



CODE COMBAT

Google



Let's Code
Thailand



THAI
PROGRAMMER



ศ. E-SAN Thailand Coding & AI Academy

โครงการวิจัยโมเดลระบบนิเวศการเรียนรู้ที่บูรณาการ CODING & AI สำหรับเยาวชน

Model of Learning Ecosystem Platform integrate with Coding & AI for Youth

โครงการย่อที่ 6

การพัฒนาเยาวชนเพื่อเข้าสู่วิชาชีพขั้นสูงด้าน Coding & AI

ร่วมกับ Coding Entrepreneur & Partnership: Personal AI

ชื่อหัวข้อ Code Clone Detector - ระบบ AI สำหรับช่วยตรวจวัดความเหมือนของโค้ดและ
ตรวจจับการคัดลอกโค้ด

ดร. ชัยยงค์ รักขิตเวชสกุล



๔ E-SAN Thailand
Coding & AI Academy

Outline



การพัฒนาเยาวชนเพื่อเข้าสู่วิชาชีพขั้นสูงด้าน Coding & AI ร่วมกับ Coding Entrepreneur & Partnership:

Personal AI

โครงการวิจัยโมเดลระบบนิเวศการเรียนรู้ที่บูรณาการ CODING & AI สำหรับเยาวชน
Model of Learning Ecosystem Platform integrate with Coding & AI for Youth

1. โค้ดที่เหมือนกันคืออะไร? (What are code clones?)
2. วิธีการสร้าง Machine Learning Model เพื่อตรวจจับโค้ดที่เหมือนกัน
3. วิธีการนำ Machine Learning Model ที่สร้างแล้ว มาตรวจจับโค้ดที่เหมือนกัน
4. การวัดประสิทธิภาพของ Machine Learning Model ที่สร้างขึ้น ในเชิงความแม่นยำ
5. การวัดประสิทธิภาพของ Machine Learning Model ที่สร้างขึ้น ในเชิงเครื่องมือ
6. บทสรุปและแนวทางการพัฒนาต่อในอนาคต

What are Code Clones ?

ສັກຊີໂນຂອງໂລດທີ່ສົດວານເຫຼືອງວິນ

“ Two code fragments form a clone pair
if they are similar enough
according to a given definition of similarity. [1]”

ແບ່ງອອກເປົ້າ 4 ຢູ່ມະນາຄ

```
public int sum(int a, int b){      public int sum(int num1, int num2){  
    int sum;                      int result;  
    sum = a + b;                  result = num1 + num2;  
    return sum;                   return result;  
}  
}
```

Clone Types: Syntactic Based [2]

Type 1

គិតមាន
មានការពិនិត្យ

```
public int sum(int a, int b){  
    int sum;  
    sum = a + b;  
    return sum;  
}
```

```
public int sum(int a, int b){  
    int sum;  
    sum = a + b;  
    return sum;  
}
```

Identical code fragments
except for layout, white space, and
comments

Type 2

```
public int sum(int a, int b){  
    int sum;  
    sum = a + b;  
    return sum;  
}
```

```
public int sum(int num1, int num2){  
    int result;  
    result = num1 + num2;  
    return result;  
}
```

Identical code fragments
except for literals, identifiers, data types,
layout, white space, and comments.

Type 3

```
public int sum(int a, int b){  
    int sum;  
    sum = a + b;  
    return sum;  
}
```

```
public int sum(int a, int b){  
    return a + b;  
}
```

Similar clone fragments
with added, changed, and removed
some statements.

Clone Types: Functionality Based [2]

TYPE 4

```
private static String getFormatByName(String name){  
    if(name != null){  
        final int j = name.lastIndexOf(".") + 1,  
            k = name.lastIndexOf("/") + 1;  
        if(j > k &&  
            j < name.length()) {  
            return name.substring(j);  
        }  
    }  
    return null  
}
```

```
public static String getExtension(final String filename){  
    if(filename == null ||  
        filename.trim().length == 0 ||  
        !filename.contains(".")){  
        return null;  
    }  
    int pos = filename.lastIndexOf(".");  
    return filename.substring(pos+1);  
}
```

Code fragments that have the same computation
but different in syntax or algorithm.

Why Code Clones need to be detect ?

- Source code plagiarism detection [3]
- Code Clones can be harmful in software maintenance [1]
- To improve code quality by reducing code redundancy (usually occur 7-23% in software) [2]
- Some clones can be beneficial (e.g. software product line, well-written code) [4]
- **We need to detect clones so that we can manage them [5]**

[1] T. Kamiya, S. Kusumoto and K. Inoue, "CCFinder: a multilingual token-based code clone detection system for large scale source code," July 2002.

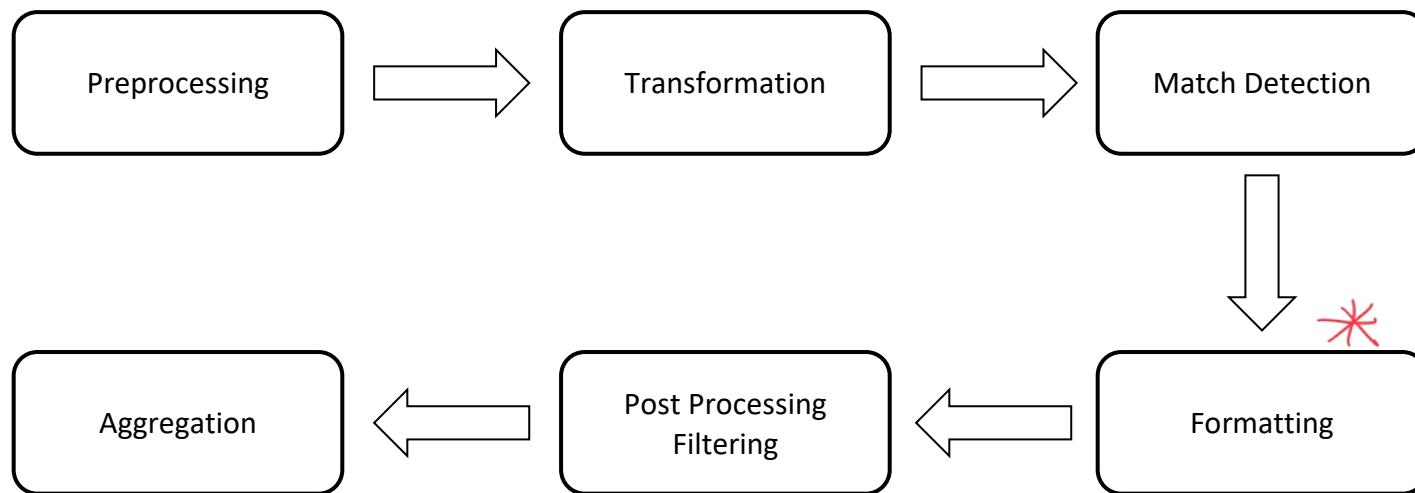
[2] Roy et al. Comparison and evaluation of code clone detection techniques and tools: A qualitative approach, *Science of Computer Programming* (2009)

[3] Prechelt, L., Malpohl, G., & Philipsen, M. (2002). Finding Plagiarisms among a Set of Programs with JPlag. *Journal Of Universal Computer Science*, 8(11), 1016–1038.

[4] Kapser, C. J., & Godfrey, M. W. (2008). Cloning considered harmful: patterns of cloning in software. *Empirical Software Engineering*, 13(6), 645–692.

[5] D. Chatterji, J. C. Carver, and N. A. Kraft. Code clones and developer behavior: results of two surveys of the clone research community. *Empirical Software Engineering*, 21(4):1476–1508, Aug 2016.

Code Clone Detection Process



Problem Statements

- The existing techniques and tools are still facing challenges when detecting clones with several modifications (e.g., added/deleted/modified statements).
- Existing clone detection and plagiarism detection tools are difficult to use because it is command line based tool.

Objectives

1. To create a code clone detection tool using machine learning techniques and study its effectiveness.
2. To enhance the user experience of code clone detection tools
 - providing code clone detection as a web application to users
 - providing visualization of clone results

