Microsoft

Understanding Code using Natural Language & Graph Neural Networks

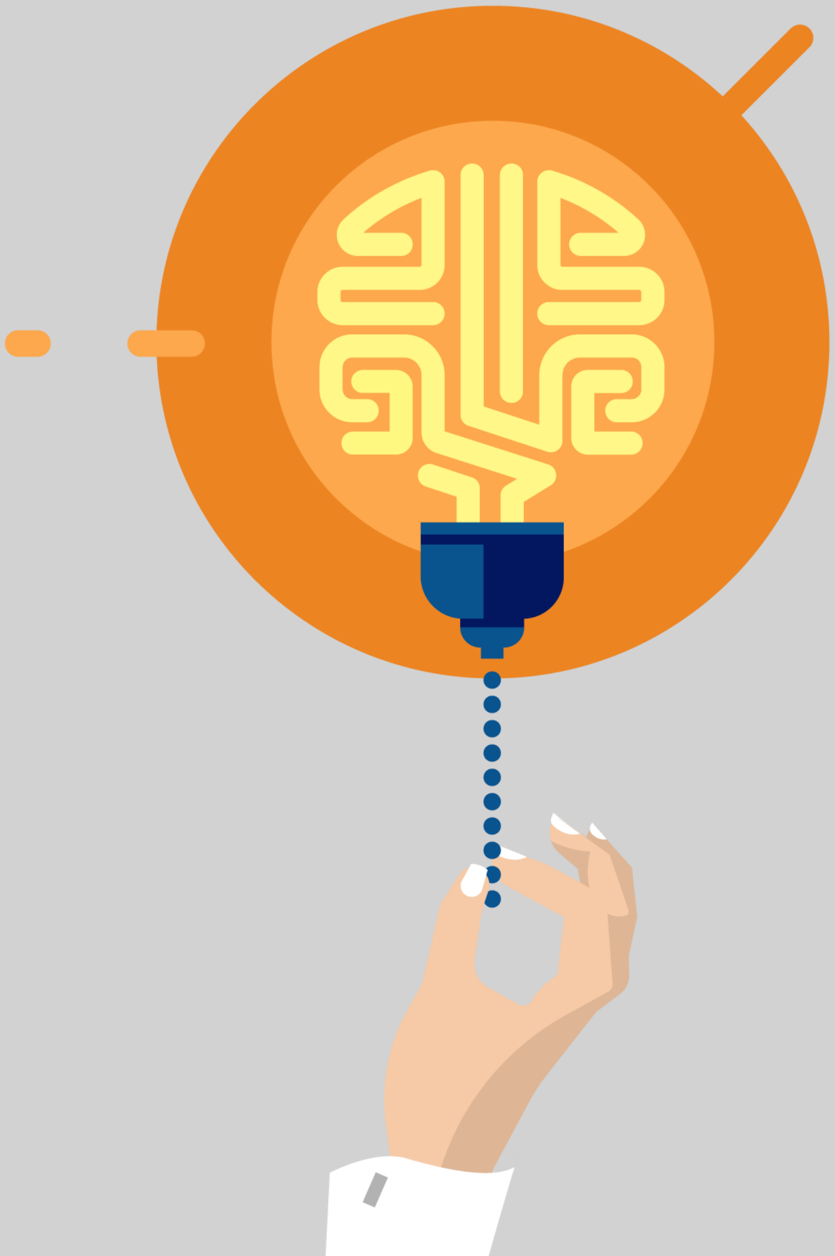
Miltos Allamanis

Microsoft Research, Cambridge

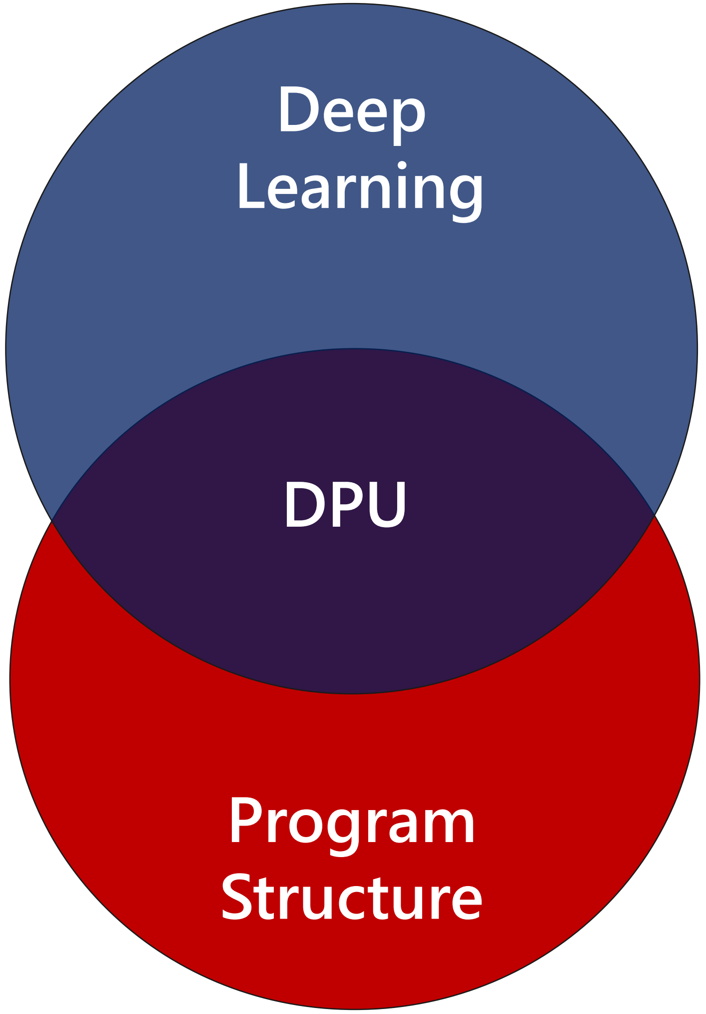
@miltos1

miltos.allamanis.com

Joint work with Marc Brockschmidt, Patrick Fernandes, Mahmoud Khademi, Hamel Husain, Ho-Hsiang Wu, Tiferet Gazit



Deep Program Understanding



- ✓ Understands images/language/speech
- ✓ Finds patterns in noisy data

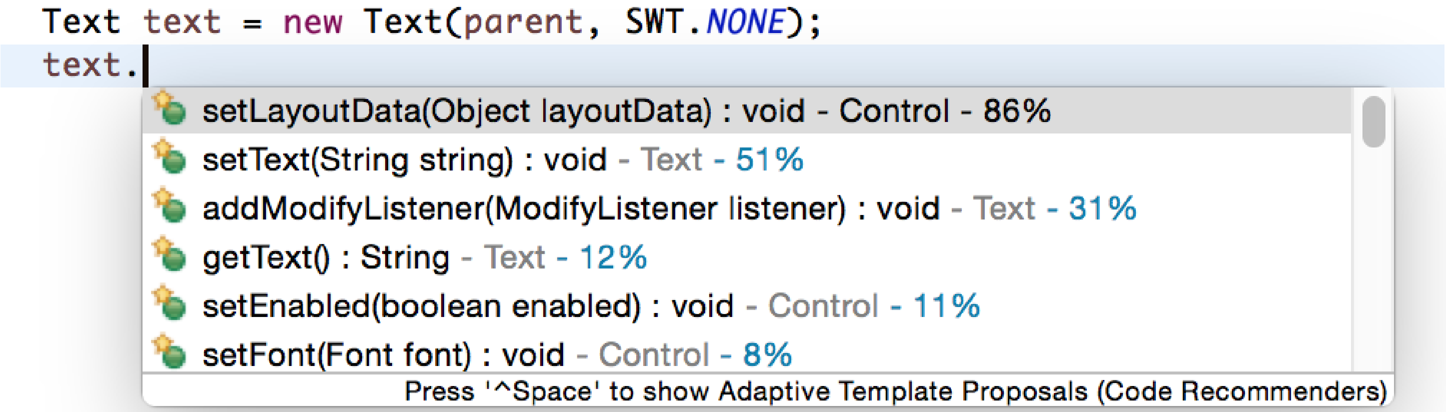
- Requires many samples
- Handling structured data is hard

- ✓ Interpretable
- ✓ Generalisation verifiable

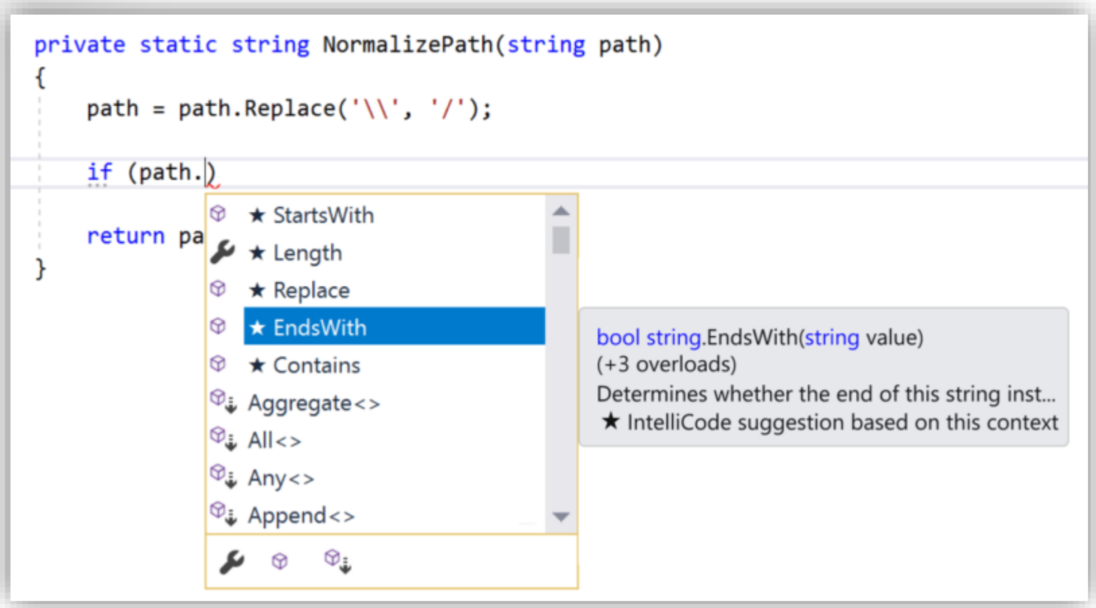
- Manual effort
- Limited to specialists

Source Code
and
Natural Language

Code Autocompletion



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



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

Argument Swapping

Declaration:

```
void foo(Duration responseTTLDuration, Duration frequencyCapDuration, List<A> slotResponse)
```

Invocation:

```
foo(frequencyCapDuration, responseTTLDuration, slotResponse)
```

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

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

Inferring Type Refinements

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.

