Understanding Source Code through Machine Learning to Create Smart Software Engineering Tools

Miltos Allamanis, University of Edinburgh

March 13th, 2016

THE UNIVERSITY *of* EDINBURGH
informatics

Joint work with: *Charles Sutton (UoE), Earl T. Barr (UCL), Chris Bird (MSR), Daniel Tarlow (MSRC), Yi Wei (MSRC), Andrew D. Gordon (MSRC)*

My PhD is supported by
Microsoft Research

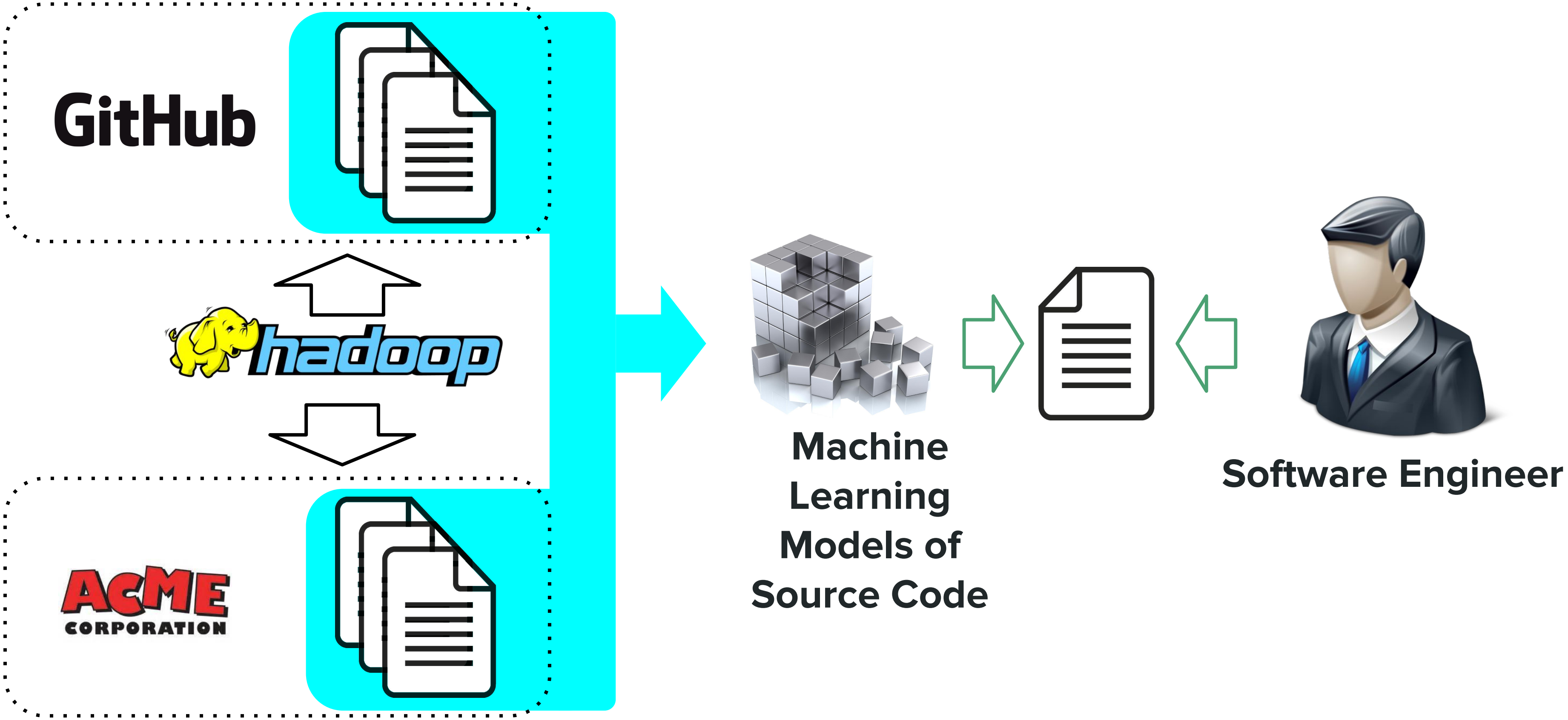
Developers implicitly embed **knowledge** in code that may be useful for the same or other projects.

GitHub

Atlassian Bitbucket

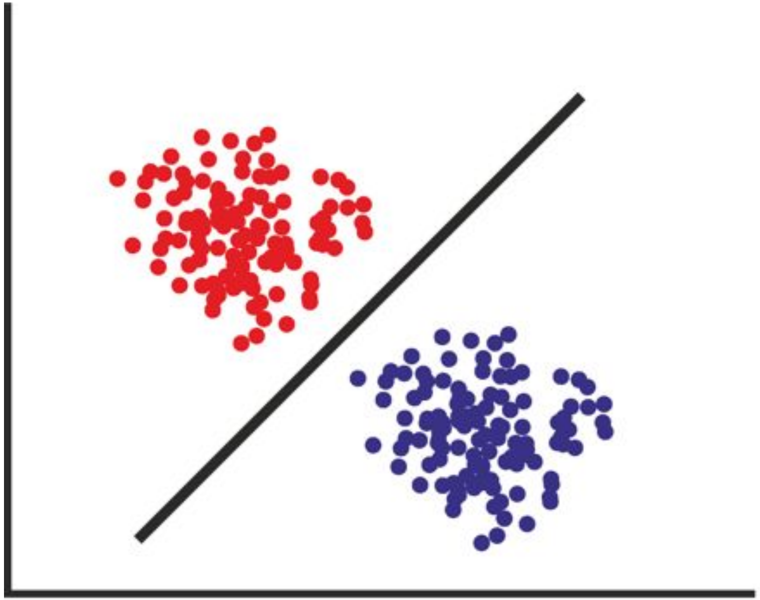
internal & external codebases

Mine the hidden knowledge to create **smart** software engineering tools.



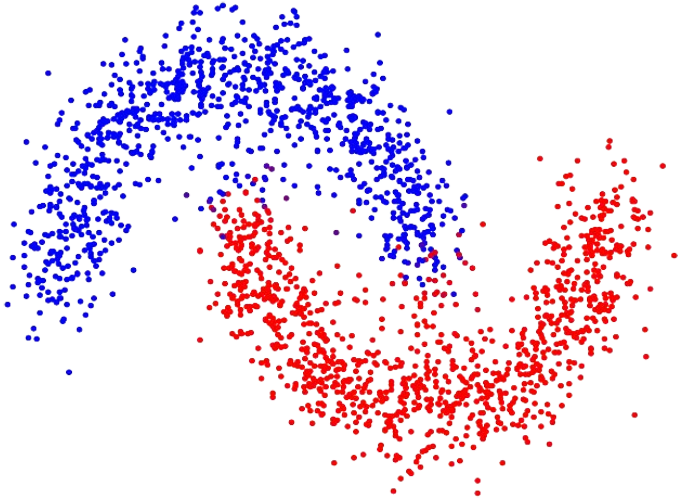
GitHub
hadoop
ACME
CORPORATION
Machine
Learning
Models of
Source Code
Software Engineer

A Spectrum of Problems for Machine Learning



Classification

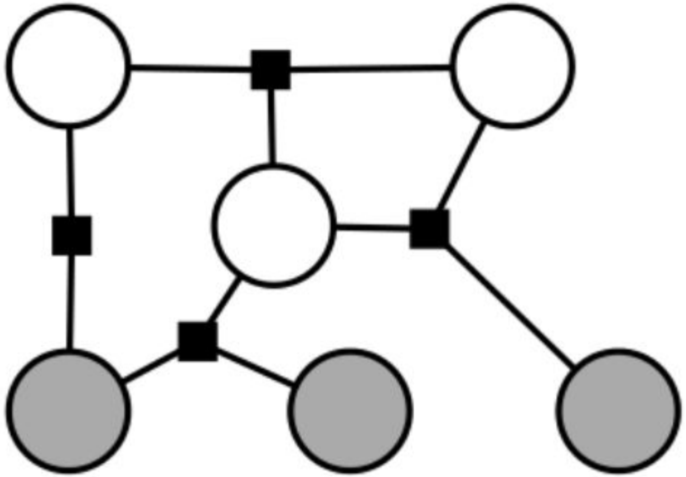
Supervised



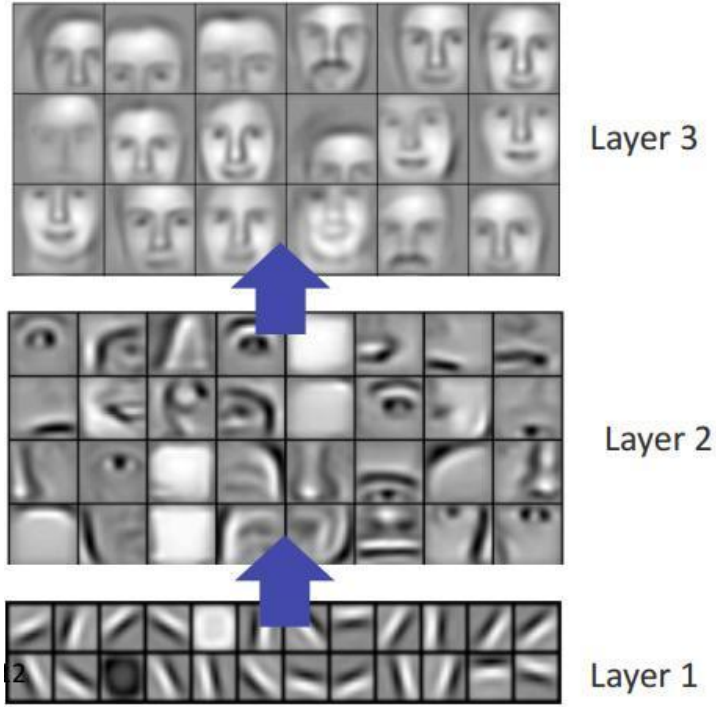
Clustering

Unsupervised

A Spectrum of Problems for Machine Learning

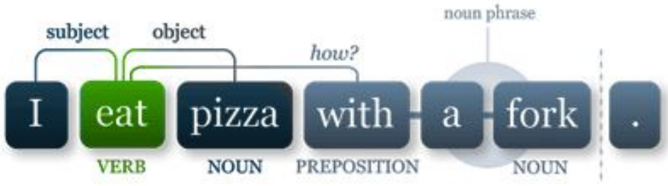


Joint Classification



Learning Features

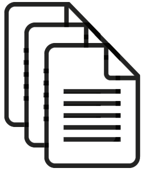
Natural Language Processing with Machine Learning



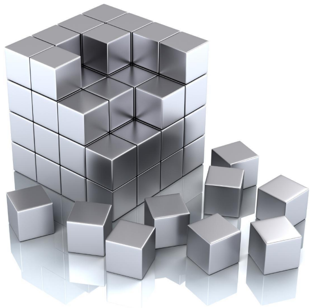
- **Resolve** language ambiguities with principled probabilistic models of language.
- Learn model parameters from annotated corpora.

Natural Language Processing (NLP)

Some **Knowledge** of **Linguistics**



Data: Corpora of Text, Speech etc



Models of Aspects of Natural Language

Parsing

Named Entity Recognition

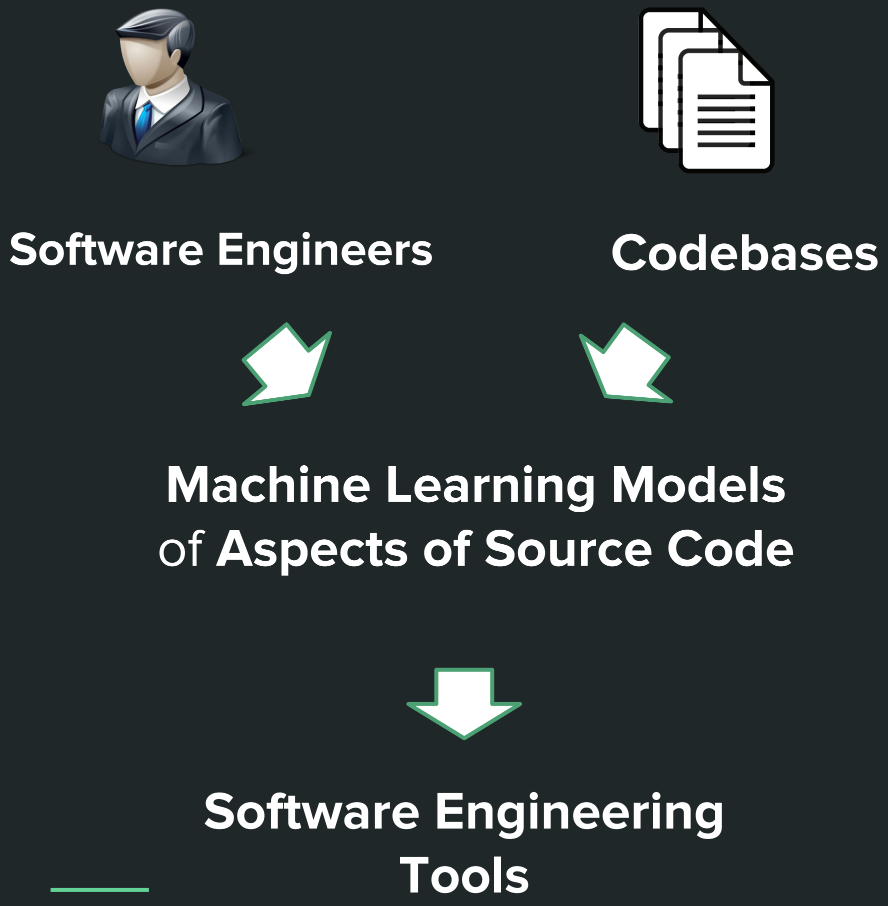
Machine Translation

....

Use Machine Learning to **model** aspects of a natural language.

Machine Learning Models of Source Code

“All models are wrong, some are useful” - George Box



Language Models for Source Code

Assign a non-zero probability to every piece of valid code

Probabilities learned from training corpus

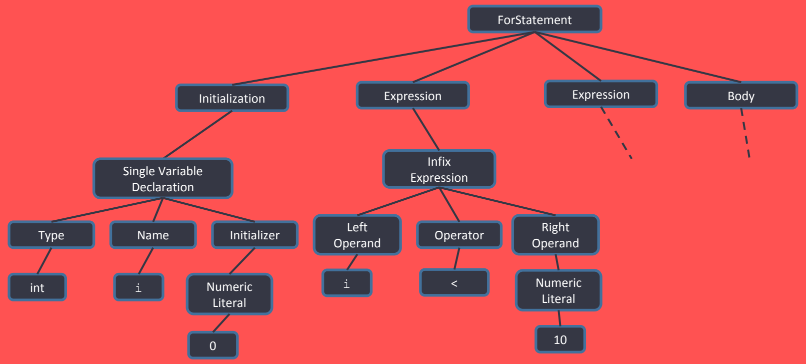
```
    for (int i = 0; i < nProperties; i++) {
        final List<TreeNode<TSGNode>> children = node
                .getChildrenByProperty().get(i);
        final int nChildren = children.size();
        ruleConsequent.nodes.add(Lists
                .<Integer> newArrayListWithCapacity(nChildren));
        for (int j = 0; j < nChildren; j++) {
            final int childNode = node.getChild(j, i).getData().nodeKey;
            ruleConsequent.nodes.get(i).add(childNode);
        }
    }

    return new CFGRule(rootId, ruleConsequent);
}

public BlockedPosteriorComputer getPosteriorComputer() {
    return samplePosteriorComputer;
}

public final CFGPrior getPrior() {
    return prior;
}

public void lockSamplerData() {
    prior.lockPrior();
    burninPosteriorComputer.getPrior().cfg = samplePosteriorComputer
            .getPrior().cfg;
}

@Override
public void sampleAllTreesOnce(final int currentIteration,
        final int totalIterations, final AtomicBoolean stop) {
    final Thread termSignalHandler = new Thread() {
        @Override
        public void run() {
            stop.set(true);
        }
    };
```

Language Models of Source Code – Design Choices

Token-level Models

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

Syntactic Models



N-gram Language Models

$$P(t_0 \ldots t_M) = \prod_{m=0}^{M} P(t_m | t_{m-1} \ldots t_{m-n+1})$$

Parameters of ML Model

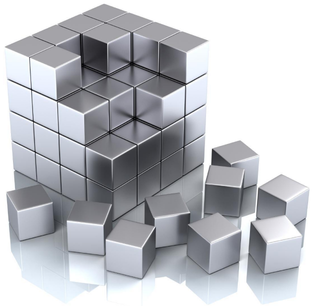
$$P(t_m | t_{m-1} \ldots t_{m-n+1})$$

e.g. `P(0|“for (int i =”)`

How n-gram models see code?

```
package org.cfeclipse.cfml.snippets;

import org.rioproject.examples.logicdesigner.model.getState ( ) {
 cdl.Choreography;
import org.apache.thrift.protocol.TProtocolUtil.skip(iprot);
event.newLineCount == 3 )
 { case '|' :
    if ( rule.FireAllRulesCommand;

  import org.apache.hadoop.conf.get(0, 0, newByteBuffer, 0,
count);
 } switch ( classifierID ) {
    pd.getName() {
        cBondNeighborsB.get(MODULE).declaringType
               = (DEREnumerated)
```

Machine **Learning**



Machine Learning Model

Designed by humans

Model Parameters

Learned from data

Learn the parameters of the model from data.

