

NeuralCCD: Integrating Multiple Features for Neural Coincidental Correctness Detection

Zhou Tao¹, Yan Lei^{1,2*}, Huan Xie¹, Jian Hu¹

¹ School of Big Data & Software Engineering, Chongqing University, Chongqing, China

²Peng Cheng Laboratory, Shenzhen, China

zhoutao@stu.cqu.edu.cn, yanlei@cqu.edu.cn, huanxie@cqu.edu.cn, jianhu@cqu.edu.cn

背景

Fault Localization(FL)

- 不具合箇所を自動的に探索する技術
- デバックコストを削減

問題点

CC (Coincidental Correctness)テストケース

- バグを実行しながらも正しい結果を出力
- ノイズとなりFLモデルの精度が低下



CCテストケースを検出する必要

背景

Spectrum-based Fault Localization(SFL)

- 一般的なFLアプローチ
- カバレッジ情報とテスト結果を入力とし、各プログラム要素の疑いスコアを算出

		<i>N statements</i>				<i>errors(e)</i>
		s_1	s_2	$\dots$	s_N	
<i>M test cases</i>	t_1	x_{11}	x_{12}	$\dots$	x_{1N}	e_1
	t_2	x_{21}	x_{22}	$\dots$	x_{2N}	e_2
	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
	t_M	x_{M1}	x_{M2}	$\dots$	x_{MN}	e_M
		カバレッジ行列				結果ベクトル

Name	formula
Dstar	$\frac{a_{ef}}{a_{nf}+a_{ep}}$
Ochiai	$\frac{a_{ef}}{\sqrt{(a_{ef}+a_{nf})(a_{ef}+a_{ep})}}$
Barinel	$2 * (a_{ef} + \sqrt{a_{np}} + \sqrt{a_{ep}})$
Optimal_P	$a_{ef} - \frac{a_{ep}}{a_{ep}+a_{np}+1}$
Russel & Rao	$\frac{a_{ep}+a_{np}+a_{ef}+a_{nf}}{a_{ef}}$
GP02	$2(a_{ef} + \sqrt{a_{np}}) + \sqrt{a_{ep}}$
GP19	$a_{ef} \sqrt{ a_{ep} - a_{ef} + a_{nf} - a_{np} }$
GP03	$\sqrt{ a_{ef}^2 - \sqrt{a_{ep}} }$
Op2	$a_{ef} - \frac{a_{ep}}{a_{ep}+a_{np}+1}$
Jaccard	$\frac{a_{ef}}{a_{nf}+a_{ep}+a_{np}}$

a_{ef} : 不合格テストケースで実行された1文の数

a_{ep} : 合格テストケースで実行された1文の数

a_{nf} : 不合格テストケースで実行されていない1文の数

a_{np} : 合格テストケースで実行されていない1文の数

SFLの例

t_1	t_2	t_3	t_4	t_5
$a = 2$	$a = 5$	$a = 8$	$a = 4$	$a = 7$
$b = 1$	$b = 3$	$b = 2$	$b = 1$	$b = 4$
$c = 2$	$c = 2$	$c = 3$	$c = 5$	$c = 4$
t_6	t_7	t_8	t_9	t_{10}
$a = 1$	$a = -4$	$a = 0$	$a = 7$	$a = 2$
$b = -2$	$b = -9$	$b = 3$	$b = 2$	$b = 9$
$c = 1$	$c = 1$	$c = 5$	$c = 0$	$c = 0$



CCテストケース

S#		t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}
1	<code>double Test(double a, double b, int c){</code>										
2	<code>double sum = a + b;</code>	●	●	●	●	●	●	●	●	●	●
3	<code>double div = 1;</code>	●	●	●	●	●	●	●	●	●	●
4	<code>double max = b;</code>	●	●	●	●	●	●	●	●	●	●
5	<code>if ((a > 0) &&(b>0))</code>	●	●	●	●	●	●	●	●	●	●
6	<code>div = a / b;</code>	●	●	●	●	●				●	●
7	<code>if (a > b)</code>	●	●	●	●	●	●	●	●	●	●
8	<code>max = b; //Correct: max = a</code>	●	●	●	●	●	●	●		●	
9	<code>if (c == 0)</code>	●	●	●	●	●	●	●	●	●	●
10	<code>return sum;</code>									●	●
11	<code>else if(c == 1)</code>	●	●	●	●	●	●	●	●		
12	<code>return div;</code>						●	●			
	<code>else return max;}</code>	●	●	●	●	●			●		
Verified test results		F	F	F	F	F	P	P	P	P	P