Merry: Web-Based Code Clone Detection System Using Machine Learning

Features

11 Syntactic
Code Metrics
+
12 Semantic
Code Metrics
(code2vec)

Machine Learning

Decision tree, Random forest, SVM,
SVM + SMO

BigCloneBench

Largest
and
Credible clone data

Web-based Tool
with
User Interface

GitHub Integration



Visualizations and
reports

วิธีการสร้าง
Machine Learning Model เพื่อ^{เพื่อ}
ตรวจจับโค้ดที่เหมือนกัน

Merry Engine Implementation

1

Building Merry Engine

2

Using Merry Engine for Clone Detection

1

Building Merry Engine

1.1

Data collection and preparation ✓

1.2

Code metrics extraction ✓

1.3

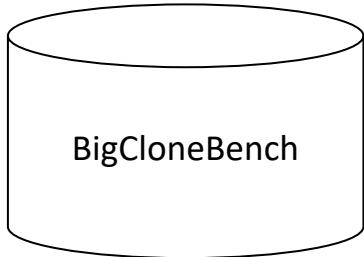
Machine learning models ✓

1.1

Data collection and preparation

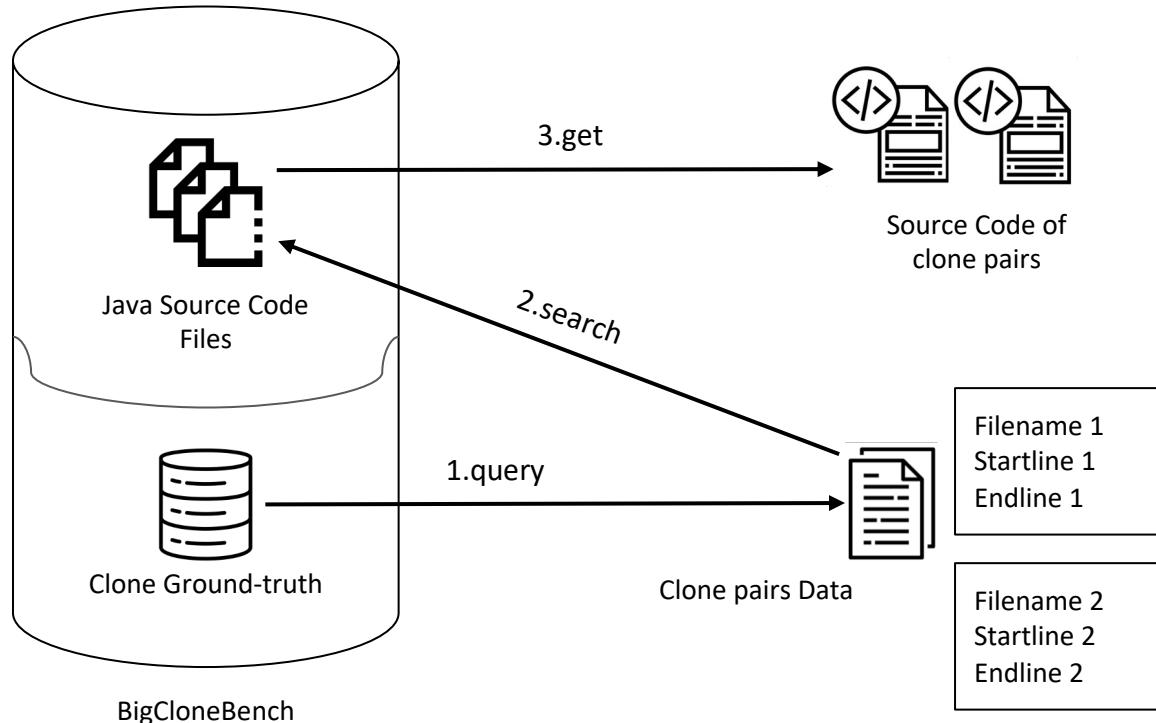
BigCloneBench [7]

- Largest and reliable clone data
- Well-known in code clone detection field
- Big data of inter-project repository IJaDataset 2.0
- Created from 25,000 Java projects
- Contains tagged true and false clone pairs



1.1

How do we get clone pairs from BCB?



1.1

Training Set and Testing Set Splitting

Training Data [8]

True Clone Pairs (22,663 pairs)				False Clone Pairs
Type 1	Type 2	Very Strong Type 3	Strongly Type 3	
13,750	3,104	1,207	4,602	22,663

Testing Data

True Clone Pairs (4,724 pairs - stratified sampling)				False Clone Pairs
Type 1	Type 2	Very Strong Type 3	Strongly Type 3	
2,383	557	307	1,477	18,893

1.2

Code Metrics Extraction

Syntactic metrics

=

Metrics extracted from syntactic
characteristics
of the source code fragments



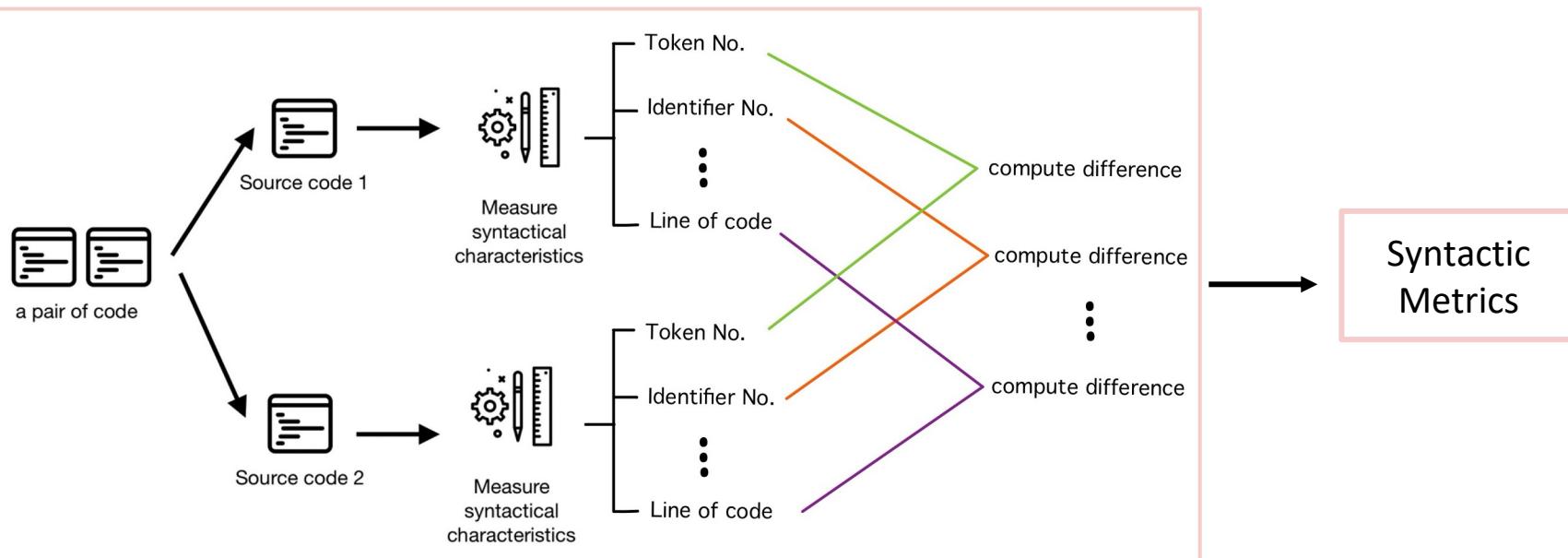
Semantic metrics

=

* Metrics that capture the
behaviour of the source code
fragments

1.2

Syntactic Code Metrics



1.2

Syntactic Code Metrics

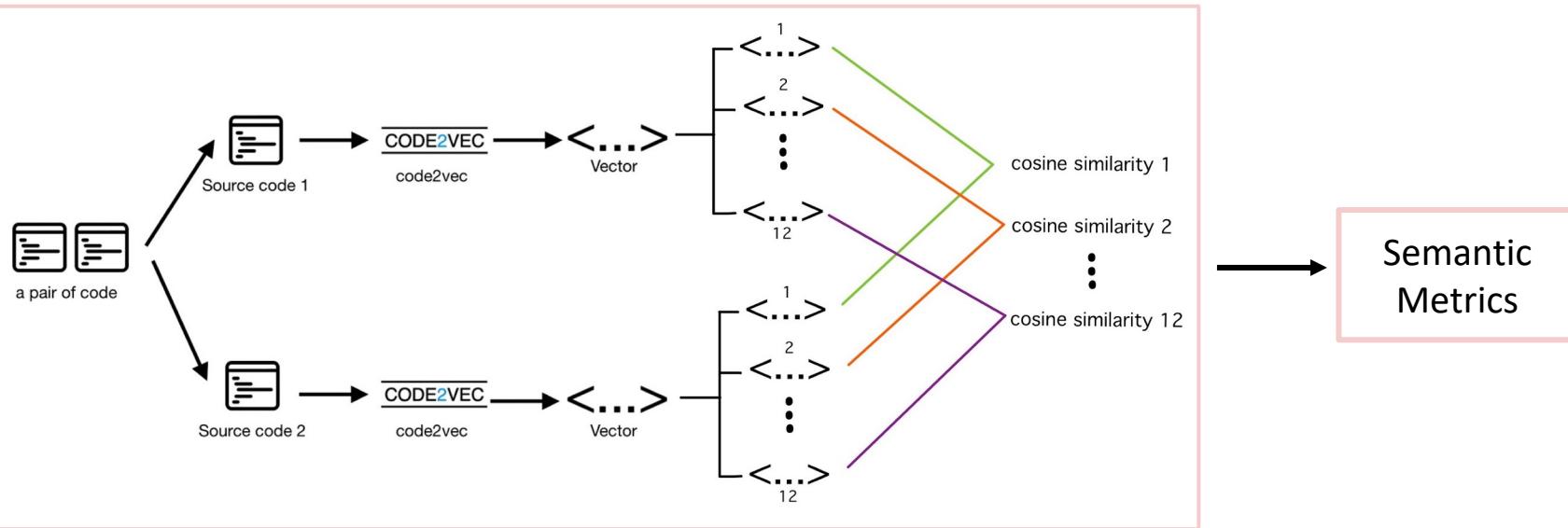
No.	Metric	Description
1	Token No [9]	Difference of number of tokens
2	Unique Token No [10]	Difference of number of unique tokens
3	Identifier No [10]	Difference of number of Identifiers
4	Unique Identifier No [10]	Difference of number of unique Identifiers
5	Operator No [10]	Difference of number of operators
6	Unique Operator No [10]	Difference of number of unique operators
7	Token Types Diversity [9]	Difference of number of unique token types
8	Diff File Name Score	File names difference score
9	Diff Method Name Score	Method names difference score
10	Similar Return Type	Same return type or not
11	DiffLOC	Difference of lines of code

[9] Koschke R., Bazrafshan S., "Software-clonerasinopen-sourceprogramswritten in C or C++", In: 2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER). vol. 3. IEEE; 2016. p. 1–7.

