




# Code Duplication and Machine Learning Models of Code

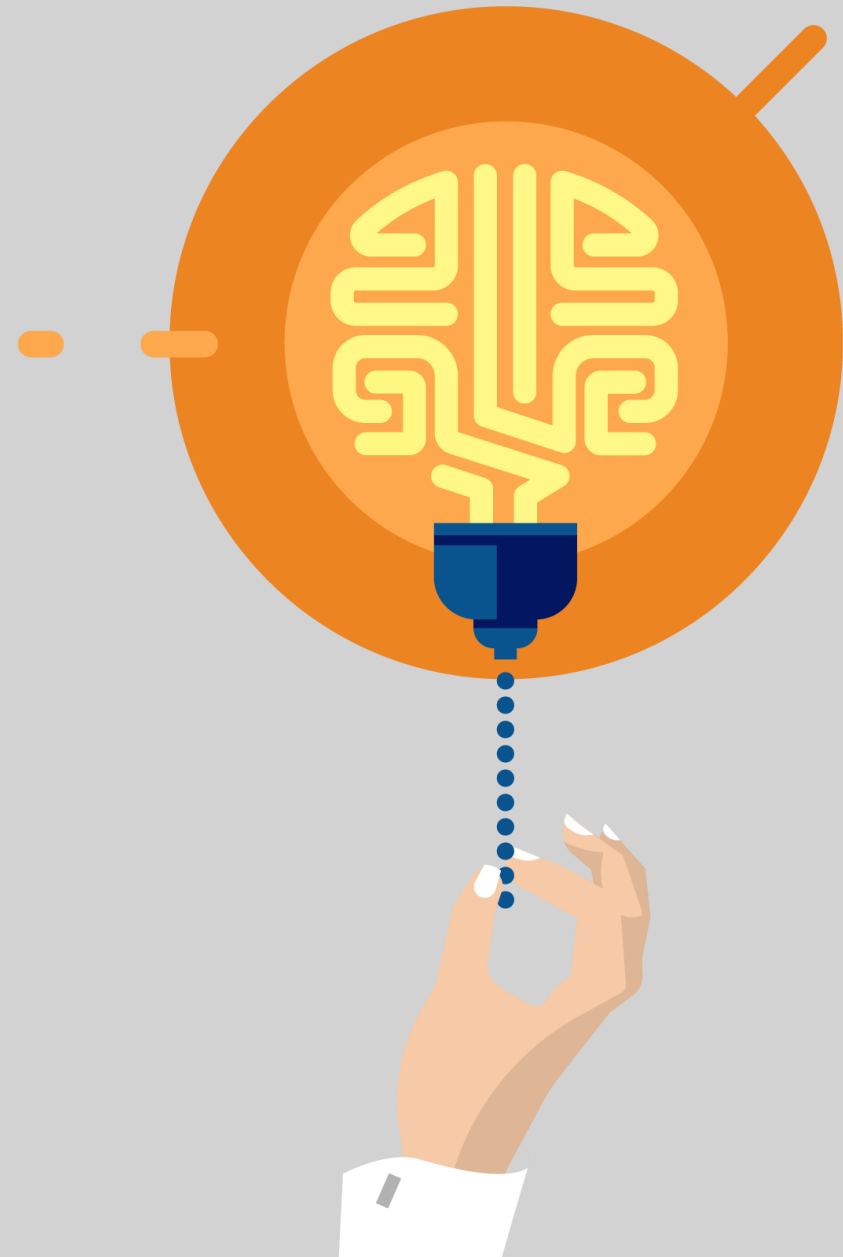
Miltos Allamanis

ML4SE 2019

 miltos1

 <https://miltos.allamanis.com>

Microsoft Research Cambridge



# What we know...



## DéjàVu: A Map of Code Duplicates on GitHub

CRISTINA V. LOPES, University of California, Irvine, USA

PETR MAJ, Czech Technical University, Czech Republic

PEDRO MARTINS, University of California, Irvine, USA

VAIBHAV SAINI, University of California, Irvine, USA

DI YANG, University of California, Irvine, USA

JAKUB ZITNY, Czech Technical University, Czech Republic

HITESH SAJNANI, Microsoft Research, USA

JAN VITEK, Northeastern University, USA

Previous studies have shown that there is a non-trivial amount of duplication in source code. This paper analyzes a corpus of 4.5 million non-fork projects hosted on GitHub representing over 428 million files written in Java, C++, Python, and JavaScript. We found that this corpus has a mere 85 million unique files. In other words, 70% of the code on GitHub consists of clones of previously created files. There is considerable variation between language ecosystems. JavaScript has the highest rate of file duplication, only 6% of the files are distinct. Java, on the other hand, has the least duplication, 60% of files are distinct. Lastly, a project-level analysis shows that between 9% and 31% of the projects contain at least 80% of files that can be found elsewhere. These rates of duplication have implications for systems built on open source software as well as for researchers interested

# My Definition for Code Duplicates

- **File-Level (smaller chunks are ok)**
- Highly-Similar
- Not all clones are duplicates

