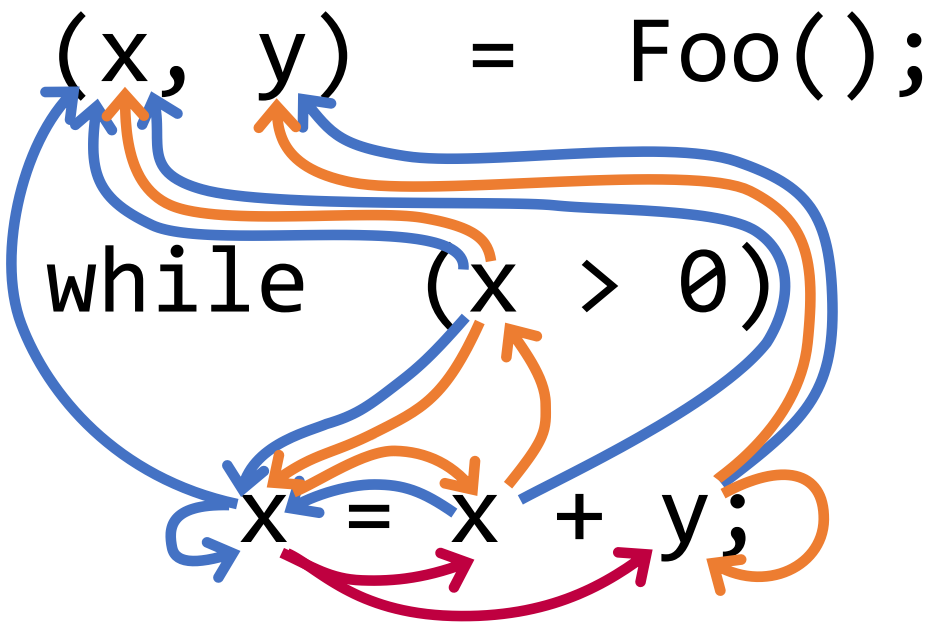
```
int SumPositive(int[] arr, int lim) {
  int sum = 0;
  for (int i = 0; i < lim; i++)
    if (arr[i] > 0)
        sum += arr[i];

  return sum;
}
```

Programs as Graphs: Syntax

```
Assert.NotNull(clazz);
```



Programs as Graphs: Data Flow



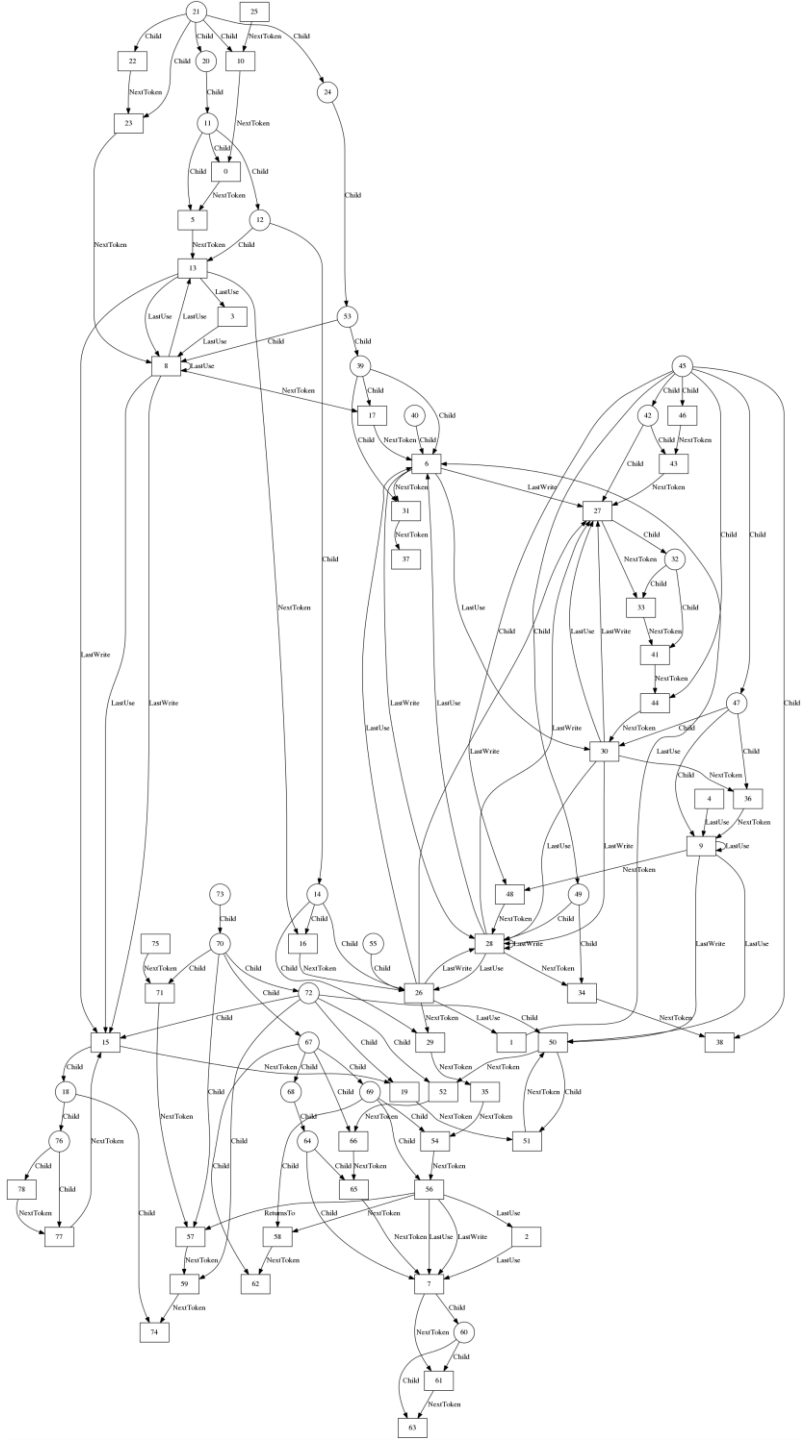
Last Write

Last Use

Computed From

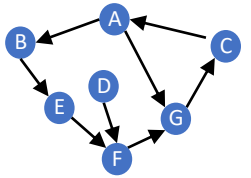
Programs as Graphs