Handle **uncertainty** and **noise**.

Learning Model Parameters

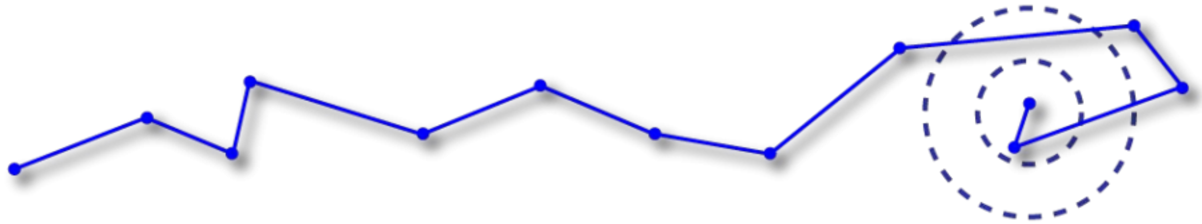
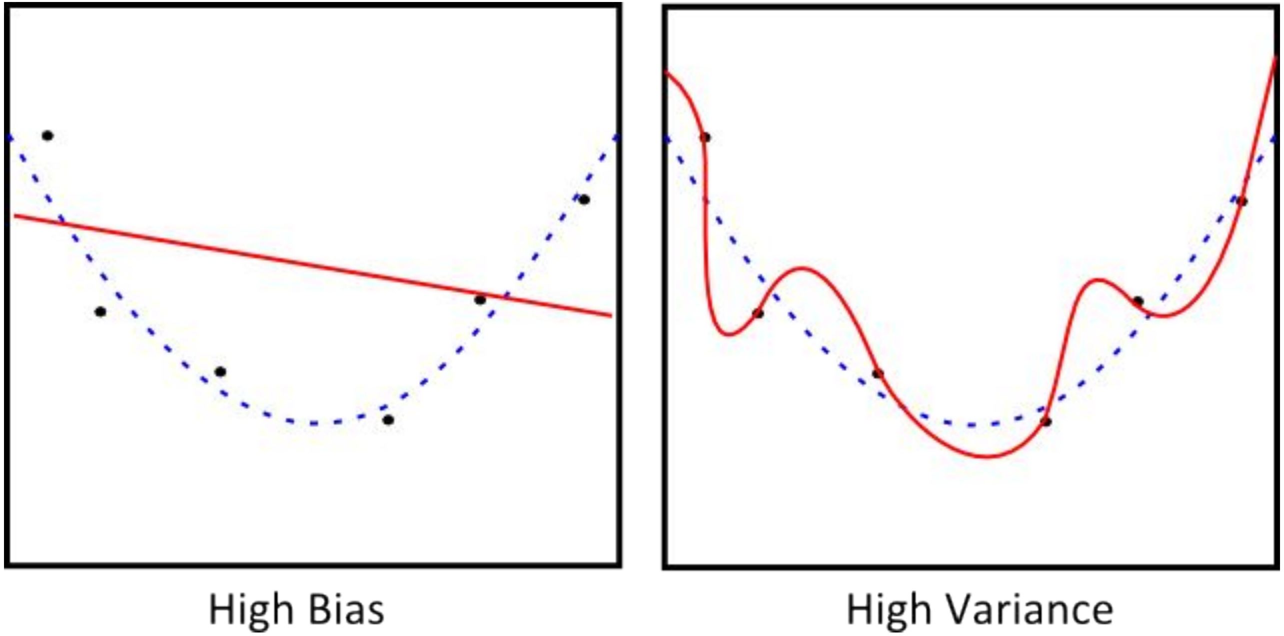


Image from marple.eeb.uconn.edu

- **Optimize** objective function in training set
- Use **computational methods** of optimization

Finding a good model



Underfitting   Overfitting

image from http://antianti.org/?p=175

Automatic Evaluation in Machine Learning

Imperfect measures of performance *such as*

- Prediction Accuracy
- Model Fit

- **Quantify** performance in a **reproducible** manner
- **Drive** improvement of systems in a **measurable** way

Source Code and Machine Learning

Coding Patterns

Mine & exploit common patterns

[*Hindle et al. 2012, Allamanis & Sutton 2014, Allamanis et al. 2014, 2015*]

Probabilistic Static Analyses

Probability Distribution of (Formal) Properties

[*Raychev et al. 2015, Mangal et al. 2015*]

Formal Methods

Probabilities over Search Space (*e.g.* Synthesis)

[*Ellis et al. 2015*]

Runtime Traces

Infer Program Properties from Traces

[*Brockschmidt et al. 2014 Yujia Li et al. 2015*]

Code & Text

Code search, NL to Code

[*Yusuke, et al. 2015, Movshovitz-Attias & Cohen, 2013, Allamanis et al. 2014*]

Outline

Learning Naming Conventions

- Lexical Patterns

Learning to Map Natural Language to Source Code

- Syntactic Patterns

“Programs must be written for people to read, and only incidentally for machines to execute.”

- Abelson & Sussman, SICP, preface to the first edition

Learning Naming Conventions

A **coding convention** is a syntactic constraint *beyond* those imposed by the language grammar.

Allamanis et. al, FSE 2014, FSE 2015
ACM Distinguished Paper Award

The Importance of Coding Conventions

	Code Review Discussions
Conventions	38%
Naming	24%
Formatting	9%

[Allamanis et al. FSE 2014]

Based on 169 *code reviews* with 1,093 discussion threads in Microsoft.

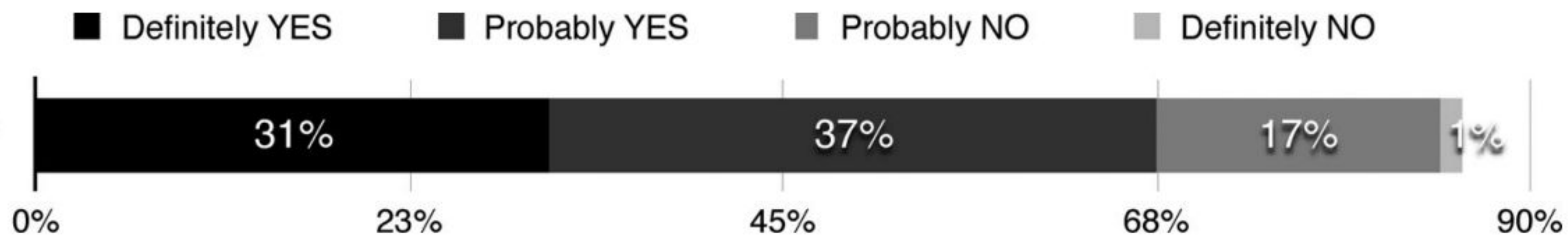
The Importance of Coding Conventions

	Code Review Discussions
Conventions	38%
Naming	24%
Formatting	9%

[Allamanis et al. FSE 2014]

Based on 169 *code reviews* with 1,093 discussion threads in Microsoft.

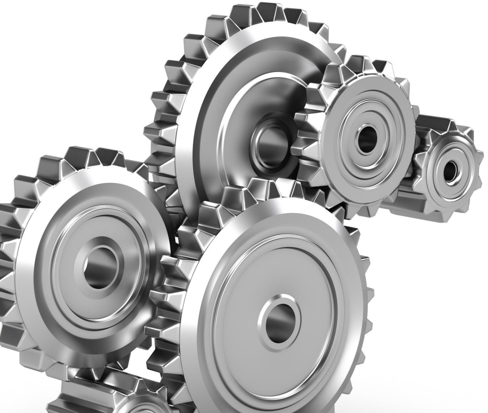
Is recommending identifier renamings useful?



94 developers

Arnaoudova, Venera, L. Eshkevari, Massimiliano Di Penta, Rocco Oliveto, Giuliano Antoniol, and Y. Gueheneuc. "REPENT: Analyzing the nature of identifier renamings." (2014)

A Machine Learning Perspective



A name reflects important aspects of code **functionality**.

Learning to name source code elements is a first step in *understanding* code through machine learning.

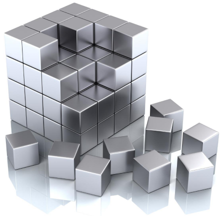
Suggestions for junit/src/test/java/junit/tests/runner/TextRunnerTest.java

```
public class TextRunnerTest extends TestCase {
    void execTest(String testClass, boolean success) throws Exception {
        ...
        InputStream i = p.getInputStream();
        while ((i.read()) != -1);
        ...
    }
    ...
}
```

Suggestions for junit/src/test/java/junit/tests/runner/TextRunnerTest.java

```
public class TextRunnerTest extends TestCase {
    void execTest(String testClass, boolean success) throws Exception {
        ...
        InputStream i = p.getInputStream();
        while ((i.read()) != -1);
        ...
    }
    ...
}
```

automatically suggest renamings

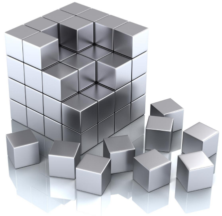


Source Code Language Model

Suggestions for junit/src/test/java/junit/tests/runner/TextRunnerTest.java

```
public class TextRunnerTest extends TestCase {
    void execTest(String testClass, boolean success) throws Exception {
        ...
        InputStream i = p.getInputStream();
        while ((i.read()) != -1);
        ...
    }
    ...
}
```

automatically suggest renamings

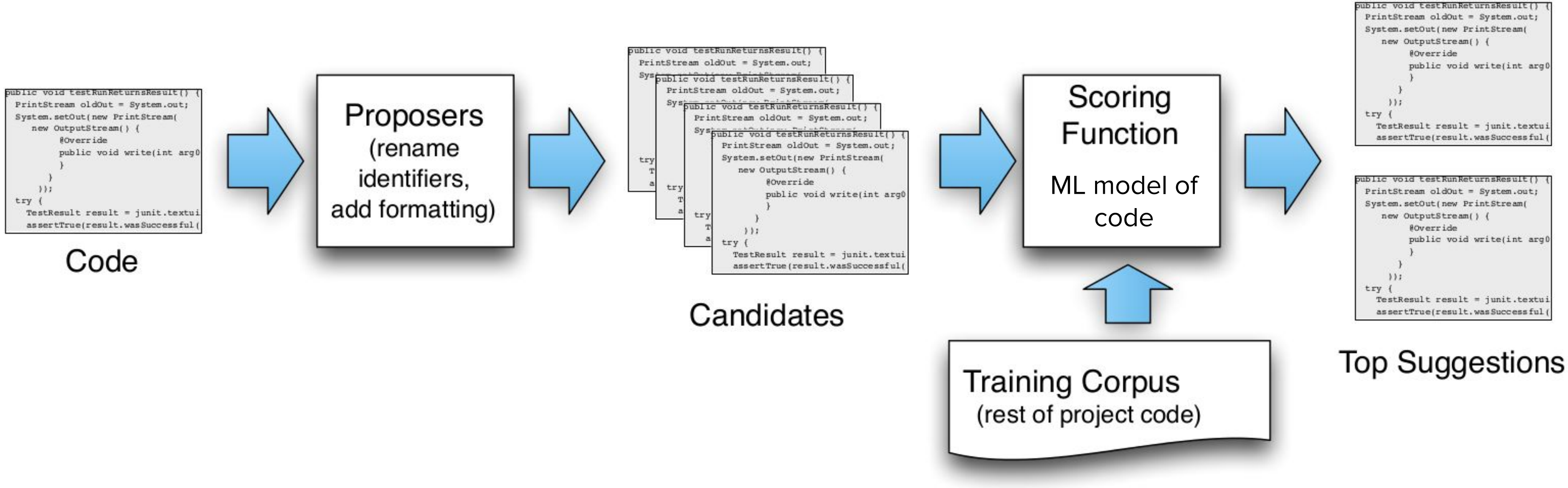


Source Code Language Model

Score by naturalness & Threshold

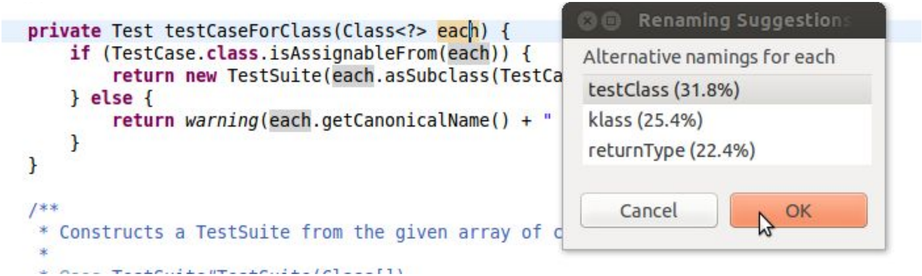
```
1.'i' (18.07%) -> {input(81.93%), }
```

Suggesting Names to Developers: The Naturalize Framework

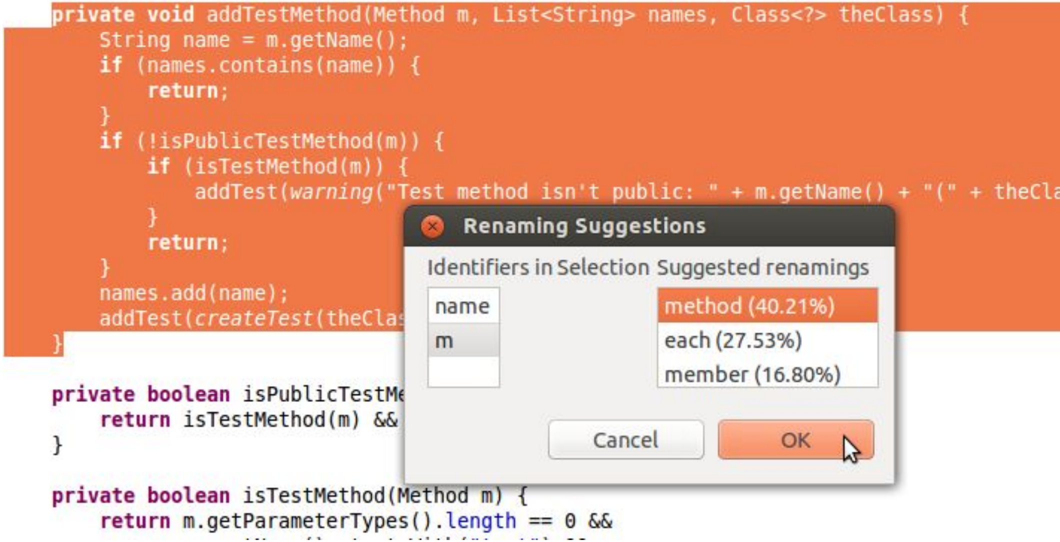


[Allamanis et al. FSE 2014, FSE 2015]