[10] Vara A., Rainer K., Chaiyong R., Morakot C., Thanwadee S., "Improving CloneDetection Precision using Machine Learning Techniques", In: Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering. IWESEP; 2019.

1.2

Semantic Code Metrics



code2vec in Detail

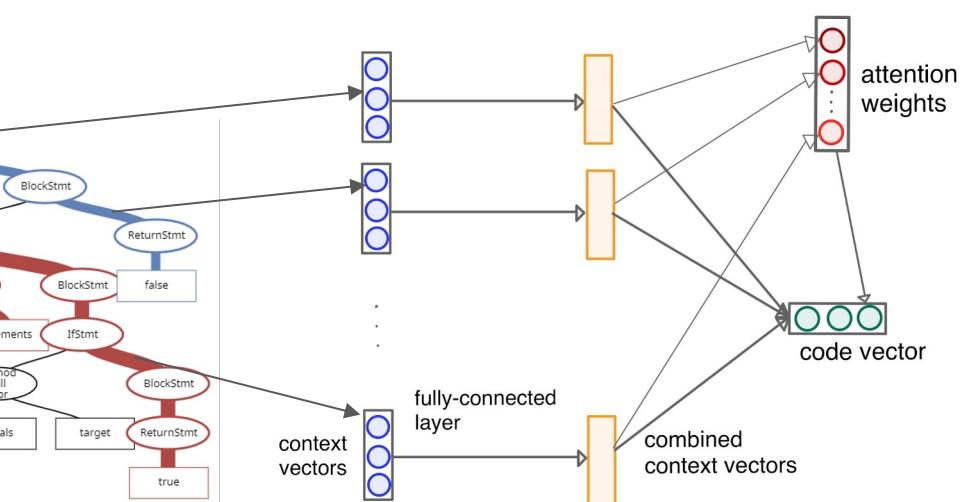
code2vec is neural model that represent code snippet as fixed-length vectors (size = 384) that capture semantic of the source code [12].

```

boolean f(object target) {
    for(Object elem: this.elements) {
        if(elem.equals(target)) {
            return true;
        }
    }
    return false;
}
  
```



We use the pretrain model that trained from 12 M. of real software methods.



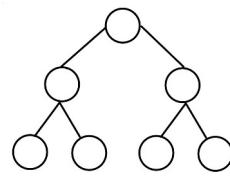
1.2

Example of Code Metrics

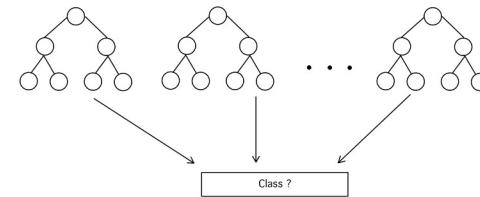
	source code 1	source code 2																																																																									
sum1.java	<pre>public int sum(int a, int b){ int sum; sum = a + b; return sum; }</pre>	<pre>sum2.java</pre>	<pre>public int sum(int a, int b){ return a + b; }</pre>																																																																								
<u>Syntactic metrics</u>																																																																											
<table> <tr> <td>Diff. No. Tokens</td><td>:</td><td style="text-align: center;">7</td><td>vector#1 similarity</td><td>:</td><td style="text-align: center;">0.5206784273085</td></tr> <tr> <td>Diff. No. Unique Token</td><td>:</td><td style="text-align: center;">1</td><td>vector#2 similarity</td><td>:</td><td style="text-align: center;">0.457455059252847</td></tr> <tr> <td>Diff. No. Identifier</td><td>:</td><td style="text-align: center;">3</td><td>vector#3 similarity</td><td>:</td><td style="text-align: center;">0.588965399230431</td></tr> <tr> <td>Diff. No. Unique Identifier</td><td>:</td><td style="text-align: center;">0</td><td>vector#4 similarity</td><td>:</td><td style="text-align: center;">0.652574760995595</td></tr> <tr> <td>Diff. No. Operator</td><td>:</td><td style="text-align: center;">1</td><td>vector#5 similarity</td><td>:</td><td style="text-align: center;">0.529570896554088</td></tr> <tr> <td>Diff. No. Unique Operator No</td><td>:</td><td style="text-align: center;">1</td><td>vector#6 similarity</td><td>:</td><td style="text-align: center;">0.490625929242188</td></tr> <tr> <td>Diff. Token Types Diversity</td><td>:</td><td style="text-align: center;">0</td><td>vector#7 similarity</td><td>:</td><td style="text-align: center;">0.792832125905209</td></tr> <tr> <td>Diff. File Name Score</td><td>:</td><td style="text-align: center;">0.25</td><td>vector#8 similarity</td><td>:</td><td style="text-align: center;">0.562600679903085</td></tr> <tr> <td>Diff. Method Name Score</td><td>:</td><td style="text-align: center;">0.0</td><td>vector#9 similarity</td><td>:</td><td style="text-align: center;">0.799728586131479</td></tr> <tr> <td>Similar Return Type</td><td>:</td><td style="text-align: center;">TRUE</td><td>vector#10 similarity</td><td>:</td><td style="text-align: center;">0.642114908324134</td></tr> <tr> <td>Diff. No. LOC</td><td>:</td><td style="text-align: center;">2</td><td>vector#11 similarity</td><td>:</td><td style="text-align: center;">0.540922173109001</td></tr> <tr> <td></td><td></td><td></td><td>vector#12 similarity</td><td>:</td><td style="text-align: center;">0.729365945085146</td></tr> </table>				Diff. No. Tokens	:	7	vector#1 similarity	:	0.5206784273085	Diff. No. Unique Token	:	1	vector#2 similarity	:	0.457455059252847	Diff. No. Identifier	:	3	vector#3 similarity	:	0.588965399230431	Diff. No. Unique Identifier	:	0	vector#4 similarity	:	0.652574760995595	Diff. No. Operator	:	1	vector#5 similarity	:	0.529570896554088	Diff. No. Unique Operator No	:	1	vector#6 similarity	:	0.490625929242188	Diff. Token Types Diversity	:	0	vector#7 similarity	:	0.792832125905209	Diff. File Name Score	:	0.25	vector#8 similarity	:	0.562600679903085	Diff. Method Name Score	:	0.0	vector#9 similarity	:	0.799728586131479	Similar Return Type	:	TRUE	vector#10 similarity	:	0.642114908324134	Diff. No. LOC	:	2	vector#11 similarity	:	0.540922173109001				vector#12 similarity	:	0.729365945085146
Diff. No. Tokens	:	7	vector#1 similarity	:	0.5206784273085																																																																						
Diff. No. Unique Token	:	1	vector#2 similarity	:	0.457455059252847																																																																						
Diff. No. Identifier	:	3	vector#3 similarity	:	0.588965399230431																																																																						
Diff. No. Unique Identifier	:	0	vector#4 similarity	:	0.652574760995595																																																																						
Diff. No. Operator	:	1	vector#5 similarity	:	0.529570896554088																																																																						
Diff. No. Unique Operator No	:	1	vector#6 similarity	:	0.490625929242188																																																																						
Diff. Token Types Diversity	:	0	vector#7 similarity	:	0.792832125905209																																																																						
Diff. File Name Score	:	0.25	vector#8 similarity	:	0.562600679903085																																																																						
Diff. Method Name Score	:	0.0	vector#9 similarity	:	0.799728586131479																																																																						
Similar Return Type	:	TRUE	vector#10 similarity	:	0.642114908324134																																																																						
Diff. No. LOC	:	2	vector#11 similarity	:	0.540922173109001																																																																						
			vector#12 similarity	:	0.729365945085146																																																																						
<u>Semantic metrics</u>																																																																											

