Active Refinement of Clone Anomaly Reports

MD 輪講

修士課程 1 年　楊　嘉晨

大阪大学大学院コンピュータサイエンス専攻楠本研究室

2012 年 7 月 4 日（火）

出典 Publication

出典
Publication

Active Refinement of Clone Anomaly Reports

- ICSE 2012
- Similarity and Classification

Lucia, David Lo, Lingxiao Jiang, and Aditya Budi

- Singapore Management University

背景 Introduction

背景
Introduction

コードクローンはソフトウェア保守に対して悪い影響

不具合 (anomaly)があるコードクローンに バグを含む可能性が高い

?

バグが含む不具合があるクローンを

正解 (True Positive)

バグがない不具合があるクローンを

誤検出 (False Positive)

背景
Introduction

コードクローンはソフトウェア保守に対して悪い影響

?

不具合 (anomaly)があるコードクローンに バグを含む可能性が高い

バグが含む不具合があるクローンを

正解 (True Positive)

バグがない不具合があるクローンを

誤検出 (False Positive)

背景
Introduction

コードクローンはソフトウェア保守に対して悪い影響

不具合 (anomaly)があるコードクローンに バグを含む可能性が高い　?

バグが含む不具合があるクローンを

正解 (True Positive)

バグがない不具合があるクローンを

誤検出 (False Positive)

背景
Introduction

コードクローンはソフトウェア保守に対して悪い影響

不具合 (anomaly)があるコードクローンに バグを含む可能性が高い ?

バグが含む不具合があるクローンを

正解 (True Positive)

バグがない不具合があるクローンを

誤検出 (False Positive)

既存研究：クローンに基づく不具合検出

Related Researches: Clone-based Anomaly Detection

クローン間の識別子 (Identifier) の不一致
(Juergens et al., 2009)

E. Juergens, F. Deissenboeck, B. Hummel, and S. Wagner, ``Do code clones matter?'' in Proceedings of the 31st International Conference on Software Engineering. IEEE Computer Society, 2009, pp. 485--495.

クローン周りのコード片の差異(Jiang et al., 2007a)

L. Jiang, Z. Su, and E. Chiu, ``Context-based detection of clone-related bugs,'' in ESEC/FSE, vol. 2007, 2007.

正解の例
Example of True Positive

Linux-2.6.19 から見つけたクローン

fs/sysfs/inode.c

```c
struct dentry * dentry = sd->s_dentry;

if (dentry) {
    /* the fol lowing parts are detected as clones */
    spin_lock(&dcache_lock);
    spin_lock(&dentry->d_lock);
    if (!( d_unhashed(dentry) && dentry->d_inode)) {
        dget_locked(dentry);
        __d_drop(dentry);
        spin_unlock(&dentry->d_lock);
        spin_unlock(&dcache_lock);
         ......
```

drivers/infiniband/hw/ipath/ipath_fs.c

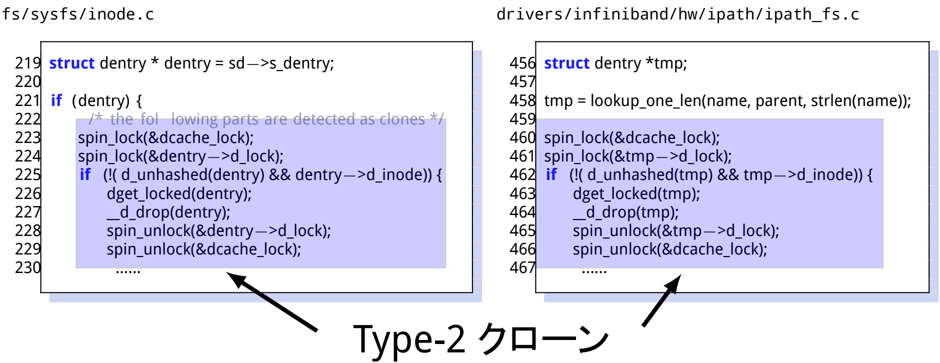
```c
struct dentry *tmp;

tmp = lookup_one_len(name, parent, strlen(name));

spin_lock(&dcache_lock);
spin_lock(&tmp->d_lock);
if (!( d_unhashed(tmp) && tmp->d_inode)) {
    dget_locked(tmp);
    __d_drop(tmp);
    spin_unlock(&tmp->d_lock);
    spin_unlock(&dcache_lock);
     ......
```

正解の例
Example of True Positive

Linux-2.6.19 から見つけたクローン



fs/sysfs/inode.c

```
struct dentry * dentry = sd->s_dentry;

if (dentry) {
    /* the following parts are detected as clones */
    spin_lock(&dcache_lock);
    spin_lock(&dentry->d_lock);
    if (!( d_unhashed(dentry) && dentry->d_inode)) {
        dget_locked(dentry);
        __d_drop(dentry);
        spin_unlock(&dentry->d_lock);
        spin_unlock(&dcache_lock);
        ......
```

drivers/infiniband/hw/ipath/ipath_fs.c

```
struct dentry *tmp;

tmp = lookup_one_len(name, parent, strlen(name));

spin_lock(&dcache_lock);
spin_lock(&tmp->d_lock);
if (!( d_unhashed(tmp) && tmp->d_inode)) {
    dget_locked(tmp);
    __d_drop(tmp);
    spin_unlock(&tmp->d_lock);
    spin_unlock(&dcache_lock);
    ......
```

正解の例
Example of True Positive

Linux-2.6.19 から見つけたクローン

fs/sysfs/inode.c

```
struct dentry * dentry = sd->s_dentry;

if (dentry) {
    /* the following parts are detected as clones */
    spin_lock(&dcache_lock);
    spin_lock(&dentry->d_lock);
    if (!( d_unhashed(dentry) && dentry->d_inode)) {
        dget_locked(dentry);
        __d_drop(dentry);
        spin_unlock(&dentry->d_lock);
        spin_unlock(&dcache_lock);
          ......
```

drivers/infiniband/hw/ipath/ipath_fs.c

```
struct dentry *tmp;

tmp = lookup_one_len(name, parent, strlen(name));

spin_lock(&dcache_lock);
spin_lock(&tmp->d_lock);
if (!( d_unhashed(tmp) && tmp->d_inode)) {
    dget_locked(tmp);
    __d_drop(tmp);
    spin_unlock(&tmp->d_lock);
    spin_unlock(&dcache_lock);
      ......
```

片方に Null であるかの判断

誤検出の例
Example of False Positive

fs/nfsd/nfs3xdr.c

```
423 if  (!( p = decode_fh(p, &args—>fh))
424 ||!(p=decode_filename(p,&args—>name,&args—>len))
425 ||!(p=decode_sattr3(p,&args—>attrs)))
426      return 0;
```

fs/nfsd/nfsxdr.c

```
344 if  (!( p = decode_fh(p, &args—>ffh))
345 ||!(p=decode_fh(p,&args—>tfh))
346 ||!(p=decode_filename(p,&args—>tname,&args—>tlen)))
347      return 0;
```

drivers/hwmon/lm87.c

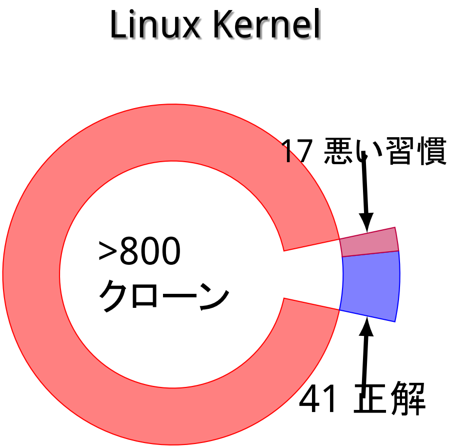
```
688 if  ((err  =  device_create_file(&new_client—>dev,
689                &dev_attr_in6_input))
690        || (err  =  device_create_file(&new_client—>dev,
691                &dev_attr_in6_min))
692        || (err  =  device_create_file(&new_client—>dev,
693                &dev_attr_in6_max)))
694          goto exit_remove;
```

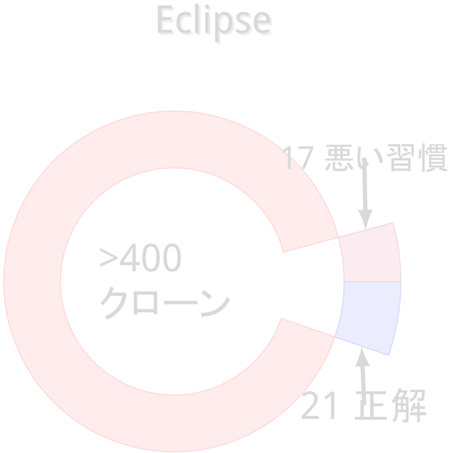
drivers/hwmon/gl520sm.c

```
615 if  ((err  =  device_create_file(&new_client—>dev,
616                &dev_attr_in4_input))
617        || (err  =  device_create_file(&new_client—>dev,
618                &dev_attr_in4_min))
619        || (err  =  device_create_file(&new_client—>dev,
620                &dev_attr_in4_max)))
621          goto exit_remove_files;
```