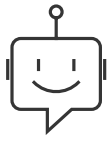
```
[  
  "0xsky/xblog/xblogroot/admin/tinymce/plugins/codemirror/CodeMirror/lib/codemirror.js",  
  "benatkin/codemirror/codemirror.js",  
  "cdnjs/cdnjs/ajax/libs/codemirror/3.16.0/codemirror.js",  
  "cdnjs/cdnjs/ajax/libs/codemirror/3.21.0/codemirror.js",  
  "cdnjs/cdnjs/ajax/libs/codemirror/3.22.0/codemirror.js",  
  "cdnjs/cdnjs/ajax/libs/codemirror/3.23.0/codemirror.js",  
  "disnet/contracts.js/js/codemirror.js",  
  "ericbarnes/wardrobe/app/assets/vendor/plugins/editor/editor.js",  
  "Paxa/postbird/lib/codemirror/codemirror.js",  
  "renz45/cs_console/demo_app/cs_console.js",  
  "tantaman/Strut/app/components/codemirror/codemirror.js",  
  "TheMightyFingers/MightyEngine/editor/client/js/plugins/sourceEditor/cm/lib/codemirror.js",  
  "yoavram/markx/static/js/codemirror.js"  
]
```

```
1- (function () {
2  ... var jasmineEnv = jasmine.getEnv();
3  ... jasmineEnv.updateInterval = 250;
4
5  ... var htmlReporter = new jasmine.HtmlReporter();
6- ... jasmineEnv.addReporter(htmlReporter);
7- ... jasmineEnv.specFilter = function ( spec ) {
8
9  ... return htmlReporter.specFilter(spec);
10 ... };
11
12 ... var currentWindowOnload = window.onload;
13 ... window.onload = function () {
14 ...     if ( currentWindowOnload ) {
15 ...         currentWindowOnload();
16 ...     }
17 ... }
18
19 ... jasmineEnv.execute();
20 ... };
21 ... }();
```

```
1+ (function() {
2  ... var jasmineEnv = jasmine.getEnv();
3  ... jasmineEnv.updateInterval = 250;
4+
5+ /**
6+  ... Create the `HTMLReporter`, which Jasmine calls to provide results of each spec and
7+  ... */
8  ... var htmlReporter = new jasmine.HtmlReporter();
9  ... jasmineEnv.addReporter(htmlReporter);
10+
11+ /**
12+  ... Delegate filtering of specs to the reporter. Allows for clicking on single suites
13+  ... */
14+ ... jasmineEnv.specFilter = function(spec) {
15+ ...     return htmlReporter.specFilter(spec);
16+ ... };
17+
18+ /**
19+  ... Run all of the tests when the page finishes loading - and make sure to run any pre
20+  ...
21+  ... ### Test Results
22+  ...
23+  ... Scroll down to see the results of all of these specs.
24+  ... */
25 ... var currentWindowOnload = window.onload;
26+ ... window.onload = function() {
27+ ...     if (currentWindowOnload) {
28+ ...         currentWindowOnload();
29+ ...     }
30+
31+ ... //document.querySelector('.version').innerHTML = jasmineEnv.versionString();
32+ ... execJasmine();
33+ ... };
34+
35+ ... function execJasmine() {
36+ ...     jasmineEnv.execute();
37+ ... }
38 ... }();
```

```
1280 *return originOrURL || null;
1281 };
1282 -
1283 -
1284 /**
1285 * Set `allowScriptAccess` based on `trustedDomains` and `window.location.host` vs.
1286 *
1287 * @returns The appropriate script access level.
1288 * @private
1289 */
1290 - var _determineScriptAccess = (function() {
1291 - var _extractAllDomains = function(origins) {
1292 - var i, len, tmp,
1293 - resultsArray = [];
1294 -
1295 - if (typeof origins === "string") {
1296 - origins = [origins];
1297 - }
1298 - if (!(typeof origins === "object" && origins && typeof origins.length === "number")) {
1299 - return resultsArray;
1300 - }
1301 - for (i = 0, len = origins.length; i < len; i++) {
1302 - if (_hasOwn.call(origins, i) && (tmp = _extractDomain(origins[i]))) {
1303 - if (tmp === "*") {
1304 - resultsArray.length = 0;
1305 - resultsArray.push("*");
1306 - break;
1307 - }
1308 - if (resultsArray.indexOf(tmp) === -1) {
1309 - resultsArray.push(tmp);
1310 - }
1311 - }
1312 - return resultsArray;
1313 - };
1314 -
1315 - return function(currentDomain, configOptions) {
1316 - // Get SWF domain
1317 - var swfDomain = _extractDomain(configOptions.swfPath);
1318 - if (swfDomain === null) {
1319 - swfDomain = currentDomain;
1320 - }
1321 - // Get all trusted domains
1322 - var trustedDomains = _extractAllDomains(configOptions.trustedDomains);
1323 -
1324 - var len = trustedDomains.length;
```

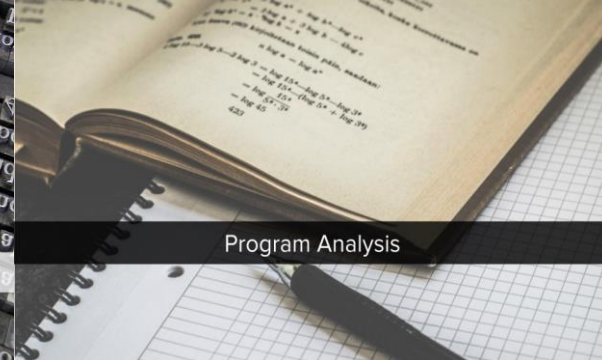
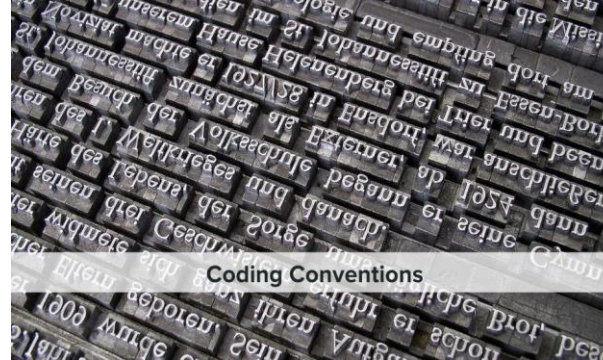
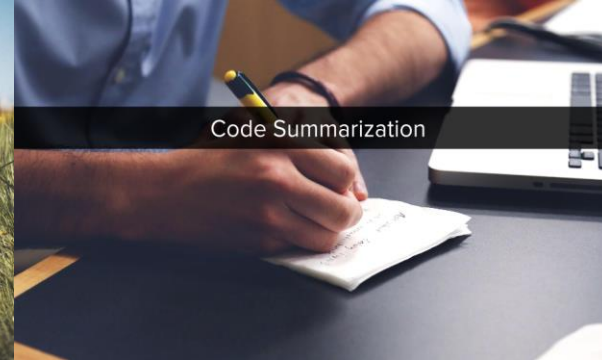
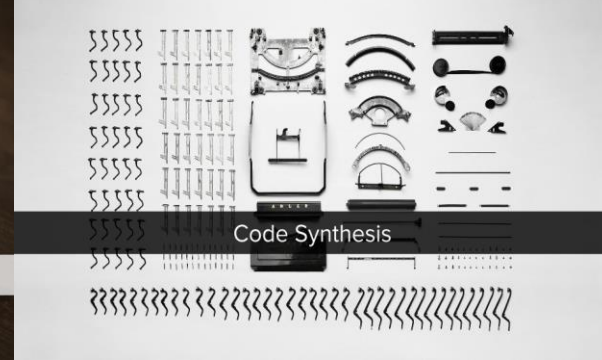
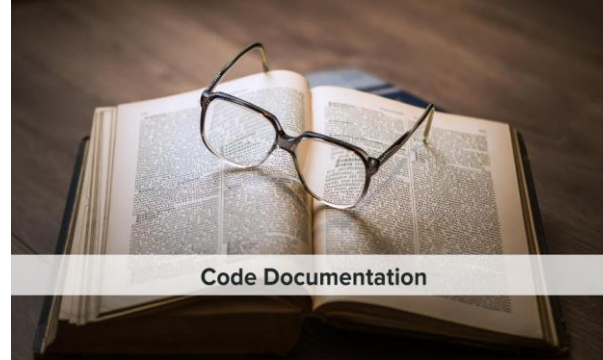
```
1200 *return originOrURL || null;
1201 };
1202 -
1203 -
1204 /**
1205 * Set `allowScriptAccess` based on `trustedDomains` and `window.location.host` vs.
1206 *
1207 * @returns The appropriate script access level.
1208 * @private
1209 */
1210 + var _determineScriptAccess = function() {
1211 + var _extractAllDomains = function(origins, resultsArray) {
1212 + var i, len, tmp;
1213 + if (origins == null || resultsArray[0] === "*") {
1214 + return;
1215 + }
1216 + if (typeof origins === "string") {
1217 + origins = [origins];
1218 + }
1219 + if (!(typeof origins === "object" && typeof origins.length === "number")) {
1220 + return;
1221 + }
1222 + for (i = 0, len = origins.length; i < len; i++) {
1223 + if (_hasOwn.call(origins, i) && (tmp = _extractDomain(origins[i]))) {
1224 + if (tmp === "*") {
1225 + resultsArray.length = 0;
1226 + resultsArray.push("*");
1227 + break;
1228 + }
1229 + if (_inArray(tmp, resultsArray) === -1) {
1230 + resultsArray.push(tmp);
1231 + }
1232 + }
1233 + return function(currentDomain, configOptions) {
1234 + // Get SWF domain
1235 + var swfDomain = _extractDomain(configOptions.swfPath);
1236 + if (swfDomain === null) {
1237 + swfDomain = currentDomain;
1238 + }
1239 + // Get all trusted domains
1240 + var trustedDomains = [];
1241 + _extractAllDomains(configOptions.trustedOrigins, trustedDomains);
1242 + _extractAllDomains(configOptions.trustedDomains, trustedDomains);
1243 +
1244 + var len = trustedDomains.length;
```