1.3

Machine Learning Models



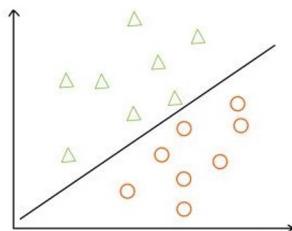
Decision Tree



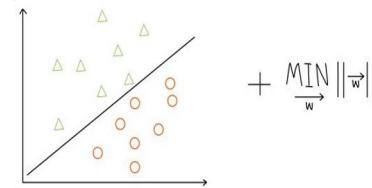
Random Forest



WEKA



Support Vector Machine

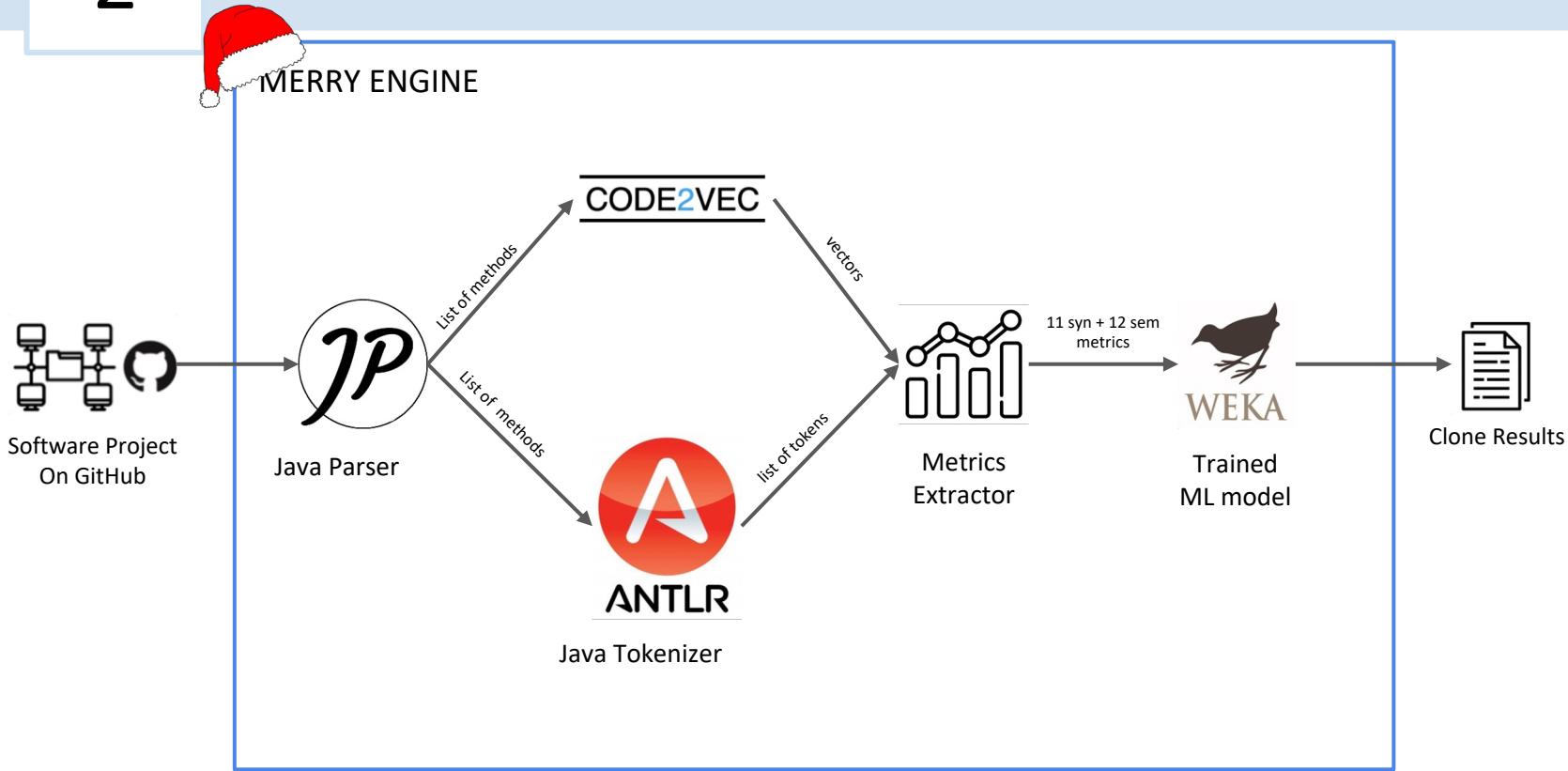


SVM using SMO

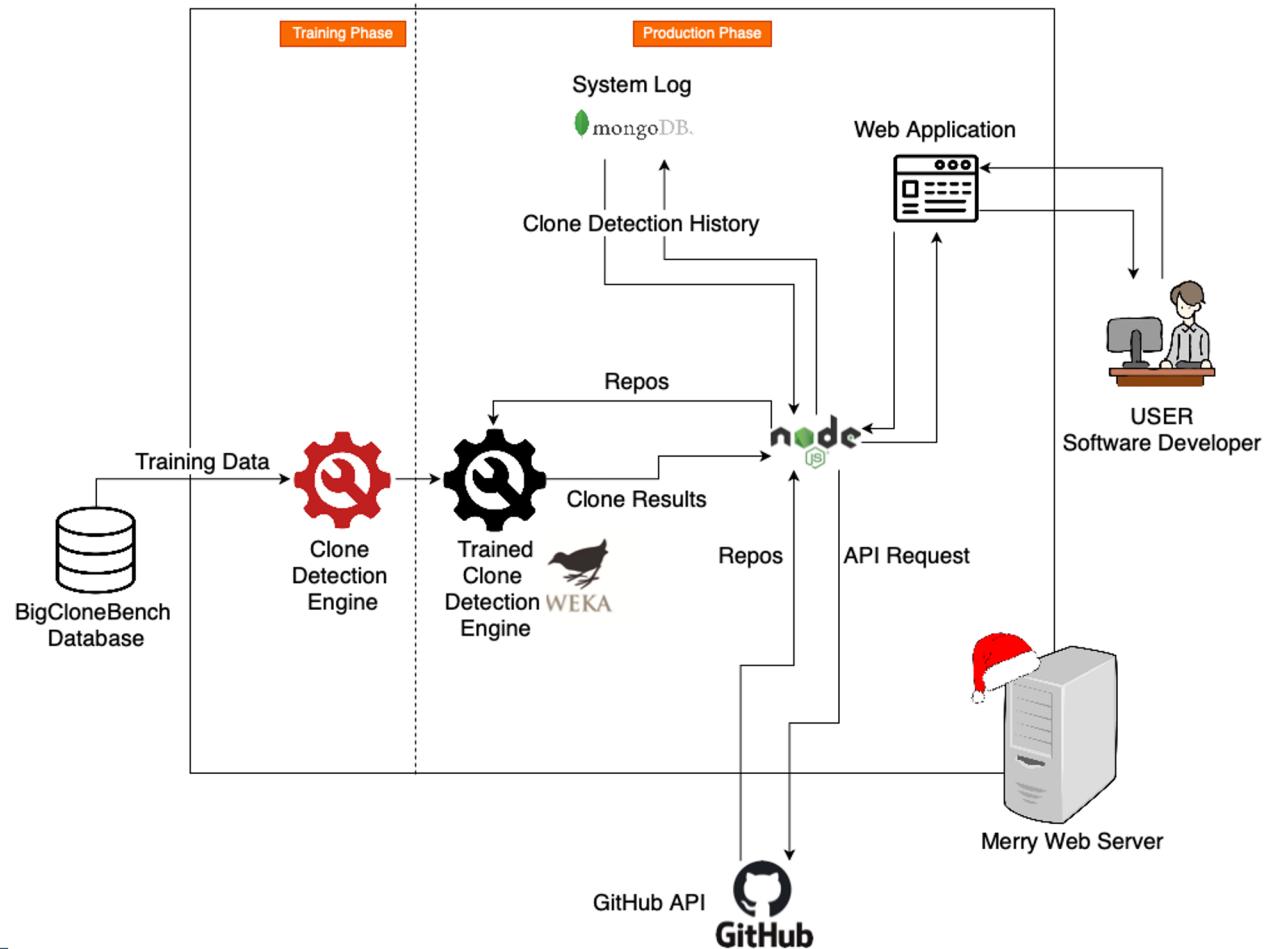
วิธีการนำ Machine Learning Model ที่สร้างแล้ว มาตรวจจับโค้ดที่ เหมือนกัน