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



~900 nodes/graph ~8k edges/graph

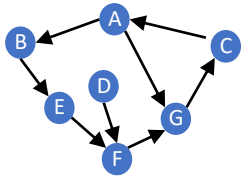
Graph Representation for Variable Misuse




```
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 );

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

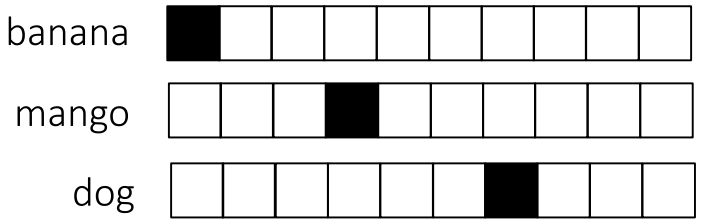
first

clazz

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

$$\left(\boldsymbol{h}_T^{SLOT}\right)^T \boldsymbol{h}_T^{first} \gg \left(\boldsymbol{h}_T^{SLOT}\right)^T \boldsymbol{h}_T^{clazz}$$

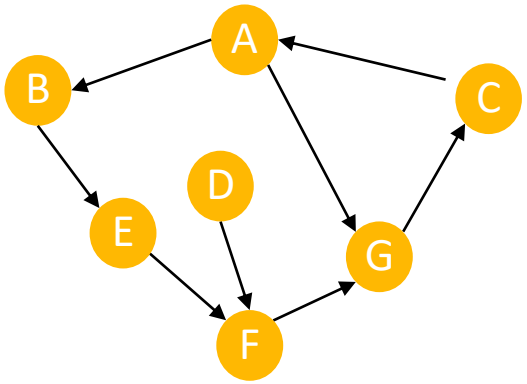
Vector Space Representations



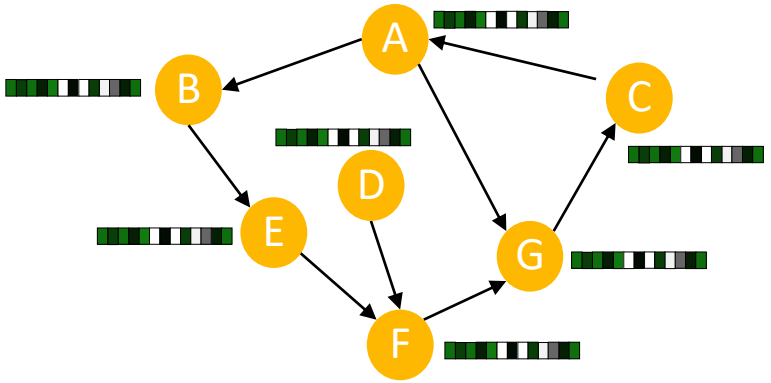
Local representation

Distributed representation

Graph Neural Networks



Graph Representation
of Problem

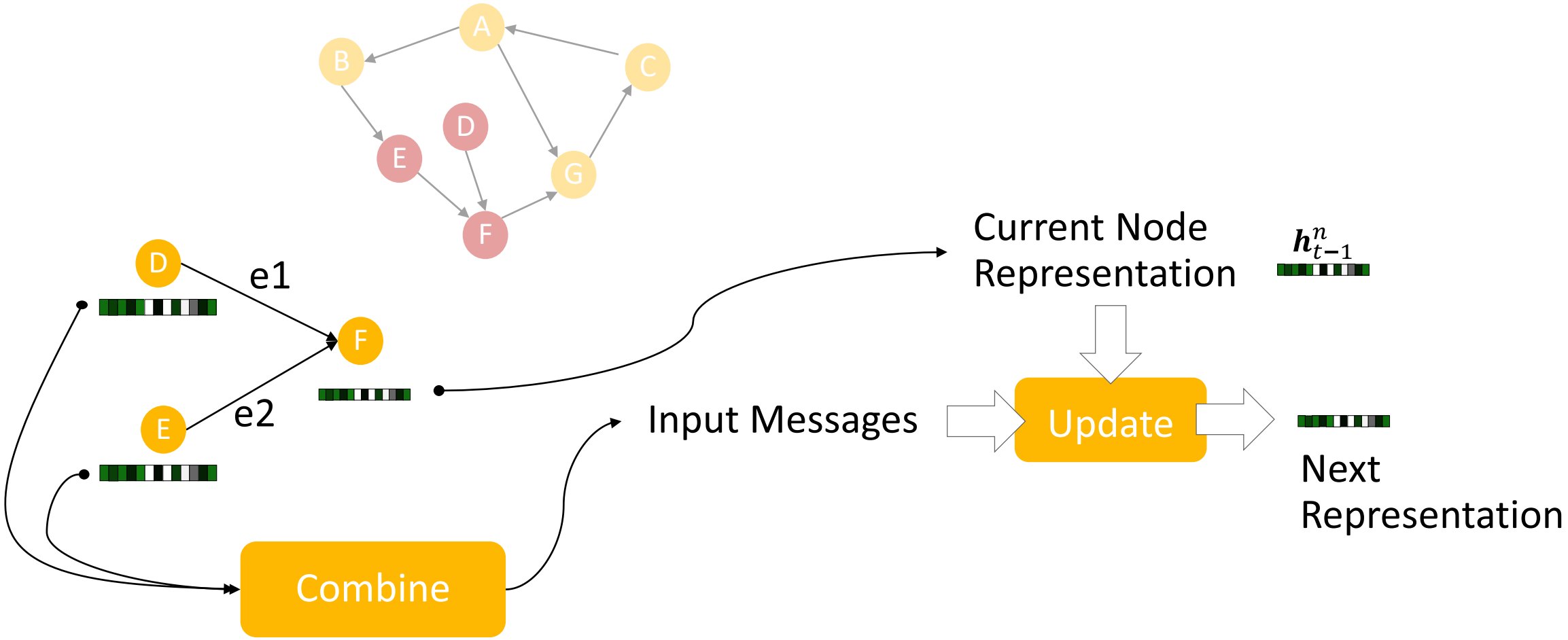


Initial Representation
of each node

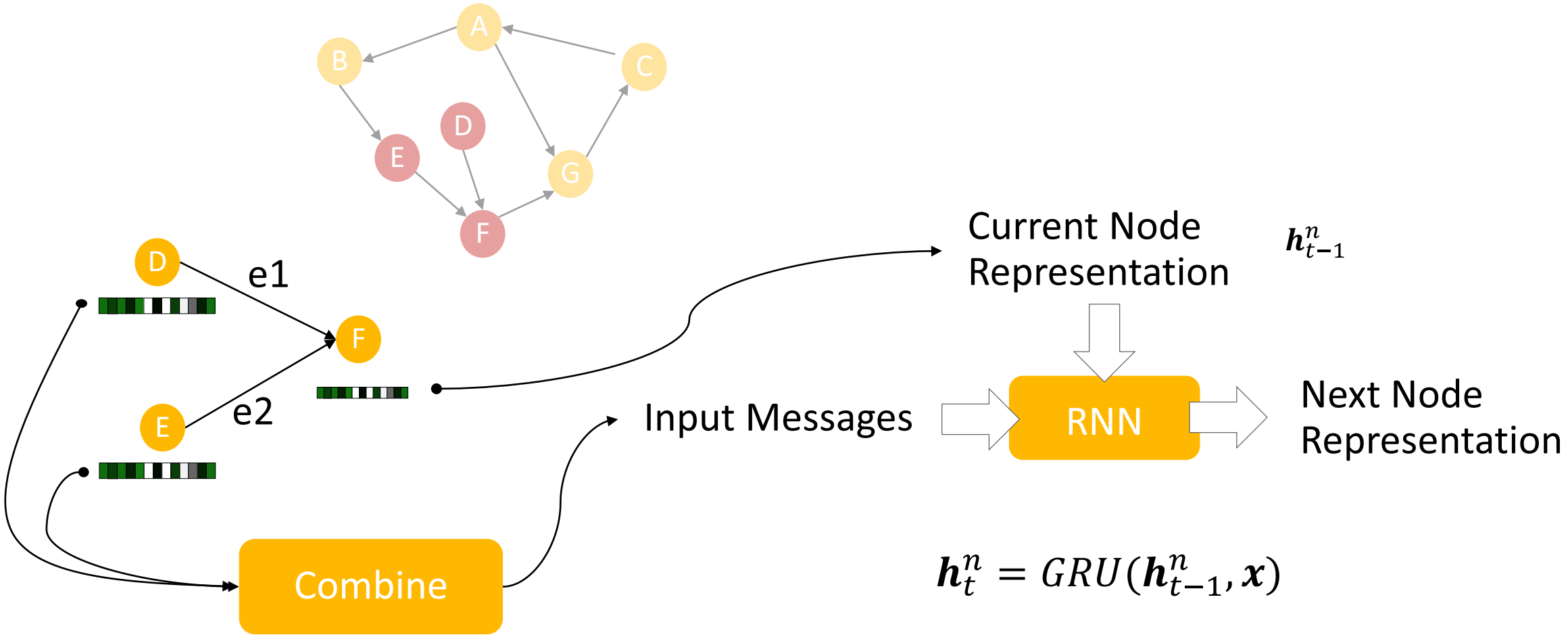
Li et al (2015). Gated Graph Sequence Neural Networks.