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

Variable Misuse

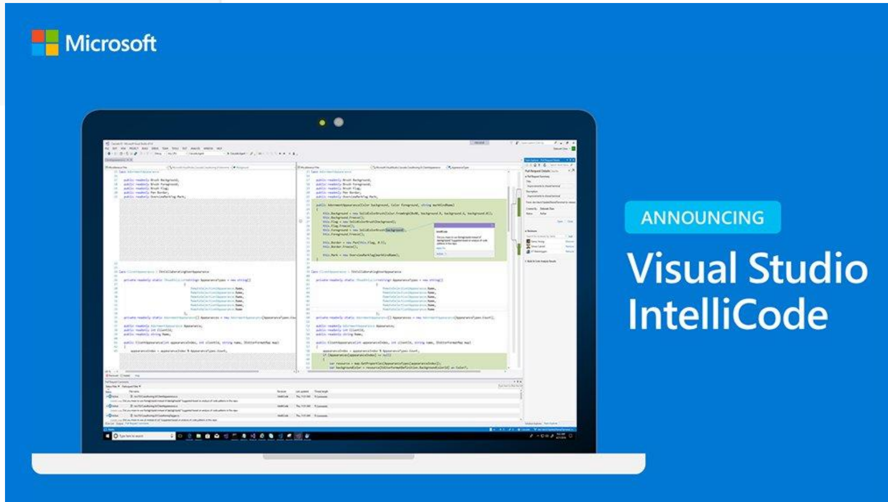
```
// Create or update the document.
var newDocument = await cosmosClient.UpsertDocumentAsync(cosmosDbCollectionUri, document);

if (updateRecord)
{
    logger.WriteLog($"Updated {existingDocument} to {newDocument}");
}
else
{
    logger.WriteLog($"Added {existingDocument}");
```

[redacted] Update 1 ♡ 1 Resolved

Based on this repo's code patterns, did you intend to use 'newDocument' (confidence 92%) rather than 'existingDocument` (confidence 7%) here? Review is recommended by Research bot's Variable Misuse analysis.

JK [redacted]

+1



Allamanis *et al.* “Learning to Represent Programs with Graphs”. 2018

Research in ML+Code

- Infer latent intent
- Ambiguous information

https://ml4code.github.io

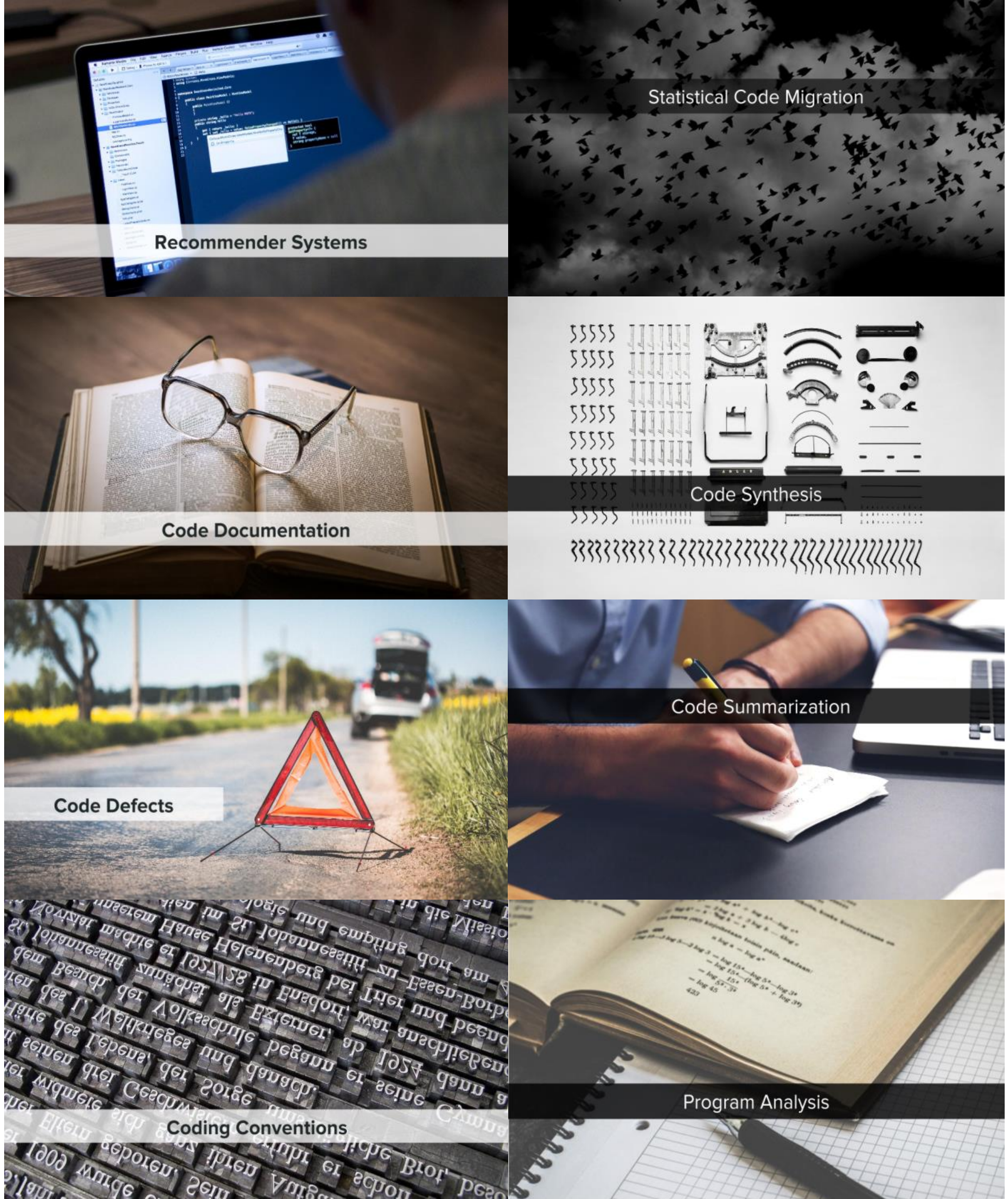
1

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-

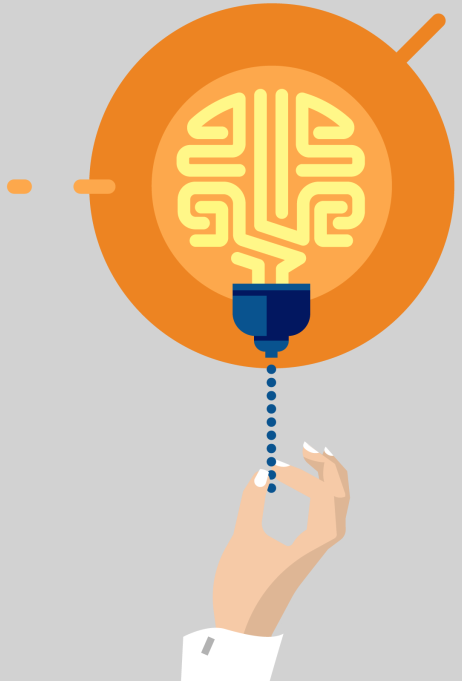


Microsoft

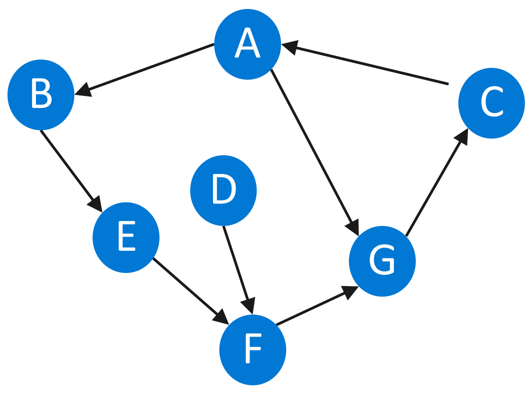
- Graph Neural Networks
- Understanding Code
- Structured Summarization
- Code Search