2

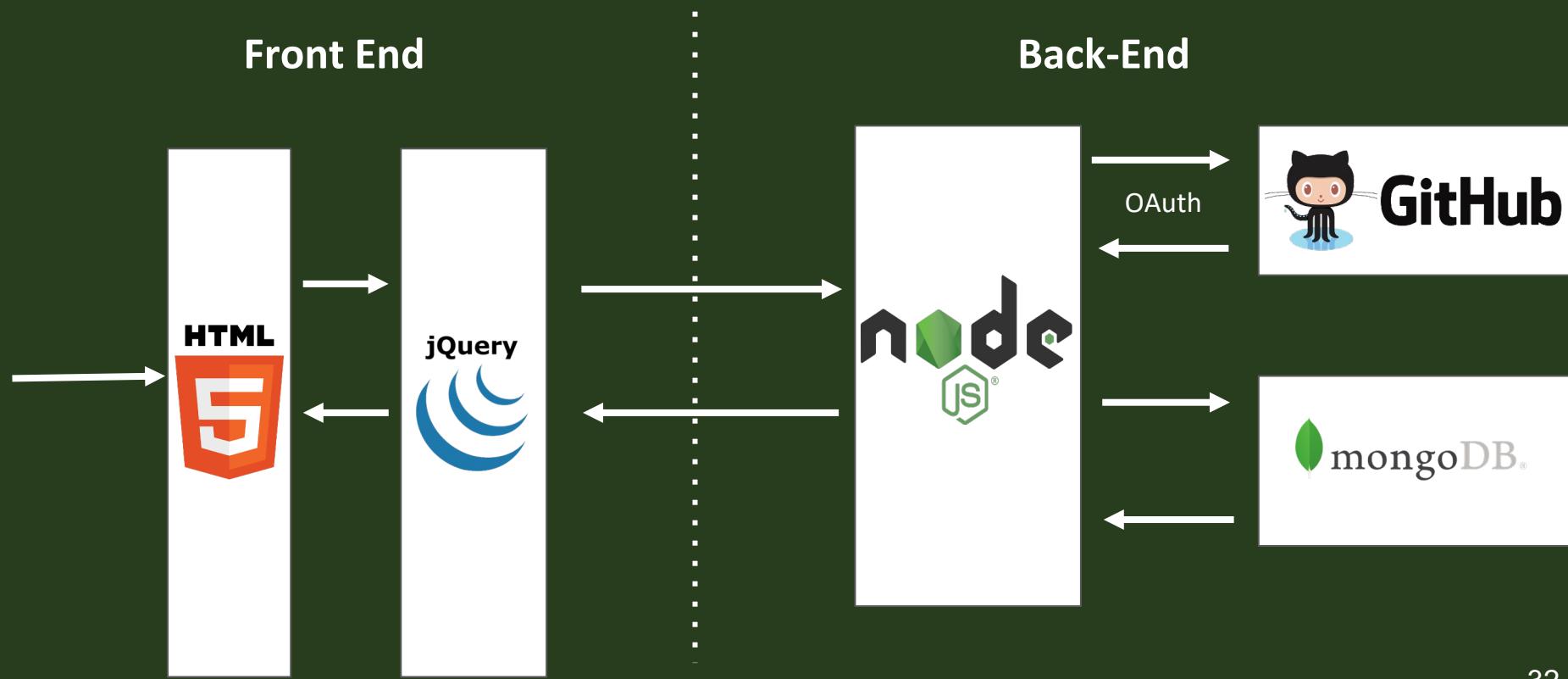
Using Merry Engine for Clone Detection



System Architecture



Frameworks and Tools



GitHub Integration



การวัดประสิทธิภาพของ Machine Learning Model ที่สร้าง ขึ้น ในเชิงความแม่นยำ

Evaluation

RQ1 How accurate is Merry code clone detection on BigCloneBench?

RQ2 How accurate is Merry code clone detection on real software projects?

RQ3 How likely are command line based tool and our Merry tool adopted by programmers?

Evaluation of Merry ML Clone Detection Engine using BCB

RQ1: How accurate is Merry code clone detection on BigCloneBench?

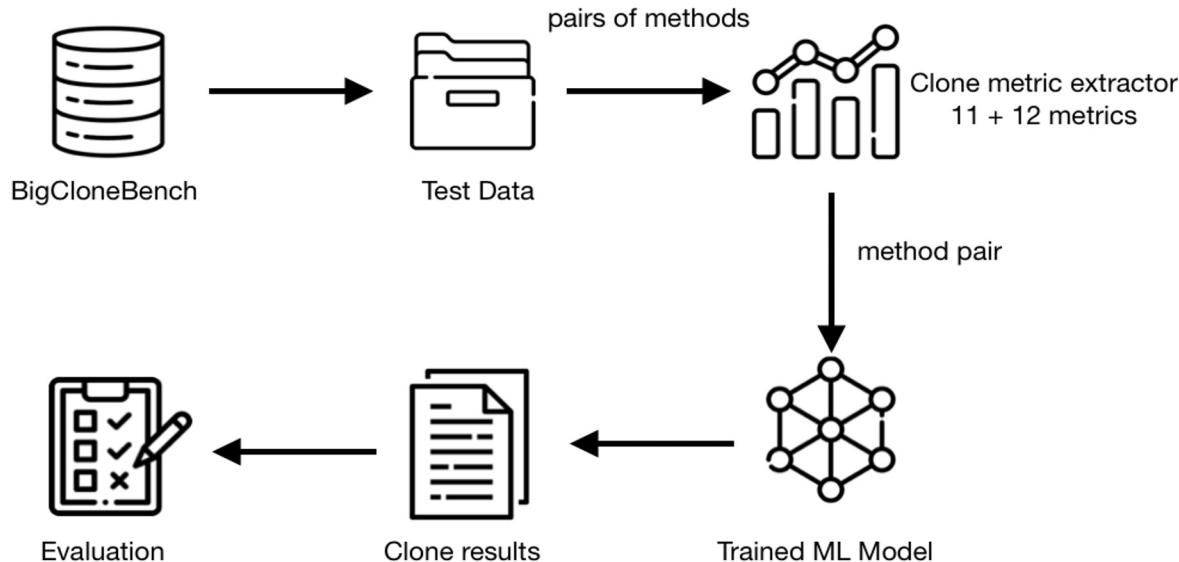
Evaluation of Merry ML Clone Detection Engine using BCB

$$\checkmark \text{precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

$$\checkmark \text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

$$\checkmark \text{F1-score} = 2 \cdot \frac{\text{Precision} - \text{Recall}}{\text{Precision} + \text{Recall}}$$

Evaluation of Merry ML Clone Detection Engine using BCB (cont.)

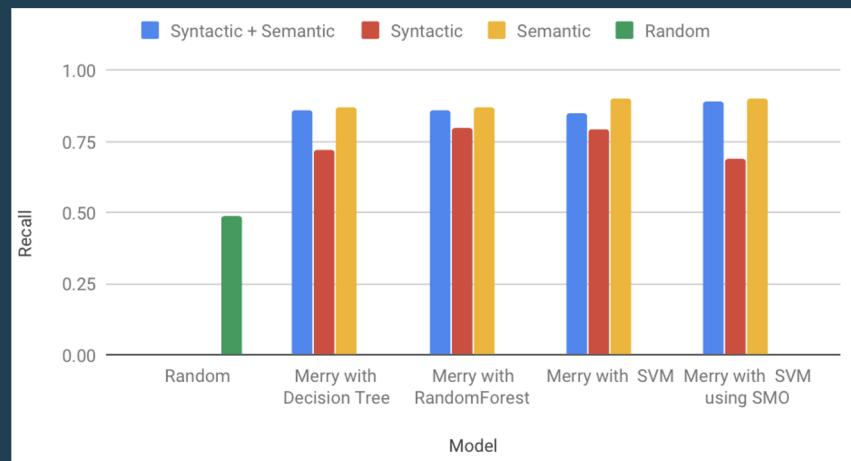


Evaluation of Merry ML Clone Detection Engine using BCB (cont.)

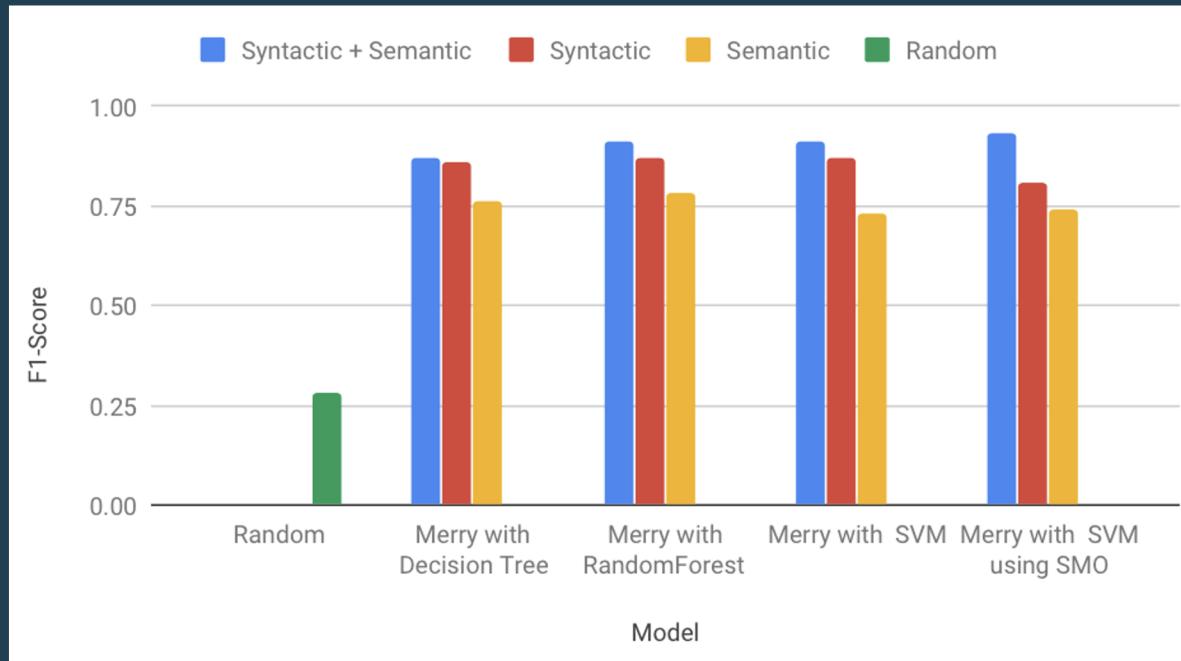
RQ1: How accurate is Merry code clone detection on BigCloneBench?

Model	Metrics	Precision	Recall	F1-Score
Randomization (baseline)		0.20	0.49	0.28
Decision Tree	Syntactic + Semantic	0.89	0.86	0.87
	Syntactic	0.95	0.72	0.86
	Semantic	0.68	0.87	0.76
Random Forest	Syntactic + Semantic	0.97	0.86	0.91
	Syntactic	0.97	0.80	0.87
	Semantic	0.70	0.87	0.78
SVM	Syntactic + Semantic	0.97	0.85	0.91
	Syntactic	0.97	0.79	0.87
	Semantic	0.62	0.90	0.73
SVM using SMO	Syntactic + Semantic	0.98	0.89	0.93
	Syntactic	0.97	0.69	0.81
	Semantic	0.63	0.90	0.74

Evaluation of Merry ML Clone Detection Engine using BCB (cont.)