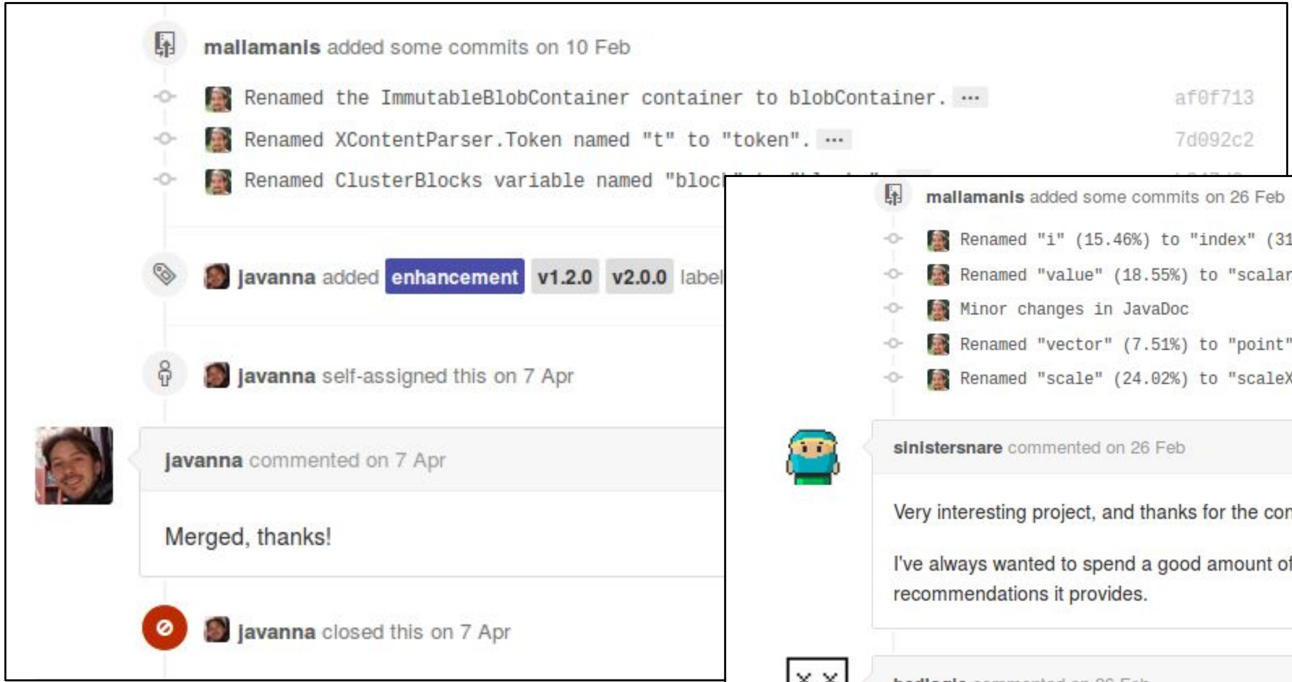
Naturalize Tools - devstyle

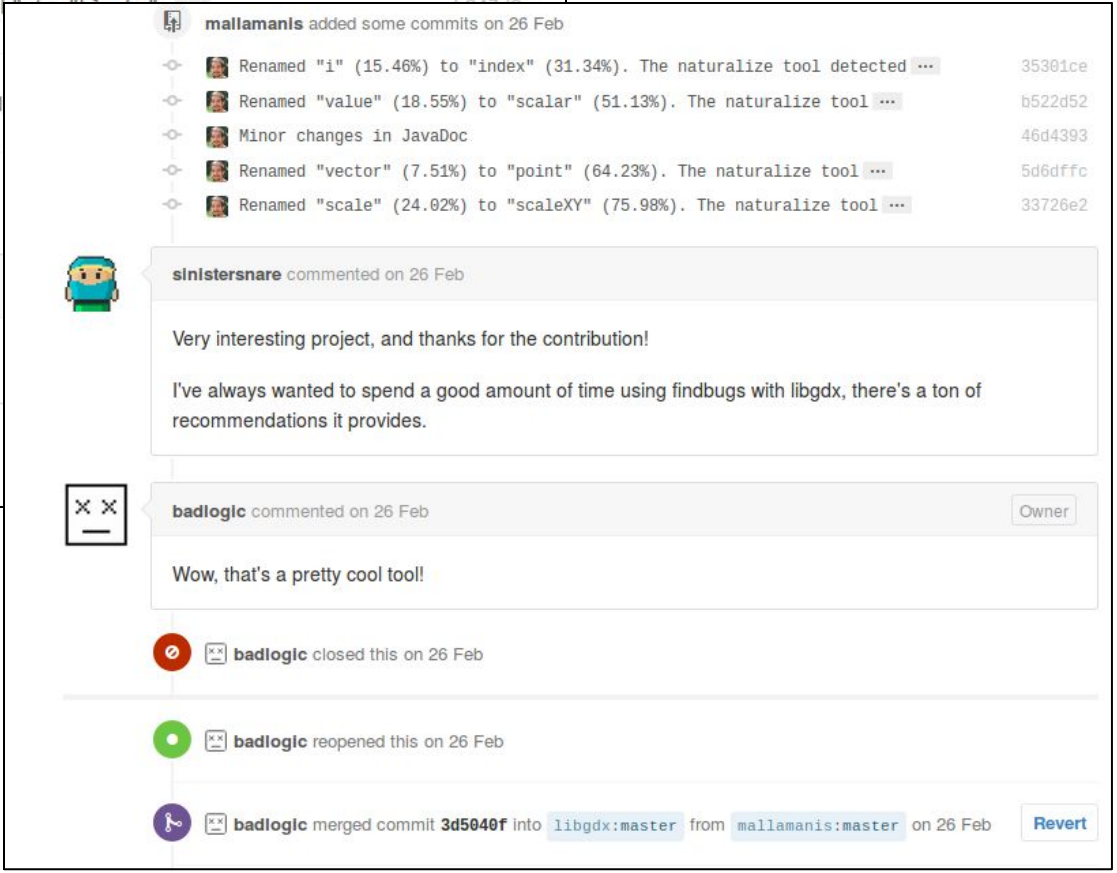


devstyle suggests identifier renamings



GitHub





18 patches for 5 well known open source projects:
14 accepted, 4 ignored

Method Naming Problem

```java
private void  () {
  String vertexShader = "literal_1";
  String fragmentShader = "literal_2";
  shader = new ShaderProgram(vertexShader,
      fragmentShader);
  if(shader.isCompiled() == false)
    throw new IllegalArgumentException(
        "literal_3" + shader.getLog());
}
```

Method Naming Problem

```
private void [illegible] () {
  String vertexShader = "literal_1";
  String fragmentShader = "literal_2";
  shader = new ShaderProgram(vertexShader,
      fragmentShader);
  if(shader.isCompiled() == false)
    throw new IllegalArgumentException(
        "literal_3" + shader.getLog());
}
```

Names describe what it *does* not what it *is*

Models need to be “non-local”

Method Naming Problem

```java
private void () {
  String vertexShader = "literal_1";
  String fragmentShader = "literal_2";
  shader = new ShaderProgram(vertexShader,
      fragmentShader);
  if(shader.isCompiled() == false)
    throw new IllegalArgumentException(
        "literal_3" + shader.getLog());
}
```

Suggestions:

- create
- create?UNK?
- init
- createShader

Method Naming Problem

```java
private void createDefaultShader () {
  String vertexShader = "literal_1";
  String fragmentShader = "literal_2";
  shader = new ShaderProgram(vertexShader,
      fragmentShader);
  if(shader.isCompiled() == false)
    throw new IllegalArgumentException(
        "literal_3" + shader.getLog());
}
```

Suggestions:

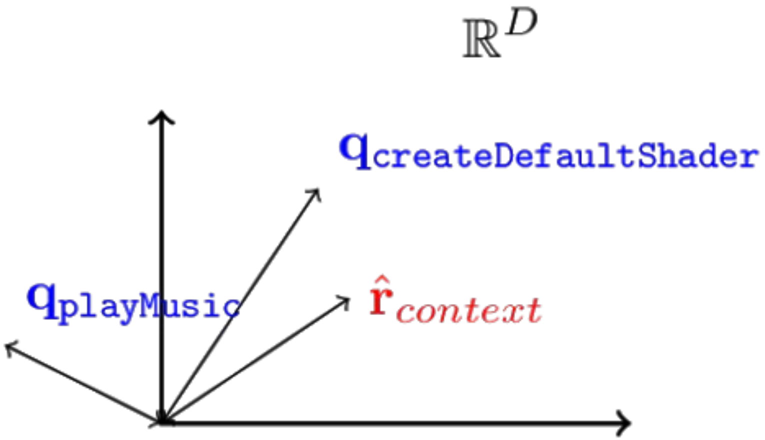
- create
- create?UNK?
- init
- createShader

A Machine Learning Model of Names

$$P(t|c) = \frac{exp(s_\theta(t,c))}{\sum_{t'} exp(s_\theta(t',c))}$$

[Mnih, Hinton, 2007; Mnih, Teh, 2012; Mnih, Kavukcuoglu, 2013; Maddison, Tarlow, 2014]

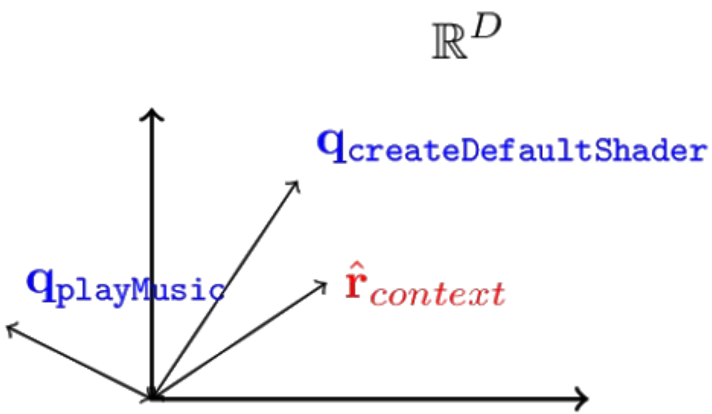
Embedding Identifiers



$\mathbf{q}_t \in \mathbb{R}^D$ are "embeddings" ::: model parameters

[Mnih, Hinton, 2007; Mnih, Teh, 2012; Mnih, Kavukcuoglu, 2013; Maddison, Tarlow, 2014]

Embedding Identifiers



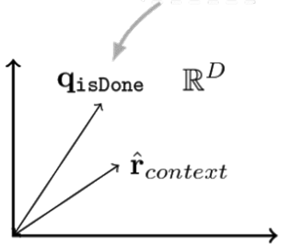
$$s_\theta(t, c) = \hat{\mathbf{r}}_{context}^\top q_t + b_t$$

$$P(t|c) = \frac{exp(s_\theta(t,c))}{\sum_{t'} exp(s_\theta(t',c))}$$

[Mnih, Hinton, 2007; Mnih, Teh, 2012; Mnih, Kavukcuoglu, 2013; Maddison, Tarlow, 2014]

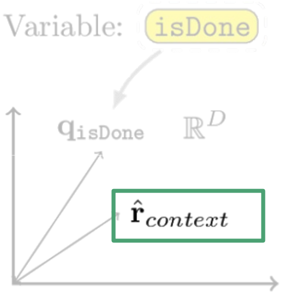
Neural Context Models of Source Code

Variable: isDone



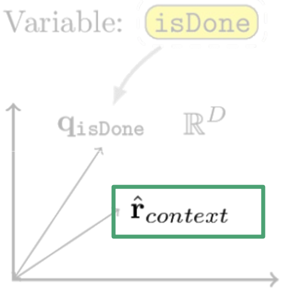
$$s_\theta(.) = \hat{\mathbf{r}}_{context}^\top \mathbf{q}_{\texttt{isDone}} + b_{\texttt{isDone}}$$

Neural Context Models of Source Code



$$s_\theta(.) = \hat{\mathbf{r}}_{context}^\top \mathbf{q}_{\texttt{isDone}} + b_{\texttt{isDone}}$$

Neural Context Models of Source Code

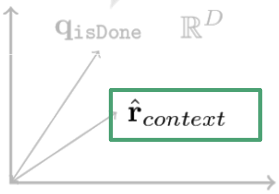


$$s_\theta(.) = \hat{\mathbf{r}}_{context}^\top \mathbf{q}_{\texttt{isDone}} + b_{\texttt{isDone}}$$

Features:
`boolean`, `in:MethodBody`, `final`

Neural Context Models of Source Code

Variable: `isDone`

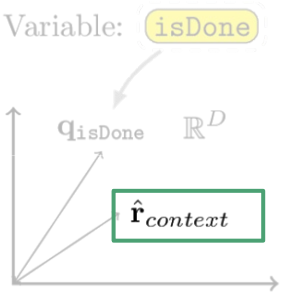


$$s_\theta(.) = \hat{\mathbf{r}}_{context}^\top \mathbf{q}_{\texttt{isDone}} + b_{\texttt{isDone}}$$

Features:
`boolean` , `in:MethodBody` , `final`

Contexts:
`while ( ! isDone ) {`

Neural Context Models of Source Code



$$s_\theta(.) = \hat{\mathbf{r}}_{context}^\top \mathbf{q}_{\texttt{isDone}} + b_{\texttt{isDone}}$$

Features:

`boolean` , `in:MethodBody` , `final`

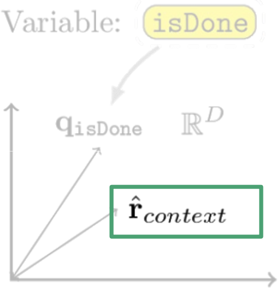
$$\mathbf{r}_{\texttt{boolean}} + \mathbf{r}_{\texttt{in:MethodBody}} + \mathbf{r}_{\texttt{final}}$$

Contexts:

`while ( ! isDone ) {`

$$C_{-2}\mathbf{r}_{(} + C_{-1}\mathbf{r}_{!} + C_{1}\mathbf{r}_{)} + C_{2}\mathbf{r}_{\}}$$

Neural Context Models of Source Code



$$s_\theta(.) = \hat{\mathbf{r}}_{context}^\top \mathbf{q}_{\texttt{isDone}} + b_{\texttt{isDone}}$$

Features:
`boolean` , `in:MethodBody` , `final`

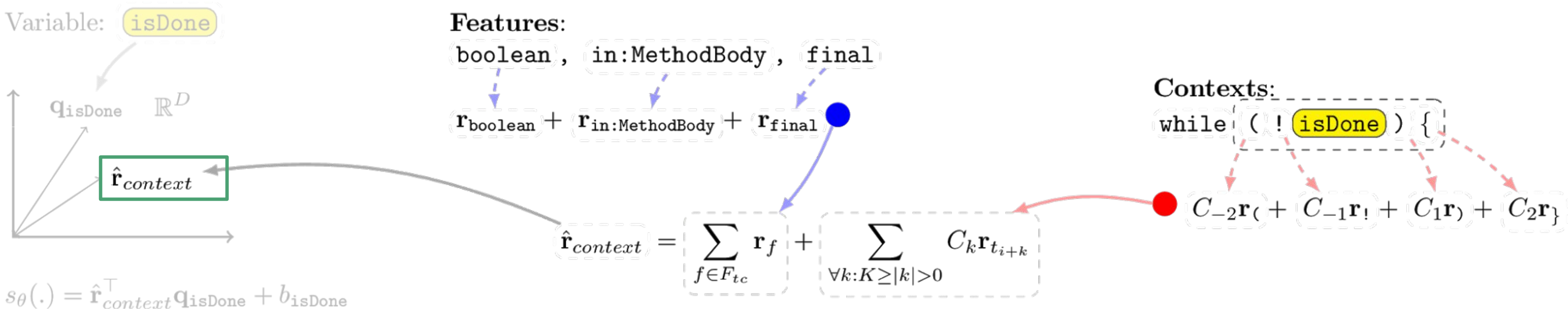
$$\mathbf{r}_{\texttt{boolean}} + \mathbf{r}_{\texttt{in:MethodBody}} + \mathbf{r}_{\texttt{final}}$$