# ML+Code

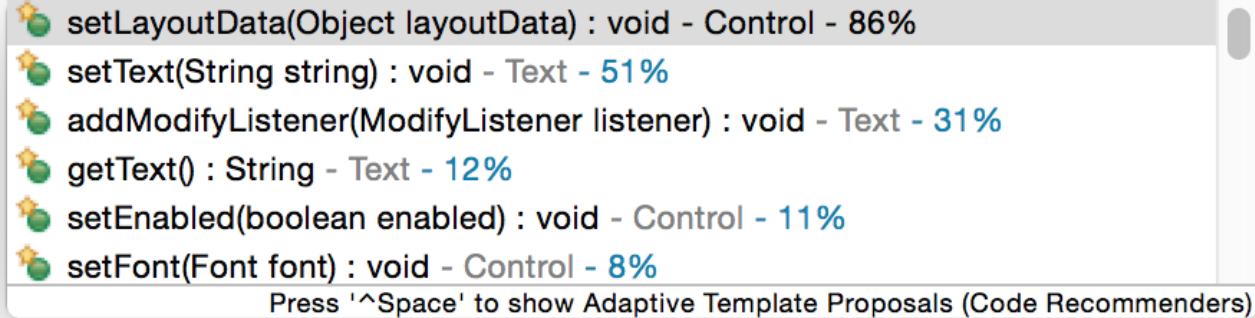
Most often: use trained models to provide recommendations and insights on new and unseen code when the software engineer is creating or maintaining it.

*"Would the tool observe code that contains duplicates?"*



# Code Autocompletion

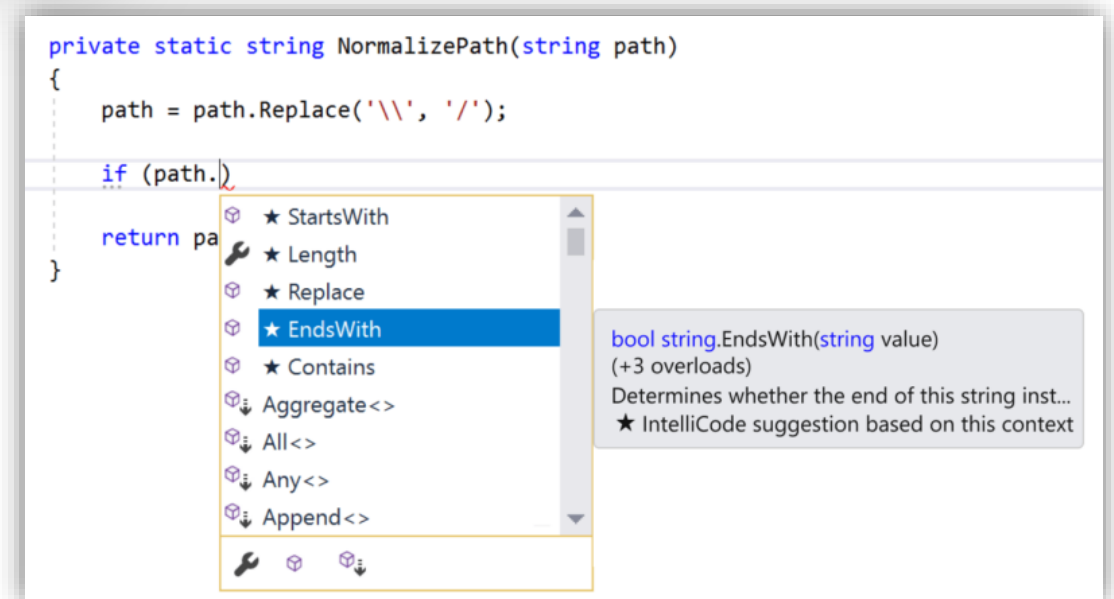
```
Text text = new Text(parent, SWT.NONE);  
text.|
```