NeuralCCD (提案手法)

手順

1. カバレッジ情報と実行結果を T_f と T_p に分割, cc_e を構成

T_f : 失敗テストケースの集合

T_p : 合格テストケースの集合

cc_e : すべての失敗テストケースと0ではないが大きすぎない割合[22]の合格テストケースで実行された文の集合

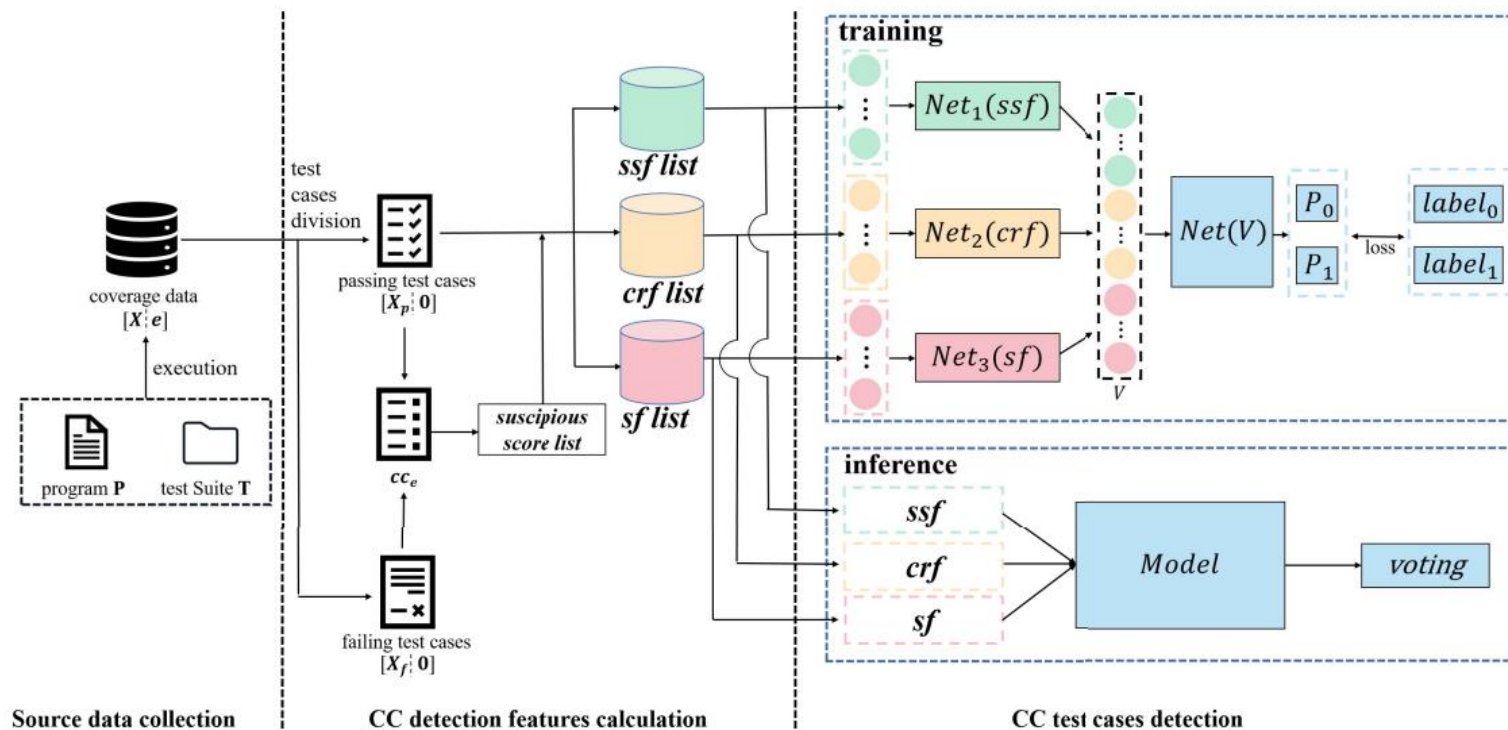


Fig. 3. Overview architecture of NeuralCCD

NeuralCCD (提案手法)

手順

2. cc_e の各文に対して疑いスコアを10パターンのSFL式を用いて算出.
疑いスコアを用い,3つの特徴スコアを算出し,3つの特徴リストにまとめる.