Contexts:
`while ( ! isDone ) {`

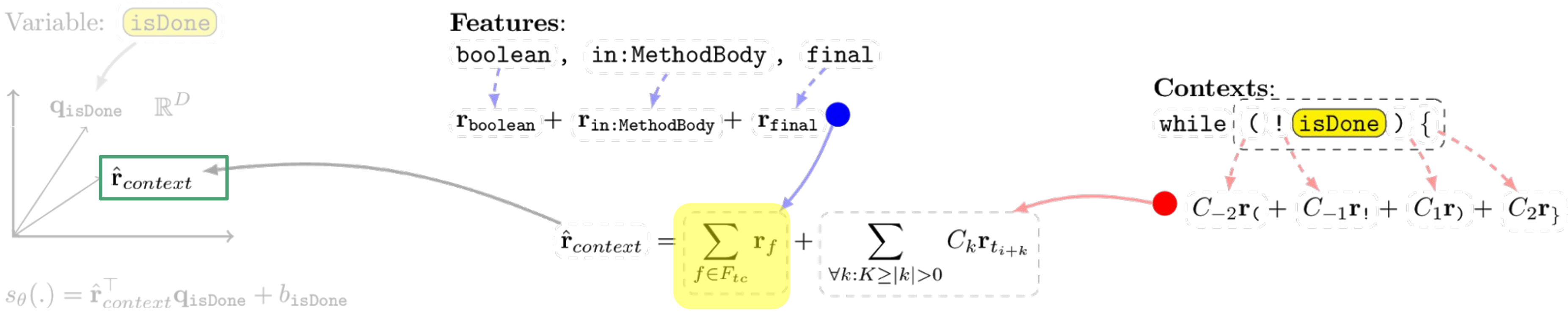
$$C_{-2}\mathbf{r}_{(} + C_{-1}\mathbf{r}_{!} + C_{1}\mathbf{r}_{)} + C_{2}\mathbf{r}_{\}}$$

$$\hat{\mathbf{r}}_{context} = \sum_{f \in F_{tc}} \mathbf{r}_f + \sum_{\forall k: K \geq |k| > 0} C_k \mathbf{r}_{t_{i+k}}$$

Neural Context Models of Source Code

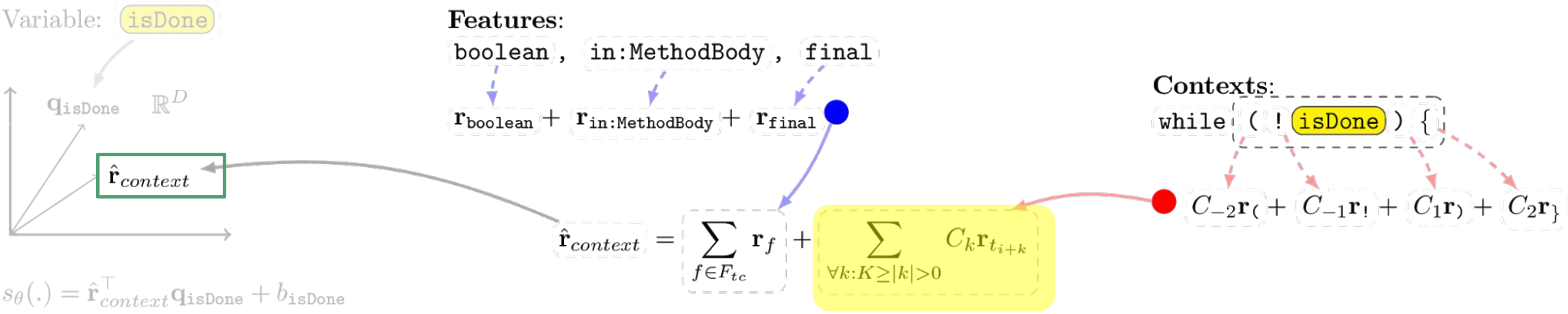


Neural Context Models of Source Code



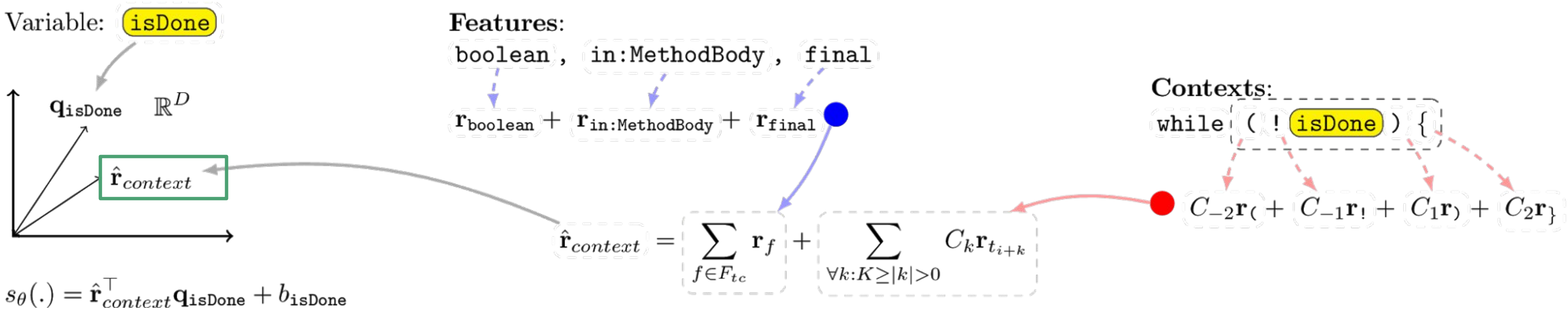
Global Information

Neural Context Models of Source Code

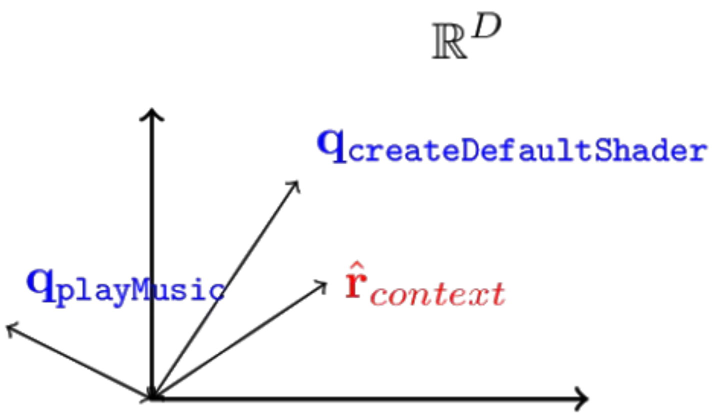


Local Information

Neural Context Models of Source Code



Embedding Identifiers



$$P(t|c) = \frac{exp(s_\theta(t,c))}{\sum_{t'} exp(s_\theta(t',c))}$$

[Mnih, Hinton, 2007; Mnih, Teh, 2012; Mnih, Kavukcuoglu, 2013; Maddison, Tarlow, 2014]

Neologisms

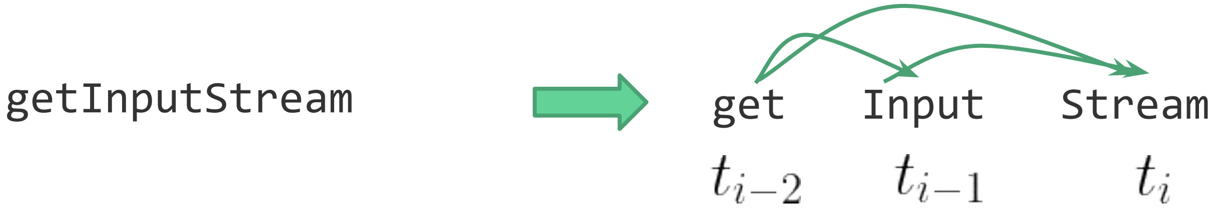
neologism

[nɪˈɒlədʒɪz(ə)m]

NOUN

a newly coined word or expression.
synonyms: new word · new expression · new term · new phrase · coinage · More

Subtoken Context Models of Code



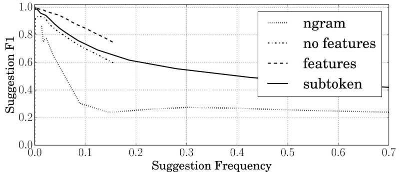
$$P\left(t_i | t_{i-1}, t_{i-2}, context\right)$$

Sequentially predict each subtoken given the context and the previous subtokens

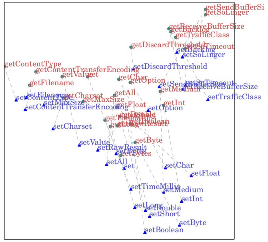
Training Data (project)

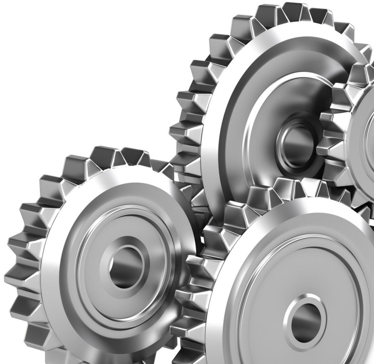
Train Neural Network

Suggest Names on Test Data



Embeddings





Evaluation Methodology

Test File

```
ForkJoinTask<?> [redacted];
if (task instanceof ForkJoinTask<?>) // avoid re-wrap
    [redacted] = (ForkJoinTask<?>) task;
else
    [redacted] = new
ForkJoinTask.AdaptedRunnableAction(task);
externalPush([redacted]);
```

Suggestions

1. job (30%)
2. task (20%)
3. tsk (15%)

Evaluation on top 10 Java GitHub projects.

Perturb existing code and **retrieve** ground truth.

Evaluation Methodology

Test File

```
ForkJoinTask<?> [redacted];
if (task instanceof ForkJoinTask<?>) // avoid re-wrap
    [redacted] = (ForkJoinTask<?>) task;
else
    [redacted] = new
ForkJoinTask.AdaptedRunnableAction(task);
externalPush([redacted]);
```

```
ForkJoinTask<?> job;
if (task instanceof ForkJoinTask<?>) // avoid re-wrap
    job = (ForkJoinTask<?>) task;
else
    job = new
ForkJoinTask.AdaptedRunnableAction(task);
externalPush(job);
```

Suggestions

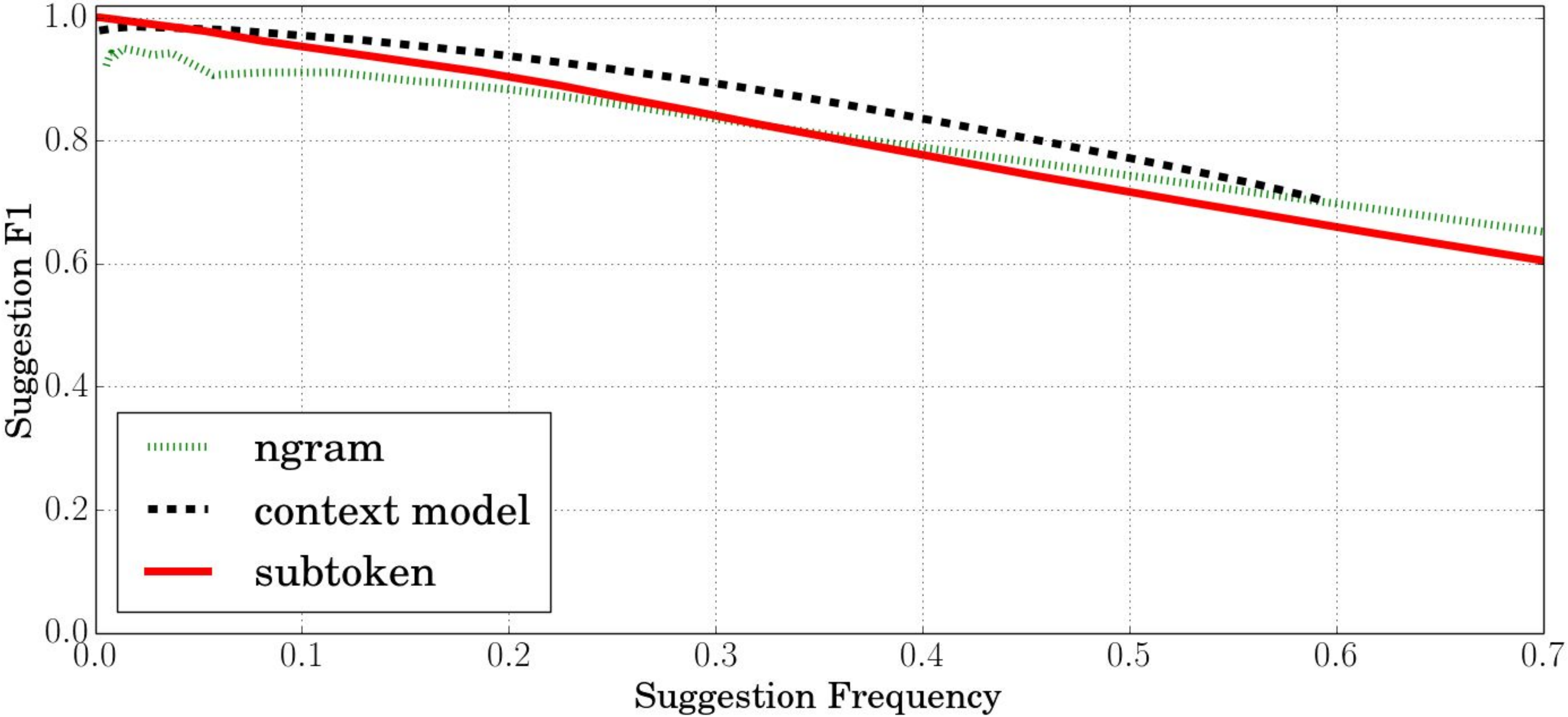
1. job (30%)
2. task (20%)
3. tsk (15%)

compare with ground truth

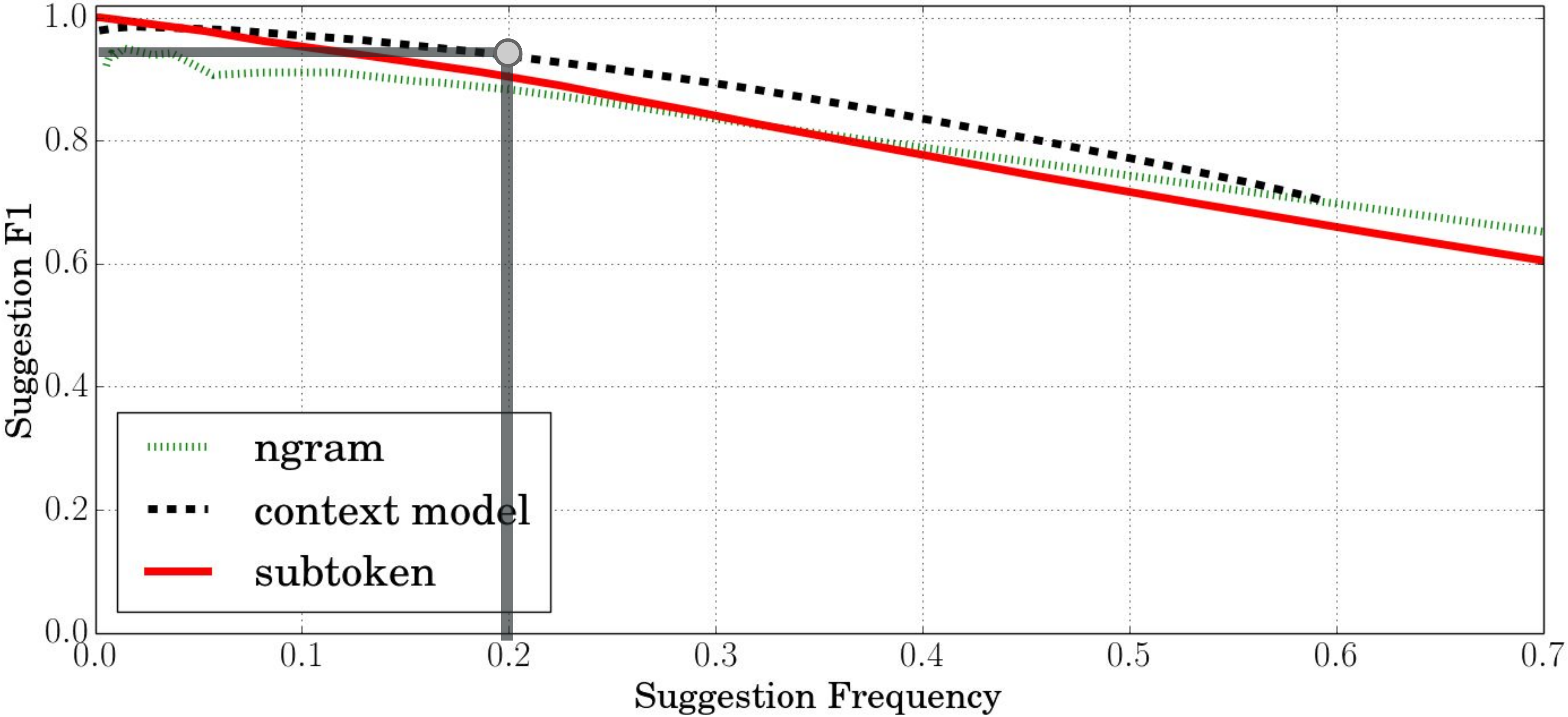
Evaluation on top 10 Java GitHub projects.