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/>



```
private static string NormalizePath(string path)  
{  
    path = path.Replace('\\', '/');  
    if (path.)  
    return pa  
}
```

★ StartsWith  
★ Length  
★ Replace  
★ EndsWith  
★ Contains  
Aggregate<>  
All<>  
Any<>  
Append<>

bool string.EndsWith(string value)  
(+3 overloads)  
Determines whether the end of this string inst...  
★ IntelliCode suggestion based on this context

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

# 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

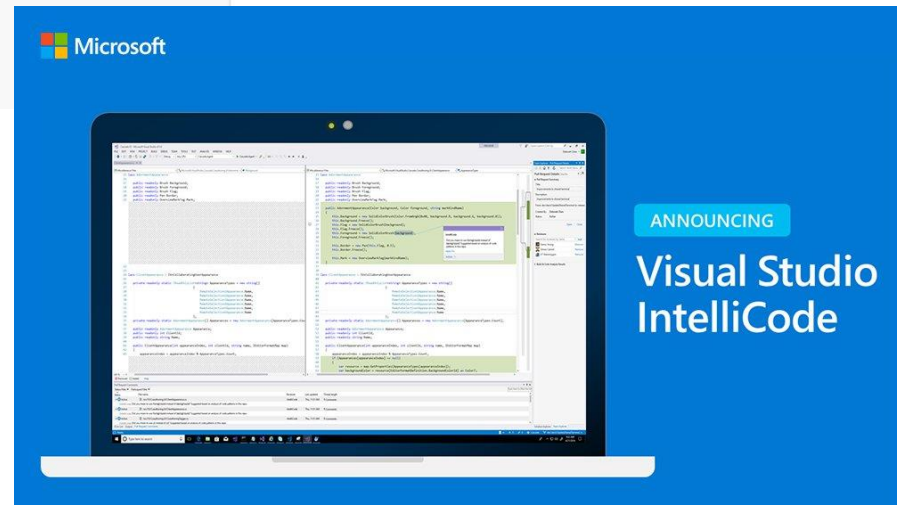
♥ 1 Resolved ▾

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



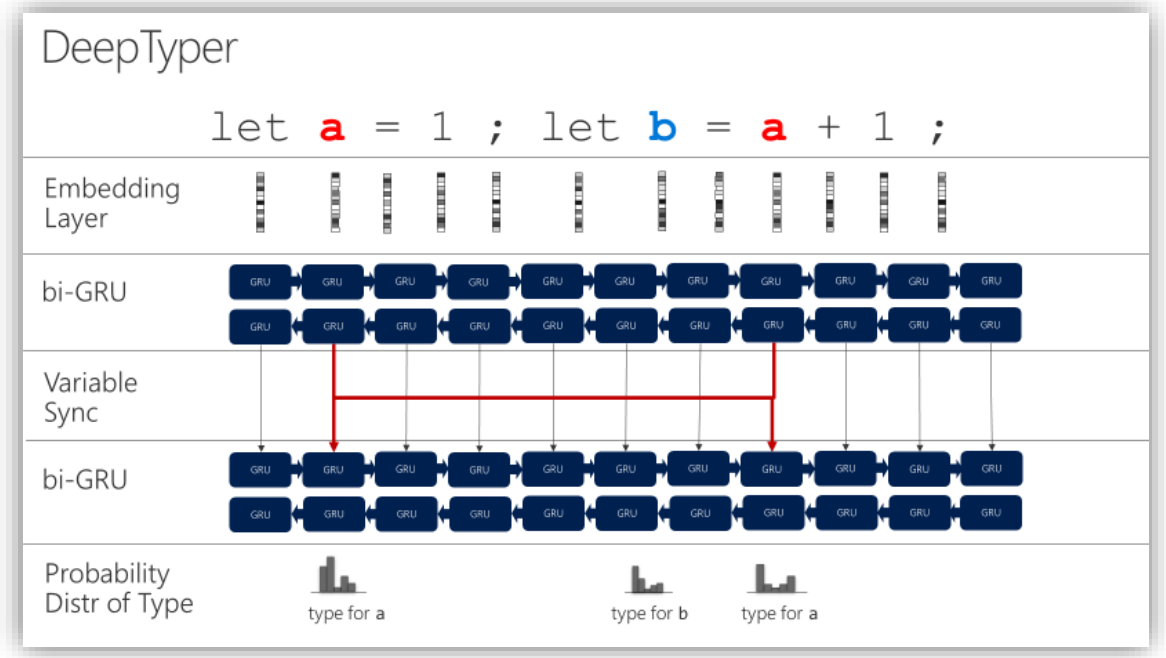
# Predicting Types

The screenshot shows the JSNice interface. On the left, under 'ENTER JAVASCRIPT', there is a code block with a function `chunkData` and a `const get` function. On the right, under 'RESULT', the transformed code is shown with JSDoc-style type annotations. A 'NICIFY JAVASCRIPT' button is visible between the two code blocks. Below the code, there are 'YES' and 'NO' buttons for user feedback.

