

Who Should Fix This Bug?

John Anvik, Lyndon Hiew and Gail C. Murphy

60191055 山崎大輝

はじめに

開発者へのレポートの割り当てを容易にするため半自動化
アプローチを用いる

機械学習アルゴリズムを用いることで、triagerはそのバグを直すのに
適した開発者に割り振ることができる。

Bugzilla-based repositoriesを用いる

Bugzilla Bug 4746

DCR: TreeItem needs removeAll() method (1GG0NL0)

Last modified: 2005-04-06 11:01:35

Bug List: (This bug is not in your last search results) [Show last search results](#) [Search page](#) [Enter new bug](#)**[Eclipse]**
Bug#: [4746](#)**Product:** Platform **Component:** SWT **Status:** RESOLVED**Resolution:** FIXED**Assigned To:****QA Contact:** **URL:** **Summary:** DCR: TreeItem needs removeAll() method (1GG0NL0)**Status Whiteboard:** **Keywords:** **Hardware:** All **OS:** All **Version:** 2.0 **Priority:** P3 **Severity:** normal **Target****Milestone:** **Reporter:****Add CC:** **CC:**

Attachment	Type	Created	Size	Actions
Create a New Attachment (proposed patch, testcase, etc.)				View All

Bug 4746 depends on: [Show dependency tree](#)**Bug 4746 blocks:** **Votes:** 0 [Show votes for this bug](#) [Vote for this bug](#)**Description:** [\[reply\]](#)**Opened:** 2001-10-11 14:22

TreeItem should really have a removeAll() method just like Tree.

NOTES:

----- *Comment #1* From [\[redacted\]](#) 2001-10-29 16:35 [\[reply\]](#) -----PRODUCT VERSION:
Build 125

機械学習について

Text categorizationがもっとも関連深い

バグレポートのことをinstanceと呼ぶ

Instanceにはカテゴリまたはラベルを付与できる

教師付き機械学習アルゴリズムは、instanceのセットを入力として受け取って分類器を生成する

これにより生成された分類器は未知のinstanceにラベルを割り当てるために使用される

A SEMI-AUTOMATED APPROACH TO BUG ASSIGNMENT

4stepから成り立つ

1. characterizing bug reports バグの特徴づけ
2. assigning a label to each report レポートにラベルをつける
3. choosing reports to train the supervised machine learning algorithm
機械学習のトレーニングのためにレポートを選ぶ
4. applying the algorithm to create the classifier for recommending assignments.
割り当て推薦を行うための分類器を作るためにアルゴリズムを適用する

A SEMI-AUTOMATED APPROACH TO BUG ASSIGNMENT

Eclipse と Firefox を対象に行う

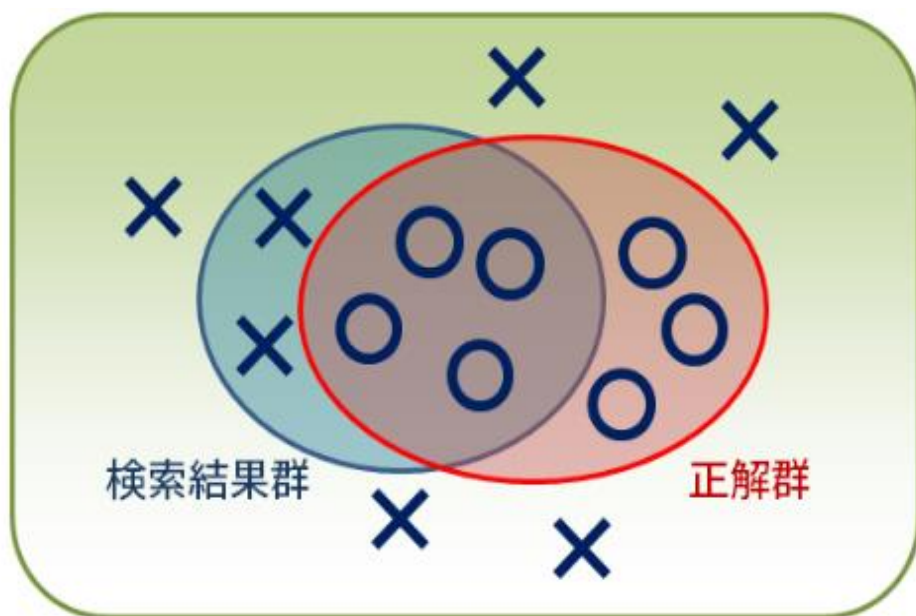
(期間は2004年9月1日から2005年5月31日)