Graph Neural Networks

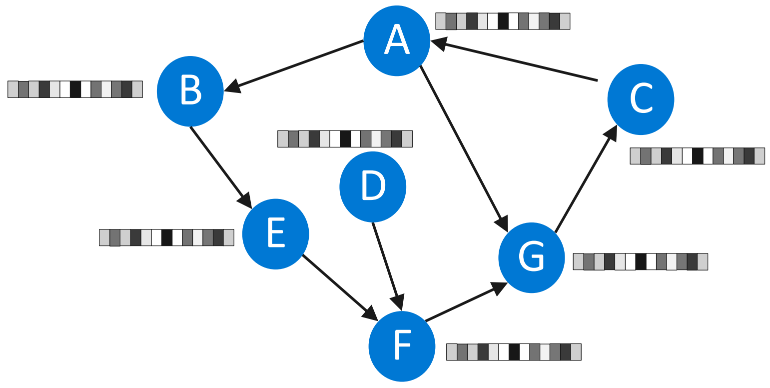
and Neural Message Passing



Graph Neural Networks



Graph Representation
of Problem

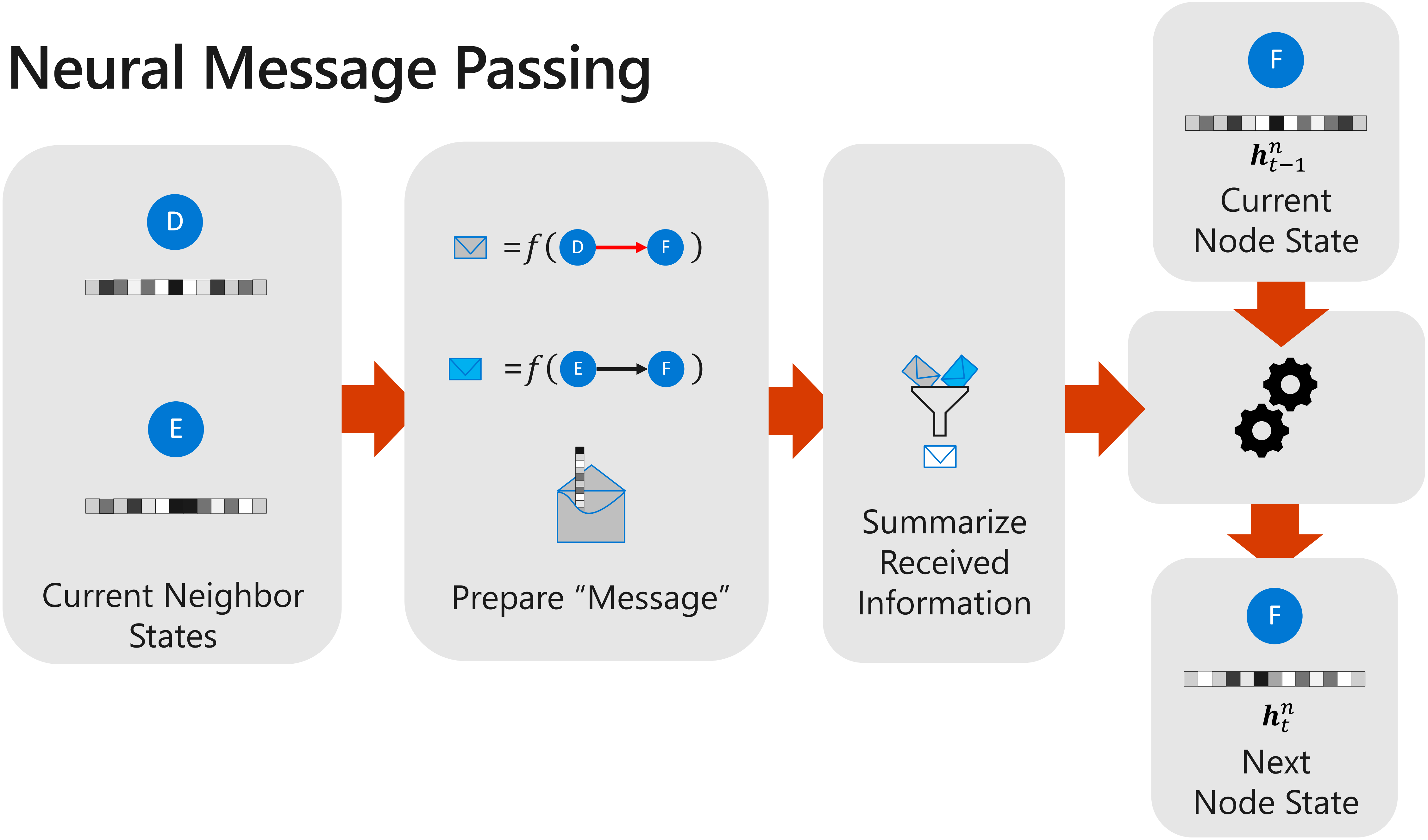


Initial Representation
of each node

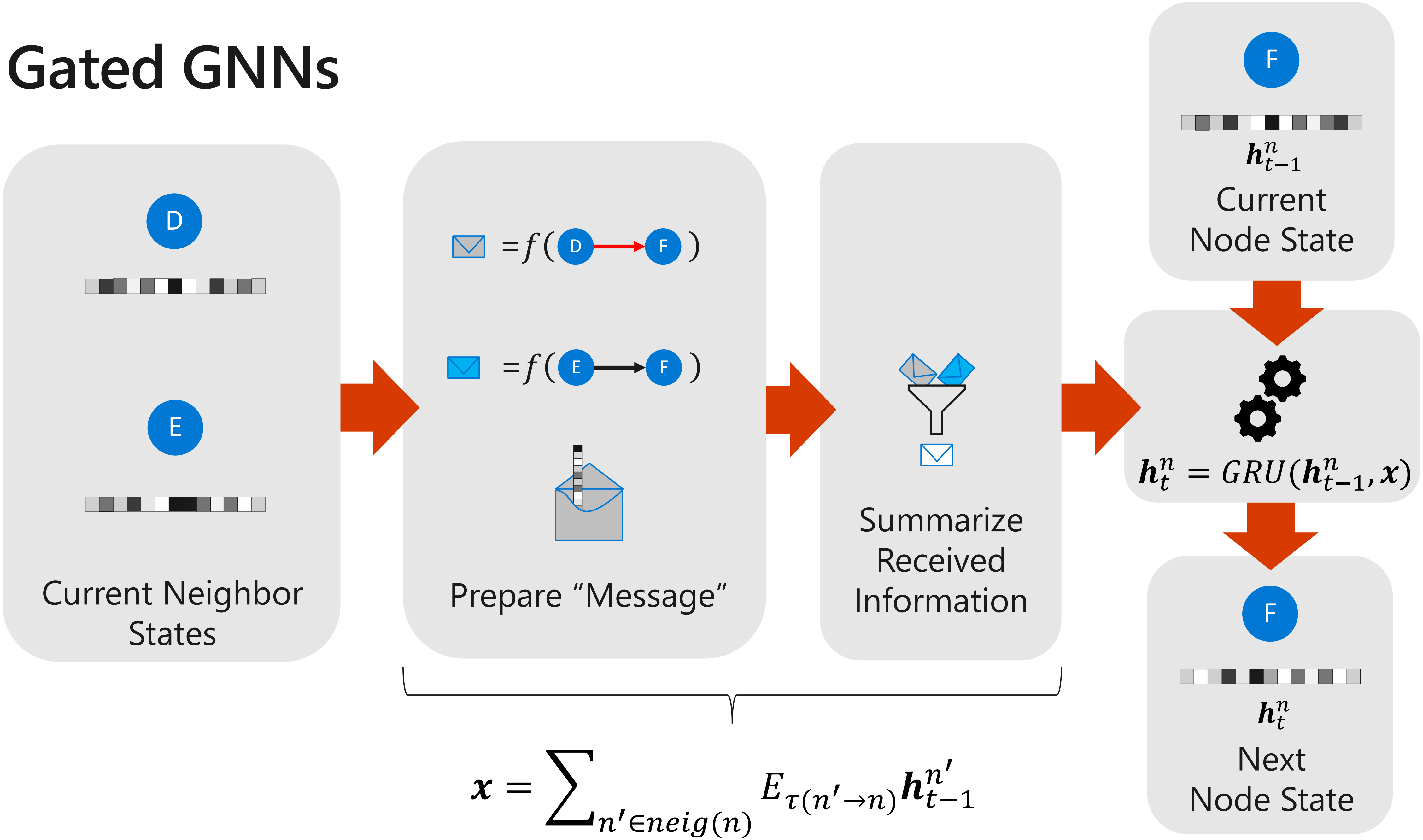
Li et al (2015). Gated Graph Sequence Neural Networks.