不具合があるクローンの誤検出率

False Positives in Anomaly Clones



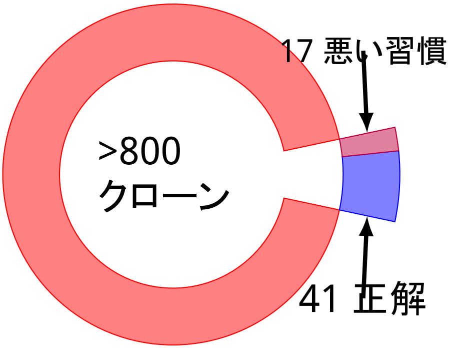


L. Jiang, Z. Su, and E. Chiu, ``Context-based detection of clone-related bugs,'' in ESEC/FSE, vol. 2007, 2007.

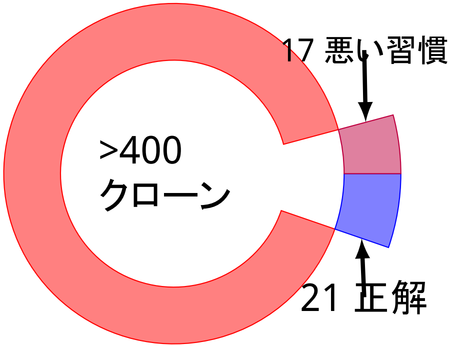
不具合があるクローンの誤検出率

False Positives in Anomaly Clones

Linux Kernel



Eclipse



L. Jiang, Z. Su, and E. Chiu, ``Context-based detection of clone-related bugs,'' in ESEC/FSE, vol. 2007, 2007.

不具合があるクローンの誤検出率 II

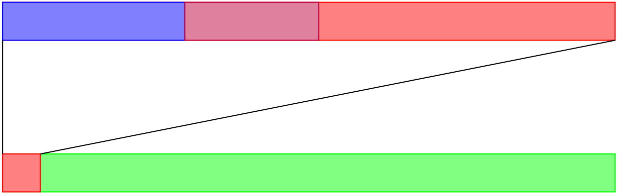
False Positives in Anomaly Clones II

商用ソフトウェア(Gabel et al., 2010)

149 バグがある

109 Code Smells

不明



500 確かめた　8103 不具合 クローン

M. Gabel, J. Yang, Y. Yu, M. Goldszmidt, and Z. Su, ``Scalable and systematic detection of buggy inconsistencies in source code,'' in ACM Sigplan Notices, vol. 45, no. 10. ACM, 2010, pp. 175--190.

コードクローンの四つの象限

4 Quadrants of Code Clone Group

		一貫性	
		Inconsistent	Consistant
可変性	厳格 (Rigid)		
	柔軟 (Flexible)		

コードクローンの四つの象限

4 Quadrants of Code Clone Group

		一貫性	
		Inconsistent	Consistant
可変性	厳格 (Rigid)	✓	
	柔軟 (Flexible)	✓	

コードクローンの四つの象限

4 Quadrants of Code Clone Group

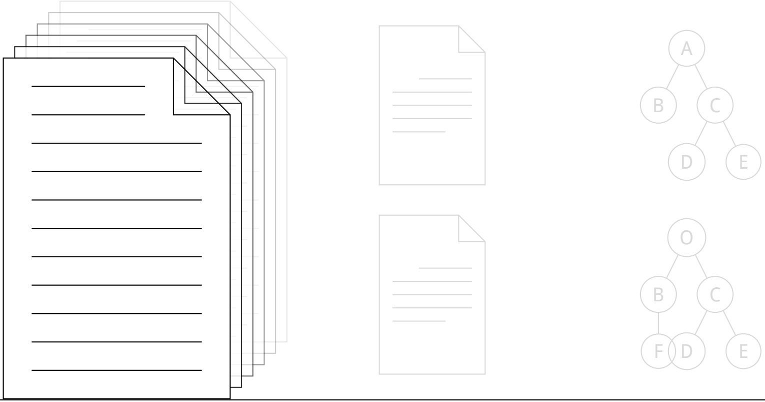
		一貫性	
		Inconsistent	Consistant
可変性	厳格 (Rigid)	✓	
	柔軟 (Flexible)	✓	

クローンに基づく不具合検出

クローンに基づく不具合検出

Clone-based Anomaly Detection

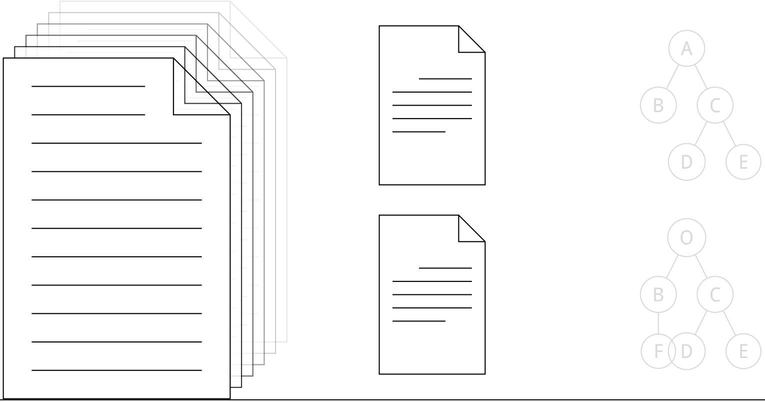
Deckard(Jiang et al., 2007b)



L. Jiang, G. Misherghi, Z. Su, and S. Glondu, ``Deckard: Scalable and accurate tree-based detection of code clones,'' in Proceedings of the 29th international conference on Software Engineering. IEEE Computer Society, 2007, pp. 96--105.

クローンに基づく不具合検出
Clone-based Anomaly Detection

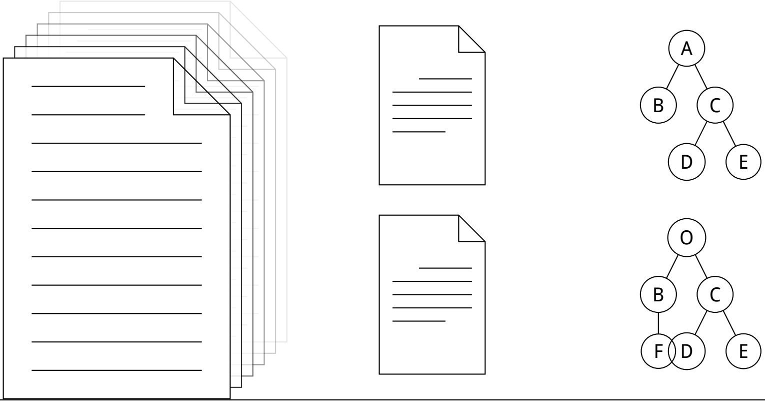
Deckard(Jiang et al., 2007b)



L. Jiang, G. Misherghi, Z. Su, and S. Glondu, ``Deckard: Scalable and accurate tree-based detection of code clones,'' in Proceedings of the 29th international conference on Software Engineering. IEEE Computer Society, 2007, pp. 96--105.

クローンに基づく不具合検出
Clone-based Anomaly Detection

Deckard(Jiang et al., 2007b)

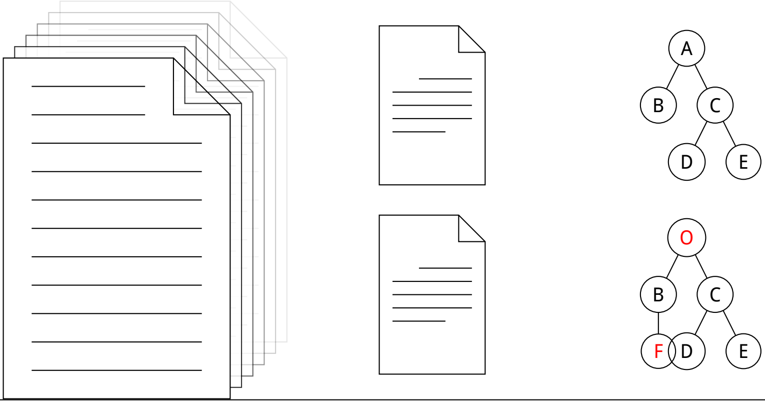


L. Jiang, G. Misherghi, Z. Su, and S. Glondu, ``Deckard: Scalable and accurate tree-based detection of code clones,'' in Proceedings of the 29th international conference on Software Engineering.　IEEE Computer Society, 2007, pp. 96--105.