ssf: テストケースの疑いスコア

crf: 疑いスコアが高い文を実行している比率

sf: 失敗テストケースとの類似性

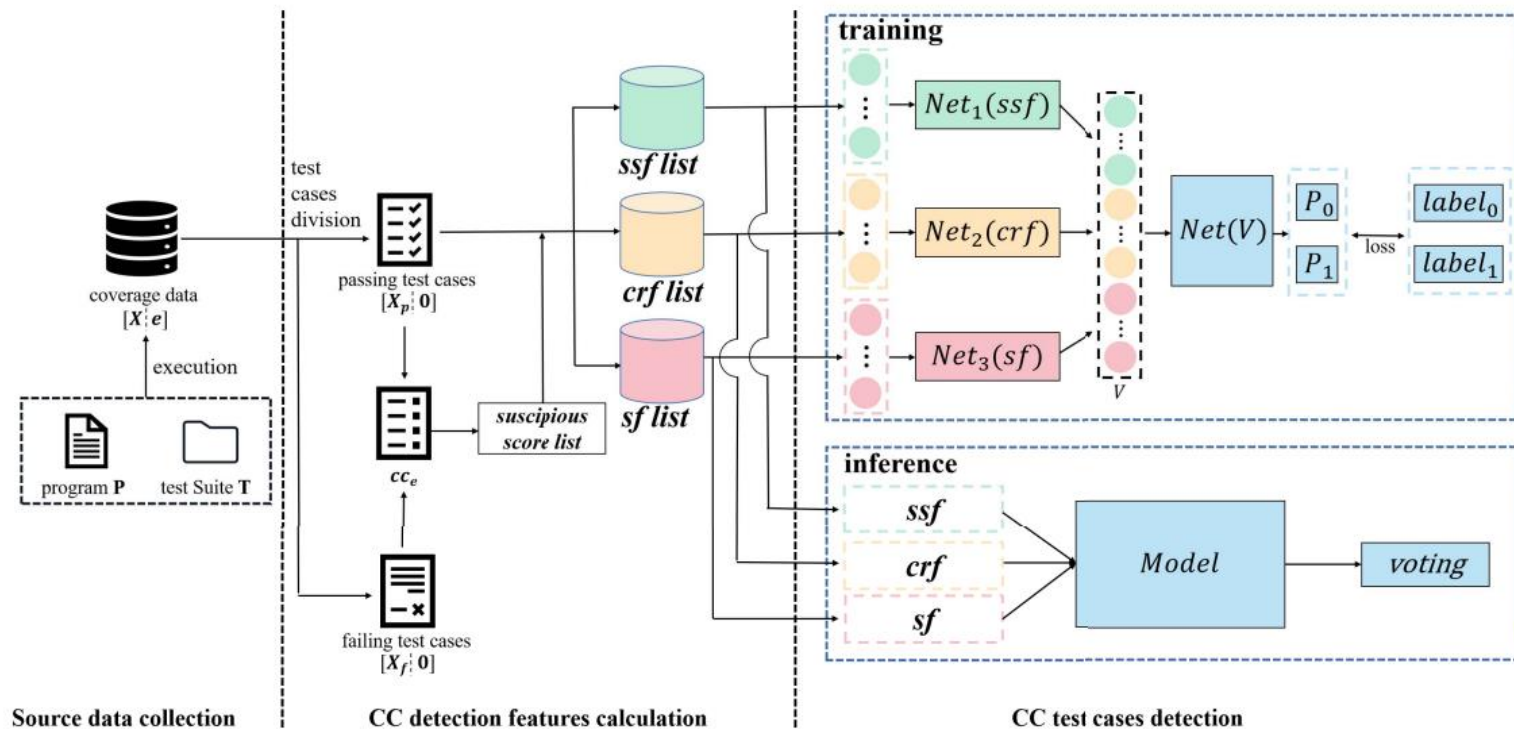


Fig. 3. Overview architecture of NeuralCCD

NeuralCCD (提案手法)