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

Neural Message Passing

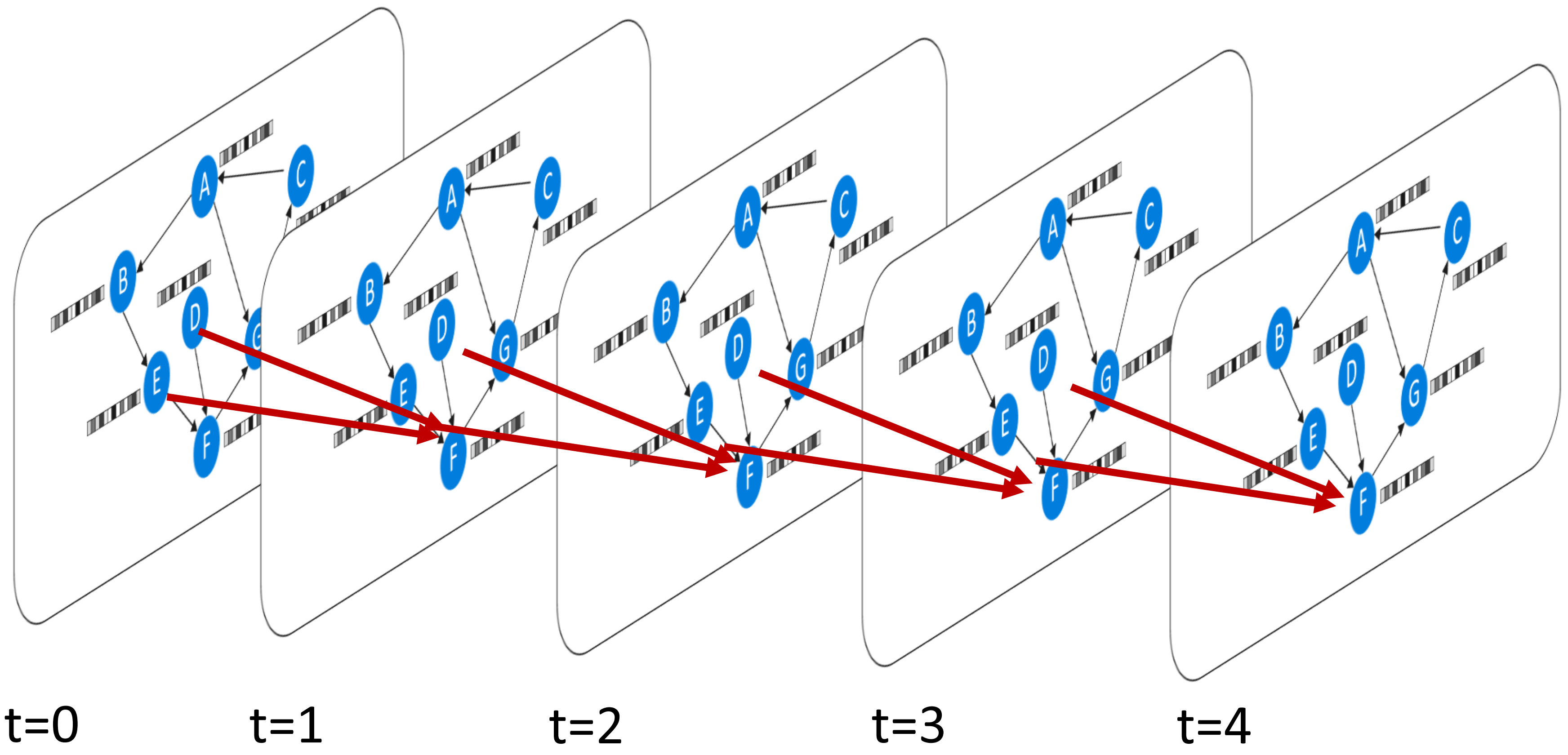


Gated GNNs

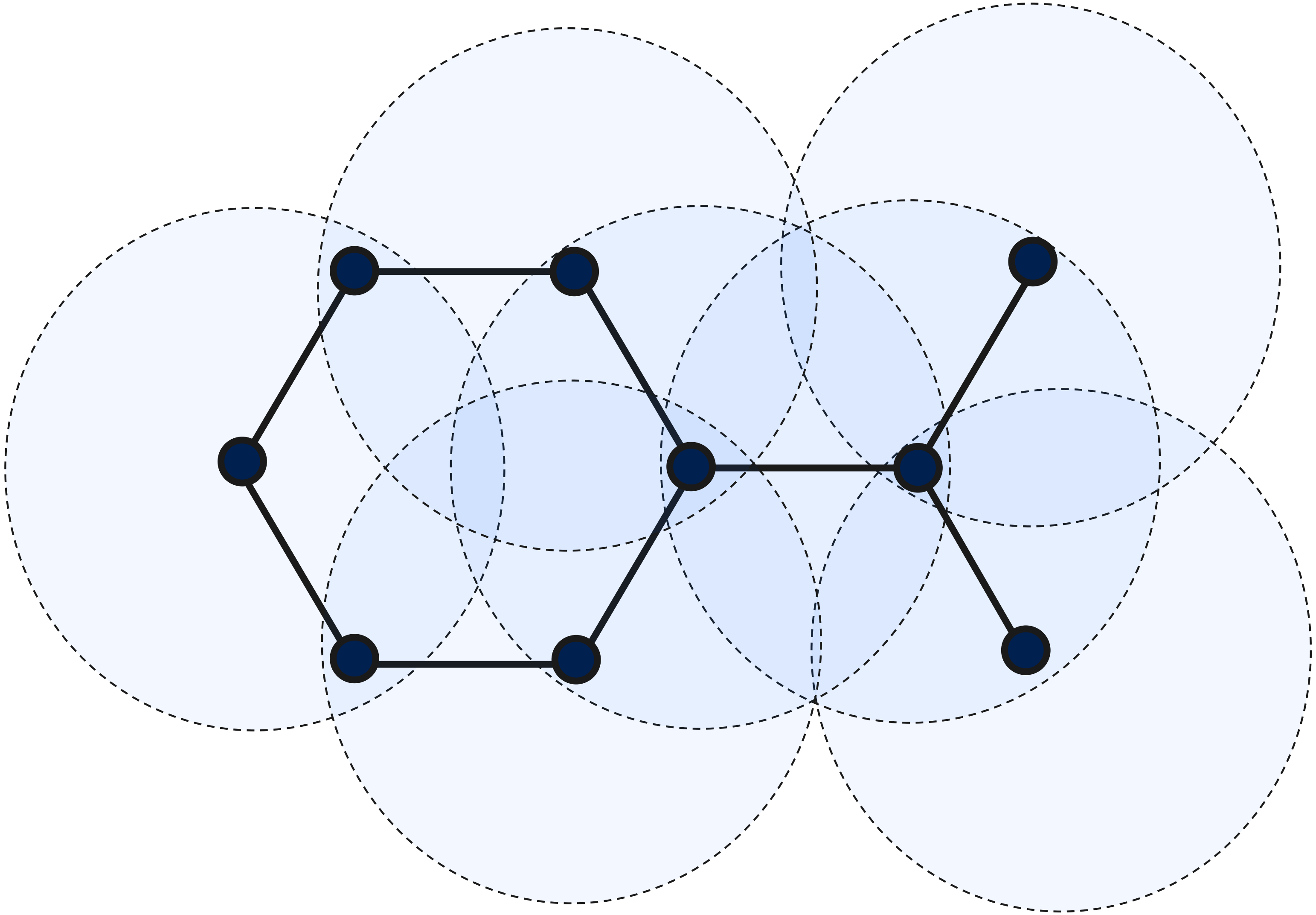


Li et al (2015). Gated graph sequence neural networks.

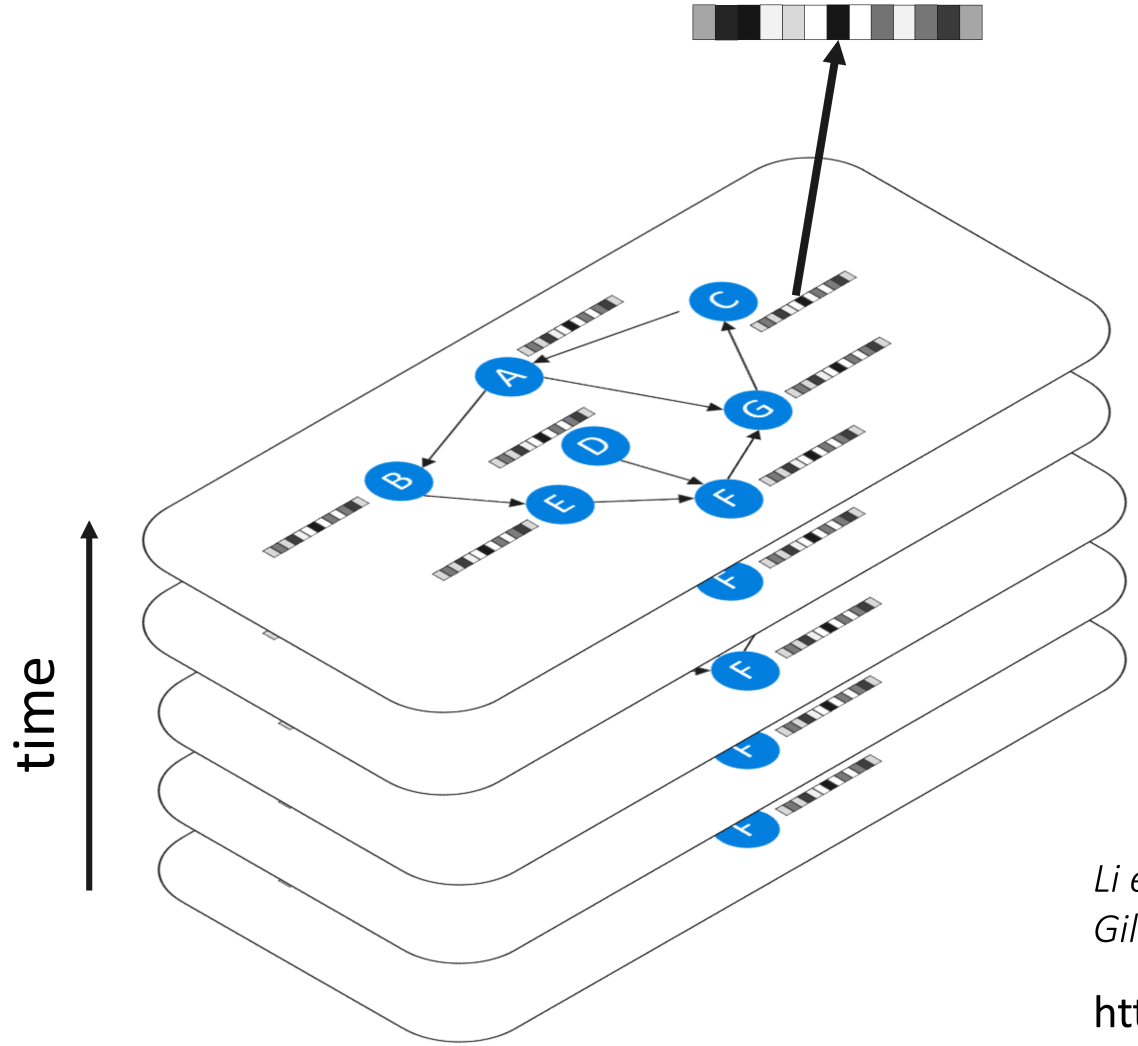
Graph Neural Networks: Message Passing



GNNs: Synchronous Message Passing (All-to-All)



Graph Neural Networks: Output



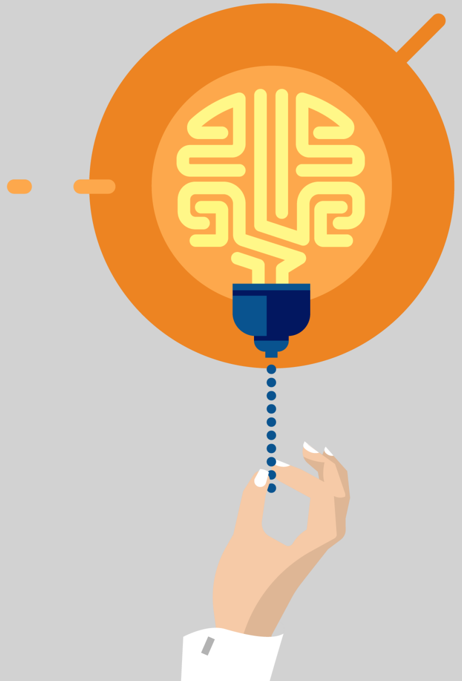
- node selection
- node classification
- graph classification

Li et al (2015). Gated Graph Sequence Neural Networks.
Gilmer et al (2017). Neural Message Passing for Quantum Chemistry.

https://github.com/microsoft/tf-gnn-samples/

Understanding & Generating Source Code

...with graph neural networks.