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

Graph Neural Networks: Message Passing



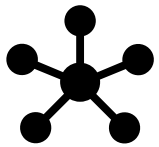
Graph Neural Networks: Message Passing



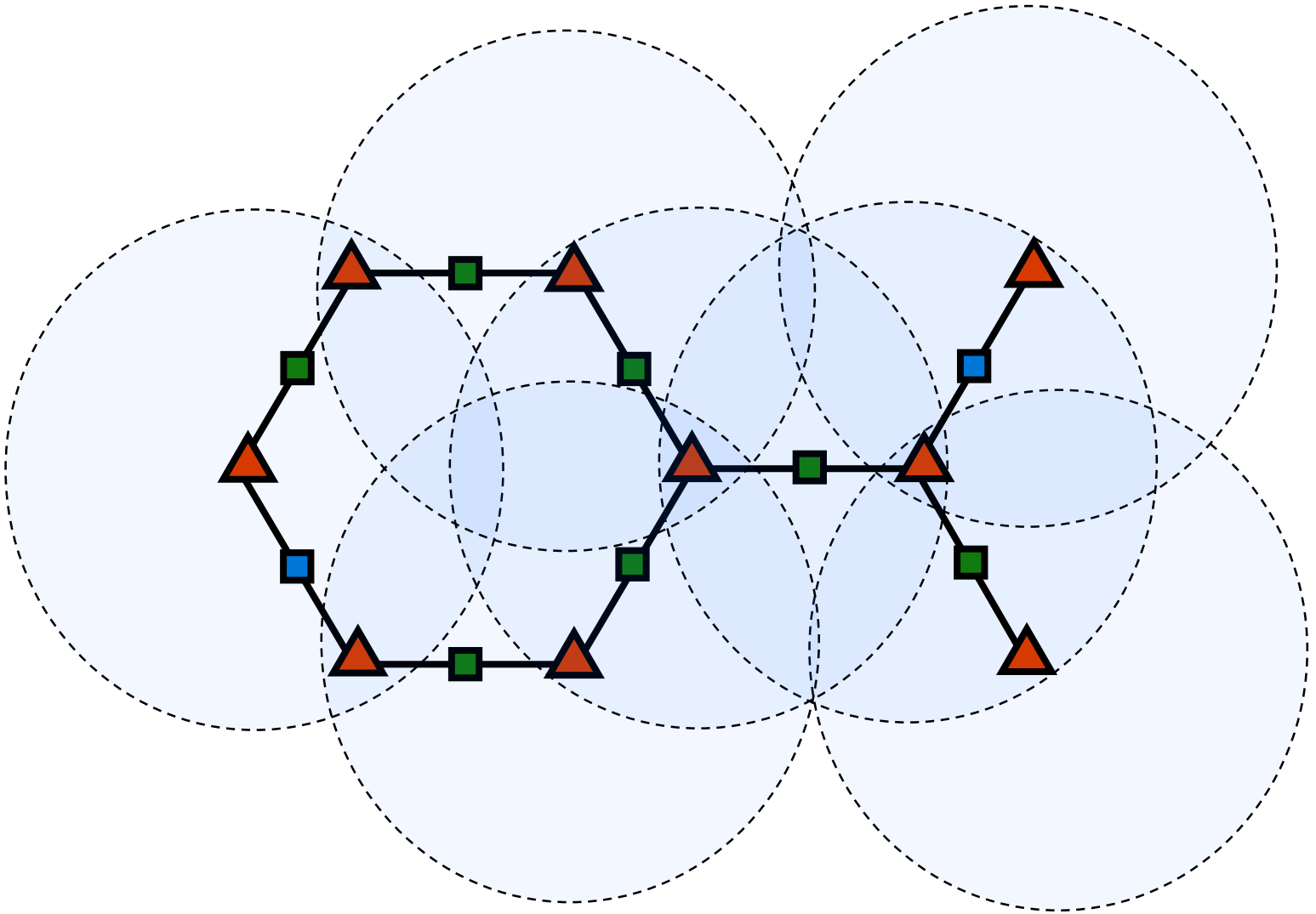