Perturb existing code and **retrieve** ground truth.

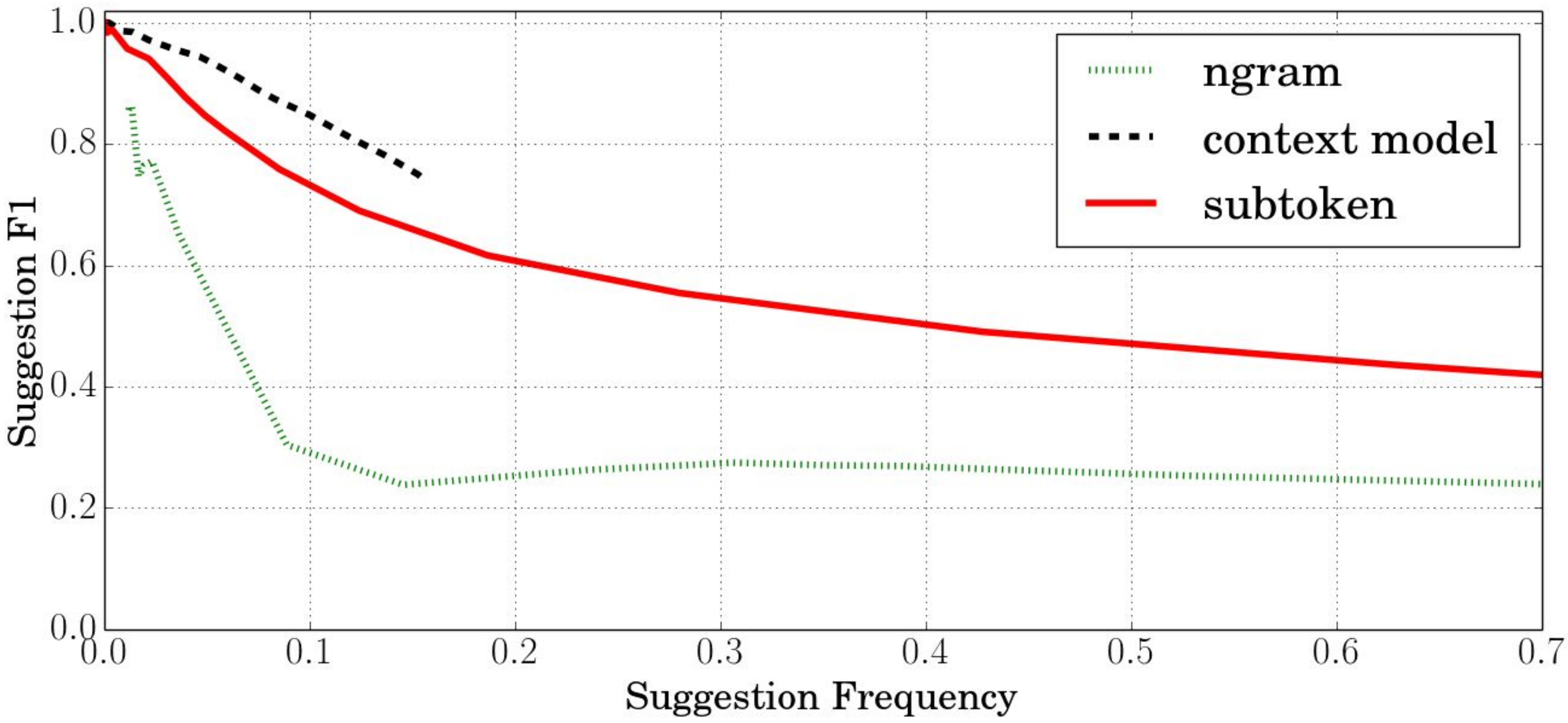
Suggesting Variable Names



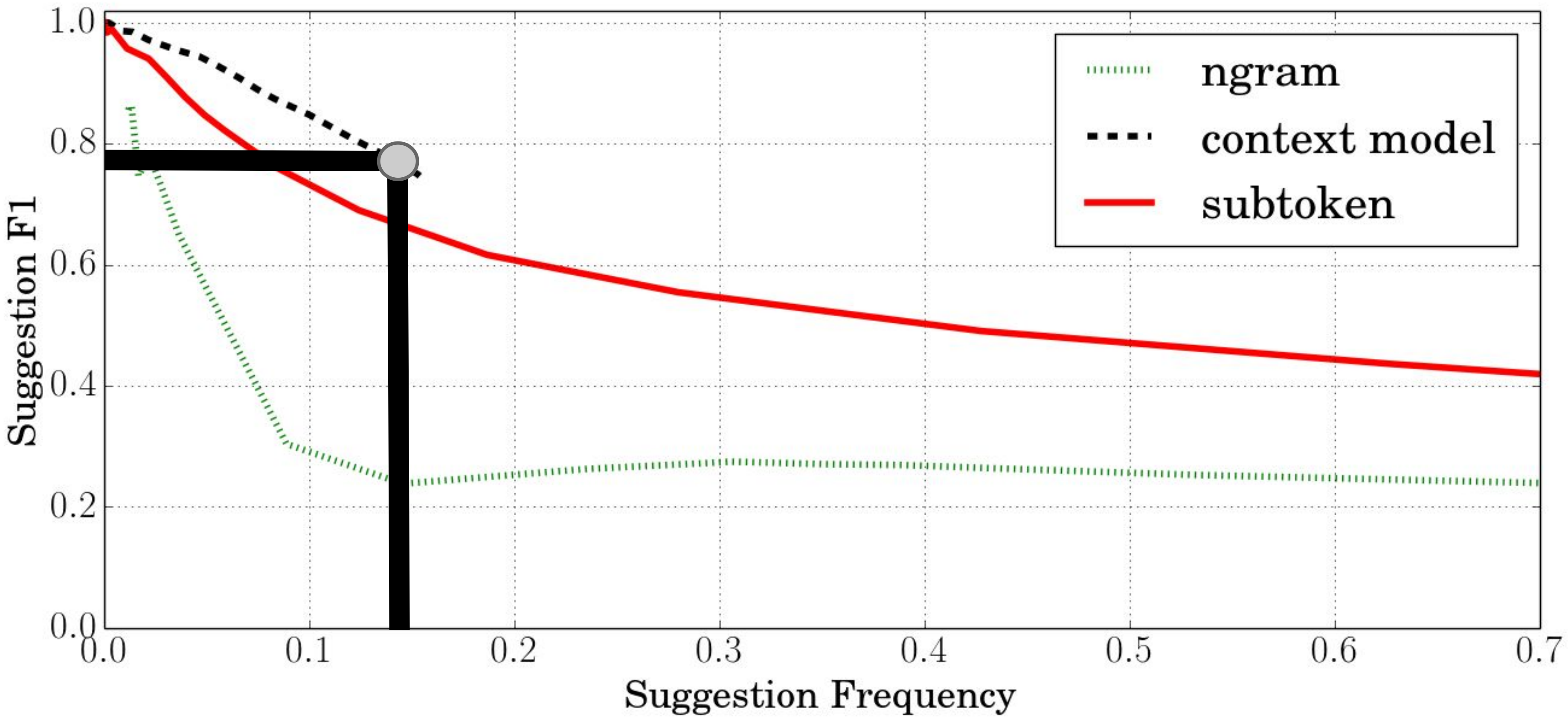
Suggesting Variable Names



Suggesting Method Names

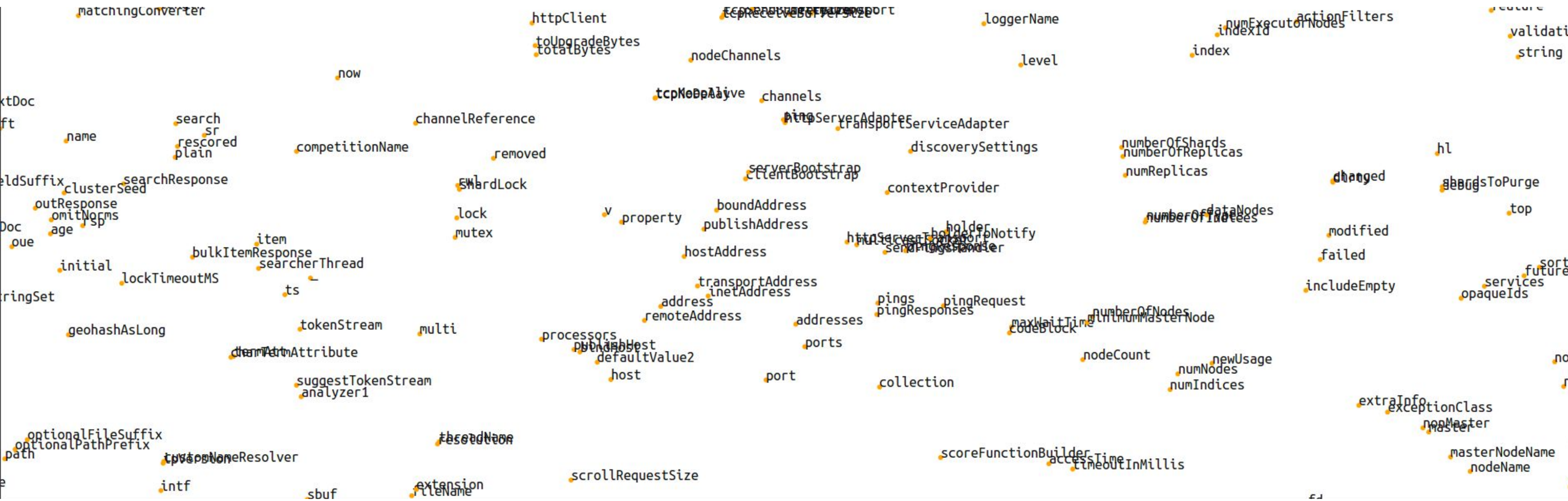


Suggesting Method Names



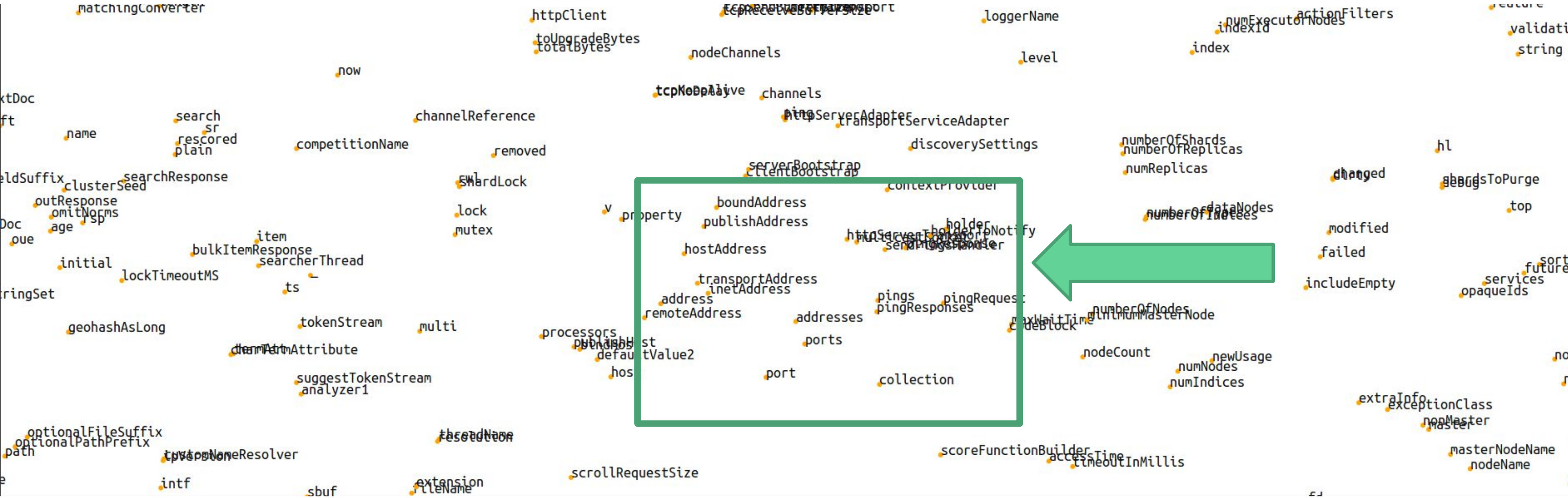
Embedding Visualization

http://groups.inf.ed.ac.uk/cup/naturalize



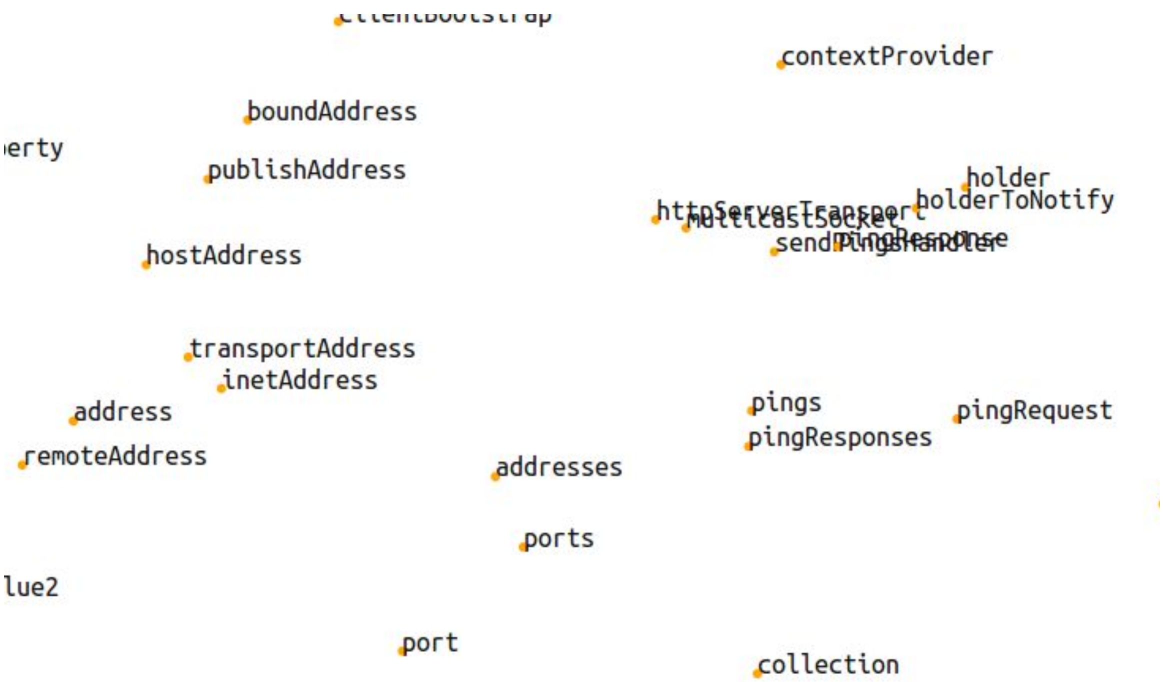
Embedding Visualization

http://groups.inf.ed.ac.uk/cup/naturalize



Embedding Visualization

http://groups.inf.ed.ac.uk/cup/naturalize



Learning to map natural language to source code

Work done in Microsoft Research - Cambridge

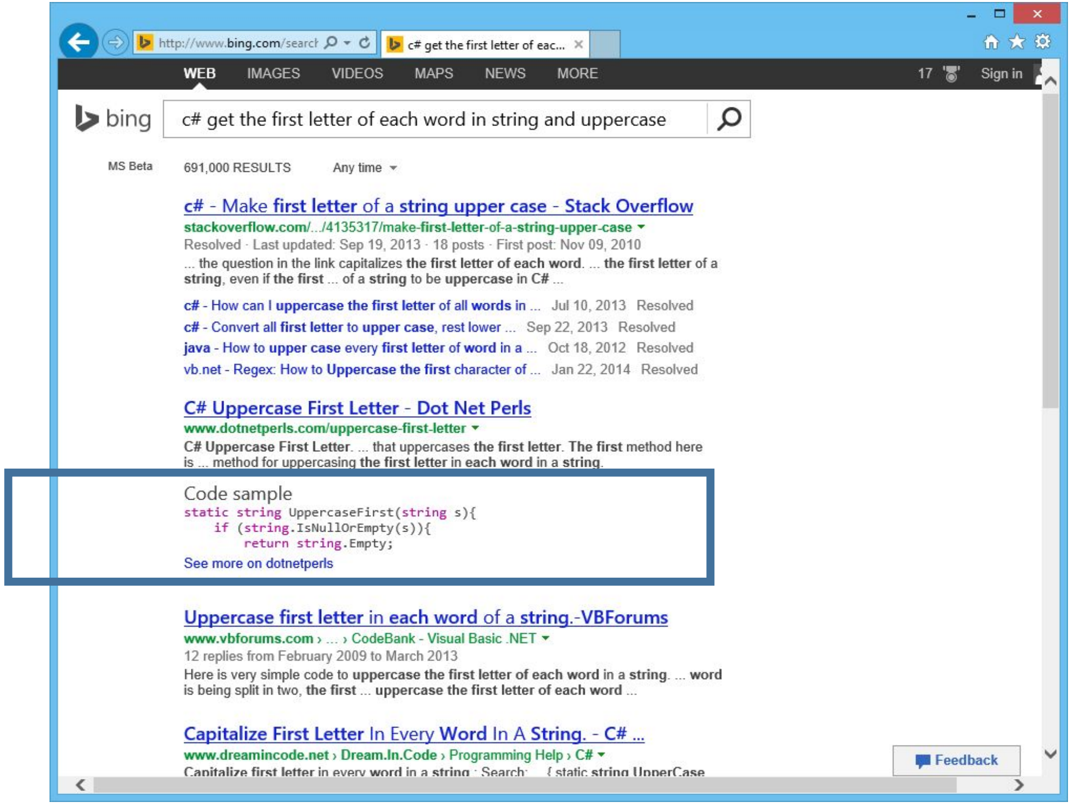
Joint work with Danny Tarlow, Yi Wei, Andy Gordon

Applications of Joint Models of Code & NL

Code Retrieval

NL Retrieval
for Source Code

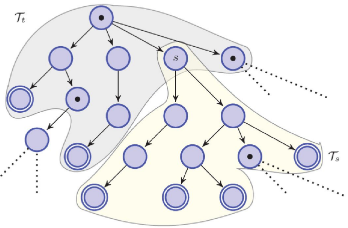
and eventually code synthesis...



A Conditional Generative Model

“get the first letter of each word in string and uppercase”