手順

3. 3つの特徴リストを取得し,学習を行う.モデルを組み合わせCCテストケースを検出する

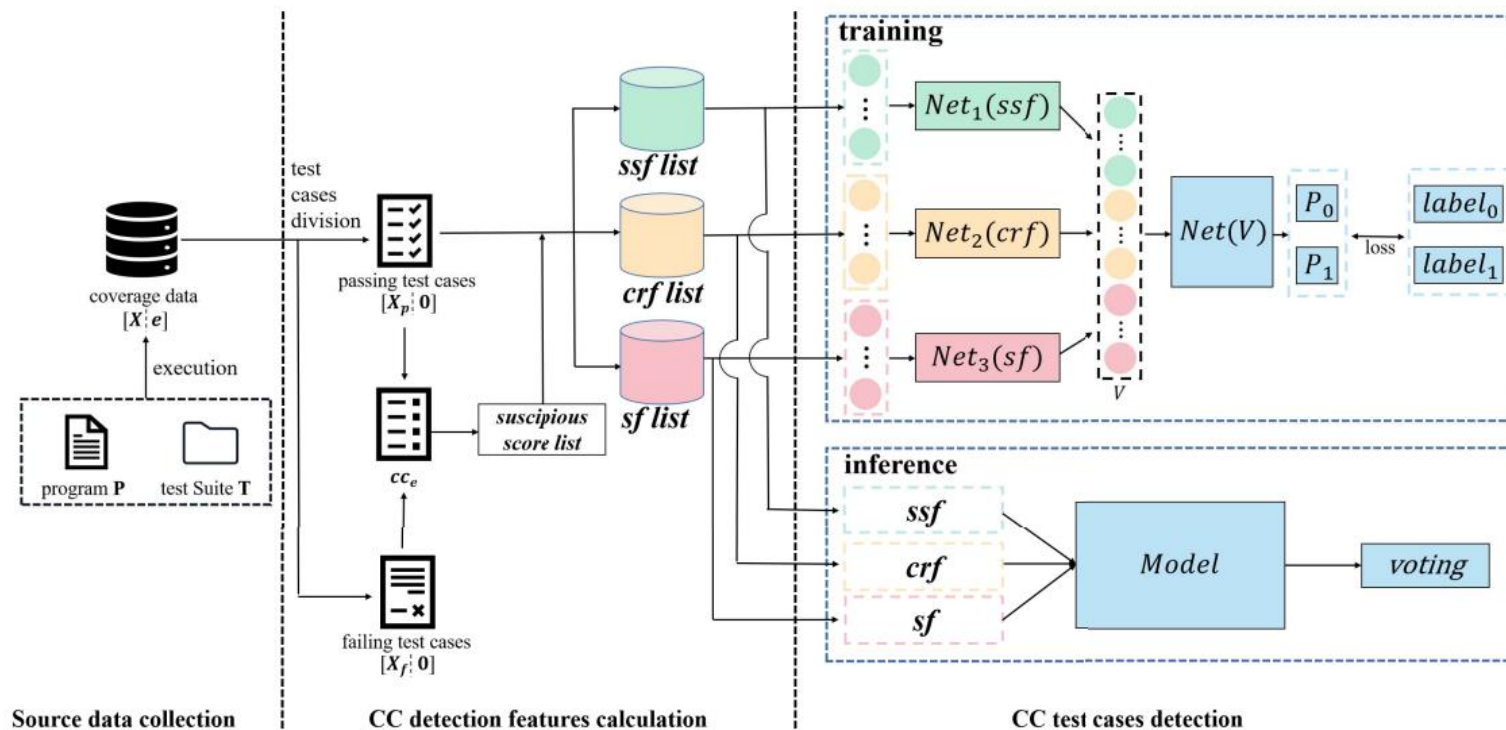


Fig. 3. Overview architecture of NeuralCCD

データセット

代表的な5つのベンチマークプログラムのエラーがある247バージョンを用いる

Program	Description	Versions	LoC (k)	Test
Chart	Java chart library	26	96	2205
Lang	Apache commons-lang	65	22	2245
Math	Apache commons-math	106	85	3602
Time	Standard date and time library	27	28	4130
Mockito	Mocking framework for Java	23	67	1075
total	-	247	298	13257

評価指標

CC検出の評価指標

- Recall: CCテストケース全体のうち検出できた割合
- Precision: 検出したCCテストケースのうち正しく検出できた割合
- F1measure: RecallとPrecisionのバランスをとったもの

FLの評価指標

- Top-K: 疑いスコアをランク付けした際、上位K位までに少なくとも1つ欠陥があるバージョンの数
- Mean First Rank(MFR): プロジェクトにおける各バージョンで最初に位置する欠陥文のランクの平均値を計算したもの
- Wilcoxon signed-rank tests(WSR)[41]