$$\boldsymbol{h}_t^n = GRU(\boldsymbol{h}_{t-1}^n, \boldsymbol{x})$$

$$\boldsymbol{x} = \sum_{n' \in neig(n)} E_{\tau(n' \to n)} \boldsymbol{h}_{t-1}^{n'} + \boldsymbol{b}_{\tau(n' \to n)}$$

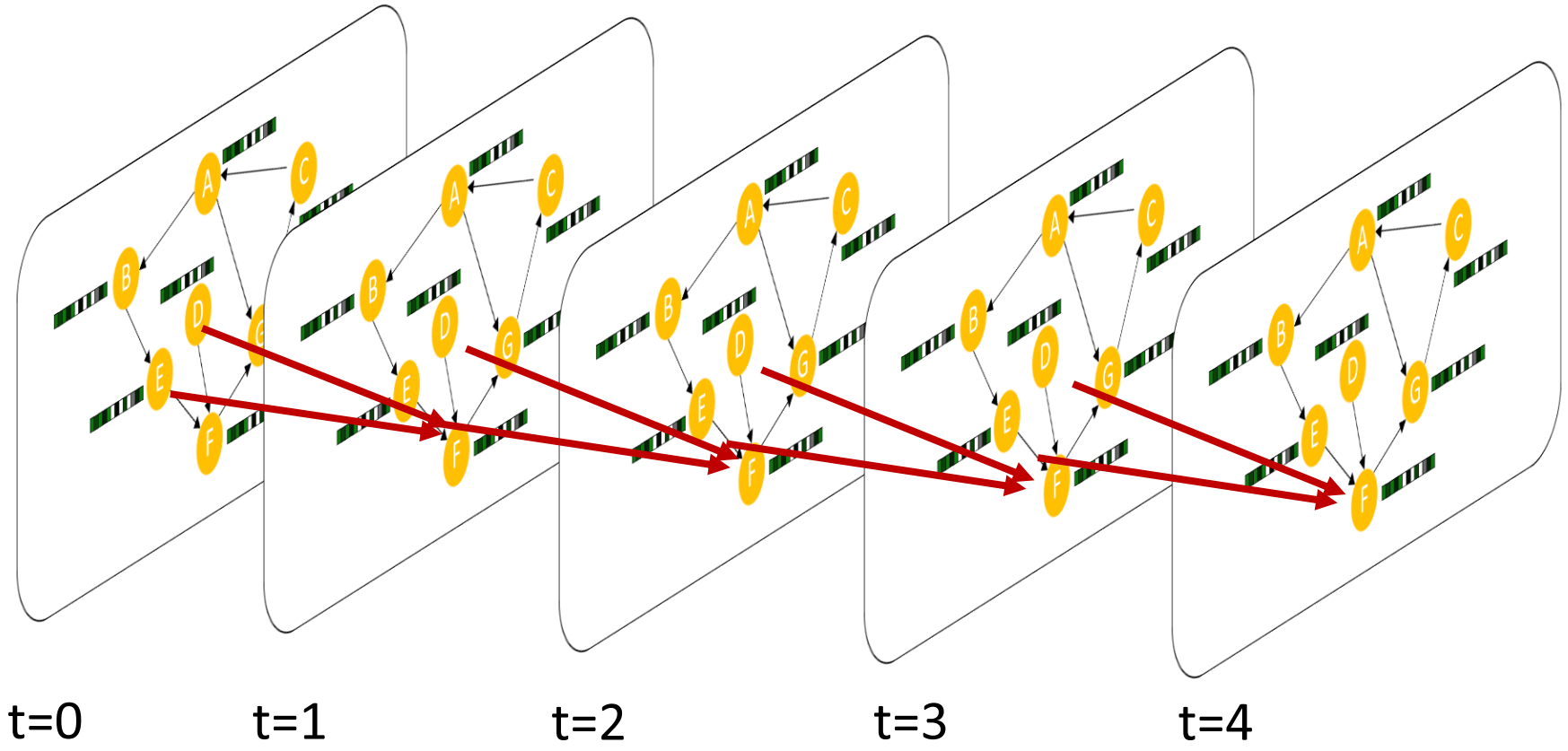
Li et al (2015). Gated graph sequence neural networks.