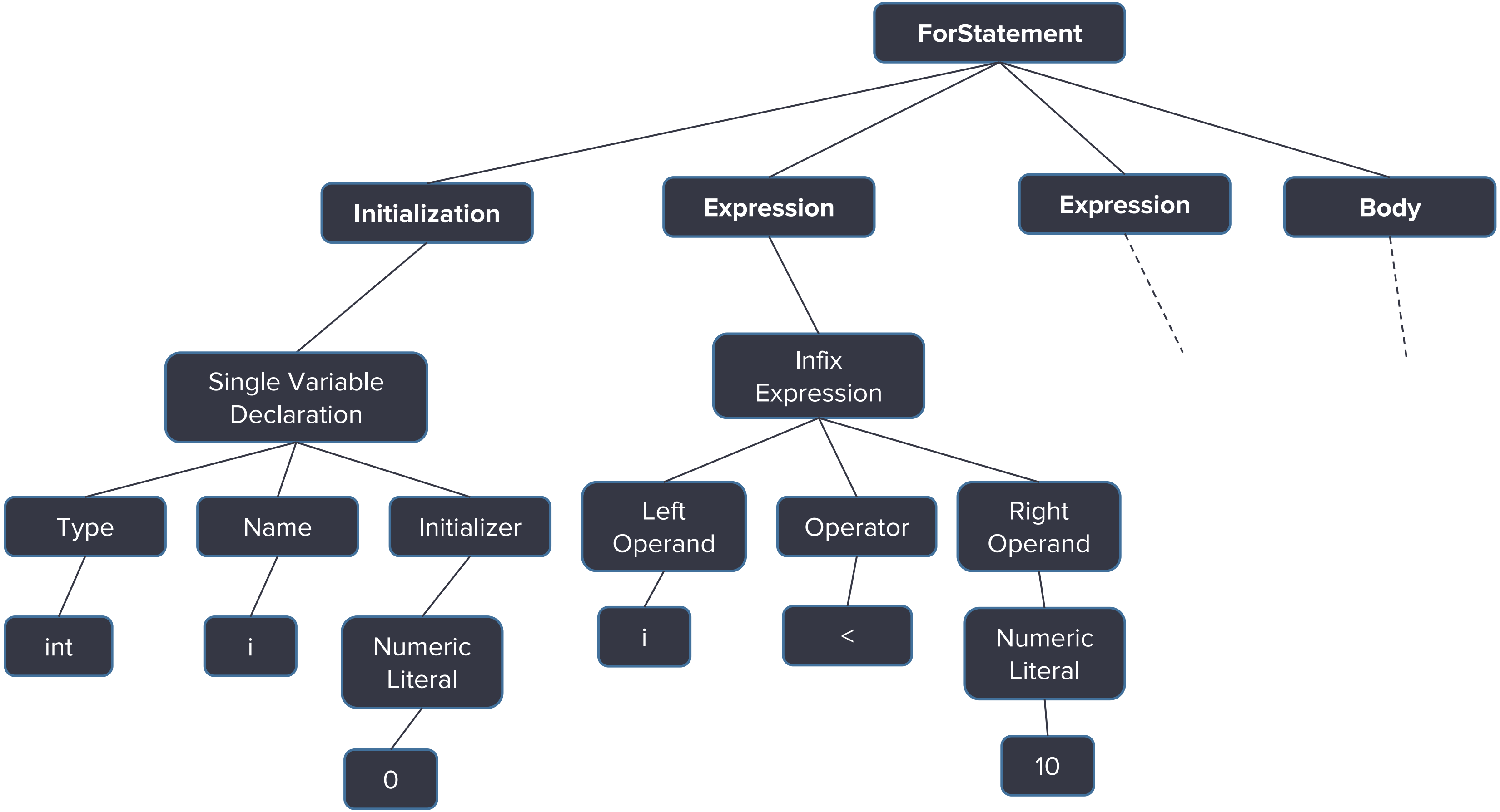
NL Query



Conditional Generative Model of Source Code

```
string s;
string[] words = s.ToUpper().split(‘ ‘);
string[] firstLetters = new string[words.Length];
for (int i=0; i < words.Length; i++) {
        firstLetters[i] = words.Substring(0,1);
}
```

Synthesize/Score Code Snippet



Syntactic model of source code, *i.e.* model how AST is generated

Tree Generation Model: Context Free Grammars (CFG)

$$E \rightarrow E + E$$

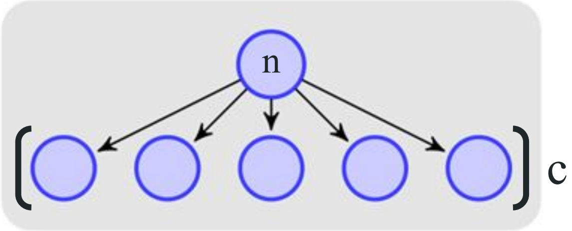
$$E \rightarrow T$$

$$F \rightarrow (E)$$

$$T \rightarrow F * F$$

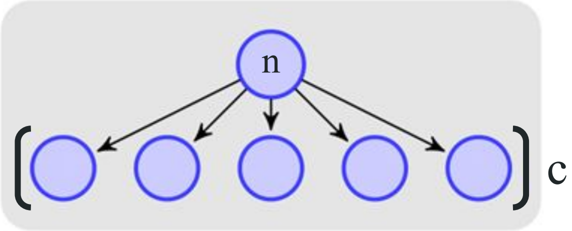
$$T \rightarrow F$$

$$F \rightarrow id$$



Tree Generation Model: Probabilistic Context Free Grammars (PCFG)

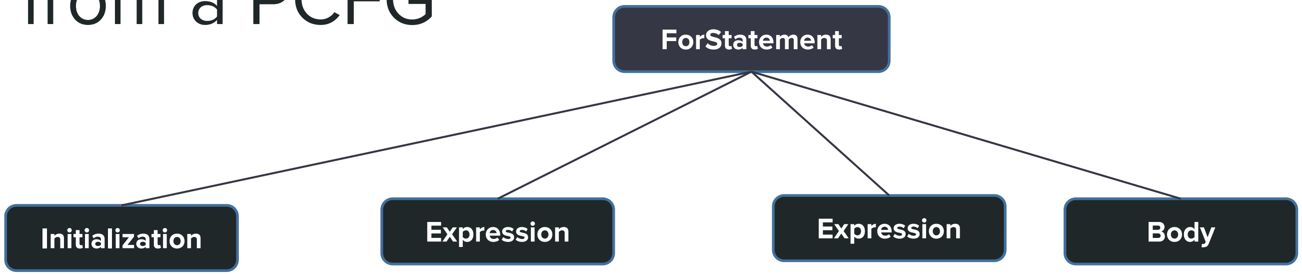
$$
\begin{array}{llll}
E \to E + E & \text{(prob 0.7)} & T \to F * F & \text{(prob 0.6)} \\
E \to T & \text{(prob 0.3)} & T \to F & \text{(prob 0.4)} \\
F \to (E) & \text{(prob 0.1)} & F \to id & \text{(prob 0.9)}
\end{array}
$$