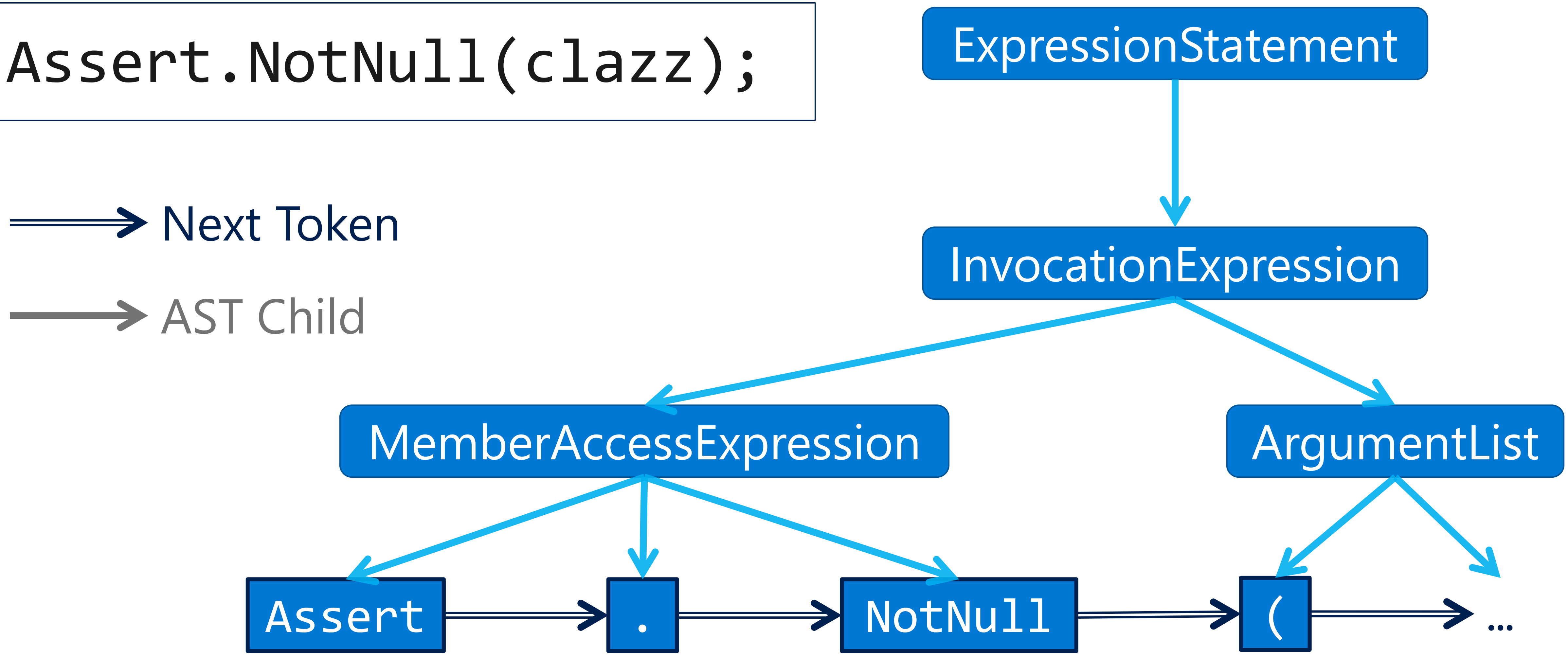


Programs as Graphs

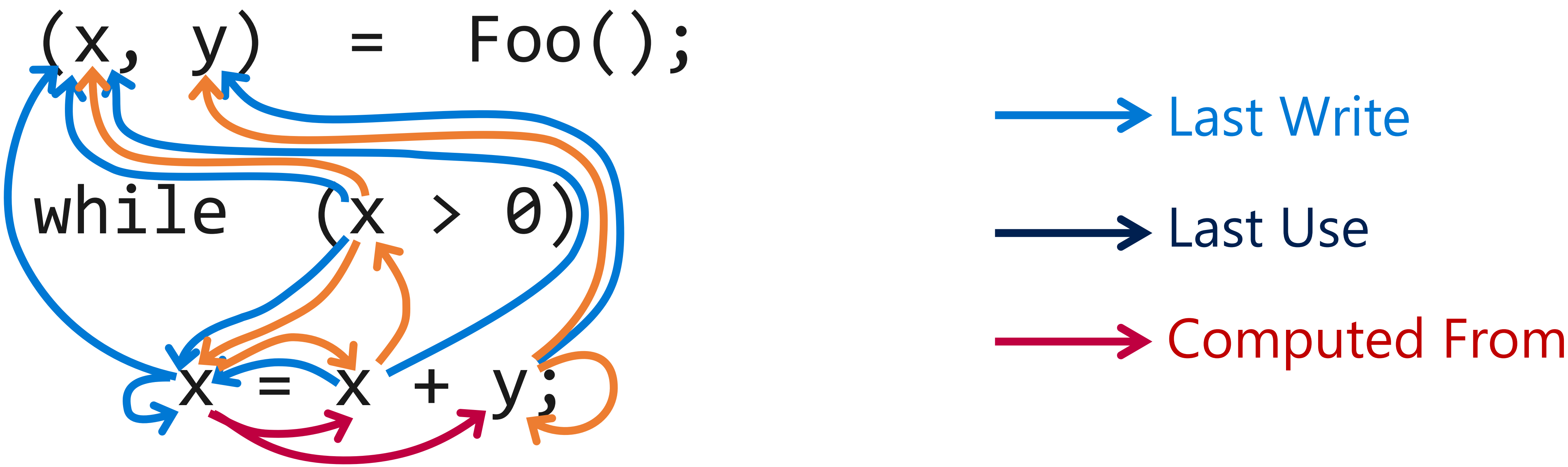
```
int SumPositive(int[] arr, int lim) {
  int sum = 0;
  for (int i = 0; i < lim; i++)
    if (arr[i] > 0)
        sum += arr[i];

  return sum;
}
```