Research Question

RQ1

- CCテストケースの検出において, NeuralCCDは, 最先端のCC検出アプローチと比較して, どのような性能を持つか？

RQ2

- NeuralCCDを故障箇所の特定に利用した場合, 最先端のCC検出アプローチと比較して, どの程度有効か？

RQ3

- 各コンポーネントはNeuralCCDの有効性に寄与しているか？

RQ1: CCテストケースの検出精度

- 提案手法はPrecisionにおいて最も優れている

Program	Approach	Recall	Precision	F1
Chart	<i>Tech-I</i>	96%	35%	52%
	<i>Tech-II</i>	98%	24%	38%
	RF	96%	13%	24%
	CBCFL	58%	18%	28%
	NeuralCCD	32%	67%	43%
Lang	<i>Tech-I</i>	70%	38%	49%
	<i>Tech-II</i>	73%	34%	47%
	RF	89%	15%	26%
	CBCFL	61%	29%	40%
	NeuralCCD	19%	93%	32%
Math	<i>Tech-I</i>	77%	43%	55%
	<i>Tech-II</i>	68%	19%	30%
	RF	95%	15%	26%
	CBCFL	56%	23%	33%
	NeuralCCD	16%	60%	25%
Mockito	<i>Tech-I</i>	82%	34%	48%
	<i>Tech-II</i>	81%	37%	51%
	RF	99%	32%	48%
	CBCFL	86%	42%	57%
	NeuralCCD	13%	47%	20%
Time	<i>Tech-I</i>	90%	4%	7%
	<i>Tech-II</i>	86%	4%	8%
	RF	99%	3%	6%
	CBCFL	84%	5%	10%
	NeuralCCD	30%	35%	32%

RQ2: 不具合箇所の特定期精度

●従来手法を大幅に上回る

TABLE VII
WSR results on FL.

program	Comparision	2-tailed	1-tailed(right)	1-tailed(left)	Conclusion
Chart	<i>NeuralCCD vs Tech-I</i>	3.51E-03	9.98E-01	1.76E-03	Better
	<i>NeuralCCD vs Tech-II</i>	6.83E-02	9.66E-01	3.42E-02	Better
	<i>NeuralCCD vs RF</i>	7.63E-06	1.00E+00	3.81E-06	Better
	<i>NeuralCCD vs CBCFL</i>	1.56E-02	9.5E-01	7.81E-02	Better
Lang	<i>NeuralCCD vs Tech-I</i>	1.59E-02	9.92E-01	7.93E-03	Better
	<i>NeuralCCD vs Tech-II</i>	2.16E-04	1.00E+00	1.08E-04	Better
	<i>NeuralCCD vs RF</i>	1.21E-04	1.00E+00	6.03E-05	Better
	<i>NeuralCCD vs CBCFL</i>	3.13E-02	1.00E+00	1.56E-02	Better
Math	<i>NeuralCCD vs Tech-I</i>	3.49E-03	9.98E-01	1.74E-03	Better
	<i>NeuralCCD vs Tech-II</i>	3.49E-01	8.25E-01	1.75E-01	Similar
	<i>NeuralCCD vs RF</i>	3.84E-04	1.00E+00	1.92E-04	Better
	<i>NeuralCCD vs CBCFL</i>	2.5E-01	1.00E+00	1.25E-01	Similar
Mockito	<i>NeuralCCD vs Tech-I</i>	3.13E-03	9.98E-01	1.56E-03	Better
	<i>NeuralCCD vs Tech-II</i>	9.59E-01	5.21E-01	4.79E-01	Similar
	<i>NeuralCCD vs RF</i>	6.16E-04	1.00E+00	3.08E-04	Better
	<i>NeuralCCD vs CBCFL</i>	2.29E-01	8.94E-01	1.14E-01	Similar
Time	<i>NeuralCCD vs Tech-I</i>	1.32E-04	1.00E+00	6.59E-05	Better
	<i>NeuralCCD vs Tech-II</i>	1.55E-04	1.00E+00	7.75E-05	Better
	<i>NeuralCCD vs RF</i>	1.91E-06	1.00E+00	9.54E-07	Better
	<i>NeuralCCD vs CBCFL</i>	7.63E-06	1.00E+00	3.81E-06	Better