クローンに基づく不具合検出

Clone-based Anomaly Detection

Deckard(Jiang et al., 2007b)



L. Jiang, G. Misherghi, Z. Su, and S. Glondu, ``Deckard: Scalable and accurate tree-based detection of code clones,'' in Proceedings of the 29th international conference on Software Engineering. IEEE Computer Society, 2007, pp. 96--105.

動的洗練法 Dynamic Refinement

従来のクローンレポート静的洗練法

Static Refinement of Clone Report in Other Researches

ID	内容	Bug?
1	AAA	
2	BBB	
3	CCC	
4	DDD	
5	EEE	
6	FFF	
7	III	
...	...	...

従来のクローンレポート静的洗練法

Static Refinement of Clone Report in Other Researches

ID	内容	Bug?
1	AAA	?
2	BBB	?
3	CCC	X
4	DDD	?
5	EEE	?
6	FFF	?
7	III	X
...	...	...

従来のクローンレポート静的洗練法

Static Refinement of Clone Report in Other Researches

ID	内容	Bug?
1	AAA	?
2	BBB	?
3	CCC	X
4	DDD	?
5	EEE	?
6	FFF	?
7	III	X
…	…	…

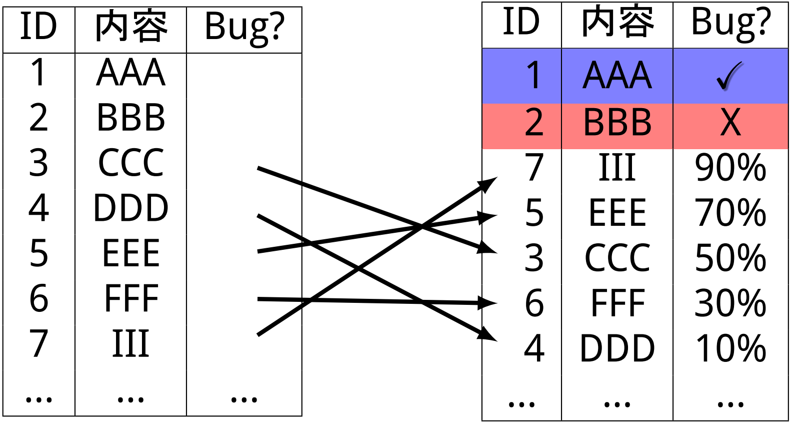
ID	内容	Bug?
1	AAA	✓
2	BBB	X
4	DDD	✓
5	EEE	X
6	FFF	X
…	…	

クローンレポートの動的洗練法
Dynamic Refinement of Clone Report

ID	内容	Bug?
1	AAA	
2	BBB	
3	CCC	
4	DDD	
5	EEE	
6	FFF	
7	III	
...	...	...

クローンレポートの動的洗練法

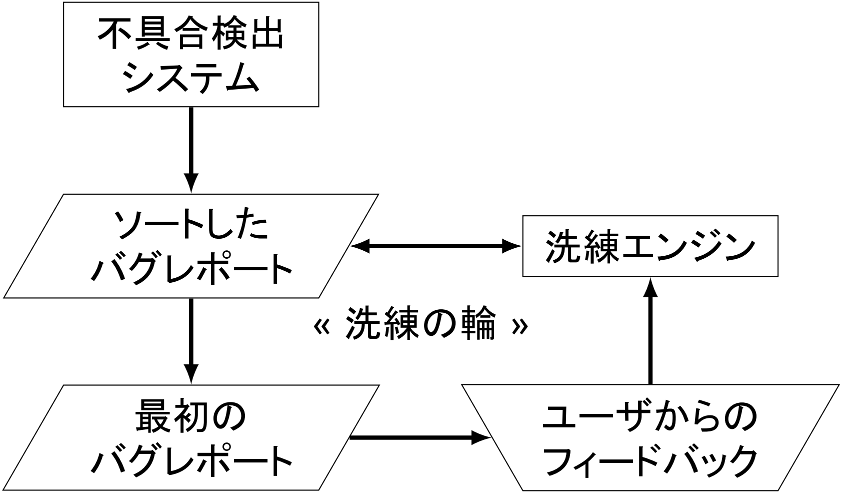
Dynamic Refinement of Clone Report



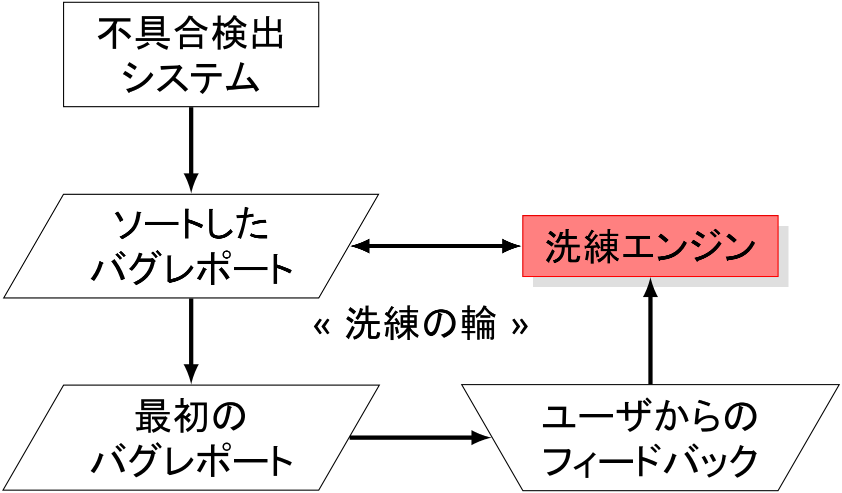
ID	内容	Bug?
1	AAA	
2	BBB	
3	CCC	
4	DDD	
5	EEE	
6	FFF	
7	III	
…	…	…