Programs as Graphs: Syntax

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

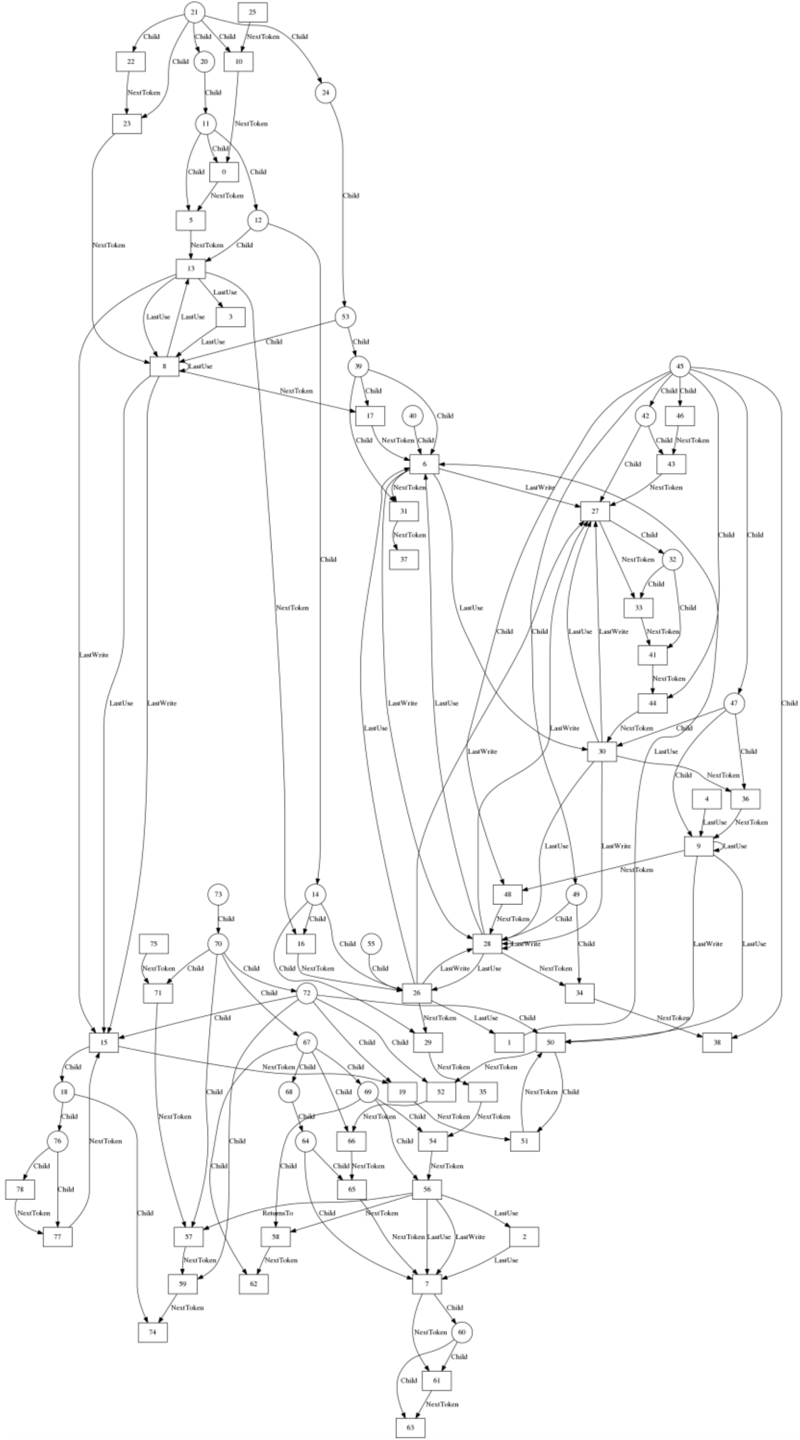


Programs as Graphs: Data Flow



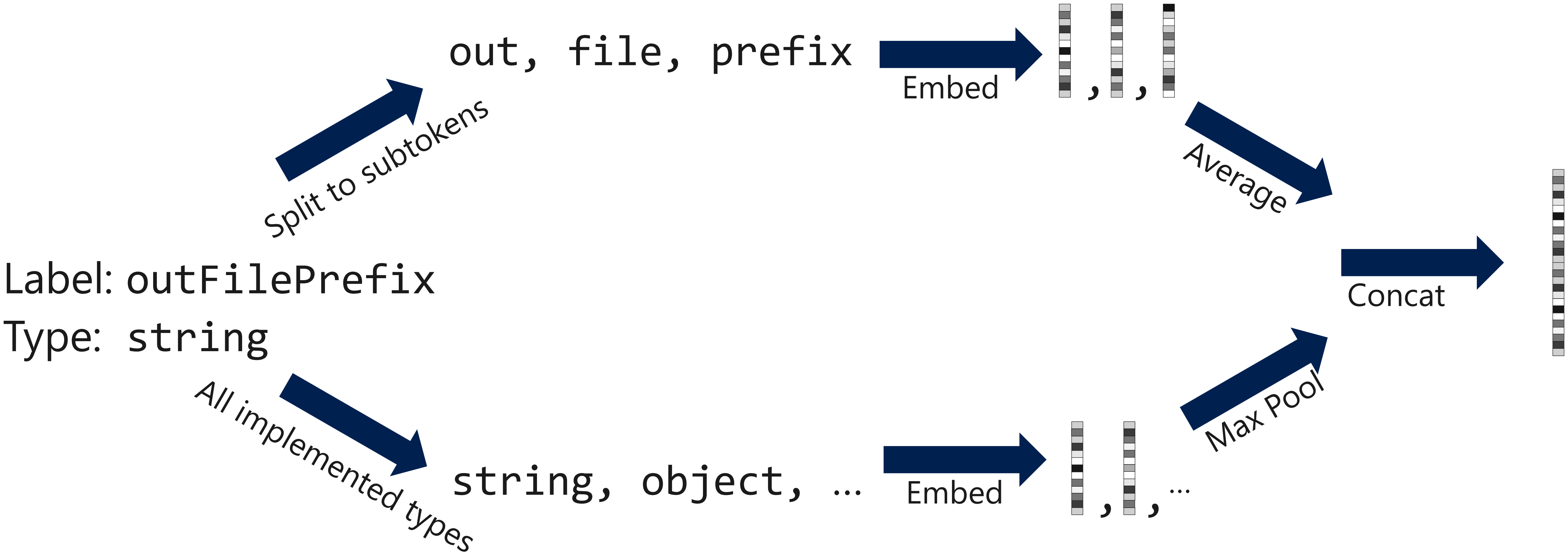
Programs as Graphs

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



~900 nodes/graph ~8k edges/graph

Initial Node Representations



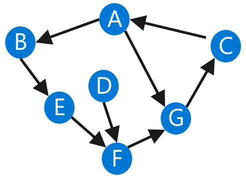
Variable Misuse Task

```
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.

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

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

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, $\sigma = 2.6$)

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)

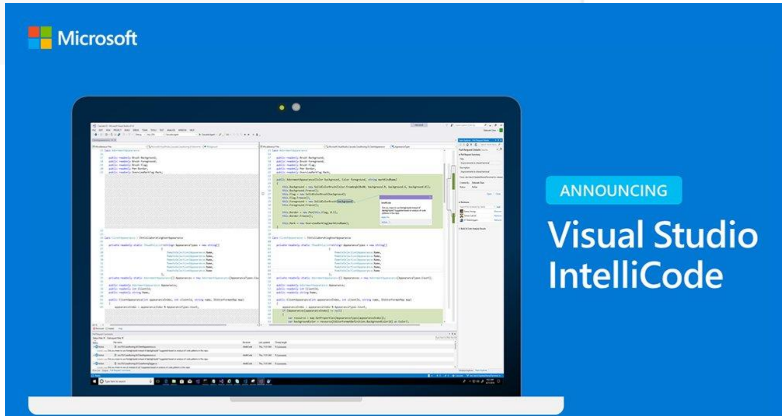
```
// Create or update the document.
var newDocument = await cosmosClient.UpsertDocumentAsync(cosmosDbCollectionUri, document);

if (updateRecord)
{
    logger.WriteLog($"Updated {existingDocument} to {newDocument}");
}
else
{
    logger.WriteLog($"Added {existingDocument}");
```

[redacted] Update 1 1 Resolved

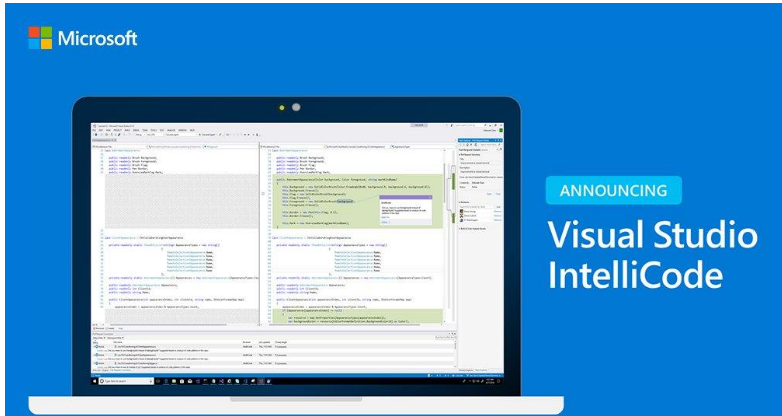
Based on this repo's code patterns, did you intend to use 'newDocument' (confidence 92%) rather than 'existingDocument` (confidence 7%) here? Review is recommended by Research bot's Variable Misuse analysis.

JK [redacted]

+1



```
var x = ComputeX()
if (some-condition-without-x) {
        UseX(x)
} else {
        UseOtherVars(y)
}
// x not used after this point
```



Code Summarization to Natural Language

Code Function