## Predicting Program Properties from Code

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

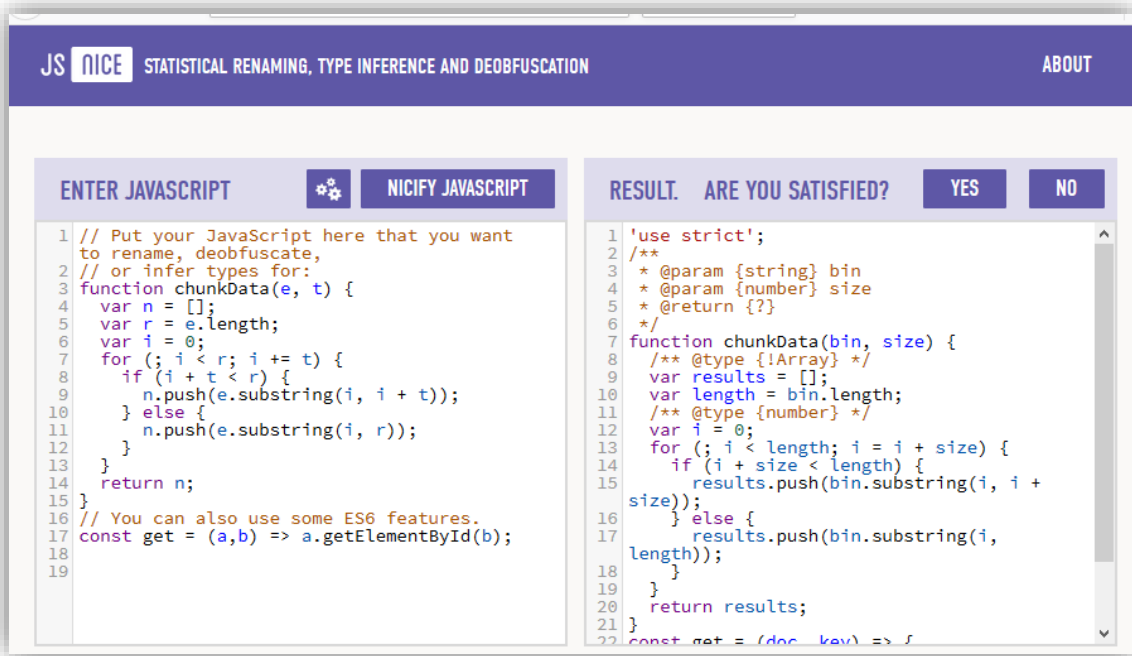
<http://jsnice.org/>



## Deep Learning Type Inference

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

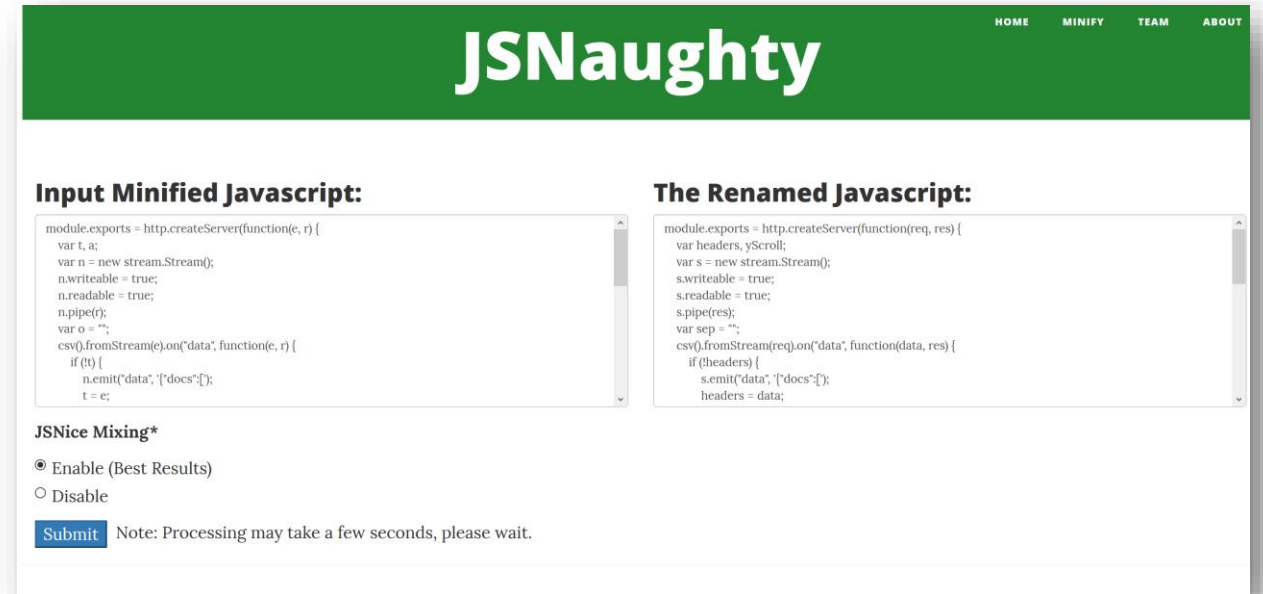
# Predicting Names (Deobfuscation)



Predicting Program Properties from Code

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

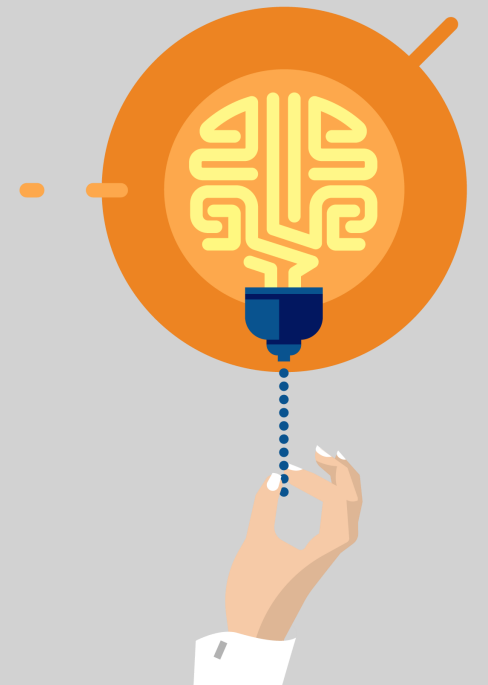
<http://jsnice.org/>



Recovering Clear, Natural Identifiers from  
Obfuscated JS Names B. Vasilescu, C. Casalnuovo, P.  
Devanbu. 2017

[http://tardigrade.andrew.cmu.edu:8000/get\\_js/](http://tardigrade.andrew.cmu.edu:8000/get_js/)

... in most cases we want to make suggestions for *new, previously unseen* code.



# But I've deduplicated the corpus!

... that's what I thought too!

```
1 public void execute(Runnable task) {                public void execute(Runnable task) {
2     if (task == null)                                if (task == null)
3         throw new NullPointerException();            throw new NullPointerException();
4     ForkJoinTask<?> job;                               ForkJoinTask<?> job;
5     if (task instanceof ForkJoinTask<?>) // avoid re-wrap  if (task instanceof ForkJoinTask<?>) // avoid re-wrap
6         job = (ForkJoinTask<?>) task;                job = (ForkJoinTask<?>) task;
7     else                                              else
8         job = new ForkJoinTask.AdaptedRunnableAction(task);  job = new ForkJoinTask.AdaptedRunnableAction(task);
9     doSubmit(job);                                    doSubmit(job);
10 }                                                    }
```