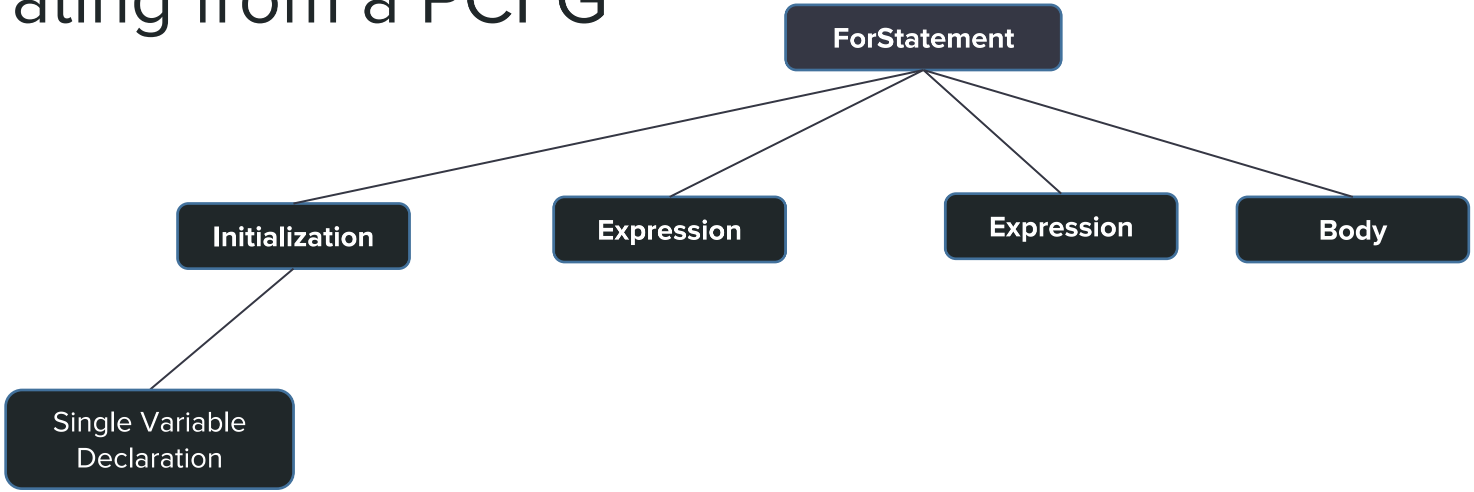
Generating from a PCFG

ForStatement

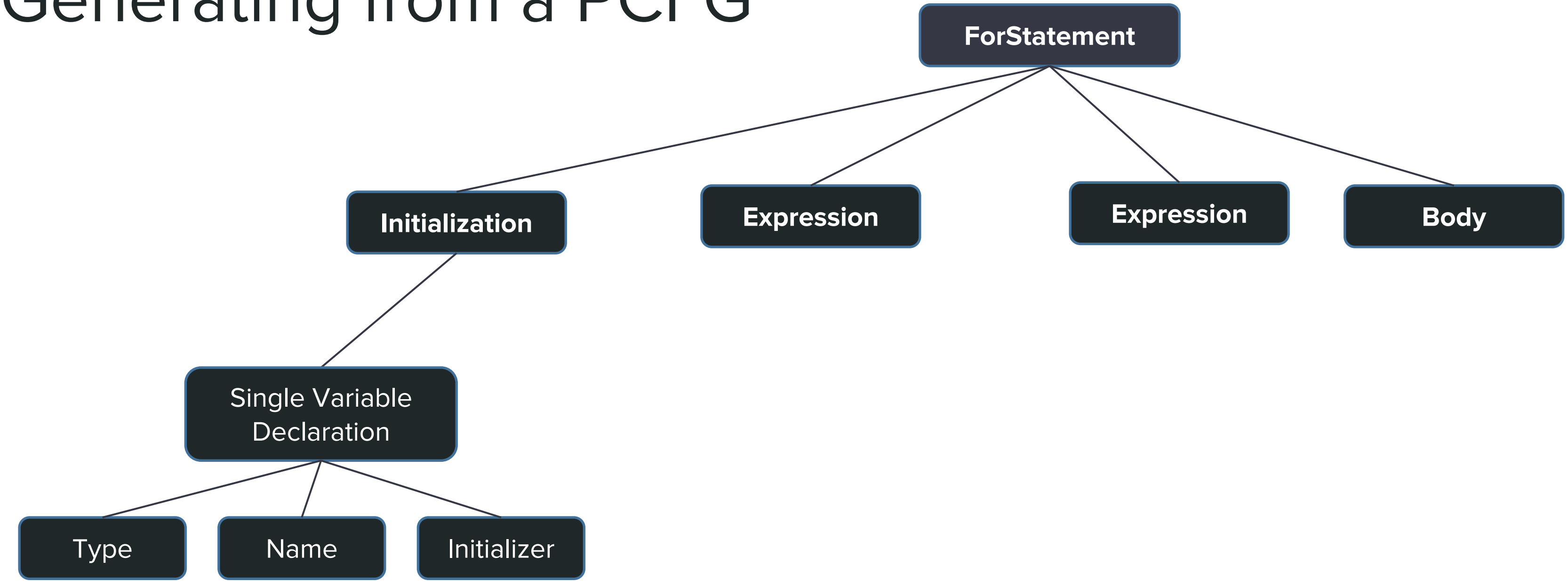
Generating from a PCFG



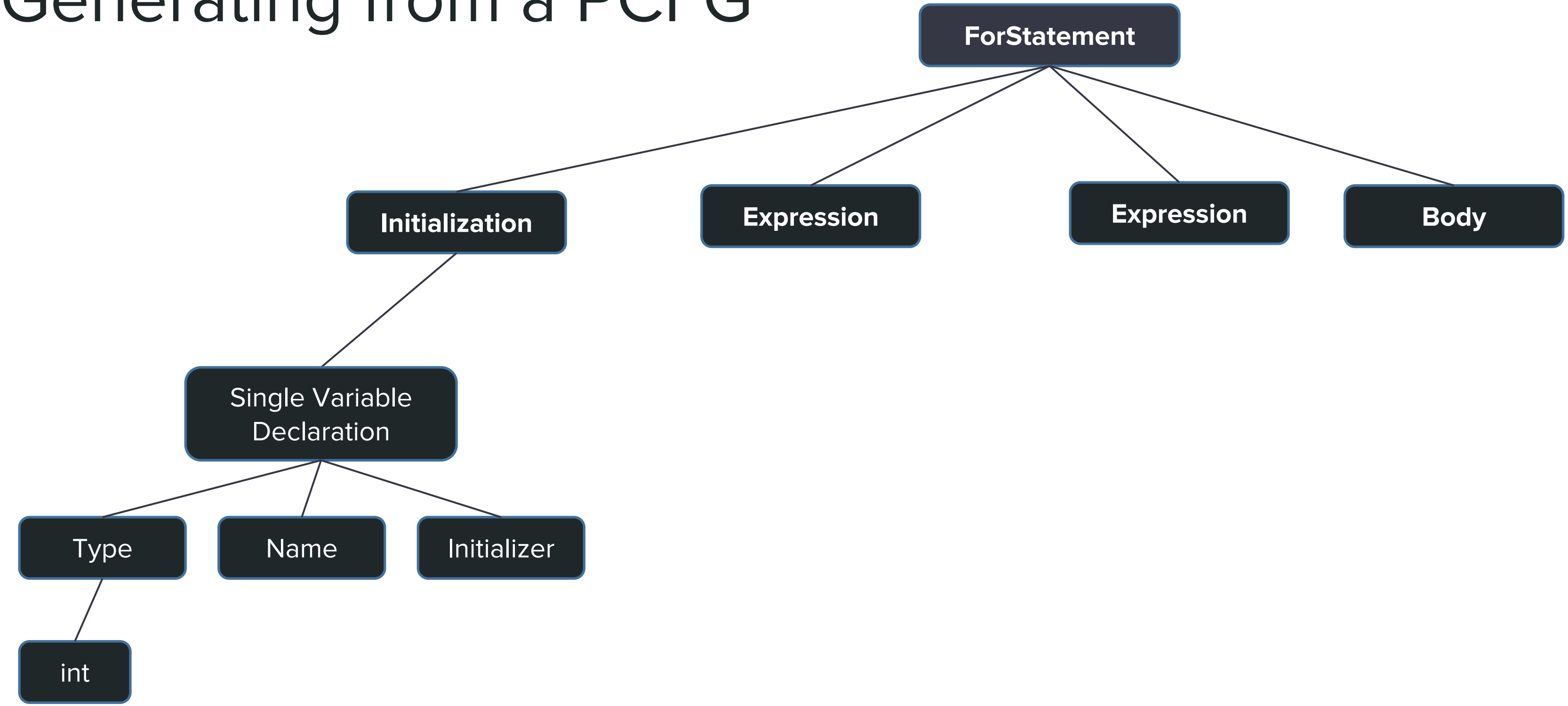
Generating from a PCFG



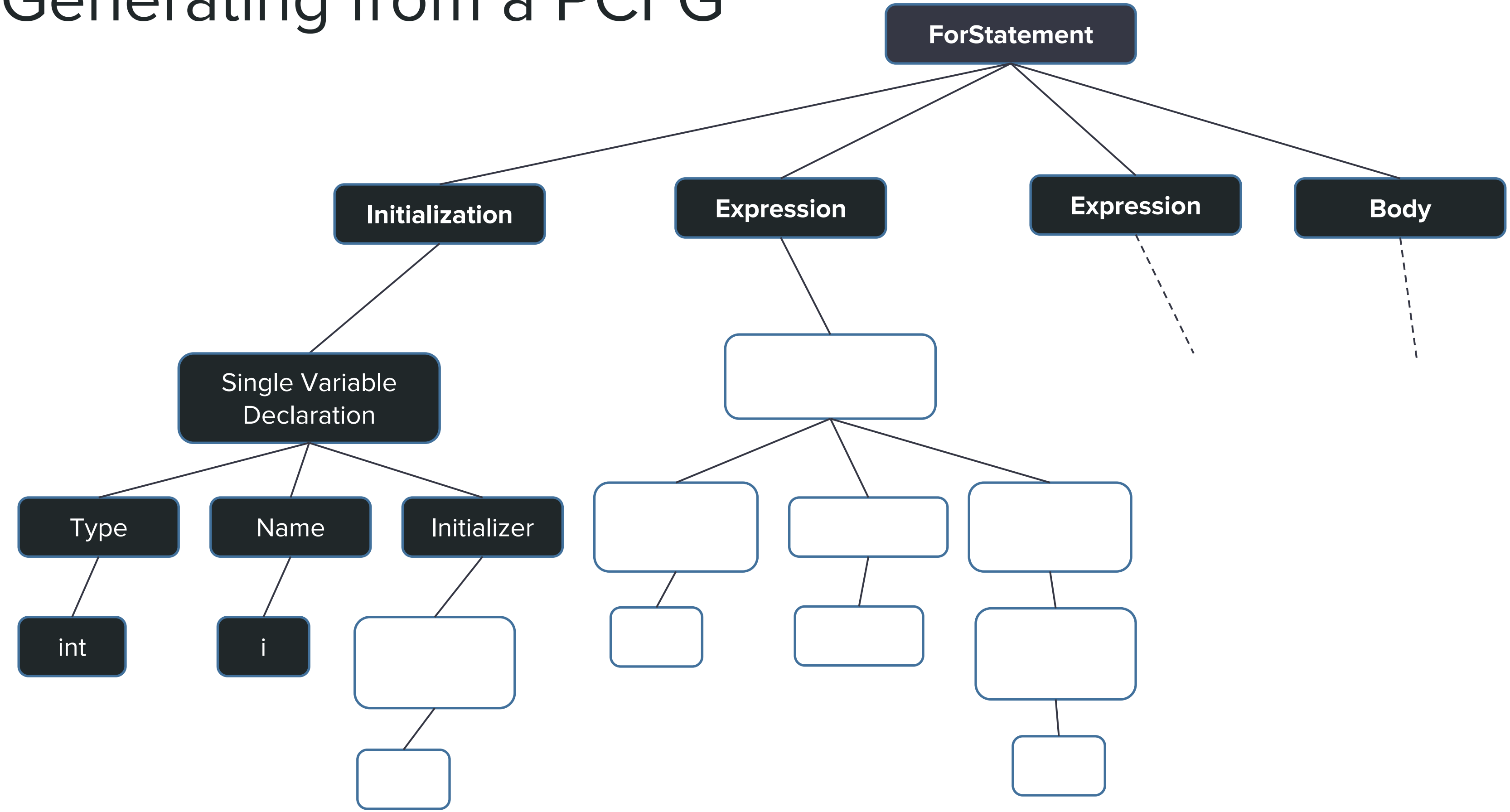
Generating from a PCFG



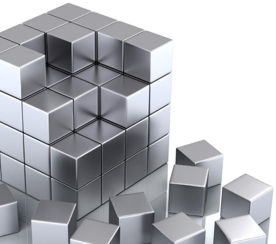
Generating from a PCFG



Generating from a PCFG



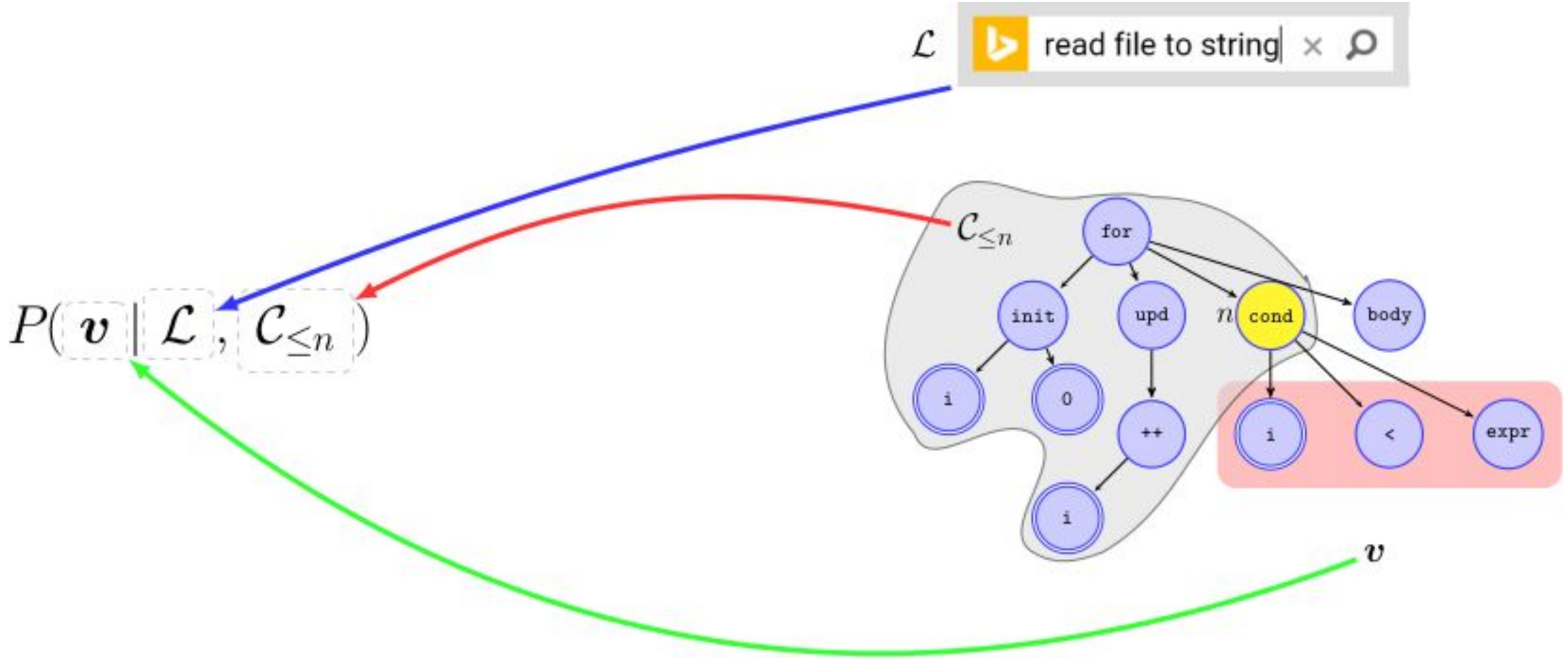
Conditional Generative Model of Source Code



Given natural language, get a model that can **generate** (probabilistically) source code, *i.e.*

P(code | natural language)

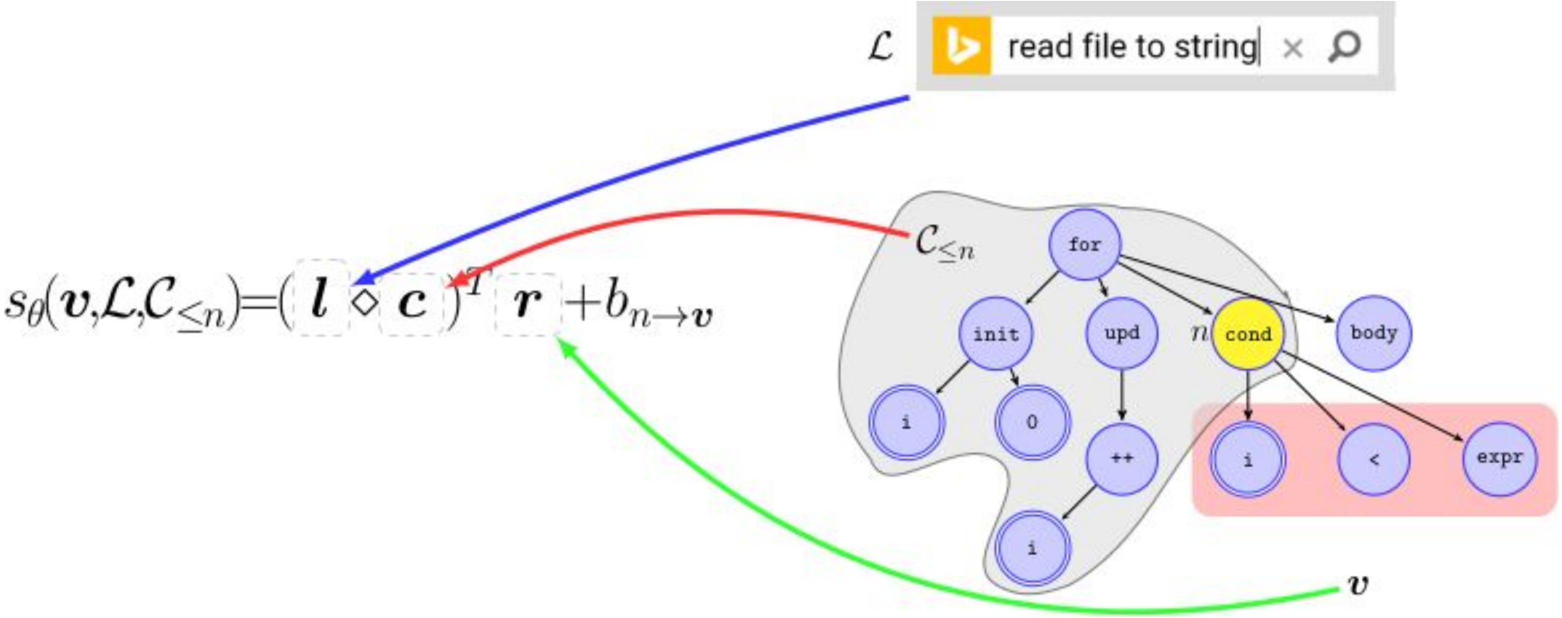
A Neural Log-Bilinear Bimodal Model of Code



Kiros, Ryan, Ruslan Salakhutdinov, and Rich Zemel. "Multimodal neural language models."