ID	内容	Bug?
1	AAA	✓
2	BBB	X
7	III	90%
5	EEE	70%
3	CCC	50%
6	FFF	30%
4	DDD	10%
…	…	…

動的洗練法の流れ
Active Refinement Process

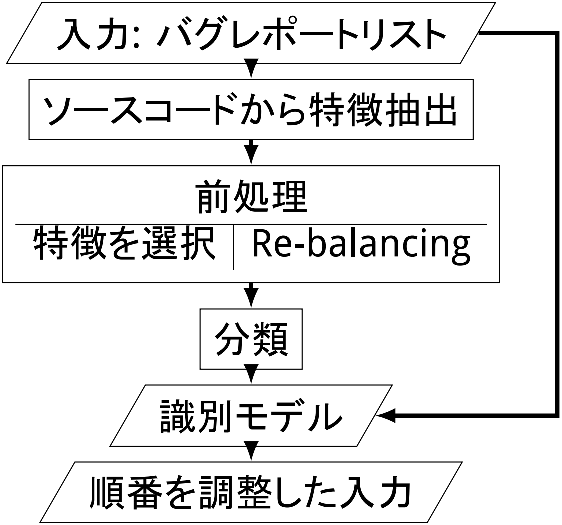


動的洗練法の流れ
Active Refinement Process



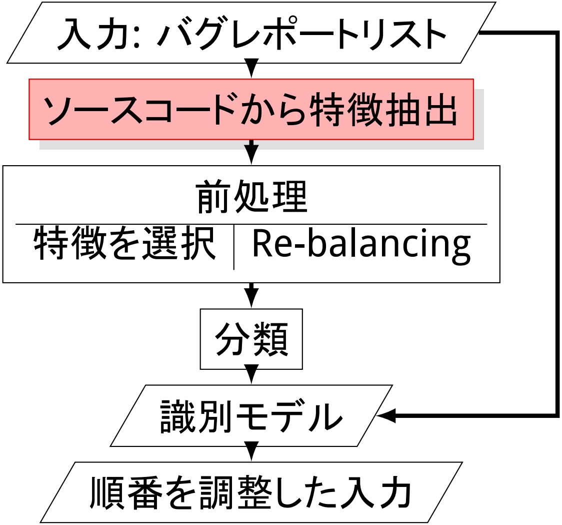
洗練エンジン
Refinement Engine

洗練エンジンの流れ
Process of Refinement Engine



洗練エンジンの流れ

Process of Refinement Engine



入力: バグレポートリスト

ソースコードから特徴抽出

前処理

特徴を選択

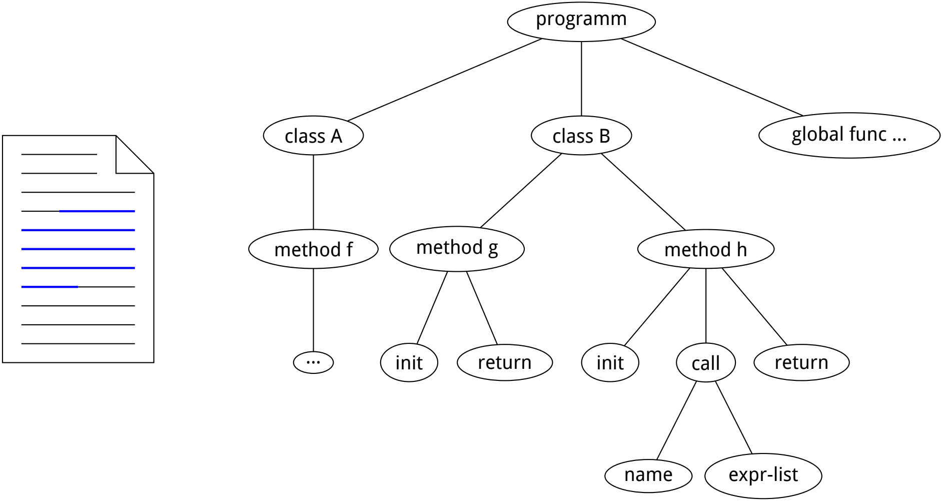
Re-balancing

分類

識別モデル

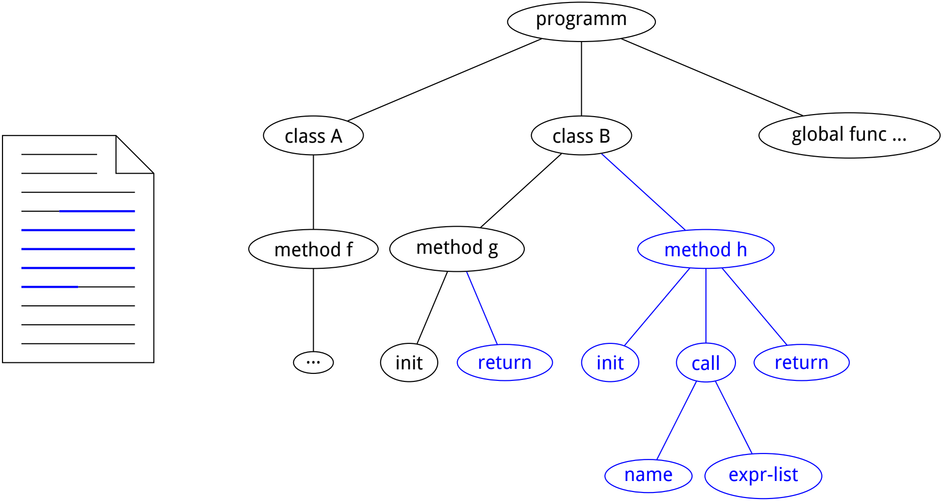
順番を調整した入力

特徴抽出: 構文木を構築
Feature Extraction: Tree Constraction



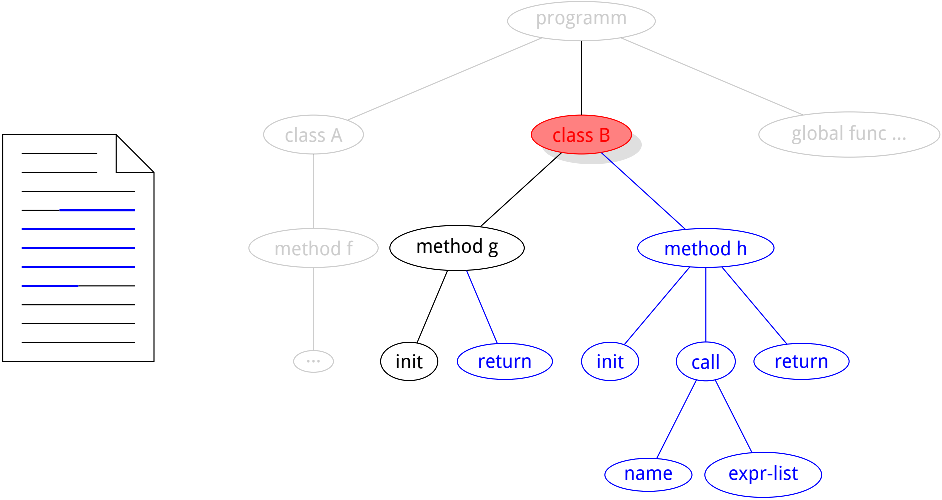
特徴抽出: 構文木を構築

Feature Extraction: Tree Constraction



特徴抽出: 構文木を構築

Feature Extraction: Tree Constraction



特徴抽出: 五つの特徴
Feature Extraction: 5 features

基本的な特徴

Basic Features

対の特徴

Pair Features

基本的な特徴の割合

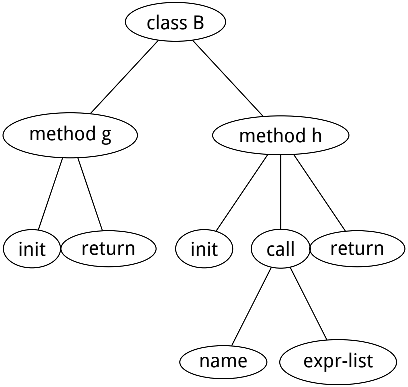
Proportional Features—Basic

対の特徴の割合

Proportional Features—Pair

他の特徴

Other Features



特徴抽出: 五つの特徴
Feature Extraction: 5 features

基本的な特徴

Basic Features

対の特徴

Pair Features

基本的な特徴の割合

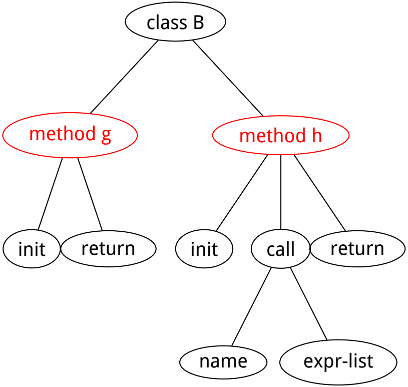
Proportional Features—Basic

対の特徴の割合

Proportional Features—Pair

他の特徴

Other Features



特徴抽出: 五つの特徴

Feature Extraction: 5 features

基本的な特徴

Basic Features

対の特徴

Pair Features

基本的な特徴の割合

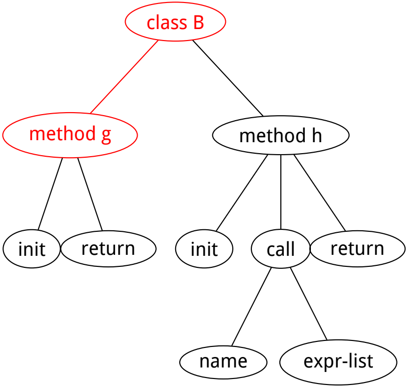
Proportional Features—Basic

対の特徴の割合

Proportional Features—Pair

他の特徴

Other Features



特徴抽出: 五つの特徴
Feature Extraction: 5 features

基本的な特徴

Basic Features

対の特徴

Pair Features

基本的な特徴の割合

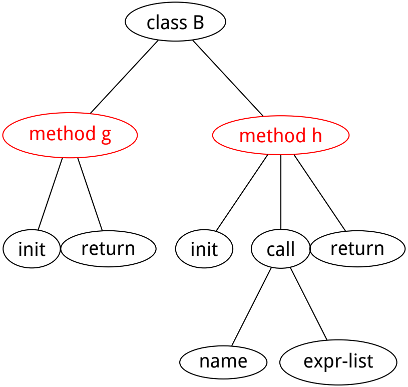
Proportional Features—Basic

対の特徴の割合

Proportional Features—Pair

他の特徴

Other Features



特徴抽出: 五つの特徴
Feature Extraction: 5 features

基本的な特徴

Basic Features

対の特徴

Pair Features

基本的な特徴の割合

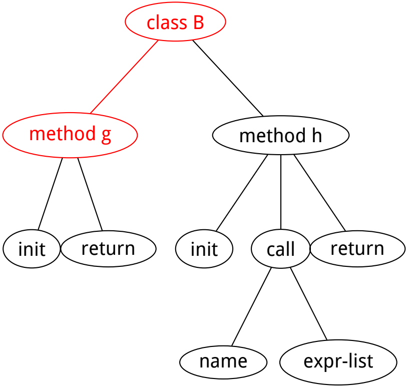
Proportional Features—Basic

対の特徴の割合

Proportional Features—Pair

他の特徴

Other Features



特徴抽出: 五つの特徴

Feature Extraction: 5 features

基本的な特徴

Basic Features

対の特徴

Pair Features

基本的な特徴の割合

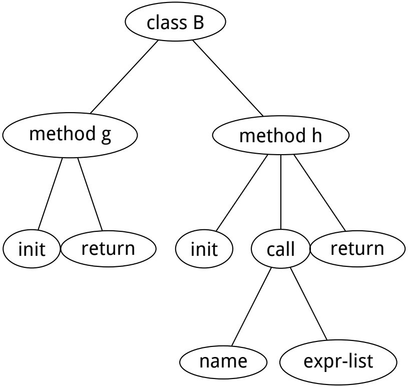
Proportional Features—Basic

対の特徴の割合

Proportional Features—Pair

他の特徴

Other Features



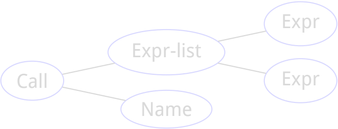
クローンの数 $|CG|$, クローンの平均サイズ $\frac{\sum_{c \in CG} |C|}{|CG|}$