```
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;
}
```

Summary

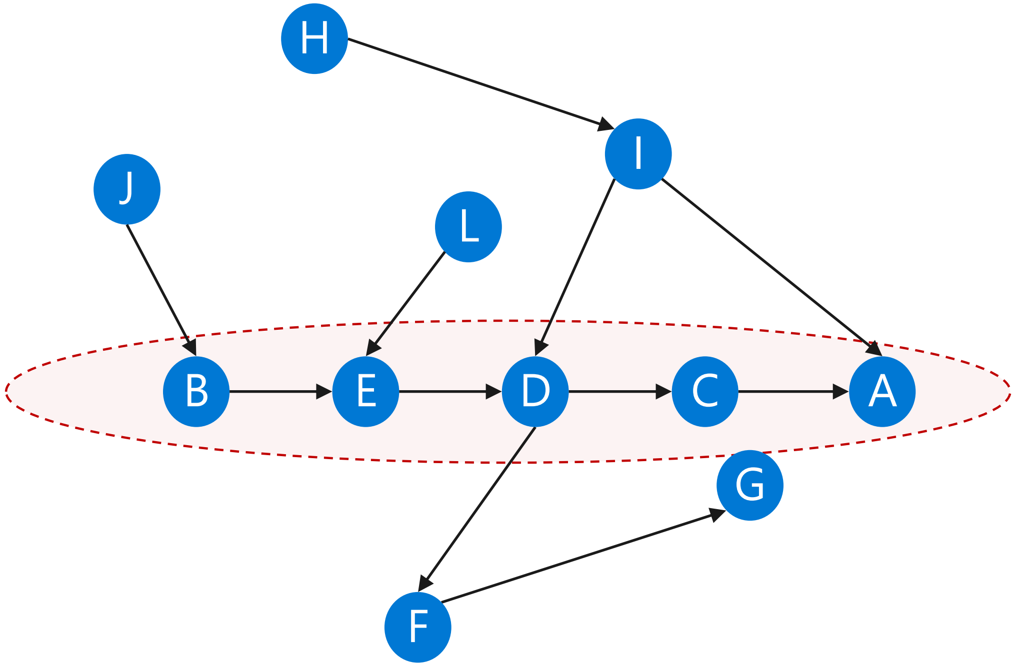
Returns the sum of the positive numbers in an array

Fernandes *et al. 2019* "Neural Structured Summarization"

SeqGraph2Seq

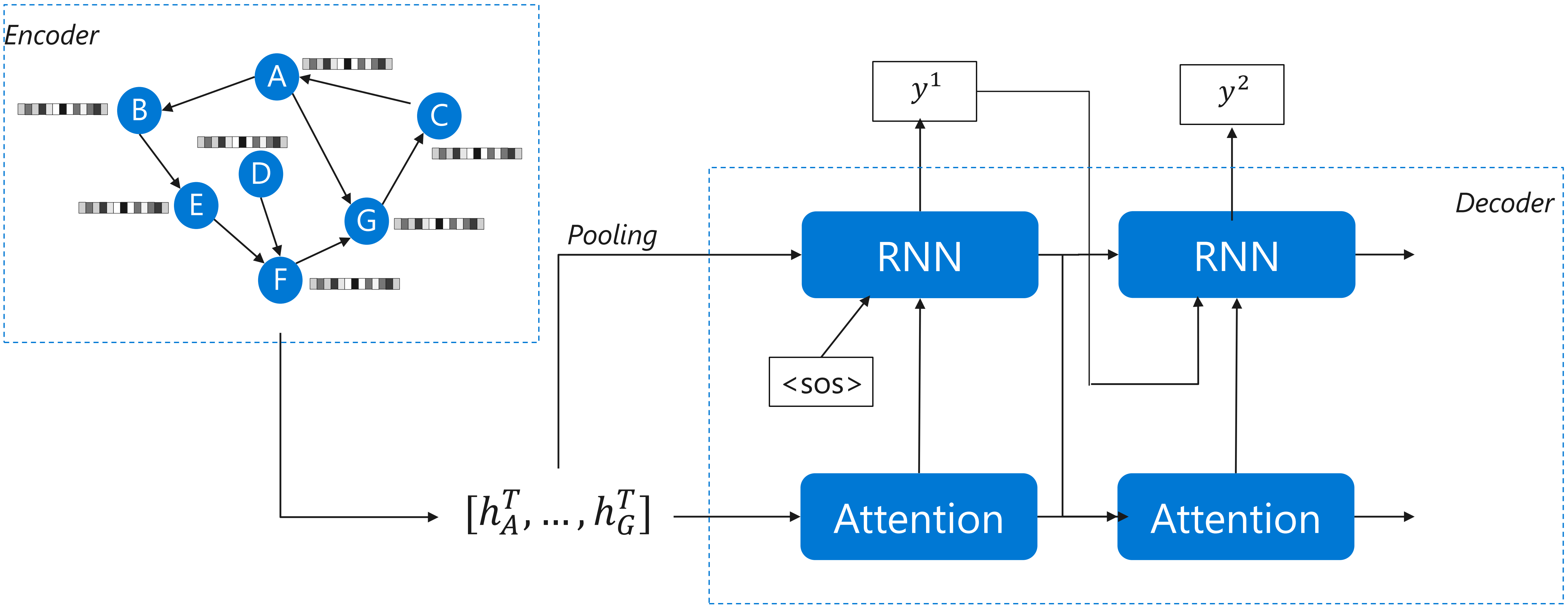
$$\widetilde{\boldsymbol{h}}_n^{(0)} = \begin{cases} ENCODER(P)_n & if\ n \in P \\ \boldsymbol{h}_n^{(0)} & else \end{cases}$$

$$\boldsymbol{h}_n^{(T)} = biGNN\left(G, \left\{\widetilde{\boldsymbol{h}}_1^{(0)}, \dots, \widetilde{\boldsymbol{h}}_N^{(0)}\right\}\right)_n$$



P is the backbone sequence

Graph to Sequence Model



Quantitative Results

C# Documentation	F1	ROUGE-2	ROUGE-L
biRNN -> RNN	35.2	20.8	36.7
GNN -> RNN	38.9	25.6	37.1
biRNN + GNN -> RNN	**45.4**	**28.3**	**41.1**

Java Method Naming	F1	ROUGE-2	ROUGE-L
Alon et al. *	43.0	-	-
biRNN -> RNN	35.8	17.9	39.7
biRNN + GNN -> RNN	**44.7**	**21.1**	**43.1**

* *Alon et al. (2018). Code2seq: Generating sequences from structured representations of code.*

Natural Language Summarization

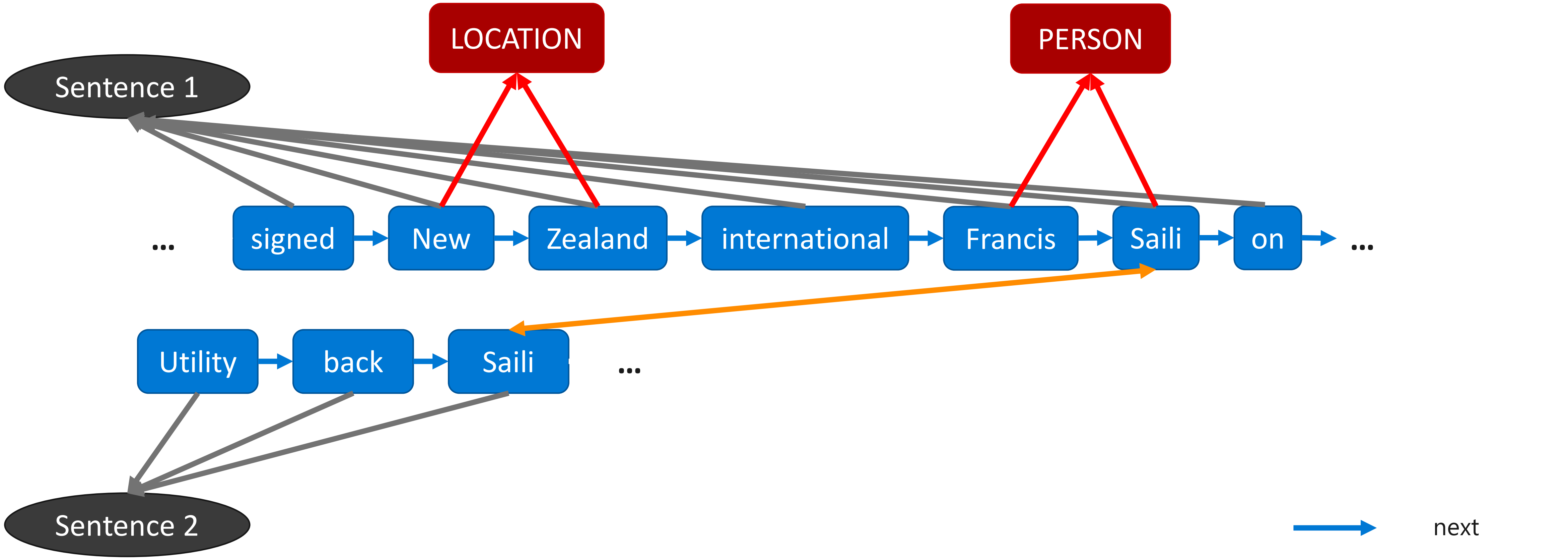
News Article