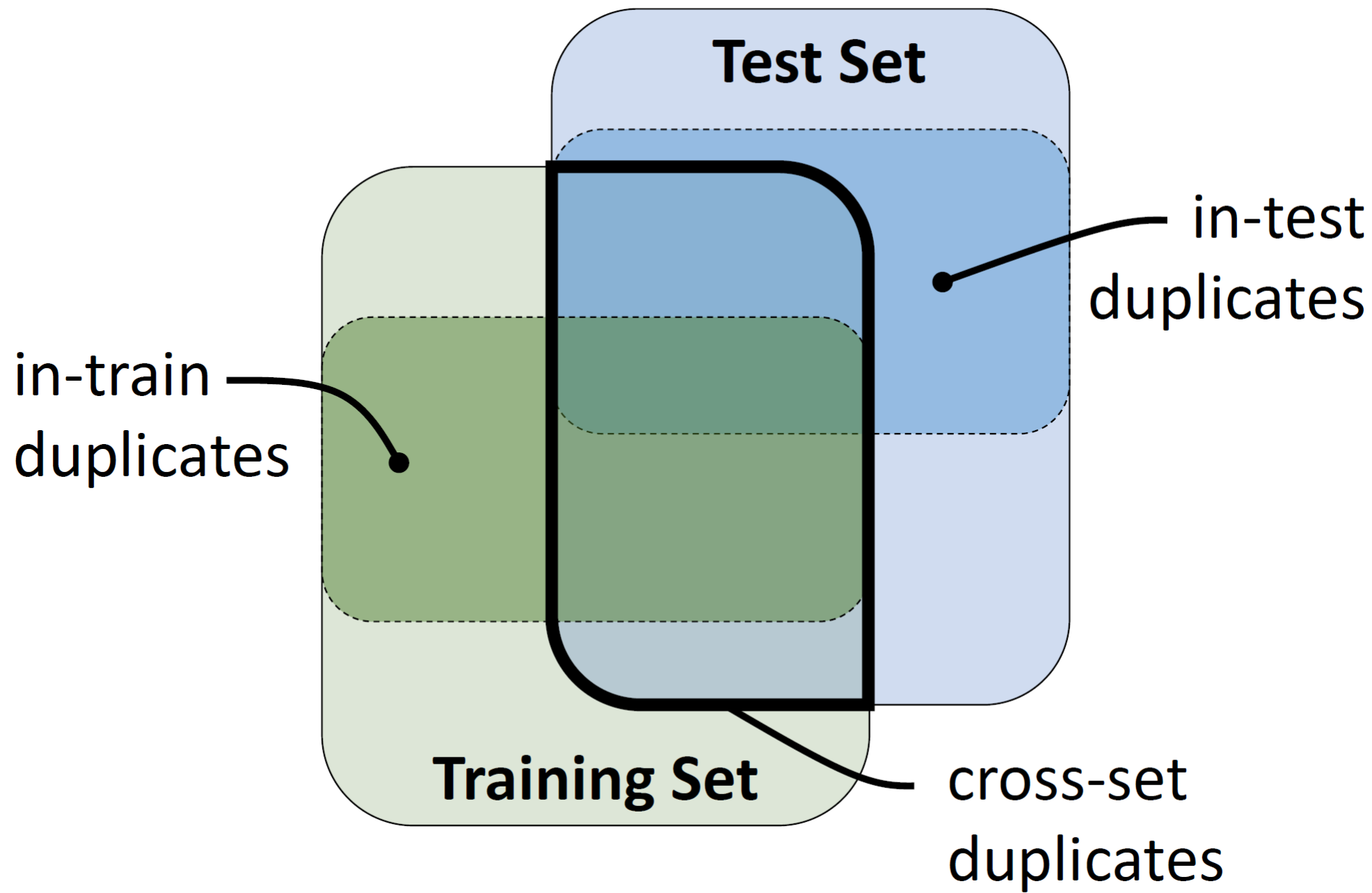
Allamanis & Sutton 2013

- 24.8% duplicates
- Each duplicate file appears ~x2




# Commonly Used Datasets

Dataset	# Files (x1000)	% duplicates
C# ICLR'19	28.3	10.6
Concode- Java*	229.3	68.7
Java GitHub Corpus	1853.7	24.8
Java-Small	79.8	4.7
Java-Large	1863.4	20.2
JavaScript-150k	112.0	20.7
Python-150k	126.0	6.6
Python docstrings v1*	105.2	9.2
Python docstrings v2*	194.6	31.5

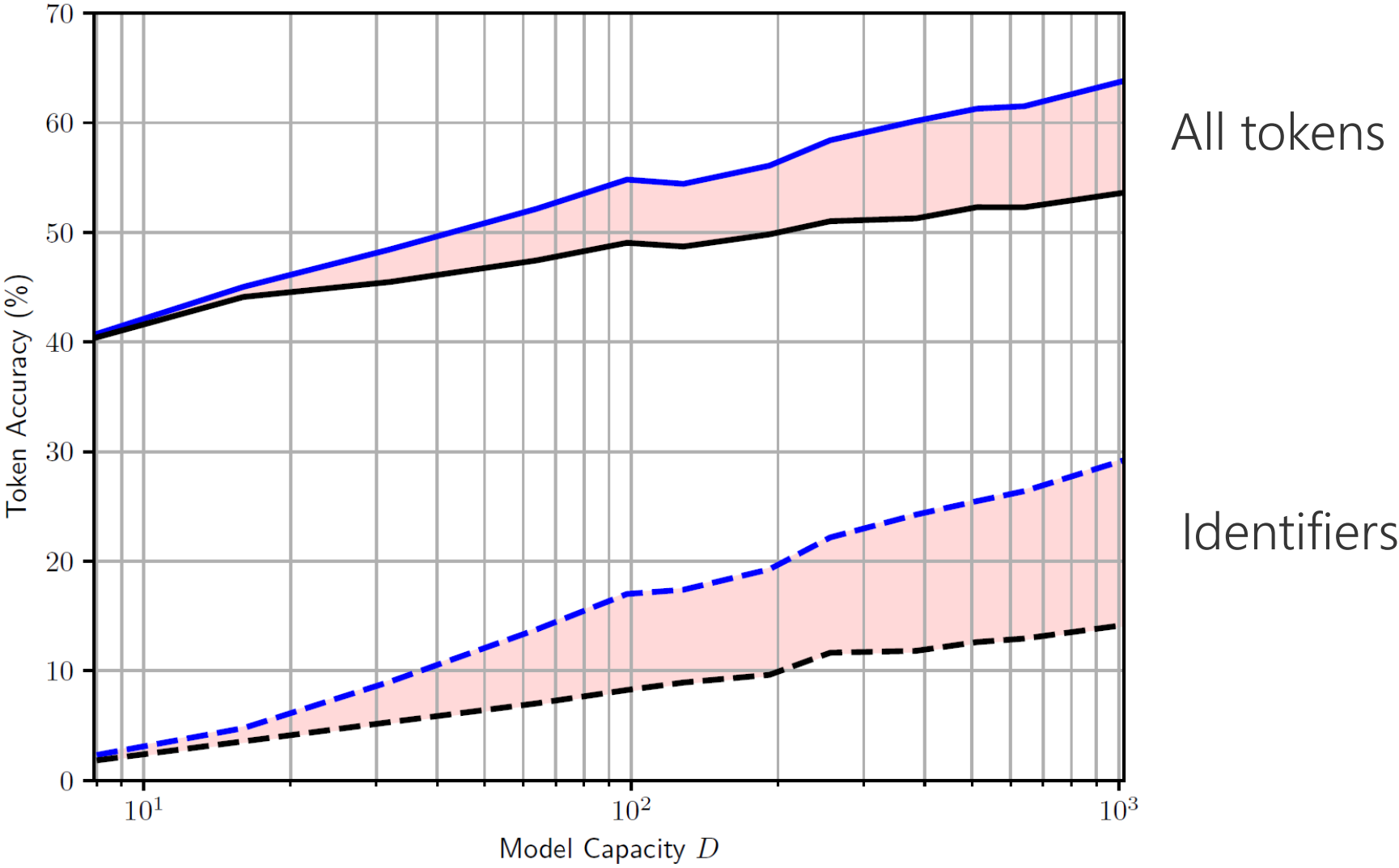
\* Dataset is per-function not per-file.