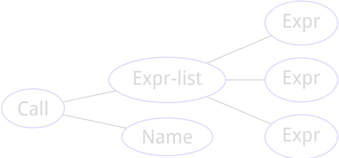
特徴抽出: 五つの特徴の例

Feature Extraction: Example of 5 features

```
decode_sattr3(p, &args—>attrs)
```

```
decode_filename(p, &args—>tname, &args—>tlen)
```





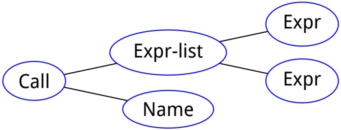
基本		対		基本割合		対割合		他	
タイプ	数	タイプ	数	タイプ	割合	タイプ	割合	タイプ	値
Call	2	Call/Name	2	Call	100%	Call/Name	100%	Num	2
Name	2	Call/ Expr-list	2	Name	100%	Call/ Expr-list	100%	Avg	5.5
Expr-list	2	Expr-list/ Expr	2	Expr-list	100%	Expr-list/ Expr	100%		
Expr	2			Expr	100%				

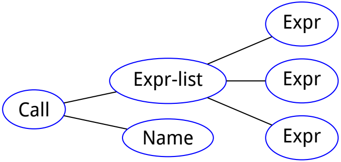
特徴抽出: 五つの特徴の例

Feature Extraction: Example of 5 features

```
decode_sattr3(p, &args—>attrs)
```

```
decode_filename(p, &args—>tname, &args—>tlen)
```





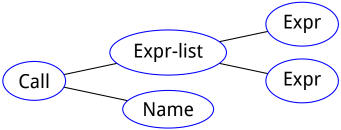
基本		対		基本割合		対割合		他	
タイプ	数	タイプ	数	タイプ	割合	タイプ	割合	タイプ	値
Call	2	Call/Name	2	Call	100%	Call/Name	100%	Num	2
Name	2	Call/ Expr-list	2	Name	100%	Call/ Expr-list	100%	Avg	5.5
Expr-list	2	Expr-list/ Expr	2	Expr-list	100%	Expr-list/ Expr	100%		
Expr	2			Expr	100%				

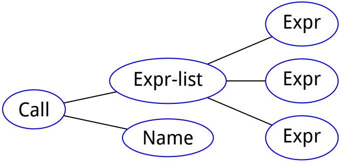
特徴抽出: 五つの特徴の例

Feature Extraction: Example of 5 features

```
decode_sattr3(p, &args−>attrs)
```

```
decode_filename(p, &args−>tname, &args−>tlen)
```

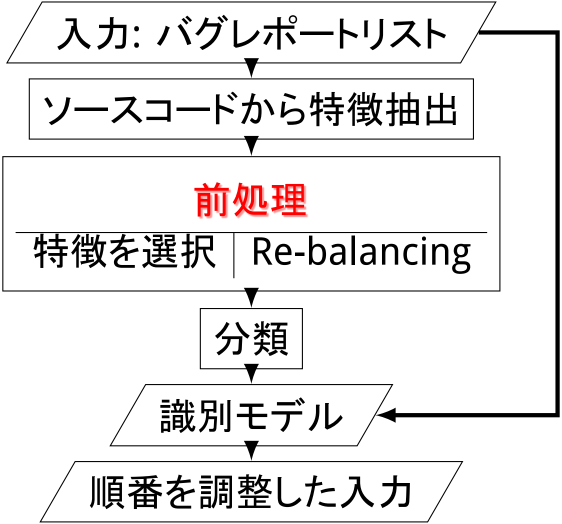




基本		対		基本割合		対割合		他	
タイプ	数	タイプ	数	タイプ	割合	タイプ	割合	タイプ	値
Call	2	Call/Name	2	Call	100%	Call/Name	100%	Num	2
Name	2	Call/ Expr-list	2	Name	100%	Call/ Expr-list	100%	Avg	5.5
Expr-list	2	Expr-list/ Expr	2	Expr-list	100%	Expr-list/ Expr	100%		
Expr	2			Expr	100%				

洗練エンジンの流れ (再掲)

Process of Refinement Engine (Again)



前処理: 特徴選択
Preprocessing: Feature Selection

- c はクローンのクラスラベル, f は一つの特徴
  - 正解は ve クラス, 誤検出は −ve クラス

情報利得 (Information Gain) は:

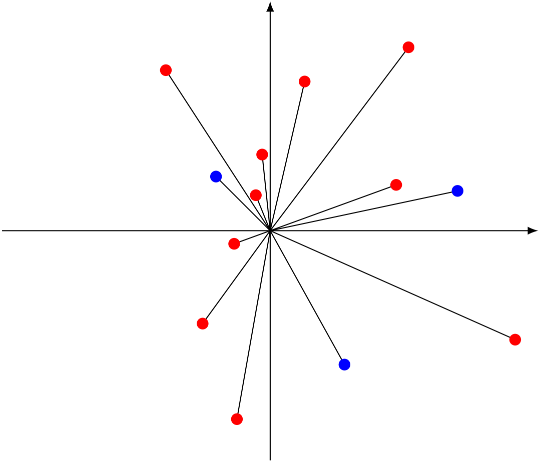
$$\begin{aligned} IG(c|f) &= H(c) - H(c|f) && (1) \\ H(c) &= - \sum_{c_i \in \{\pm ve\}} P(c_i) \log P(c_i) && (2) \\ H(c|f) &= - \sum P(f) \sum_{c_i \in \{\pm ve\}} P(c_i|f) \log P(c_i|f) && (3) \end{aligned}$$

情報利得を基づいて Weka(Holmes et al., 1994) を用いて特徴選択

G. Holmes, A. Donkin, and I. Witten, ``Weka: A machine learning workbench,'' in Intelligent Information Systems, 1994. Proceedings of the 1994 Second Australian and New Zealand Conference on. Ieee, 1994, pp. 357--361.

前処理: Data Re-balancing

Preprocessing: Data Re-balancing



Cosine-similarity
(Kantardzic, 2011)

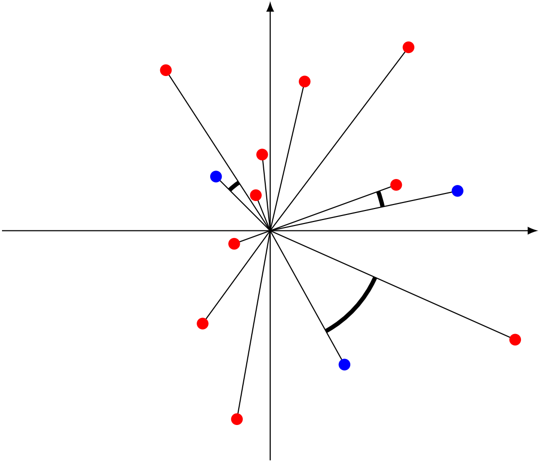
M. Kantardzic, Data mining: concepts, models, methods, and algorithms. Wiley-IEEE Press, 2011.

nearest neighbor approach
(Renieres and Reiss, 2003)

M. Renieres and S. Reiss, ``Fault localization with nearest neighbor queries,'' in Automated Software Engineering, 2003. Proceedings. 18th IEEE International Conference on. IEEE, 2003, pp. 30--39.

前処理: Data Re-balancing

Preprocessing: Data Re-balancing



Cosine-similarity
(Kantardzic, 2011)

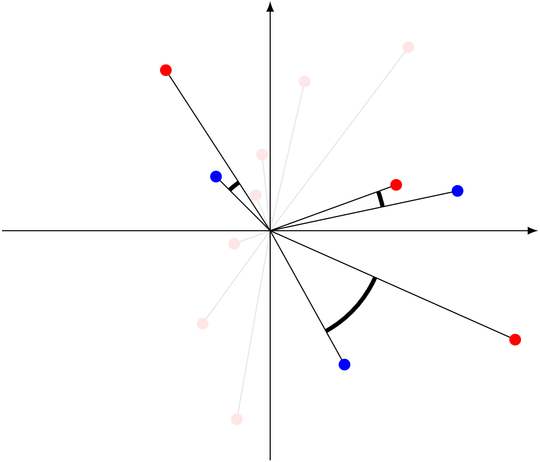
M. Kantardzic, Data mining: concepts, models, methods, and algorithms. Wiley-IEEE Press, 2011.

nearest neighbor approach
(Renieres and Reiss, 2003)

M. Renieres and S. Reiss, ``Fault localization with nearest neighbor queries,'' in Automated Software Engineering, 2003. Proceedings. 18th IEEE International Conference on. IEEE, 2003, pp. 30--39.

