AI for detecting code plagiarism

1. What are Code Clones

คือ โค้ดที่มีความเหมือนหรือคล้ายกันมาก

Type 1 เหมือนกัน 100 %

```
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;
}
```

Type 2 มีการแก้ไขเล็กน้อย

```
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 มีการแก้ไข ลบ เพิ่มเติมออก

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

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

TYPE 4 ได้ผลลัพธ์เหมือนกัน แต่ เป็นคนละระดับ

```
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.contain(".") ){
        return null;
    }
    int pos = filename.lastIndexOf(".");
    return filename.substring(pos+1);
}
```

Preprocessing

ทำให้โค้ดเหมือนกัน

Transformation

เปลี่ยน โค้ด ให้อยู่ในเวเมตอร์

Match Detection

ดูว่า โค้ดในตอนเหมือน กับ รีปร่าง ตั้งนี้ยังคงเหมือน

Formatting

ตรวจสอบผลลัพธ์ให้สวยงาม

Post Processing Filtering

ตัดข้อมูลที่ไม่จำเป็น

Aggregation

รวมผลลัพธ์ให้เป็นใช้

Merry Engine

2. สร้างModel

Building Merry Engine อีฮีสร้าง

มี 3 ขั้นตอน

1. เราเก็บData

ใช้ข้อมูลจาก BigClone Bench

Java Source Code Files → เอาออกมาใช้

เอา Files โค้ด

clone Ground-truth → จับคู่ ไฟล์ที่มีโค้ดเหมือนกัน (โคลนกัน)

เฉลย

ใช้ Train 22,66 ไฟล์

ใช้ Test 18893 ไฟล์

2. สร้าง โค้ด metrics

มี 2 แบบ

Syntactic metrics

ดูจากข้อกำหนด, กฎ, ธีซีตาร์

โค้ด 2 ตัว → ค่าต่างๆของโค้ด → ลบกัน ดูความแตกต่าง

ค่าต่างๆของโค้ด

เช่น

Token No [9]
Unique Token No [10]
Identifier No [10]
Unique Identifier No [10]
Operator No [10]
Token Types Diversity [9]
Diff File Name Score
Diff Method Name Score
Similar Return Type
DiffLOC

Semantic metrics

ดูธีซีตารที่ทำงานของโค้ด

ยุคใหม่ลึก

โค้ด 2 ตัว → ใส่ code2vec สร้าง เวกเตอร์ออกมา

ใส่ code2vec สร้าง เวกเตอร์ออกมา

ดูความแตกต่างของเวกเตอร์

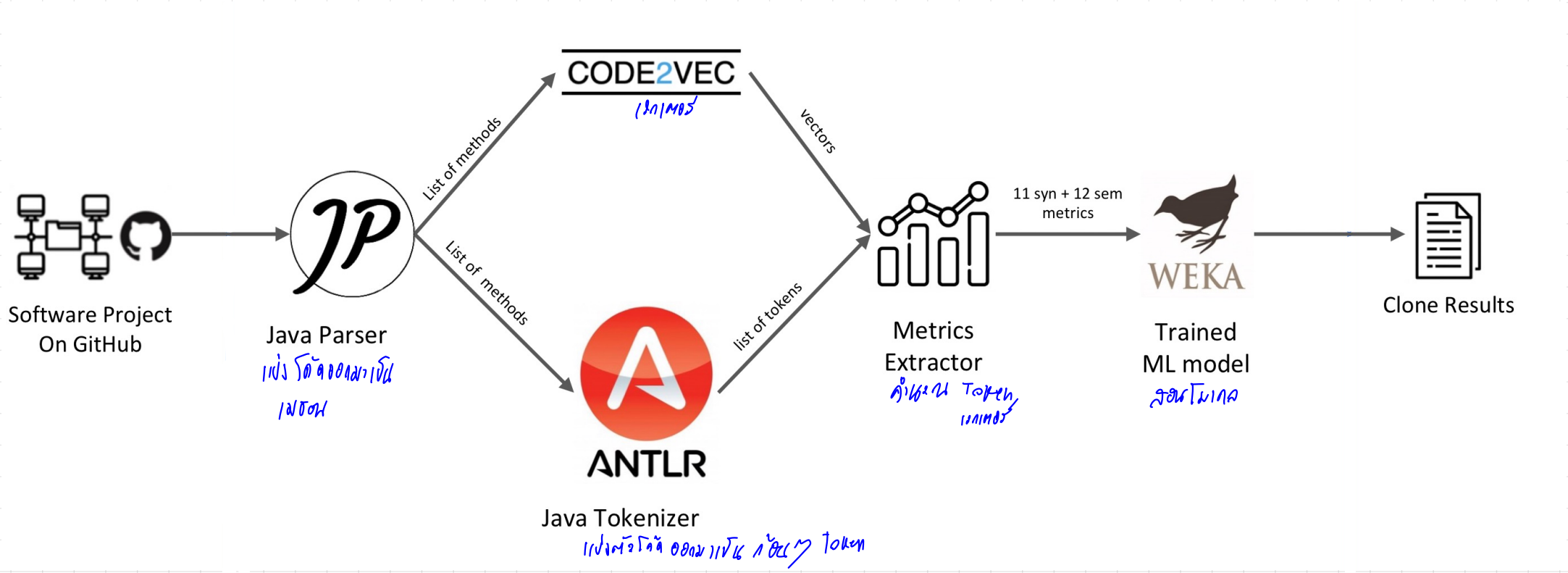