Maddison, Chris and Daniel Tarlow. "Structured generative models of natural source code."

A Neural Log-Bilinear Bimodal Model of Code



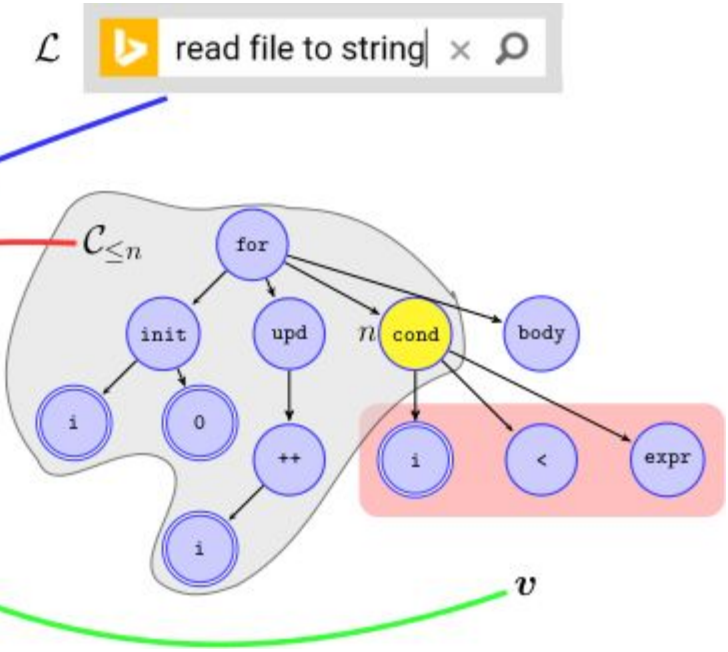
Kiros, Ryan, Ruslan Salakhutdinov, and Rich Zemel. "Multimodal neural language models."

Maddison, Chris and Daniel Tarlow. "Structured generative models of natural source code."

A Neural Log-Bilinear Bimodal Model of Code

$$P(\boldsymbol{v}|\,\mathcal{L},\mathcal{C}_{\leq n}) \propto \exp s_\theta(\boldsymbol{v},\mathcal{L},\mathcal{C}_{\leq n})$$

$$s_\theta(\boldsymbol{v},\mathcal{L},\mathcal{C}_{\leq n}) = (\boldsymbol{l} \diamond \boldsymbol{c})^T \boldsymbol{r} + b_{n\to\boldsymbol{v}}$$



Kiros, Ryan, Ruslan Salakhutdinov, and Rich Zemel. "Multimodal neural language models."

Maddison, Chris and Daniel Tarlow. "Structured generative models of natural source code."

StackOverflow Data & Augmenting Data with bing Queries

Natural Language “Query”

Code Snippets

stackoverflow

How do I enumerate an enum

1866

```
foreach (Suit suit in (Suit[]) Enum.GetValues(typeof(Suit)))
{
}
```

share | improve this answer

edited Nov 7 '13 at 0:00
Zain Rizvi
4,418 ●8 ●24 ●57

answered Sep 19 '08 at 20:37
jop
32.9k ●8 ●36 ●50

C# Configuration Manager . ConnectionStrings

21

Check your `machine.config` . If you only want your entry, you can add a `<clear />` element to the `<connectionStrings>` element like so ...

```
<connectionStrings>
  <clear />
  <add name="Target"
       connectionString=
             "server=MYSERVER; Database=MYDB; Integrated Security=SSPI;" />
</connectionStrings>
```

share | improve this answer

edited Jul 16 '12 at 14:36
SteveC
2,783 ●5 ●33 ●66

answered May 13 '10 at 15:36
Ryan Rinaldi
1,681 ●8 ●19

30K C# Questions
40K C# Snippets

stackoverflow + bing

How do I enumerate an enum

1866

```
foreach (Suit suit in (Suit[]) Enum.GetValues(typeof(Suit)))
{
}
```

share | improve this answer

edited Nov 7 '13 at 0:00
Zain Rizvi
4,418 •8 •24 •57

answered Sep 19 '08 at 20:37
jop
32.9k •8 •36 •50

http://stackoverflow.com/questions/105372/how-do-i-enumerate-an-enum

C# enum

C# foreach enum

C# enumerate enum

How to order enum values in C#

foreach enum C#

C# enumerate enumerations

C# enumeration

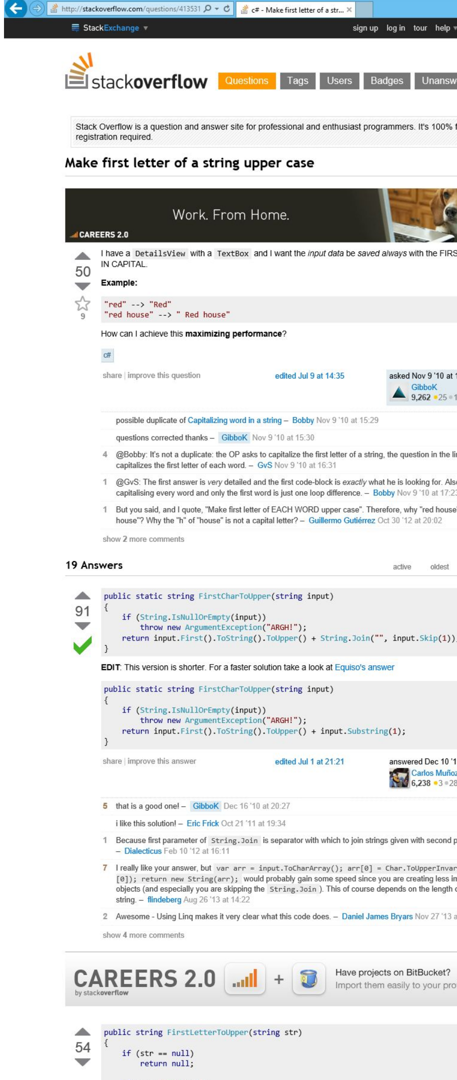
C# enumerator

C# foreach enum values

C# enumerate

stackoverflow ✚ bing

40,092 C# Snippets
6,355,393 Natural Language Queries



Performance Metric: Mean Reciprocal Rank

$$\text{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\textsc{Rank}(i)}$$

Measures how well we rank the *correct* answer

Retrieval Evaluation - MRR Performance

Code Retrieval	Model	StackOverflow Test 1	StackOverflow Test 2
	NL+Code	**0.18**	**0.17**
	NL only	0.12	0.13

Query Retrieval	Model	StackOverflow Test 1	StackOverflow Test 2
	Multiplicative	**0.43**	**0.41**
	NL only	0.25	0.26

Test 1: *Code snippets from training set with new natural language queries.*
Test 2: *New code snippets and new natural language queries.*

Synthesis Samples

> timespan day the week

```
DateTime DateTime=DateTime.Now(0);
```

> file exists on directory

```
var path = new File(directory)
```

Synthesis Samples

> timespan day the week

```
DateTime DateTime=DateTime.Now(0);
```

```
foreach(string s in Days(new DateTime(2010, 1, 1), new DateTime(2010, 2, 1))) {
    Console.WriteLine(s);
}
```

> file exists on directory

```
var path = new File(directory)
```

```
System.IO.File.Exists(path)
```

Retrieval Sample

stackoverflow

```
path=Path.GetFullPathInternal(path);
new FileIOPermission(
   FileIOPermissionAccess.Read,
   new string[] { path },
     false, false).Demand();
flag = InternalExists(path);
```

1. wpf get directory name from path
2. **determine a file exist on shared folder**
3. open file dialog class
4. create directory pathname
5. load binary file to variable

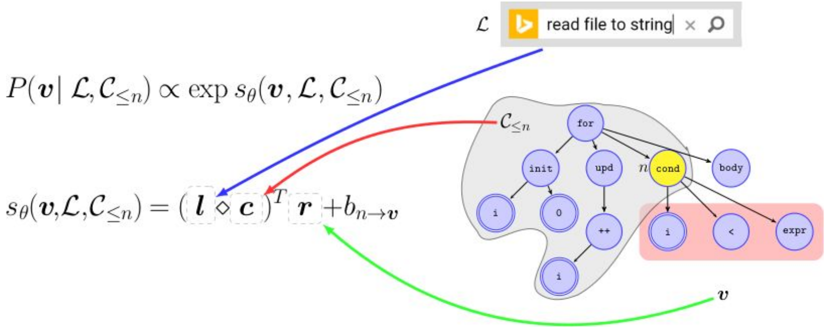
Challenges

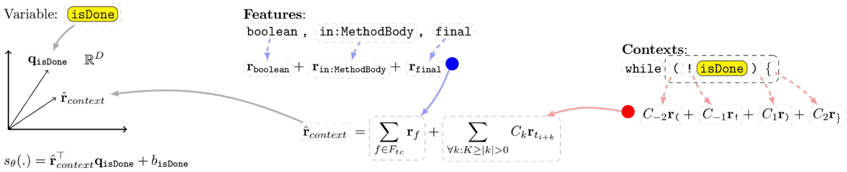
❯ Code Representations in Machine Learning

❯ Define Representative Evaluation Metrics for Software Engineering Tasks

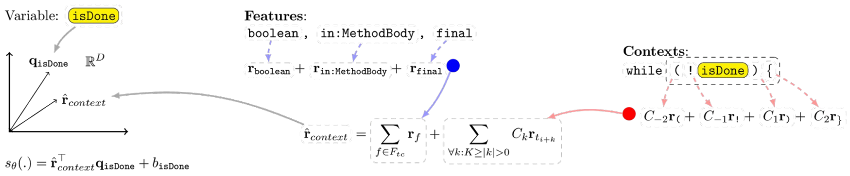
❯ Create Useful and Efficient Software Engineering Tools

Understanding Source Code through Machine Learning to Create Smart Software Engineering Tools



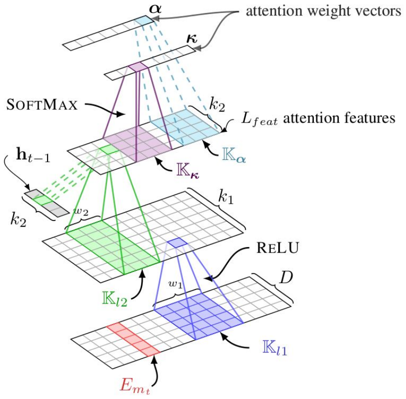


(end)

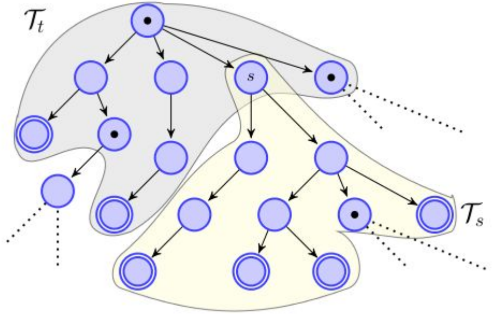


Learning Naming Conventions
Allamanis et al, 2014; 2015

n-gram LMs for Code
Allamanis et al, 2013



Learning to name source code



Mining Source Code Idioms
Allamanis and Sutton, 2014

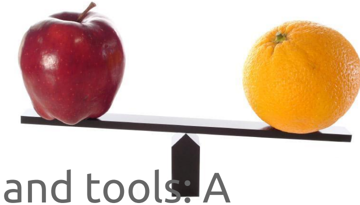
The Sympathetic Uniqueness Principle

Rare names often usefully signify unusual functionality, and need to be preserved.

- Prune rare words
- Repurpose special UNK token
- Allows Naturalize to decide when it should *not* suggest

```
public void execute(Runnable task) {
        if (task == null)
            throw new NullPointerException();
        ForkJoinTask<?> job;
        if (task instanceof ForkJoinTask<?>) // avoid
            job = (ForkJoinTask<?>) task;
        else
            job = new ForkJoinTask.AdaptedRunnableActi
        externalPush(job);
    }
```

Idioms vs. the Rest



Code Clones copy-paste code fragments

C. K. Roy et al. Comparison and evaluation of code clone detection techniques and tools: A qualitative approach. Science of Computer Programming, 2009.

L. Jiang et al. Deckard: Scalable and accurate tree-based detection of code clones. ICSE 2007.

H. A. Basit and S. Jarzabek. A data mining approach for detecting higher-level clones in software. IEEE Transactions on Software Engineering, 2009.

API Patterns usage patterns of methods

T. T. Nguyen et al. Graph-based mining of multiple object usage patterns. ESEC/FSE 2009.

J. Wang et al. Mining succinct and high-coverage API usage patterns from source code. MSR 2013.

H. Zhong et al. MAPO: Mining and recommending API usage patterns. ECOOP, 2009.

Idioms syntactic code fragments

The Distributional Hypothesis

“*You shall know a word by the company it keeps*”.

John Rupert Firth, 1957

The Distributional Hypothesis

The [redacted] is walking

“You shall know a word by the company it keeps”.

John Rupert Firth, 1957

For IWESEP, in a way I'm inviting you as a ML/NLP person that can teach software engineering people the cool things you can do with ML/NLP techniques. I'm not sure if you see yourself this way, but I think a quick "intro to ML/NLP" for the first 1/3 or so, then "look at all the cool things you can do" for the second 2/3 would be one potential way to give the presentation.

Sanity Check:
String Manipulation Synthetic Data

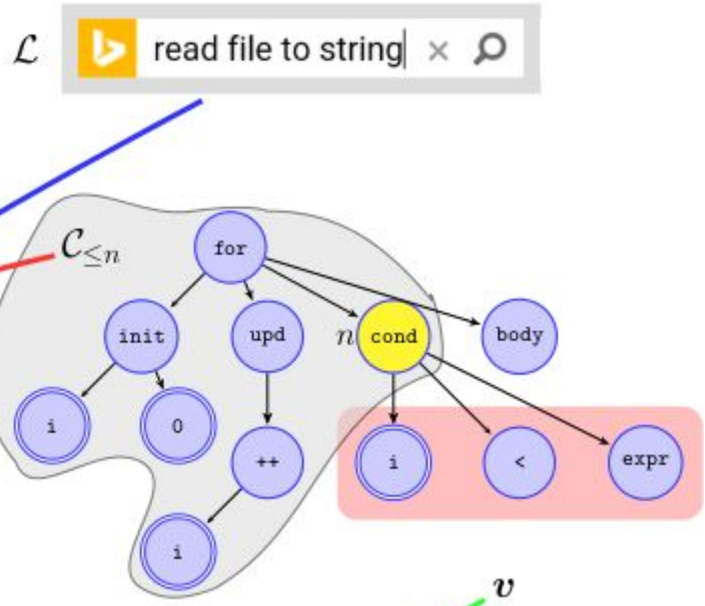
```
var result = input_string.Split(' ').Select((string x) =>
Double.parse(x))).Average();
```

each element parse double separated by a space and get mean
each element parse double separated by a space and get average
each element convert to double separated by a space and get mean
each element convert to double separated by a space and get average
each element parse to double separated by a space and get mean

A Neural Log-Bilinear Bimodal Model of Code

$$\boldsymbol{c} = \sum_{j=1}^{J} \mathbf{H}_j \boldsymbol{c}_{\phi_j}$$

$$\boldsymbol{l} = \frac{1}{|\mathcal{L}|} \sum_{w \in \mathcal{L}} \boldsymbol{l}_w$$



$$s_\theta(\boldsymbol{v}, \mathcal{L}, \mathcal{C}_{\leq n}) = (\boldsymbol{l} \diamond \boldsymbol{c})^T \boldsymbol{r} + b_{n \to \boldsymbol{v}}$$

Kiros, Ryan, Ruslan Salakhutdinov, and Rich Zemel. "Multimodal neural language models."

Maddison, Chris and Daniel Tarlow. "Structured generative models of natural source code."