前処理: Data Re-balancing

Preprocessing: Data Re-balancing



Cosine-similarity
(Kantardzic, 2011)

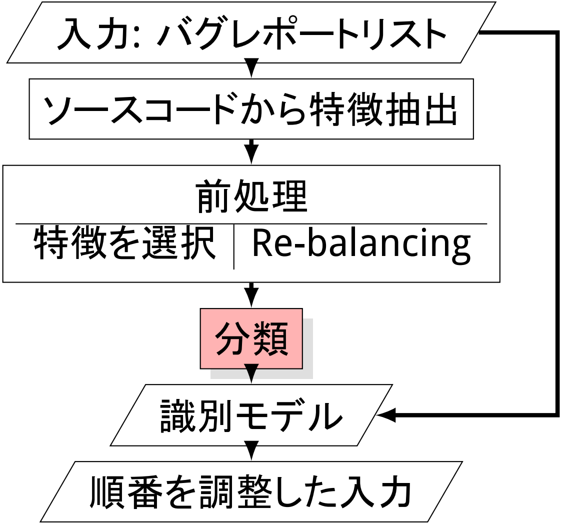
M. Kantardzic, Data mining: concepts, models, methods, and algorithms. Wiley-IEEE Press, 2011.

nearest neighbor approach
(Renieres and Reiss, 2003)

M. Renieres and S. Reiss, ``Fault localization with nearest neighbor queries,'' in Automated Software Engineering, 2003. Proceedings. 18th IEEE International Conference on. IEEE, 2003, pp. 30--39.

洗練エンジンの流れ (再掲)

Process of Refinement Engine (Again)



入力: バグレポートリスト

ソースコードから特徴抽出

前処理

特徴を選択 | Re-balancing

分類

識別モデル

順番を調整した入力

非ネスト汎化された最も近隣分類法

NNGe: Nearest Neighbor Classification with Non-Nested Generalization

非ネスト
(Wettschereck and Dietterich, 1995)

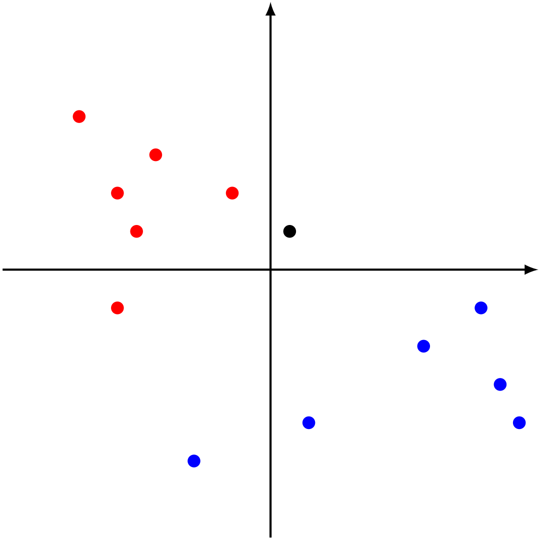
汎化された (Salzberg, 1991)
最も近隣 (Tan, 2006)
分類法 (NNGe) (Martin, 1995)

D. Wettschereck and T. Dietterich, ``An experimental comparison of the nearest-neighbor and nearest-hyperrectangle algorithms,'' Machine Learning, vol. 19, no. 1, pp. 5--27, 1995.

S. Salzberg, ``A nearest hyperrectangle learning method,'' Machine learning, vol. 6, no. 3, pp. 251--276, 1991.

S. Tan, ``An effective refinement strategy for knn text classifier,'' Expert Systems with Applications, vol. 30, no. 2, pp. 290--298, 2006.

B. Martin, ``Instance-based learning: nearest neighbour with generalisation,'' Ph.D. dissertation, University of Waikato, 1995.



非ネスト汎化された最も近隣分類法
NNGe: Nearest Neighbor Classification with Non-Nested Generalization

非ネスト
(Wettschereck and Dietterich, 1995)

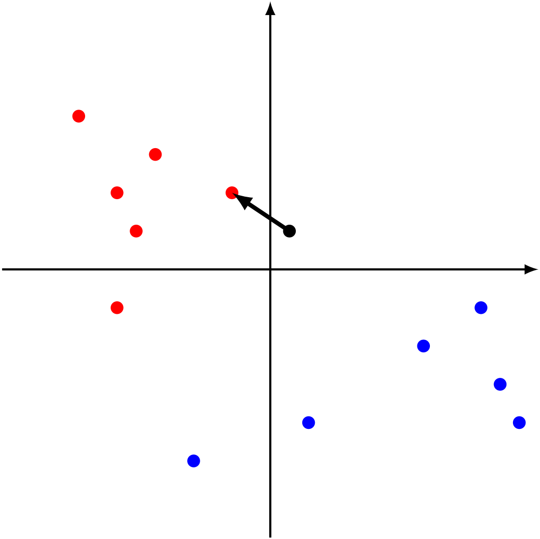
汎化された (Salzberg, 1991)
最も近隣 (Tan, 2006)
分類法 (NNGe) (Martin, 1995)

D. Wettschereck and T. Dietterich, ``An experimental comparison of the nearest-neighbor and nearest-hyperrectangle algorithms,'' Machine Learning, vol. 19, no. 1, pp. 5--27, 1995.

S. Salzberg, ``A nearest hyperrectangle learning method,'' Machine learning, vol. 6, no. 3, pp. 251--276, 1991.

S. Tan, ``An effective refinement strategy for knn text classifier,'' Expert Systems with Applications, vol. 30, no. 2, pp. 290--298, 2006.

B. Martin, ``Instance-based learning: nearest neighbour with generalisation,'' Ph.D. dissertation, University of Waikato, 1995.



非ネスト汎化された最も近隣分類法

NNGe: Nearest Neighbor Classification with Non-Nested Generalization

非ネスト
(Wettschereck and Dietterich, 1995)

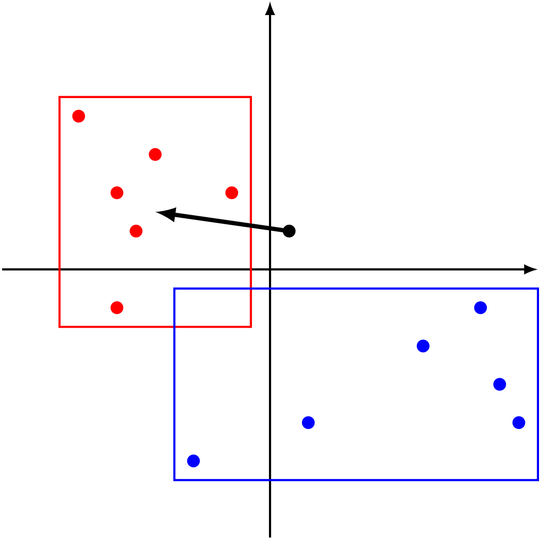
汎化された (Salzberg, 1991)
最も近隣 (Tan, 2006)
分類法 (NNGe) (Martin, 1995)

D. Wettschereck and T. Dietterich, ``An experimental comparison of the nearest-neighbor and nearest-hyperrectangle algorithms,'' Machine Learning, vol. 19, no. 1, pp. 5--27, 1995.

S. Salzberg, ``A nearest hyperrectangle learning method,'' Machine learning, vol. 6, no. 3, pp. 251--276, 1991.

S. Tan, ``An effective refinement strategy for knn text classifier,'' Expert Systems with Applications, vol. 30, no. 2, pp. 290--298, 2006.

B. Martin, ``Instance-based learning: nearest neighbour with generalisation,'' Ph.D. dissertation, University of Waikato, 1995.



非ネスト汎化された最も近隣分類法

NNGe: Nearest Neighbor Classification with Non-Nested Generalization

非ネスト
(Wettschereck and Dietterich, 1995)

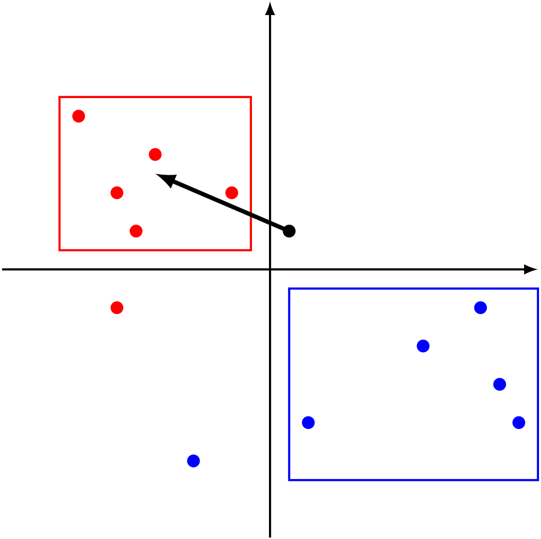
汎化された (Salzberg, 1991)
最も近隣 (Tan, 2006)
分類法 (NNGe) (Martin, 1995)

D. Wettschereck and T. Dietterich, ``An experimental comparison of the nearest-neighbor and nearest-hyperrectangle algorithms,'' Machine Learning, vol. 19, no. 1, pp. 5--27, 1995.