Graph Neural Networks: Unrolling

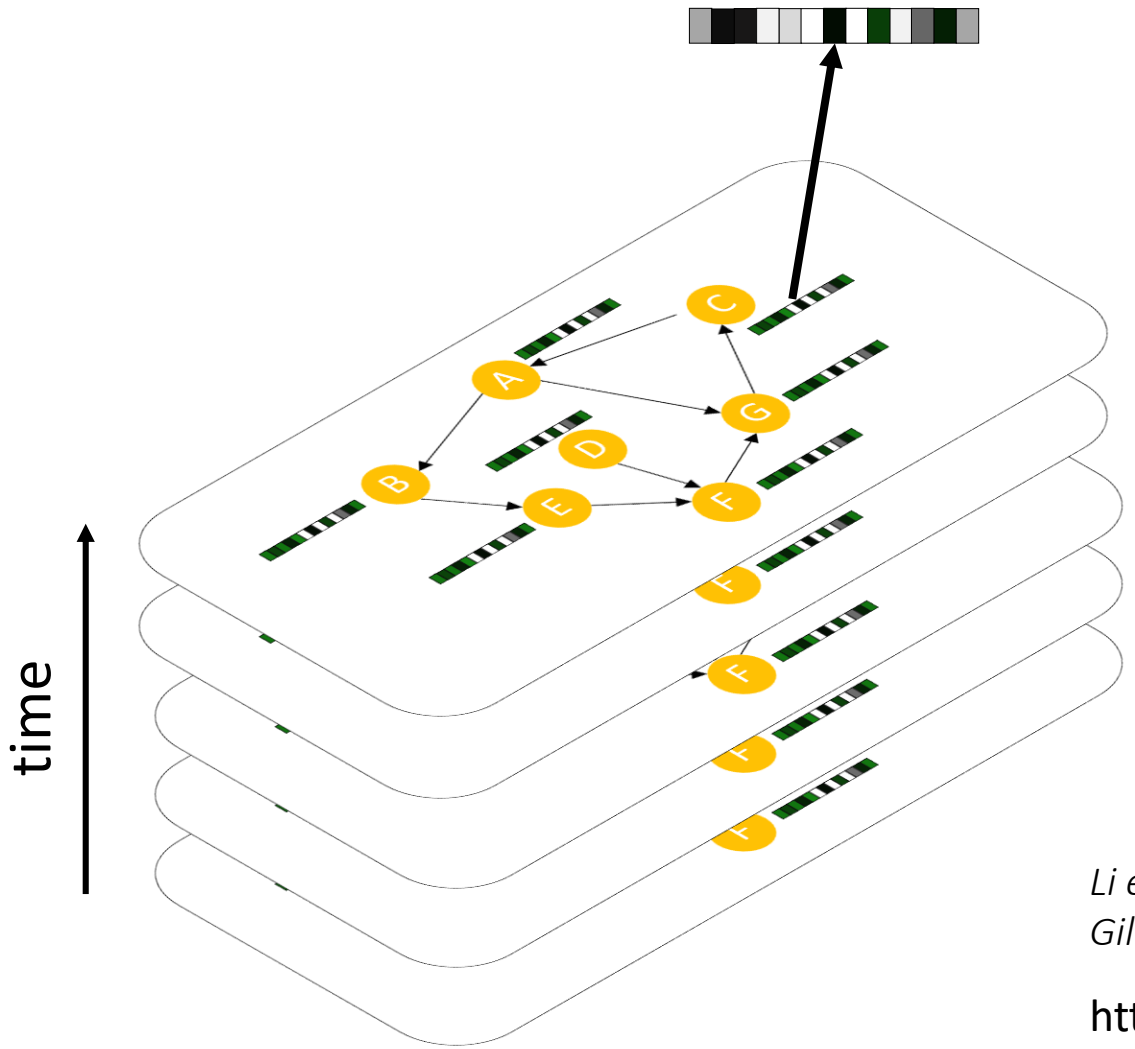


Graph Neural Networks: Unrolling



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

Quantitative Results – **Variable Misuse**

Accuracy (%)	BiGRU	BiGRU+Dataflow	GGNN
Seen Projects	50.0	73.7	**85.5**

Seen Projects: 24 F/OSS C# projects (2060 kLOC): Used for train and test

3.8 type-correct alternative variables per slot (median 3, σ= 2.6)

Quantitative Results – **Variable Misuse**

Accuracy (%)	BiGRU	BiGRU+Dataflow	GGNN
Seen Projects	50.0	73.7	**85.5**
Unseen Projects	28.9	60.2	**78.2**

Seen Projects: 24 F/OSS C# projects (2060 kLOC): Used for train and test
Unseen Projects: 3 F/OSS C# projects (228 kLOC): Used only for test
3.8 type-correct alternative variables per slot (median 3, σ= 2.6)

```
bool TryFindGlobalDirectivesFile(string baseDirectory, string fullPath, out string path) {