Evaluation of Merry ML Clone Detection Engine using BCB (cont.)

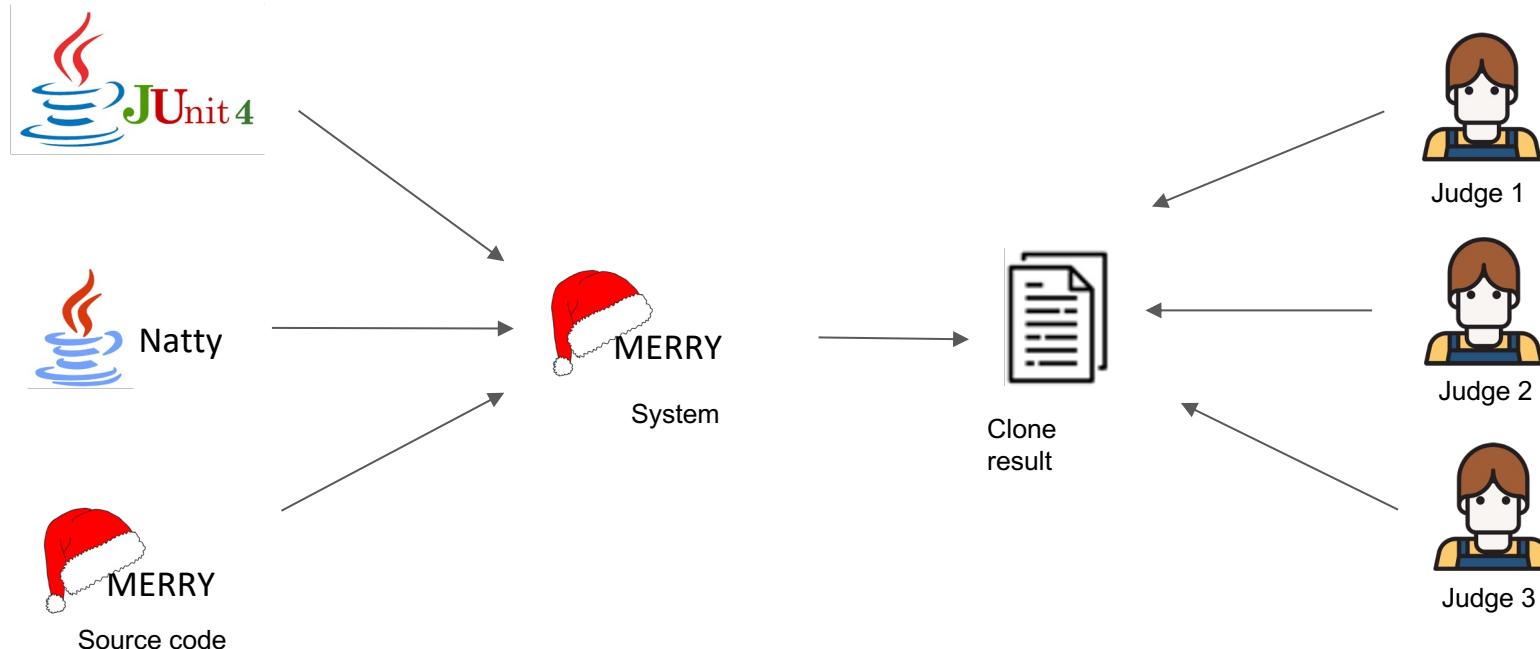


F1-score

Evaluation of Merry Clone Detection Engine on Real Software Projects

RQ2: How accurate is Merry code clone detection on real software projects?

Evaluation of Merry Clone Detection Engine on Real Software Projects



Percent of overall agreement = 69.34%
Fleiss' kappa = 0.39

Evaluation of Merry Clone Detection Engine on Real Software Projects

RQ2: How accurate is Merry code clone detection on real software projects?

Project	LOC	Methods	Clone pairs	TP	FP	Precision	Runtime (sec)
JUnit4	7550	1529	187	77	110	0.41	552
Natty	1761	146	9	7	2	0.77	99
Merry	931	102	8	6	2	0.75	115

The result of evaluation on real software project

Example of Detected Type-1 Clone Pair

Project : JUnit4

```
1. private void runBefores() throws FailedBefore {
2.     try {
3.         try {
4.             List<Method> befores =
testMethod.getBefores();
5.             for (Method before : befores) {
6.                 before.invoke(test);
7.             }
8.         } catch (InvocationTargetException e) {
9.             throw e.getTargetException();
10.        }
11.    } catch (AssumptionViolatedException e) {
12.        throw new FailedBefore();
13.    } catch (Throwable e) {
14.        addFailure(e);
15.        throw new FailedBefore();
16.    }
17. }
```

```
1. private void runBefores() throws FailedBefore {
2.     try {
3.         try {
4.             List<Method> befores =
testMethod.getBefores();
5.             for (Method before : befores) {
6.                 before.invoke(test);
7.             }
8.         } catch (InvocationTargetException e) {
9.             throw e.getTargetException();
10.        }
11.    } catch (AssumptionViolatedException e) {
12.        throw new FailedBefore();
13.    } catch (Throwable e) {
14.        addFailure(e);
15.        throw new FailedBefore();
16.    }
17. }
```

Filename: ClassRoadie.java

Startline: 51

Endline: 67

Filename: MethodRoadie.java

Startline: 128

Endline: 144

Example of Detected Type-2 Clone Pair

Project : JUnit4

```
1. @Override
2. protected Collection<FrameworkMethod>
   getSingleDataPointMethods (ParameterSignature sig) {
3.   Collection<FrameworkMethod> methods =
   super.getSingleDataPointMethods(sig);
4.   String requestedName =
   sig.getAnnotation(FromDataPoints.class).value();
5.   List<FrameworkMethod> methodsWithMatchingNames = new
   ArrayList<FrameworkMethod> ();
6.   for (FrameworkMethod method : methods) {
7.     String[] methodNames =
   method.getAnnotation(DataPoint.class).value();
8.     if
   (Arrays.asList(methodNames).contains(requestedName)) {
9.       methodsWithMatchingNames.add(method);
10.    }
11.  }
12.  return methodsWithMatchingNames;
13. }
```

```
1. @Override
2. protected Collection<Field>
   getDataPointsFields (ParameterSignature sig) {
3.   Collection<Field> fields =
   super.getDataPointsFields(sig);
4.   String requestedName =
   sig.getAnnotation(FromDataPoints.class).value();
5.   List<Field> fieldsWithMatchingNames = new
   ArrayList<Field> ();
6.   for (Field field : fields) {
7.     String[] fieldNames =
   field.getAnnotation(DataPoints.class).value();
8.     if
   (Arrays.asList(fieldNames).contains(requestedName)) {
9.       fieldsWithMatchingNames.add(field);
10.    }
11.  }
12.  return fieldsWithMatchingNames;
13. }
```

Filename: SpecificDataPointsSupplier.java

Startline: 56

Endline: 71

Filename: SpecificDataPointsSupplier.java

Startline: 39

Endline: 54

Example of True Type 3 Clone Pair

Project : JUnit4

```
1. @Override
2. public void evaluate() throws Throwable {
3.     before();
4.     List<Throwable> errors = new
5.         ArrayList<Throwable>();
6.     try {
7.         base.evaluate();
8.     } catch (Throwable t) {
9.         errors.add(t);
10.    } finally {
11.        after();
12.    } catch (Throwable t) {
13.        errors.add(t);
14.    }
15. }
16. MultipleFailureException.assertEmpty(errors);
17. }
```

Filename: ExternalResource.java

Startline: 48

Endline: 65

```
1. @Override
2. public void evaluate() throws Throwable {
3.     List<Throwable> errors = new
4.         ArrayList<Throwable>();
5.     try {
6.         next.evaluate();
7.     } catch (Throwable e) {
8.         errors.add(e);
9.     } finally {
10.        for (FrameworkMethod each : afters) {
11.            try {
12.                invokeMethod(each);
13.            } catch (Throwable e) {
14.                errors.add(e);
15.            }
16.        }
17.     MultipleFailureException.assertEmpty(errors);
18. }
```

Filename: RunAfters.java

Startline: 23

Endline: 40

Example of False Clone Pair

Project : JUnit4

```
1. protected void runChild(final FrameworkMethod
   method, RunNotifier notifier) {
2.     Description description =
   describeChild(method);
3.     if (isIgnored(method)) {
4.         notifier.fireTestIgnored(description);
5.     } else {
6.         Statement statement = new Statement() {
7.
8.             @Override
9.             public void evaluate() throws Throwable
10.            {
11.                methodBlock(method).evaluate();
12.            }
13.        };
14.        runLeaf(statement, description, notifier);
15.    }
}
```

Filename: BlockJUnit4ClassRunner.java

Startline: 91

Endline: 105

```
1. public Result run(Runner runner) {
2.     Result result = new Result();
3.     RunListener listener = result.createListener();
4.     notifier.addFirstListener(listener);
5.     try {
6.
7.         notifier.fireTestRunStarted(runner.getDescription()
   );
8.         runner.run(notifier);
9.         notifier.fireTestRunFinished(result);
10.    } finally {
11.        removeListener(listener);
12.    }
13. }
```

Filename: BlockJUnit4ClassRunner.java

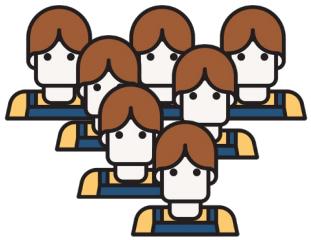
Startline: 131

Endline: 143

การวัดประสิทธิภาพของ
Machine Learning Model
ที่สร้างขึ้น ในเชิงเครื่องมือ

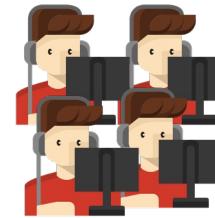
Evaluation of Merry Web Application by Users

RQ3: How likely are command line based tool and our Merry tool adopted
by programmers?