S. Salzberg, ``A nearest hyperrectangle learning method,'' Machine learning, vol. 6, no. 3, pp. 251--276, 1991.

S. Tan, ``An effective refinement strategy for knn text classifier,'' Expert Systems with Applications, vol. 30, no. 2, pp. 290--298, 2006.

B. Martin, ``Instance-based learning: nearest neighbour with generalisation,'' Ph.D. dissertation, University of Waikato, 1995.



分類法: 類似度の計算

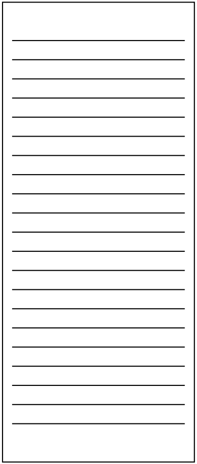
Classification: Calculation of Likelihood

$$
\begin{aligned}
\mathrm{LH}(\mathsf{dp}) &= 0.5 + \frac{\mathrm{RS}(\mathsf{dp})}{2} \\
\mathrm{RS}(\mathsf{dp}) &= \frac{\left|\sum_{\mathsf{d_T} \in \mathsf{D_T}} \mathsf{sim}(\mathsf{d_p}, \mathsf{d_T})\right|}{\mathsf{D_T}} \\
&\quad - \frac{\left|\sum_{\mathsf{d_F} \in \mathsf{D_F}} \mathsf{sim}(\mathsf{d_p}, \mathsf{d_F})\right|}{\mathsf{D_F}} \\
\mathsf{sim}(\mathsf{d_p}, \mathsf{d}) &= 1 - \mathsf{dist}(\mathsf{d_p}, \mathsf{d}) \\
\mathsf{dist}(\mathsf{d_p}, \mathsf{d}) &\in [0, 1]
\end{aligned}
$$

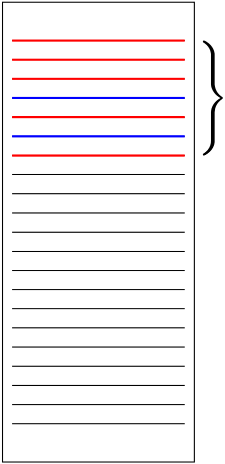
LH(dp) によって Weka の NNGe 分類法を用いて分類

具体的な洗練の振舞い

Concrete Refinement Process



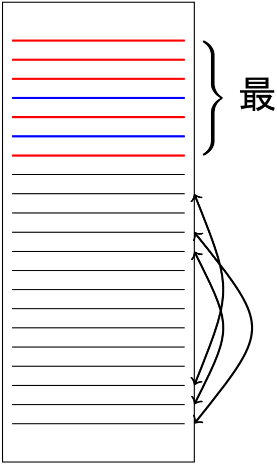
具体的な洗練の振舞い
Concrete Refinement Process



最初の k 個のレポート, 正解を含む

具体的な洗練の振舞い

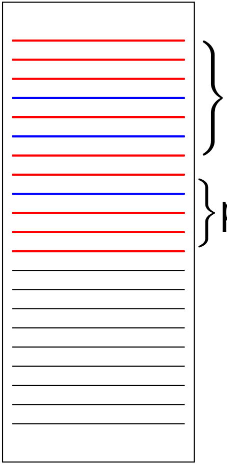
Concrete Refinement Process



最初の k 個のレポート, 正解を含む

具体的な洗練の振舞い

Concrete Refinement Process

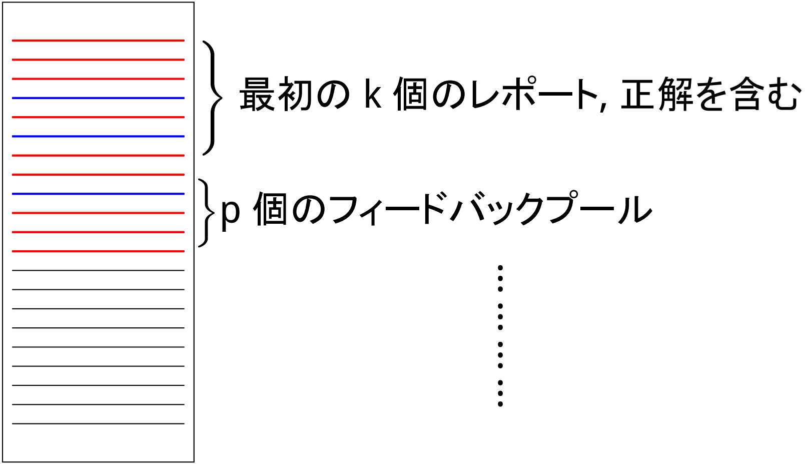


最初の k 個のレポート, 正解を含む

p 個のフィードバックプール

具体的な洗練の振舞い

Concrete Refinement Process



最初の k 個のレポート, 正解を含む

p 個のフィードバックプール

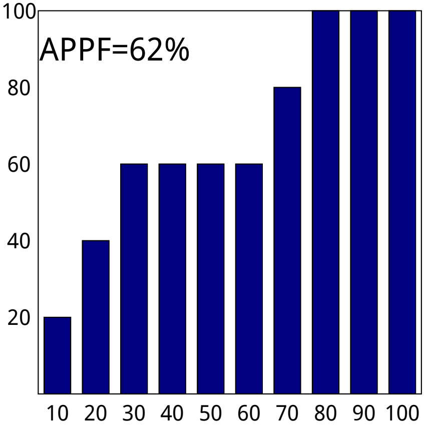
⋮

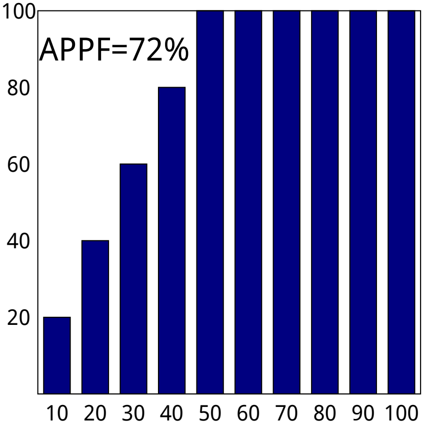
実験評価 Experiment and Evaluation

評価方法: 平均正解発見率
Evaluation Criteria: APPF: Average Percentage true Positives Found

テストケースの優先順位付け領域から借りた概念.

平均障害発見率 (APFD: Average Percentage Faults Detected).





実験の対象と設定
Settings of Empirical Evaluation

名前	バージョン	不具合数	正解数	k	p
Linux	2.6.19	>800	57	50	1
Eclipse	20070108	>400	38	50	1
ArgoUML		>50	15	10	1

実験の結果
Results of the Empirical Evaluation

名前	APPF⤴	Top-5 順番の調整
Linux	11%	$694 \mapsto 18, 672 \mapsto 64, 760 \mapsto 131, 770 \mapsto 179{,}792 \mapsto 206$
Eclipse	87%	$373 \mapsto 4, 348 \mapsto 11, 394 \mapsto 29, 388 \mapsto 43, 370 \mapsto 49$
ArgoUML	86%	$40 \mapsto 12, 35 \mapsto 15, 34 \mapsto 11, 29 \mapsto 9, 23 \mapsto 8$

Linux の結果は低いのは, この手法では改名に関するバグが取れない.