    baseDirectory = baseDirectory.TrimEnd(Path.DirectorySeparatorChar);
    var directivesDirectory = Path.GetDirectoryName(          )
                                  .TrimEnd(Path.DirectorySeparatorChar);
    while (directivesDirectory != null && directivesDirectory.Length >= baseDirectory.Length)
{
        path = Path.Combine(directivesDirectory, GlobalDirectivesFileName);
        if (File.Exists(path)) return true;

        directivesDirectory = Path.GetDirectoryName(directivesDirectory)
                                  .TrimEnd(Path.DirectorySeparatorChar);
    }
    path = null;
    return false;
}
```

What the model sees…

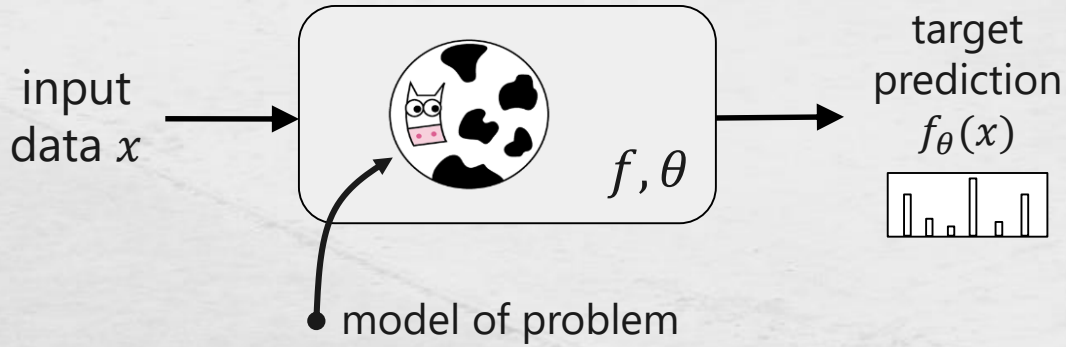
UI/UX

ML Capabilities

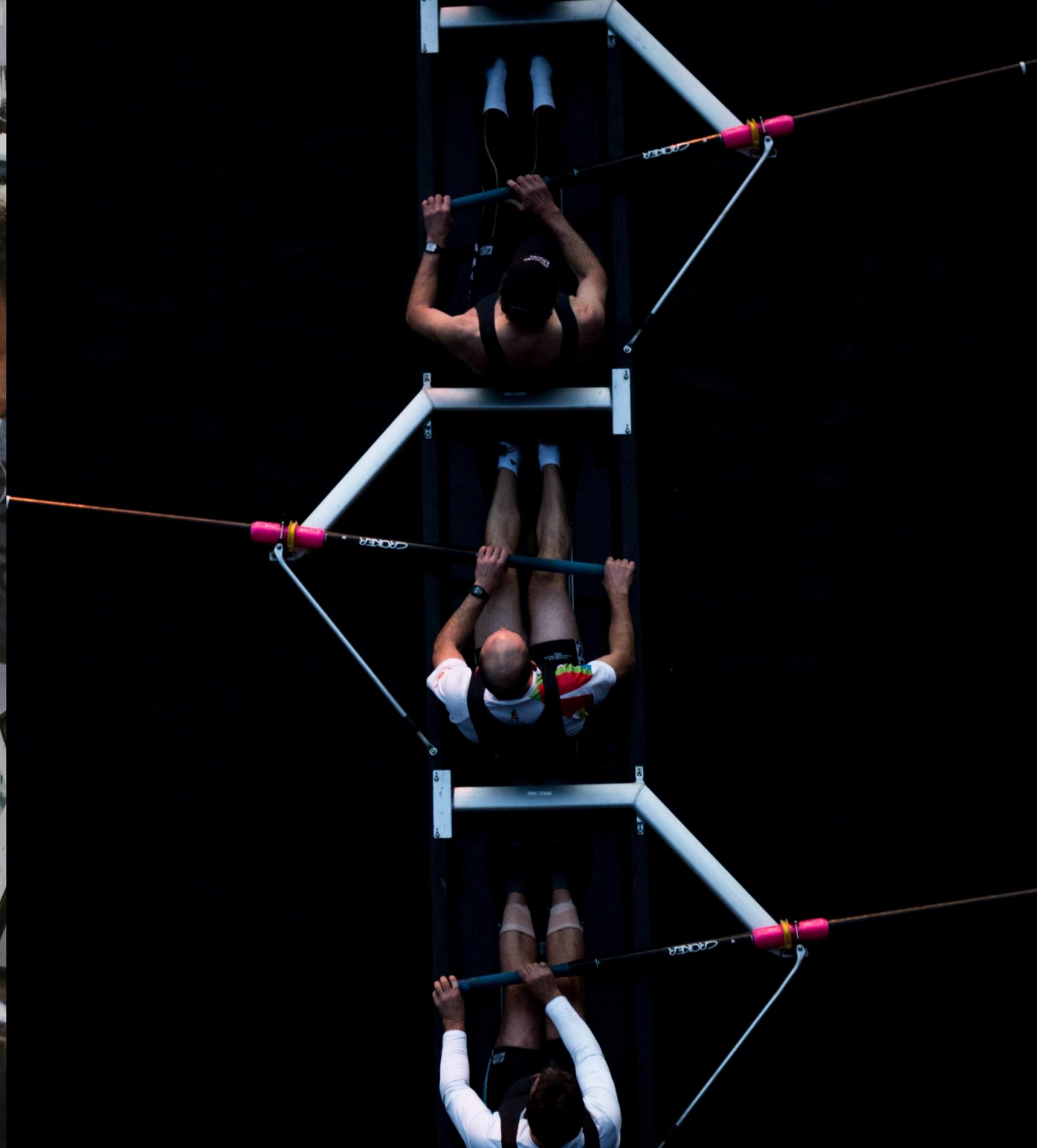
Metrics

Low resources

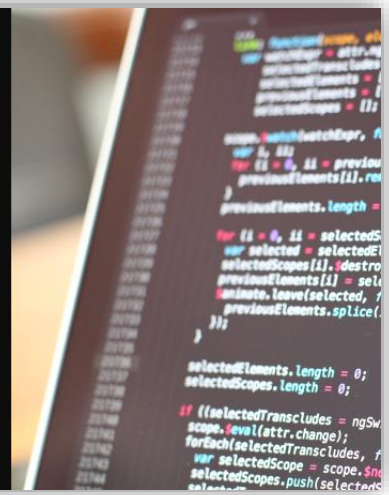
Learning Signals



- Given dataset $\{(x_1, y_0), \ldots, (x_N, y_N)\}$