# Case Study: A Simple Language Model

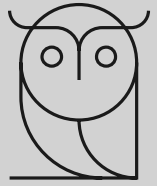
Metric	Performance			
			$\Delta(\text{blue square icon}, \text{blue square icon with red square})$	
Acc (%)	49.1	55.1	-10.9%	49.2
Acc-ID (%)	8.6	17.7	-51.4%	8.3
MRR	0.674	0.710	-5.1%	0.674
MRR-ID	0.136	0.224	-39.3%	0.132
PPL	9.4	7.5	+25.3%	9.4
PPL-ID	76.1	55.4	+37.4%	82.3

# Case Study: A Simple Language Model

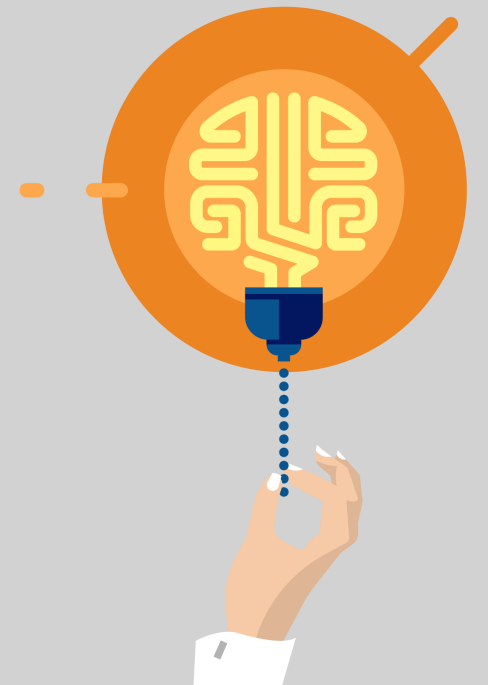


# Other Models

Metric	Performance			
			$\Delta(\text{blue}, \text{red})$	
<u>Task</u> : Method Naming <u>Model</u> : code2vec [6]				
<u>Dataset</u> : Reshuffled Java-Large [5]				
F1 (%)	44.71	50.98	-12.3%	46.04
Precision (%)	53.00	58.92	-10.5%	54.51
Recall (%)	38.67	44.93	-13.9%	39.85
<u>Task</u> : Variable Naming <u>Model</u> : JsNICE [22]				
<u>Dataset</u> : Reshuffled & Reduced JavaScript-150k [21]				
Accuracy (%)	34.44	55.04	-37.4%	29.41
<u>Task</u> : Code Autocompletion <u>Model</u> : PHOG [9]				
<u>Dataset</u> : Reshuffled & Reduced JavaScript-150k [21]				
Accuracy (%) – Types	71.80	75.69	-5.1%	72.95
Accuracy (%) – Values	71.19	77.75	-8.4%	71.35
– Identifiers	48.94	61.43	-20.3%	49.05
– String Literal	25.62	43.89	-41.6%	24.51
<u>Task</u> : Docstring Prediction <u>Model</u> : Seq2Seq [7]				
<u>Dataset</u> : Python Docstrings v1 [7]				
BLEU	12.32	13.86	-11.1%	—



# Some Best Practices to Avoid Issues with Duplication



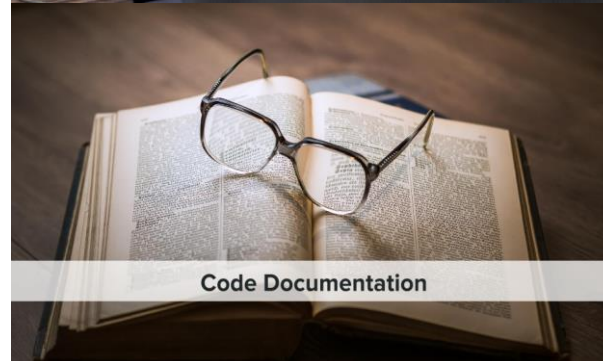
# Does the target application contain duplicates?