Computer Science
students

Target users



Programmers,
Developers

User study methodology

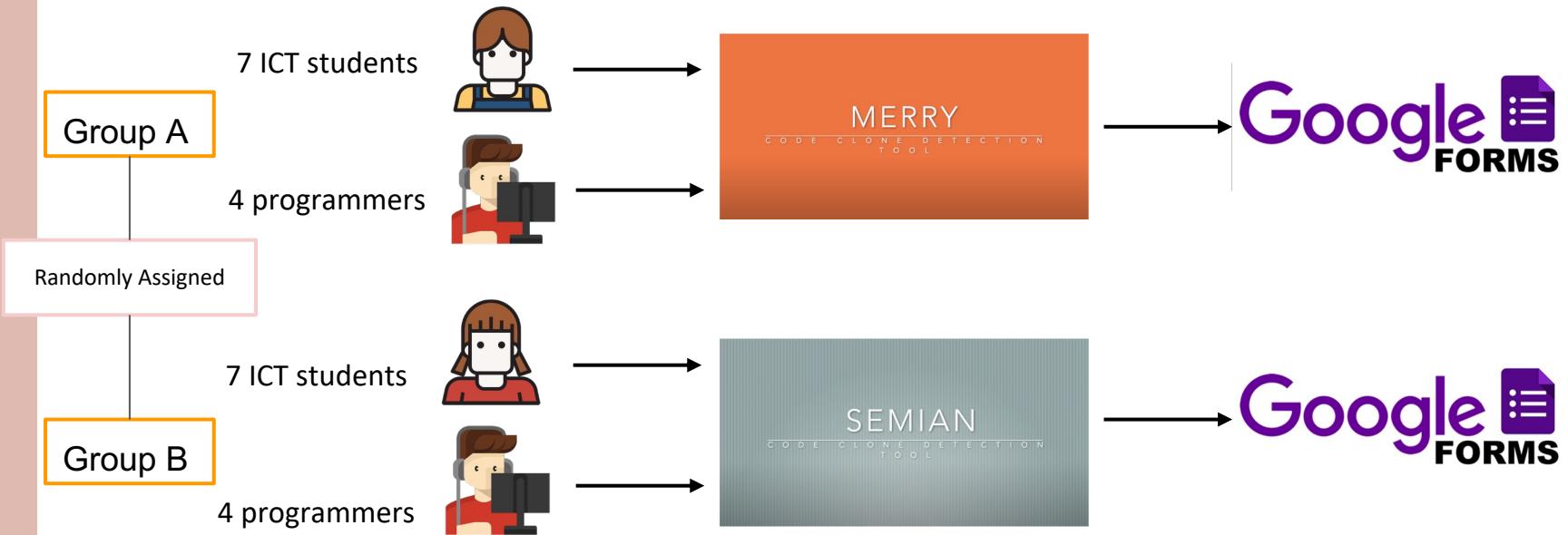
Between-Subjects study design



Reduce the impact of transfer across conditions

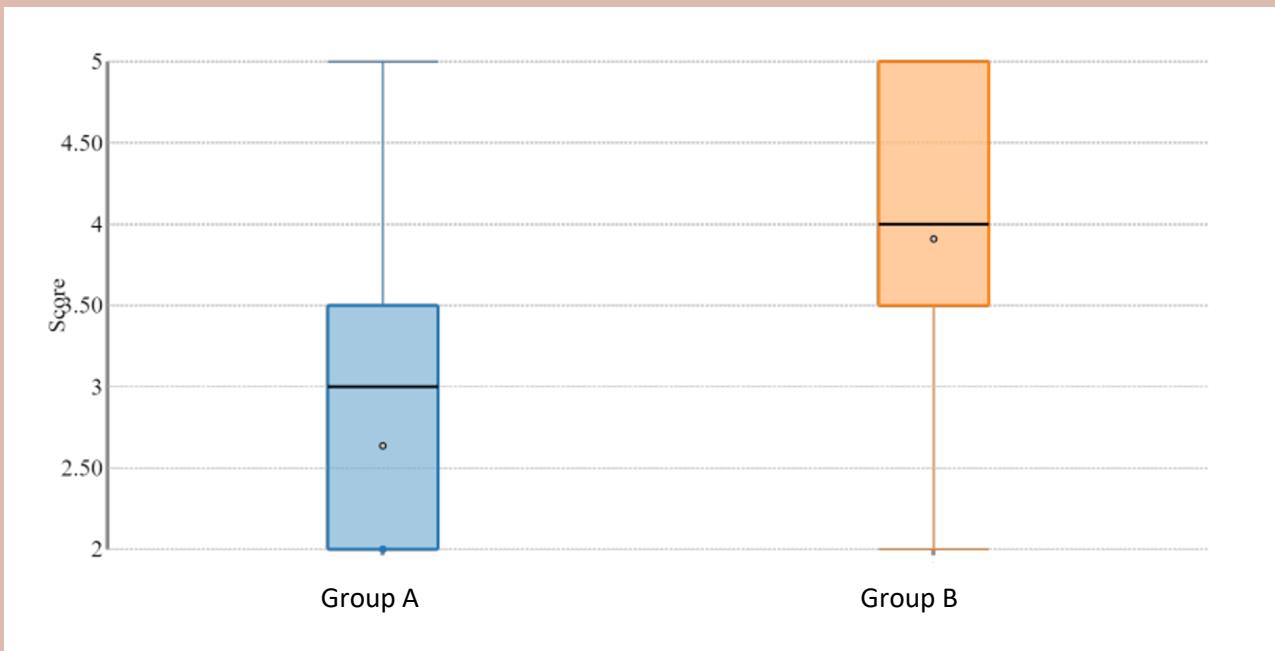
Evaluation of Merry Web Application by Users

RQ3: How likely are command line based tool and our Merry tool adopted by programmers?



Likeliness of using the tool

If this tool will be applied to one of your software project,
how likely will you use it?

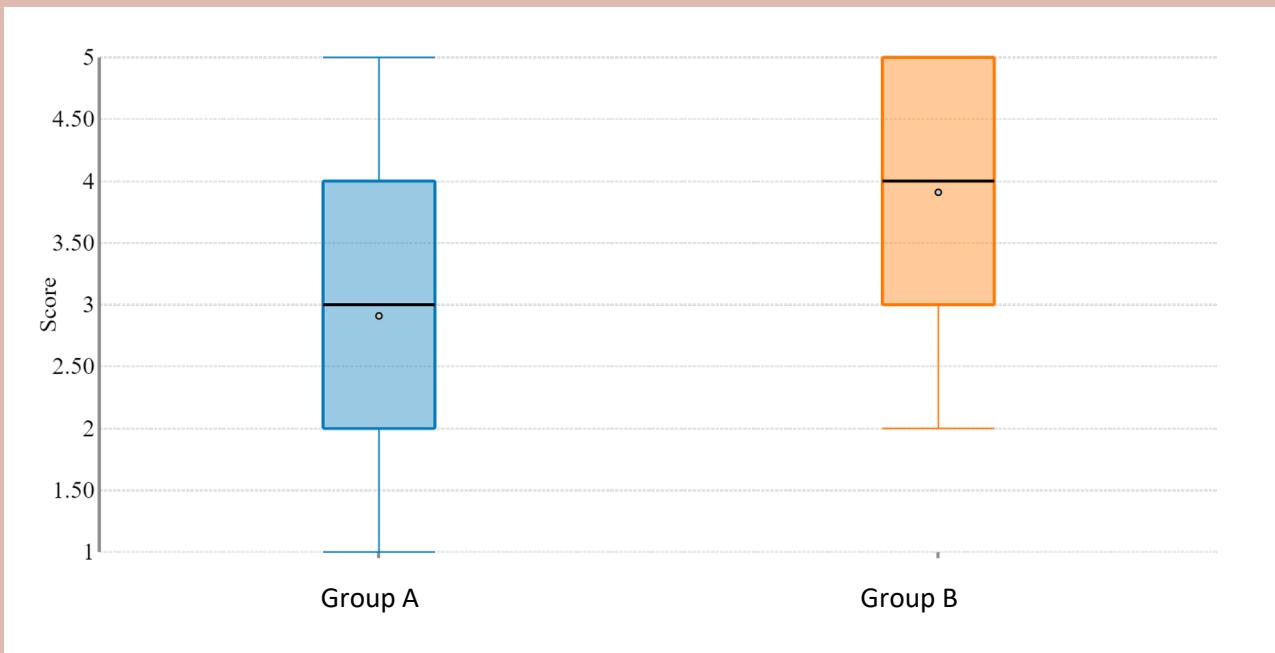


Group A = Simian command line based tool

Group B = Merry web based clone detection tool

Ease of Understanding

Did you find the tool easy to understand
(how to run, how to interpret results)?