Top-3 情報利得

Top-3 Information Gain

Top	特徴	情報利得
	Linux kernel	
1	extern_ definition^P	0.015941
2	extern_ definition _1^P	0.015941
3	program##extern_ definitions^P	0.015941
	Eclipse	
1	BOOL_OR_TK^P	0.01898
2	conditional_ or_ expression ## conditional_ or_ expression ^{P}	0.01898
3	BOOL_OR_TK^B	0.01898
	ArgoUML	
1	local_ variable_ declaration_ statement ^{B}	0.145772
2	variable_ initializer ^{B}	0.145772
3	block_ statement ## local_ variable_ declaration_ statement ^{B}	0.145772

順番を調整した例: Linux 696 ↦ 18

Example of Re-ordering: Linux 694 ↦ 18

drivers/net/wireless/bcm43xx/

bcm43xx_sysfs.c

```
347 struct bcm43xx_private *bcm = dev_to_bcm(dev);
348 mutex_lock(&(bcm)—>mutex);
349 switch (bcm43xx_current_phy(bcm)—>type) {
350     case BCM43xx_PHYTYPE_A:
351     ...
```

```
362     default:
363         assert(0);
364 }
365 mutex_unlock(&(bcm)—>mutex);
```

drivers/net/wireless/bcm43xx/bcm43xx_wx.c

```
615 struct bcm43xx_private *bcm = bcm43xx_priv(net_dev);
616 ...
```

```
618 mutex_lock(&bcm—>mutex);
619 mode = bcm43xx_current_radio(bcm)—>interfmode;
620 mutex_unlock(&bcm—>mutex);
621 switch (mode) {
622     case BCM43xx_RADIO_INTERFMODE_NONE:
623     ...
```

```
632     default:
633         assert(0);
```

順番を調整した例: Eclipse 373 ↦ 4

Example of Re-ordering: Eclipse 373 ↦ 4

debug/internal/core/LaunchConfiguration.java

```
if ( file  != null) {
    // validate edit
    if ( file .isReadOnly()) {
        IStatus  status  = ResourcesPlugin.
         getWorkspace().validateEdit(new IFile[] { file },
           null);
        if  (! status.isOK()) {
            throw new CoreException(status);
```

debug/internal/core/

LaunchConfigurationWorkingCopy.java

```
// validate edit
if  ( file .isReadOnly()) {
    IStatus  status  = ResourcesPlugin.getWorkspace().
         validateEdit(new IFile[] { file },  null);
    if  (! status.isOK())  {
        throw new CoreException(status);
```

順番を調整した例: ArgoUML 40 ↦ 12

Example of Re-ordering: ArgoUML 40 ↦ 12

argouml/uml/diagram/UMLMutableGraphSupport.java

```
if (edge instanceof CommentEdge) {
    ...
} else if (Model.getFacade().isARelationship(edge)
          || Model.getFacade().isATransition(edge)
          || Model.getFacade().isAAssociationEnd(edge))
           {
    return Model.getUmlHelper().getDestination(edge)
           ;
} else if (Model.getFacade().isALink(edge)) {
    ...
}
```

argouml/uml/diagram/UMLMutableGraphSupport.java

```
if (edge instanceof CommentEdge) {
    ...
} else if (Model.getFacade().isAAssociation(edge)) {
    List conns = new
    ArrayList(Model.getFacade().getConnections(edge)
           );
    return conns.get(1);
} else if (Model.getFacade().isARelationship(edge)
          || Model.getFacade().isATransition(edge)
          || Model.getFacade().isAAssociationEnd(edge))
           {
    return Model.getUmlHelper().getDestination(edge)
           ;
} else if (Model.getFacade().isALink(edge)) {
    ...
}
```

考察, 結論及び今後の課題

妥当性の考察 (と質問)
Threats to Validity (and Questions)

- APPF は APFD から借りた概念
  - 他に使われていない
- 本当に Bug であるかとうかは検証していない
  - 人間で目で見て判断した
  - 後のバージョンに修正したかを検証していない (質問)
- 3つのシステムしかない
  - C と Java しかない, 他の言語に対応できるか?
  - 古いバージョンを使った原因は? (質問)
    - Linux と Eclipse の実験の一部は 2007 に発表した研究. 今回追加したのは ArgoUML.
    - Long-term-service ではないから, バグが多いかも

結論と今後の課題
Conclusion and Future Work

結論

- クローンバグレポートの動的な洗練法を提案した
- この方法について, 評価方法 APPF を採用された
- Linux, Eclipse, ArgoUML の3つのシステムで実験を行った.
    - 結果はそれぞれ 11%, 87%, 86%

今後の課題

- 他のソフトウェアに対応してみたい
- バグレポート以外の不具合レポートに対応してみたい

私の研究との比較
Comparing with My Research

	この手法	私の研究
着目点	バグを不具合クローンから見つけ	人によって判断の差異
クローン	構文木 Type-3	トークン列 Type-2
学習特徴	木の Node，Edge の数と割合	トークンの型の tf-idf
分類法	非ネスト汎化された最も近隣 NNGe	Cosine-Similarity の重み付き平均数
評価方法	平均正解発見率 APPF	訓練集合 -正確率の図

特徴抽出: 五つの特徴の定義

Feature Extraction: Defination of 5 features

定義: 基本的な特徴 (Basic Features)

$(\mathsf{t}, |\mathsf{CS}|)$, 若し $\mathsf{CS} = \{\mathsf{c} \in \mathsf{CG} | \mathsf{c}$にtタイプが存在する$\} \wedge |\mathsf{CS}| > 0$

定義: 対の特徴 (Pair Features)

$(\mathsf{t}_1, \mathsf{t}_2, |\mathsf{CS}|)$, 若し $\mathsf{CS} = \{\mathsf{c} \in \mathsf{CG} | \exists_{\mathsf{n}_1, \mathsf{n}_2 \in \mathsf{c}}\ \mathsf{n}_1$ と n_2 は連結されている $\wedge\ \mathsf{n}_1$ のタイプは $\mathsf{t}_1 \wedge\ \mathsf{n}_2$ のタイプは $\mathsf{t}_2\} \wedge |\mathsf{CS}| > 0$

定義: 基本的な特徴の割合 (Proportional Features—Basic)

$(\mathsf{t}, \frac{|\mathsf{CS}|}{|\mathsf{CG}|})$, 若し $\mathsf{CS} = \{\mathsf{c} \in \mathsf{CG} | \mathsf{c}$にtタイプが存在する$\} \wedge |\mathsf{CS}| > 0$

定義: 対の特徴の割合 (Proportional Features—Pair)

$(\mathsf{t}_1, \mathsf{t}_2, \frac{|\mathsf{CS}|}{|\mathsf{CG}|})$, 若し $\mathsf{CS} = \{\mathsf{c} \in \mathsf{CG} | \exists_{\mathsf{n}_1, \mathsf{n}_2 \in \mathsf{c}}\ \mathsf{n}_1$ と n_2 は連結されている $\wedge\ \mathsf{n}_1$ のタイプは $\mathsf{t}_1 \wedge\ \mathsf{n}_2$ のタイプは $\mathsf{t}_2\} \wedge |\mathsf{CS}| > 0$

定義: 他の特徴

クローンの数 $|\mathsf{CG}|$, クローンの平均サイズ $\frac{\sum_{\mathsf{c} \in \mathsf{CG}} |\mathsf{C}|}{|\mathsf{CG}|}$

参考文献 I
Reference

E. Juergens, F. Deissenboeck, B. Hummel, and S. Wagner, ``Do code clones matter?'' in Proceedings of the 31st International Conference on Software Engineering. IEEE Computer Society, 2009, pp. 485--495.

L. Jiang, Z. Su, and E. Chiu, ``Context-based detection of clone-related bugs,'' in ESEC/FSE, vol. 2007, 2007.

M. Gabel, J. Yang, Y. Yu, M. Goldszmidt, and Z. Su, ``Scalable and systematic detection of buggy inconsistencies in source code,'' in ACM Sigplan Notices, vol. 45, no. 10. ACM, 2010, pp. 175--190.

L. Jiang, G. Misherghi, Z. Su, and S. Glondu, ``Deckard: Scalable and accurate tree-based detection of code clones,'' in Proceedings of the 29th international conference on Software Engineering. IEEE Computer Society, 2007, pp. 96--105.

G. Holmes, A. Donkin, and I. Witten, ``Weka: A machine learning workbench,'' in Intelligent Information Systems, 1994. Proceedings of the 1994 Second Australian and New Zealand Conference on. Ieee, 1994, pp. 357--361.

M. Kantardzic, Data mining: concepts, models, methods, and algorithms. Wiley-IEEE Press, 2011.

参考文献 II

Reference

M. Renieres and S. Reiss, ``Fault localization with nearest neighbor queries,'' in Automated Software Engineering, 2003. Proceedings. 18th IEEE International Conference on. IEEE, 2003, pp. 30--39.

D. Wettschereck and T. Dietterich, ``An experimental comparison of the nearest-neighbor and nearest-hyperrectangle algorithms,'' Machine Learning, vol. 19, no. 1, pp. 5--27, 1995.

S. Salzberg, ``A nearest hyperrectangle learning method,'' Machine learning, vol. 6, no. 3, pp. 251--276, 1991.

S. Tan, ``An effective refinement strategy for knn text classifier,'' Expert Systems with Applications, vol. 30, no. 2, pp. 290--298, 2006.

B. Martin, ``Instance-based learning: nearest neighbour with generalisation,'' Ph.D. dissertation, University of Waikato, 1995.