Precision と Recall を尺度に性能を評価していく

Precision・・・適合率

Recall・・・再現率

適合率と再現率について



適合率 Precision = $\frac{4}{6} = 0.67$

再現率 Recall = $\frac{4}{7} = 0.57$

それぞれトレードオフの関係

Step 1

Characterizing Bug Reports

バグレポートがどれに似ているかを理解する必要がある



そのために特徴づけが必要



テキストを特徴付けするためには、まず特徴ベクトルへの変換が不可欠

Step 2

Labeling Bug Reports

分類器をトレーニングするには、
レポートに割り当てられた開発者の名前
または
解決した開発者の名前でラベル付けされた一連のレポート
を提供する必要がある。

Step 2

Labeling Bug Reports

ヒューリスティックスを用いて解決を図る。

そのために4つの項目を定める

- (Firefox)「FIXED」・・・最後に承認されたパッチを出した人
- (Eclipse)「FIXED」・・・レポートを解決済みとした人
- (両方)「DUPLICATE」・・・重複済みのレポートを解決した人
- (Firefox)「WORKSFORME」・・・分類不能

Step 3

Selecting Training Reports

はじめに 有用でないラベルを除外する

次に、プロジェクトでもう作業していない開発者や少数のバグしか修正していない開発者に関する情報を提供するレポートを除外する。

効果的と思われるフィルタは

3ヶ月で9以上のバグを発見したかどうか

Step 3

Selecting Training Reports

Table 2: The effect of developer profile filtering on recommender accuracy and recall.

		# Dev.		Precision/Recall (%)	
		Firefox	Eclipse	Firefox	Eclipse
No Profile		414	146	23/1	58/7
>1 Fix in 3 mo.		94	82	59/2	57/7
Avg. Fixes Per Month Over 3 mo.	1	66	50	64/2	57/7
	2	33	42	59/2	57/7
	3	26	40	64/2	57/7
	4	21	40	64/2	58/7
	5	18	39	59/2	59/7
	6	13	37	45/1	57/7

Step 4

Applying a Machine Learning Algorithm

3つのアルゴリズムの結果を比較後、SVMを選択した。

Naïve Bayes , SVM, and C4.5

VALIDATING WITH GCC

GCCコンパイラプロジェクトを用いて検証してみる。

結果は...

1つは6パーセント、2つは18パーセントと期待よりも低い数値に

Table 4: Precision and recall for gcc.

Predictions	Precision/Recall
1	6/0.3
2	18/2
3	18/3

VALIDATING WITH GCC

なぜこんな結果になったのか？

プロジェクトにあわなかった

PrecisionとRecallを正確に測定するのが困難だった

この2つと考えられる

DISCUSSION

- Is >50% Precision Good Enough?

このアプローチでは、これらの再割り当ての一部を減らすことができ、開発者がシステム上ですでに割り当てられたタスクを完了するための時間を取り戻せる

- Why is the Recall so Low?

トレードオフの関係のため

Unsupervised Machine Learning

教師なし学習アルゴリズムならどうなるか

→EMアルゴリズムを用いてテストの結果

Naïve Bayesよりもパフォーマンスが悪く、使用を中止した

Summary

Eclipse, Firefoxではpresicionは高かったが、gccでは低かった

将来的に

オープンソースシステム上でのバグトリージャーによるアプローチの使用

追加の情報源の調査およびそのアプローチがいつ適用可能かを判断するための手段などが含まれている