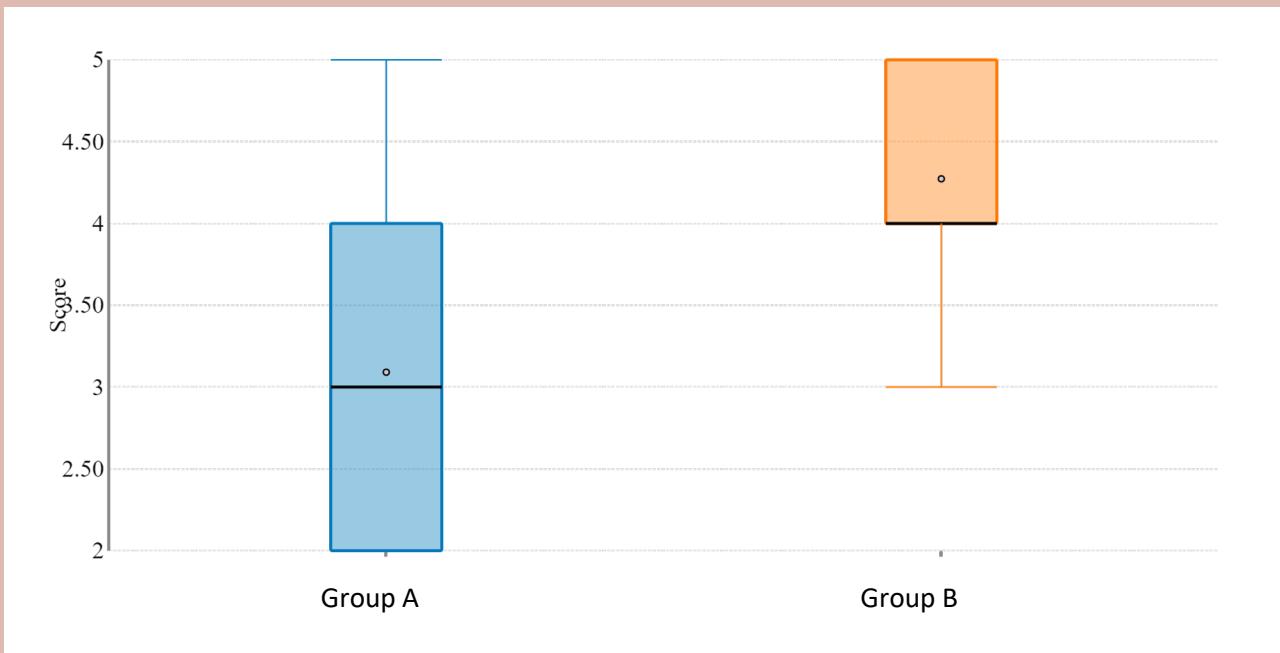


Group A = Simian command line based tool

Group B = Merry web based clone detection tool

Ease of Use

Did you find the tool easy to use
(environment needed to run to the tool and see the results)?



Group A = Simian command line based tool

Group B = Merry web based clone detection tool

บทสรุปและแนวทางการพัฒนา ต่อในอนาคต

Conclusion

Conclusion

- We create a web-based code clone detection tool that use machine learning techniques, called Merry.
- The aim is to accurately detect clones and improve user experience.

Merry Engine

- The clone detection ML engine is based on 4 machine learning models.
- Evaluation of the performance of Merry by using BigCloneBench shows high precision and recall.
- Evaluation on the real software projects has varied performance.

Merry Web Application

- The web application is integrated with GitHub.
- It offers a user friendly and convenient code clone detection function.

Problems and Limitations

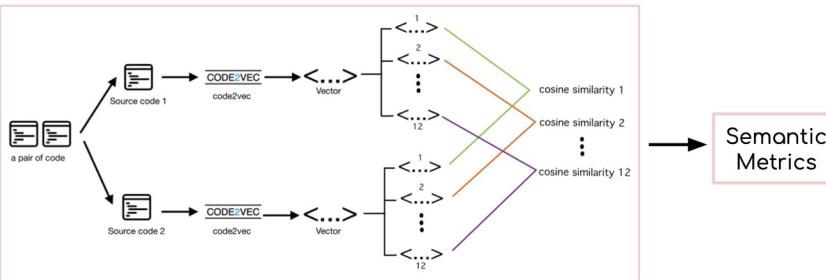
- Supports only Java language
- code2vec poor run-time performance
- The evaluated precision and recall might not reflect on real software projects.
- Current MongoDB configuration has an issue where there is large amount.
- The user study results may be biased due to the sample size is small.

Future Work

- Expand the tool to detect clones in other languages
- Improve the code2vec run-time
- Create dedicated machine learning model per clone type
- Solve the MongoDB limitation by query a part of MongoDB document at a time
- Expand the number of participants in the user study.

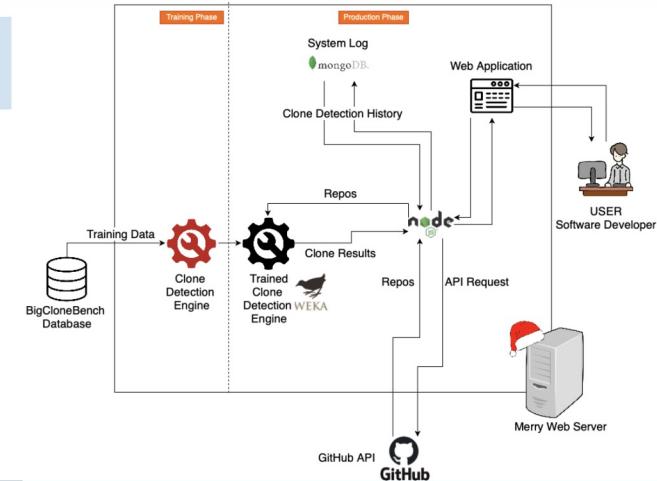
1.2

Semantic Code Metrics



20

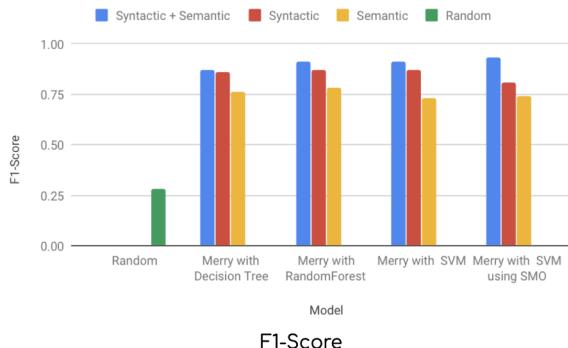
System Architecture



27

Evaluation of Merry ML Clone Detection Engine using BCB (cont.)

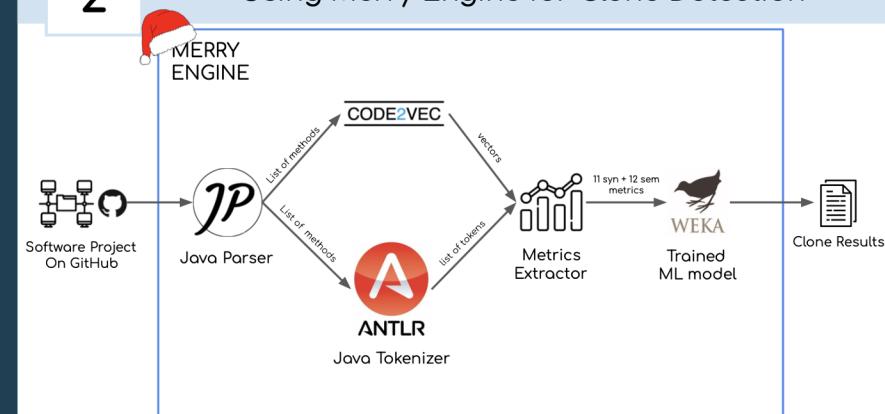
RQ1: How accurate is Merry code clone detection on BigCloneBench?



37

2

Using Merry Engine for Clone Detection



24



CODE COMBAT

Google



Let's Code
Thailand



THAI
PROGRAMMER



ศ. E-SAN Thailand Coding & AI Academy

โครงการวิจัยโมเดลระบบนิเวศการเรียนรู้ที่บูรณาการ CODING & AI สำหรับเยาวชน

Model of Learning Ecosystem Platform integrate with Coding & AI for Youth

โครงการย่อยที่ 6

การพัฒนาเยาวชนเพื่อเข้าสู่วิชาชีพขั้นสูงด้าน Coding & AI

ร่วมกับ Coding Entrepreneur & Partnership: Personal AI

ชื่อหัวข้อ Code Clone Detector - ระบบ AI สำหรับช่วยตรวจวัดความเหมือนของโค้ดและ
ตรวจจับการคัดลอกโค้ด

ดร. ชัยยงค์ รักขิตเวชสกุล





Coding Workshop

- ดาวน์โหลดแบบฝึกหัดได้ที่ <https://colab.research.google.com/drive/1qZdJlYtSeLt3D2czSyFSch824-9AnJvu?usp=sharing> หรือ Scan QR Code ด้านล่าง