- Minimize Loss $\mathcal{L}(\theta) = \frac{1}{N}\sum_i L(f_\theta(x_i), y_i)$



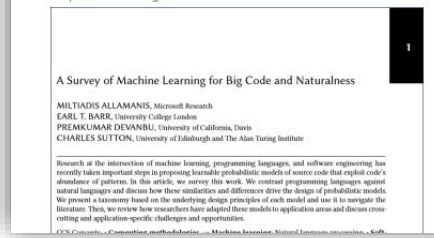
Source Code is Bimodal



Research in ML+Code

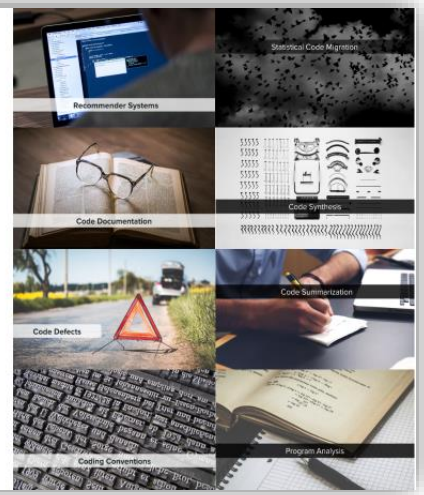
- Infer latent intent
- Ambiguous information
- Learned heuristics

https://ml4code.github.io



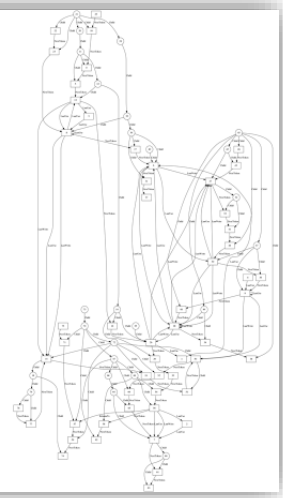
A Survey of Machine Learning for Big Code and Naturalness

MILTIADIS 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



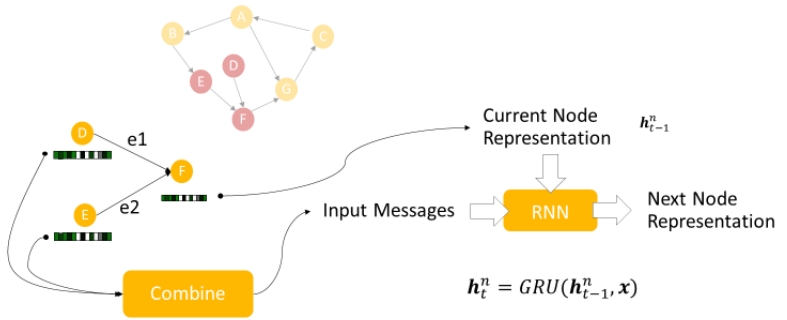
Programs as Graphs

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



~900 nodes/graph ~8k edges/graph

Graph Neural Networks: Message Passing



$$h^n_t = GRU(h^n_{t-1}, x)$$

$$x = \sum_{n' \in neig(n)} E_{\tau(n' \to n)} h^{n'}_{t-1} + b_{\tau(n' \to n)}$$

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

Microsoft

@miltos1

miltos.allamanis.com