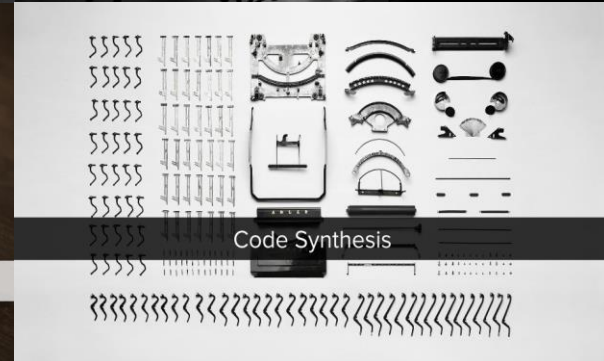
Recommender Systems



Statistical Code Migration



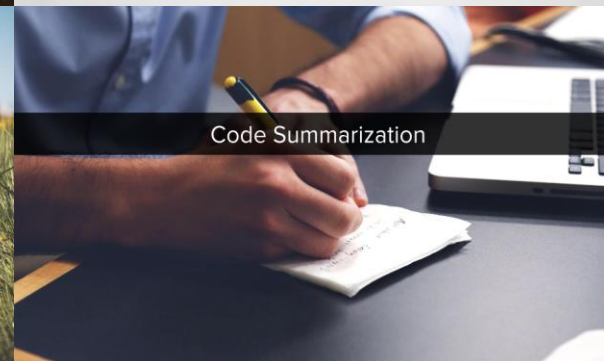
Code Documentation



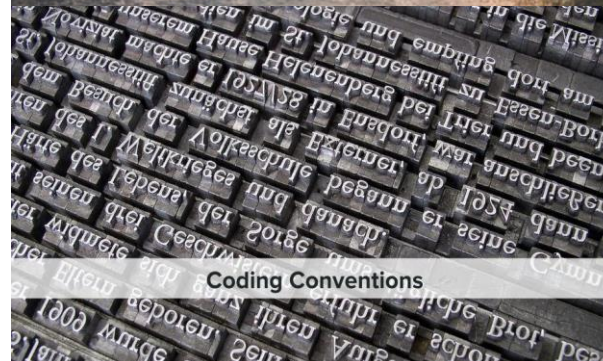
Code Synthesis



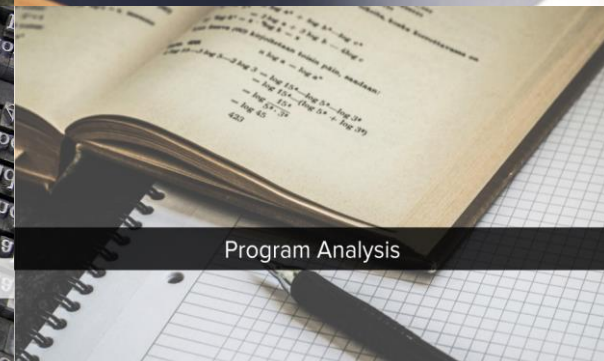
Code Defects



Code Summarization



Coding Conventions



Program Analysis

# Careful Data Collection / Use of Existing Data

- Use a method to deduplicate collected data:
  - SourcererCC
  - [github.com/Microsoft/near-duplicate-code-detector](https://github.com/Microsoft/near-duplicate-code-detector)
  - [github.com/Microsoft/dpu-utils](https://github.com/Microsoft/dpu-utils)
- Use duplication index

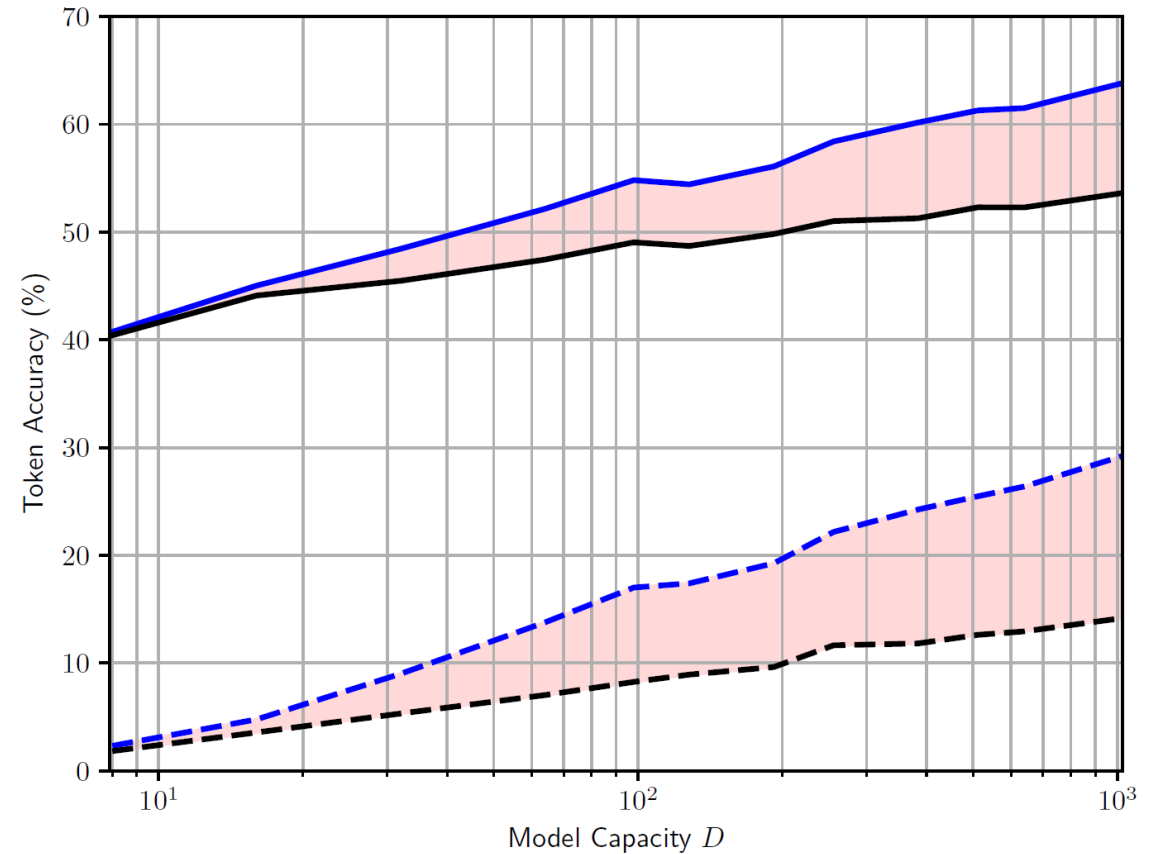
```
$ pip install dpu-utils
```

```
$ run-tokenizer-for-lang
```

```
$ deduplicationcli DATA_PATH OUT_JSON
```

# Model Capacity

- Use strong baselines.
- Try baselines that memorize well.





## Some Best Practices to Avoid Issues with Duplication

- Does the target application contain duplicates?
- Careful data collection or use of existing data.
- Be cautious of model capacity.