3 มาเลือกใช้ Model ML

ที่ 4 โดยมี

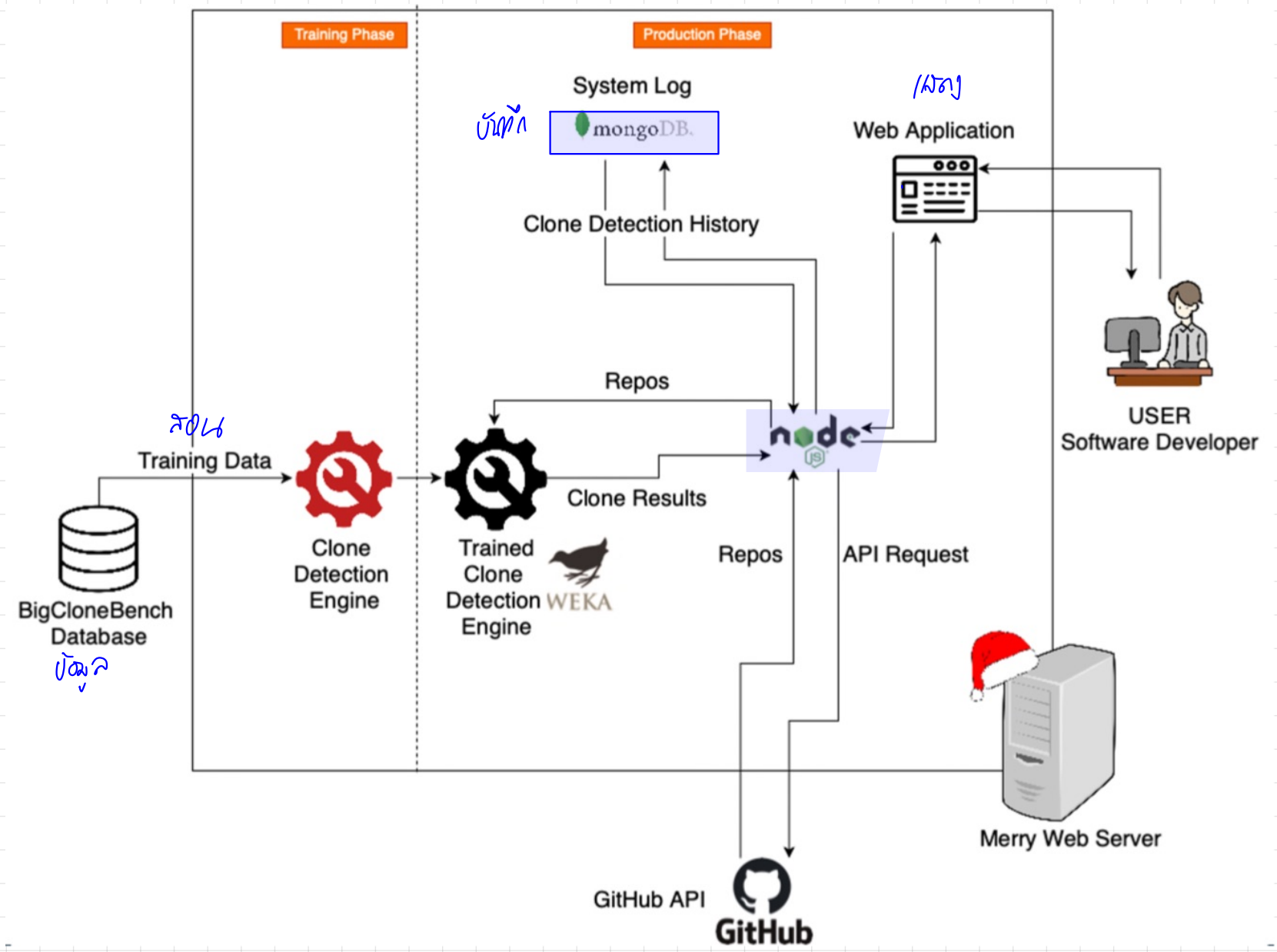
1 Decision Tree
2 Random Forest
3 SVM
4 SVM using SMO

3. วิธีใช้ Model

using Merry Engine for Clone Detection ดังนี้



System Architecture



4. วัดประสิทธิภาพ

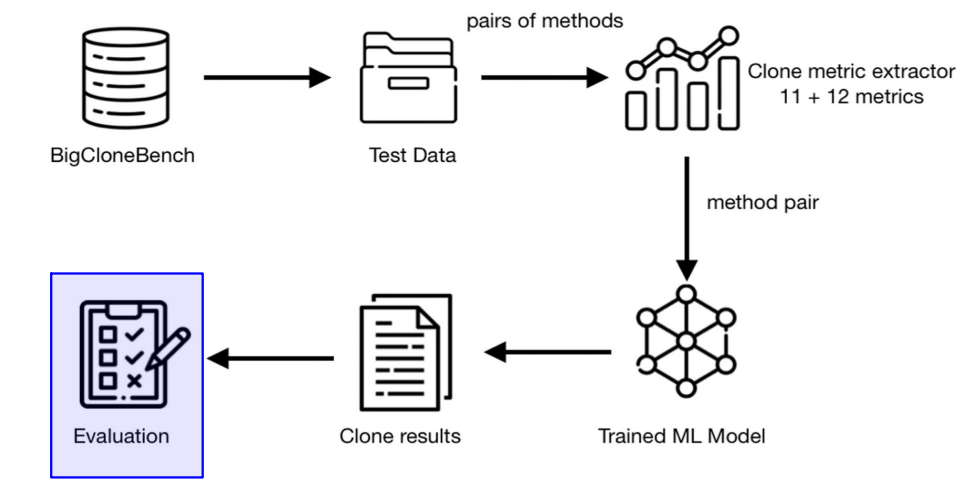
1. ดูค่าความแม่นยำ

มีตัววัด 3 ตัว

1. precision = $\frac{\text{True Positive}}{\text{True Positive + False Positive}}$ เทสเราไม่เน้นว่า จากข้อมูลที่ทำนายว่าใช่ ว่าใช่เหมือนกันกี่เท่าไหร่

2. Recall = $\frac{\text{True Positive}}{\text{True Positive + False Negative}}$ เทสเราไม่เน้นว่า จากข้อมูลที่เป็นคลีนจริงๆ ถูกกี่ตัว ว่าถูกแบบมาเท่าไหร่

3. F1 score = $2 \cdot \frac{\text{Precision} - \text{Recall}}{\text{Precision + Recall}}$ ค่าเฉลี่ยความแม่น



พวก

Model	Metrics	Precision	Recall	F1-Score
Randomization (baseline)	สุ่ม	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

แม่นยำๆ น้อย

แม่นยำที่สุด