```
Gunshots were fired at rapper Lil Wayne ’s tour bus early Sunday
in Atlanta. No one was injured in the shooting, and no arrests
have been made, Atlanta Police spokeswoman Elizabeth Espy said.
Police are still looking for suspects. Officers were called to a
parking lot in Atlanta ’s Buckhead neighbourhood, Espy said.
They arrived at 3:25 a.m. and located two tour buses that had
been shot multiple times. The drivers of the buses said the
incident occurred on Interstate 285 near Interstate 75, Espy
said. Witnesses provided a limited description of the two
vehicles suspected to be involved: a “Corvette-style vehicle”
and an SUV. Lil Wayne was in Atlanta for a performance at
Compound nightclub Saturday night. CNN’s Carma Hassan
contributed to this report.
```

Summary

Rapper Lil Wayne not injured after shots fired at his tour bus on an Atlanta interstate, police say. No one has been arrested in the shooting.

Structure in Natural Language



Quantitative Results

CNN/DailyMail	ROUGE-1	ROUGE-2	ROUGE-L
RNN -> RNN *	31.3	11.8	**28.8**
RNN + GNN -> RNN	**33.0**	**13.3**	28.3

Encoder -> Decoder

Higher is better

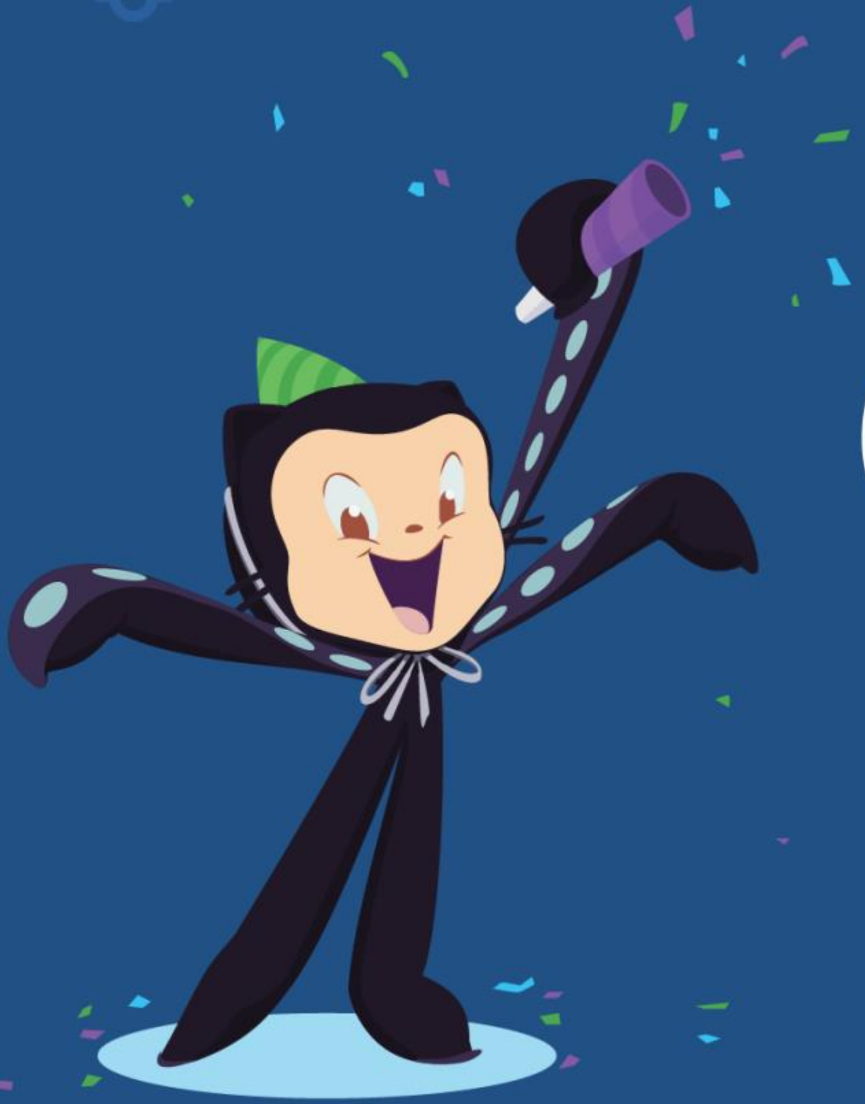
* *See et al. (2015). Get To The Point: Summarization with Pointer-Generator Networks*

Quantitative Results

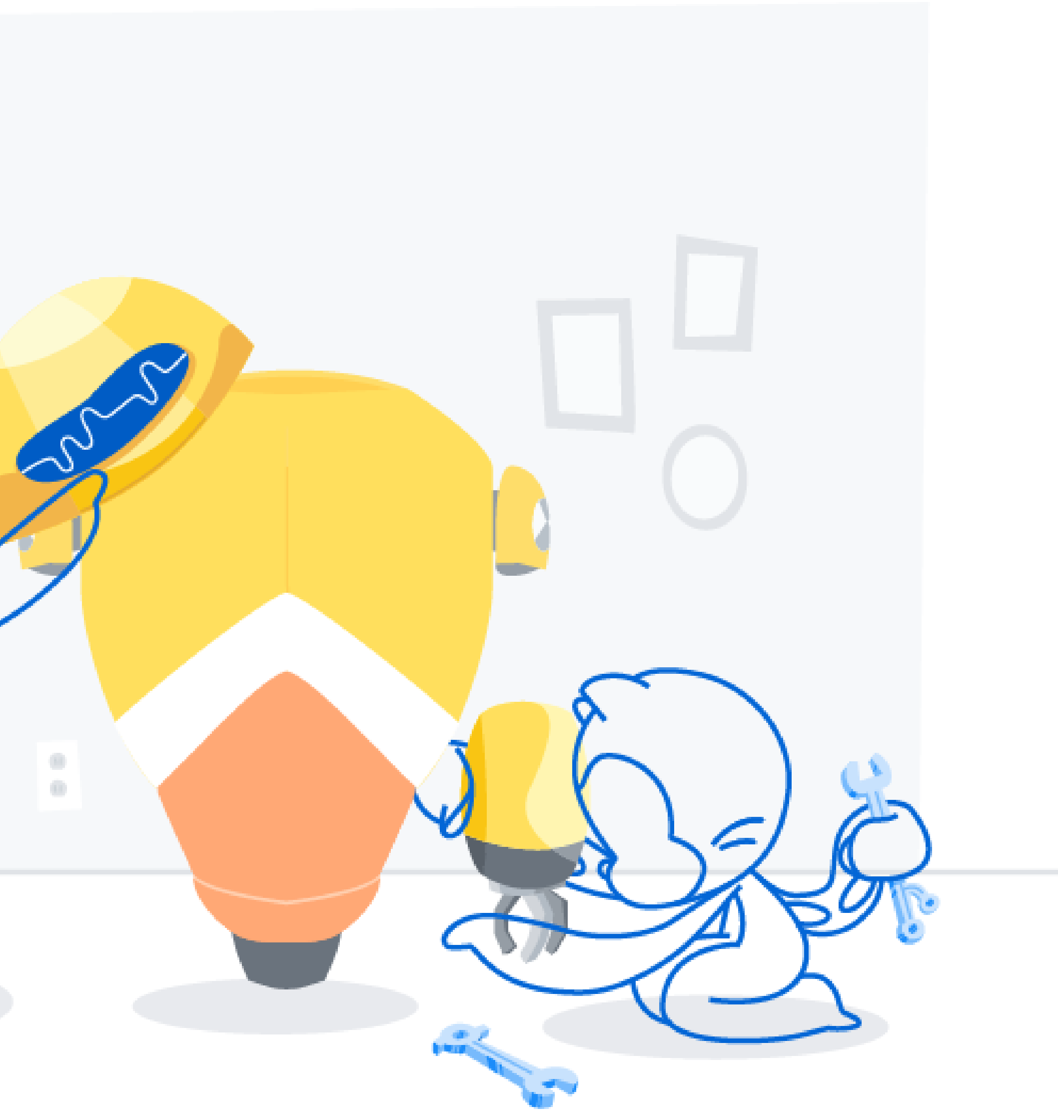
CNN/DailyMail	ROUGE-1	ROUGE-2	ROUGE-L
RNN -> RNN *	31.3	11.8	28.8
RNN + GNN -> RNN	33.0	13.3	28.3
RNN -> RNN + pointer *	36.4	15.7	**33.4**
RNN + GNN -> RNN + pointer	**38.1**	**16.1**	33.2

* *See et al. (2017). Get To The Point: Summarization with Pointer-Generator Networks*

Microsoft



GitHub

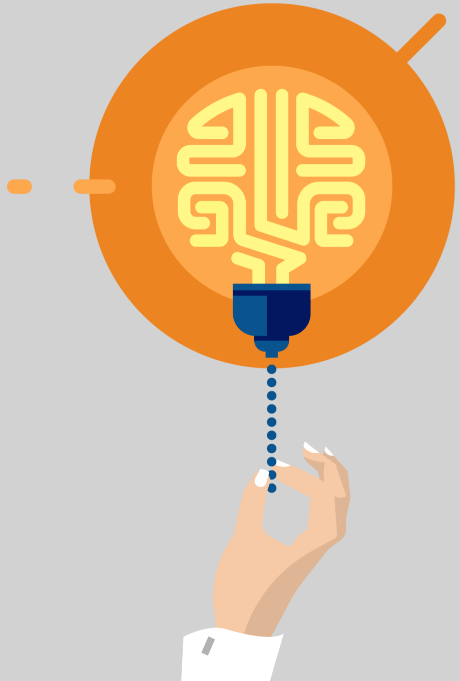


CodeSearchNet

- A corpus of 6 millions functions with metadata.
- A small human-annotated set of relevance annotations.
- A semantic code search challenge.

https://github.com/github/CodeSearchNet

Closing Thoughts



NLP with GNNs

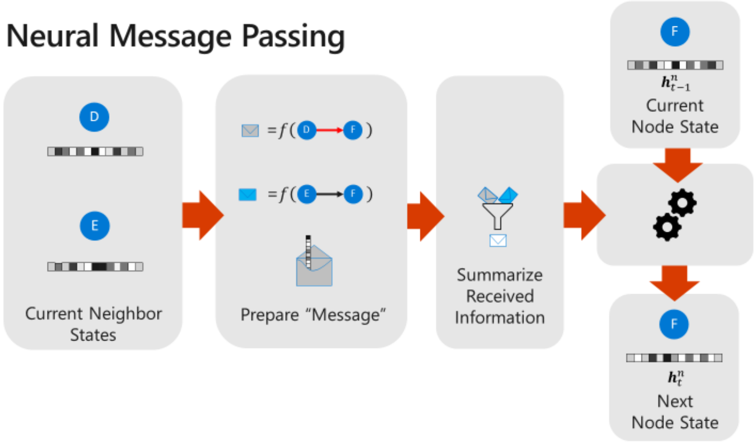
- Grounded Language
- Procedural Text
- Semantic Parsing

Source Code & Natural Language

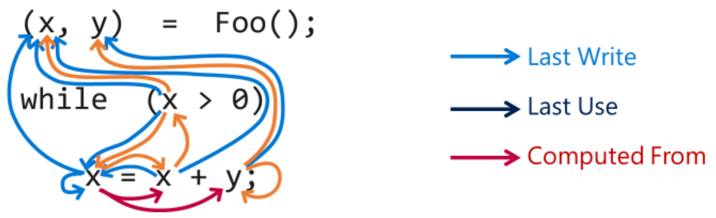
Rich Structure

Reasoning

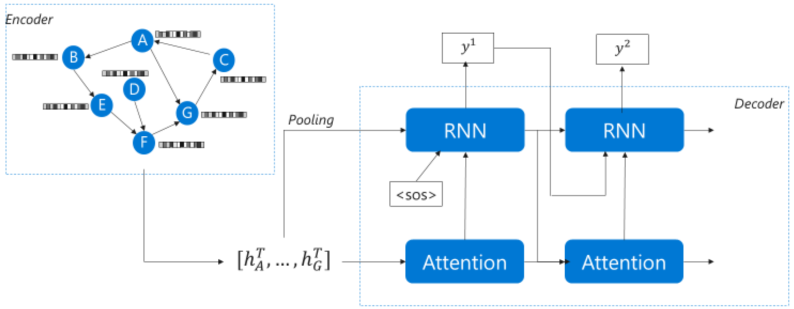
Neural Message Passing



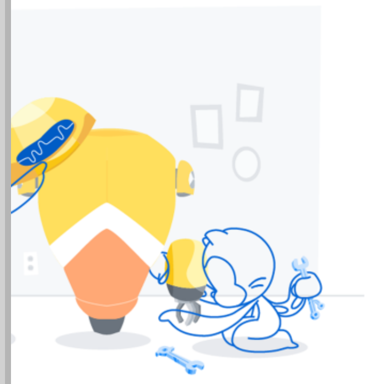
Programs as Graphs: Data Flow



Graph to Sequence Model



CodeSearchNet



- A corpus of 6 millions functions with metadata.
- A small human-annotated set of relevance annotations.
- A semantic code search challenge.

https://github.com/github/CodeSearchNet