program	Approach	Top-1			Top-3			Top-5			MFR		
		Dstar	Ochiai	Barinel	Dstar	Ochiai	Barinel	Dstar	Ochiai	Barinel	Dstar	Ochiai	Barinel
Chart	<i>Tech-I</i>	2	2	2	2	2	2	4	4	4	783.26	725.68	469.11
	<i>Tech-II</i>	2	2	1	2	2	2	3	3	4	878.84	831.00	504.53
	RF	0	0	0	0	0	0	0	0	1	1032.53	1034.74	1529.47
	CBCFL	1	1	1	1	1	1	2	3	3	467.60	455.40	258.27
	NeuralCCD	2	2	2	5	5	3	7	7	4	344.53	284.58	300.05
Lang	<i>Tech-I</i>	1	1	0	7	7	1	13	13	6	78.35	75.48	77.93
	<i>Tech-II</i>	2	2	0	10	10	3	13	13	8	82.65	82.17	82.46
	RF	0	0	0	1	1	1	3	3	1	182.87	213.80	330.24
	CBCFL	0	1	2	0	1	2	1	4	5	77.08	77.00	55.96
	NeuralCCD	2	2	2	6	6	4	15	17	12	22.57	22.43	37.39
Math	<i>Tech-I</i>	5	5	3	8	8	3	9	9	6	237.99	250.71	253.58
	<i>Tech-II</i>	3	3	2	6	6	4	8	8	6	220.39	224.43	220.61
	RF	2	2	0	3	3	0	4	4	0	267.36	267.88	391.50
	CBCFL	1	2	6	1	2	6	1	2	6	227.73	226.51	168.37
	NeuralCCD	8	8	7	18	17	9	19	19	12	114.95	127.30	157.50
Mockito	<i>Tech-I</i>	0	0	0	1	1	2	1	1	2	481.73	470.90	519.53
	<i>Tech-II</i>	0	0	0	2	2	2	2	2	2	510.60	455.77	421.53
	RF	1	2	2	1	2	2	1	2	3	503.30	508.03	1044.40
	CBCFL	1	1	0	2	2	2	2	2	3	312	302.84	143.24
	NeuralCCD	3	3	2	5	5	5	6	6	5	282.53	209.57	176.3
Time	<i>Tech-I</i>	0	0	0	0	0	0	0	0	0	1802.05	1792.10	838.90
	<i>Tech-II</i>	0	0	0	1	1	1	1	1	1	1779.70	1792.10	838.80
	RF	0	0	0	0	0	0	0	0	0	1920.50	1928.10	1710.20
	CBCFL	0	0	0	0	0	0	0	0	1	1090.18	1080.29	262.12
	NeuralCCD	0	0	1	1	1	4	1	1	5	509.75	281.50	176.00
Total	<i>Tech-I</i>	8	8	5	10	10	8	27	27	18	475.87	471.09	348.21
	<i>Tech-II</i>	7	7	3	20	20	13	27	27	21	487.27	472.54	322.95
	RF	3	4	2	5	6	3	11	12	8	564.89	571.75	755.67
	CBCFL	3	5	9	4	6	11	6	11	18	359.74	354.5	166.85
	NeuralCCD	15	15	14	35	34	25	48	50	38	188.81	149.23	148.32

RQ3: 各コンポーネントの影響度

各特徴をそれぞれ削除し,比較を行った

- 各要素がCC検出に寄与している
- sfが最もCC検出に有効

Program	Approach	Recall	Precision	F1
Chart	<i>NeuralCCD_{ssf-}</i>	17%	90%	29%
	<i>NeuralCCD_{crf-}</i>	0%	NaN	NaN
	<i>NeuralCCD_{sf-}</i>	5%	71%	9%
	<i>NeuralCCD</i>	32%	67%	43%
Lang	<i>NeuralCCD_{ssf-}</i>	17%	92%	29%
	<i>NeuralCCD_{crf-}</i>	27%	91%	42%
	<i>NeuralCCD_{sf-}</i>	7%	91%	13%
	<i>NeuralCCD</i>	19%	93%	32%
Math	<i>NeuralCCD_{ssf-}</i>	14%	59%	23%
	<i>NeuralCCD_{crf-}</i>	17%	62%	27%
	<i>NeuralCCD_{sf-}</i>	3%	65%	6%
	<i>NeuralCCD</i>	16%	60%	25%
Mockito	<i>NeuralCCD_{ssf-}</i>	3%	73%	6%
	<i>NeuralCCD_{crf-}</i>	12%	46%	19%
	<i>NeuralCCD_{sf-}</i>	1%	87%	2%
	<i>NeuralCCD</i>	13%	47%	30%
Time	<i>NeuralCCD_{ssf-}</i>	11%	42%	17%
	<i>NeuralCCD_{crf-}</i>	20%	42%	27%
	<i>NeuralCCD_{sf-}</i>	3%	34%	6%
	<i>NeuralCCD</i>	30%	35%	32%

まとめ

- FLにおいてCCがノイズとなる
- CC検出モデルNeuralCCDを提案
- 本アプローチがCC検出に対して有効