

出典 Publication

出典 Publication

背景 Introduction

背景 Introduction

既存研究 Related Researches

正解の例 Example of True Positive

誤検出の例 Example of False Positive

不具合があるクローンの誤検出率

コードクローンの四つの象限

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

動的洗練法 Dynamic Refinement

従来のクローンレポートの静的洗練法

クローンレポートの動的洗練法

動的洗練法の流れ

洗練エンジン Refinement Engine

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

特徴抽出: 構文木を構築

特徴抽出: 五つの特徴

特徴抽出: 五つの特徴の例

前処理: 特徴選択

前処理: Data Re-balancing

非ネスト汎化された最も近隣分類法

分類法: 類似度の計算

具体的な洗練の振舞い

実験評価 Experiment and Evaluation

評価方法: 平均正解発見率 APPF

実験の対象と設定 Settings of Empirical Evaluation

実験の結果 Results of the Empirical Evaluation

Top-3 情報利得 Top-3 Information Gain

順番を調整した例 Example of Re-ordering

考察, 結論及び今後の課題

妥当性の考察 Threats to Validity

結論と今後の課題 Conclusion and Future Work

出典 Publication

出典 Publication

出典 Publication

背景 Introduction

背景 Introduction

既存研究 Related Researches

正解の例 Example of True Positive

誤検出の例 Example of False Positive

不具合があるクローンの誤検出率

コードクローンの四つの象限

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

動的洗練法 Dynamic Refinement

従来のクローンレポートの静的洗練法

クローンレポートの動的洗練法

動的洗練法の流れ

洗練エンジン Refinement Engine

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

特徴抽出: 構文木を構築

特徴抽出: 五つの特徴

特徴抽出: 五つの特徴の例

前処理: 特徴選択

前処理: Data Re-balancing

非ネスト汎化された最も近隣分類法

分類法: 類似度の計算

具体的な洗練の振舞い

実験評価 Experiment and Evaluation

評価方法: 平均正解発見率 APPF

実験の対象と設定 Settings of Empirical Evaluation

実験の結果 Results of the Empirical Evaluation

Top-3 情報利得 Top-3 Information Gain

順番を調整した例 Example of Re-ordering

考察, 結論及び今後の課題

妥当性の考察 Threats to Validity

結論と今後の課題 Conclusion and Future Work

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

出典 Publication

出典 Publication

背景 Introduction

背景 Introduction

既存研究 Related Researches

正解の例 Example of True Positive

誤検出の例 Example of False Positive

不具合があるクローンの誤検出率

コードクローンの四つの象限

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

動的洗練法 Dynamic Refinement

従来のクローンレポートの静的洗練法

クローンレポートの動的洗練法

動的洗練法の流れ

洗練エンジン Refinement Engine

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

特徴抽出: 構文木を構築

特徴抽出: 五つの特徴

特徴抽出: 五つの特徴の例

前処理: 特徴選択

前処理: Data Re-balancing

非ネスト汎化された最も近隣分類法

分類法: 類似度の計算

具体的な洗練の振舞い

実験評価 Experiment and Evaluation

評価方法: 平均正解発見率 APPF

実験の対象と設定 Settings of Empirical Evaluation

実験の結果 Results of the Empirical Evaluation

Top-3 情報利得 Top-3 Information Gain

順番を調整した例 Example of Re-ordering

考察, 結論及び今後の課題

妥当性の考察 Threats to Validity

結論と今後の課題 Conclusion and Future Work

背景 Introduction

出典 Publication

出典 Publication

背景 Introduction

背景 Introduction

既存研究 Related Researches

正解の例 Example of True Positive

誤検出の例 Example of False Positive

不具合があるクローンの誤検出率

コードクローンの四つの象限

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

動的洗練法 Dynamic Refinement

従来のクローンレポートの静的洗練法

クローンレポートの動的洗練法

動的洗練法の流れ

洗練エンジン Refinement Engine

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

特徴抽出: 構文木を構築

特徴抽出: 五つの特徴

特徴抽出: 五つの特徴の例

前処理: 特徴選択

前処理: Data Re-balancing

非ネスト汎化された最も近隣分類法

分類法: 類似度の計算

具体的な洗練の振舞い

実験評価 Experiment and Evaluation

評価方法: 平均正解発見率 APPF

実験の対象と設定 Settings of Empirical Evaluation

実験の結果 Results of the Empirical Evaluation

Top-3 情報利得 Top-3 Information Gain

順番を調整した例 Example of Re-ordering

考察, 結論及び今後の課題

妥当性の考察 Threats to Validity

結論と今後の課題 Conclusion and Future Work

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)



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

```
219 struct dentry * dentry = sd->s_dentry;
220
221 if (dentry) {
222     /* the following parts are detected as clones */
223     spin_lock(&dcache_lock);
224     spin_lock(&dentry->d_lock);
225     if (!(d_unhashed(dentry) && dentry->d_inode)) {
226         dget_locked(dentry);
227         __d_drop(dentry);
228         spin_unlock(&dentry->d_lock);
229         spin_unlock(&dcache_lock);
230         .....
```

drivers/infiniband/hw/ipath/ipath_fs.c

```
456 struct dentry *tmp;
457
458 tmp = lookup_one_len(name, parent, strlen(name));
459
460 spin_lock(&dcache_lock);
461 spin_lock(&tmp->d_lock);
462 if (!(d_unhashed(tmp) && tmp->d_inode)) {
463     dget_locked(tmp);
464     __d_drop(tmp);
465     spin_unlock(&tmp->d_lock);
466     spin_unlock(&dcache_lock);
467     .....
```

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

```
219 struct dentry * dentry = sd->s_dentry;
220
221 if (dentry) {
222     /* the following parts are detected as clones */
223     spin_lock(&dcache_lock);
224     spin_lock(&dentry->d_lock);
225     if (!(d_unhashed(dentry) && dentry->d_inode)) {
226         dget_locked(dentry);
227         __d_drop(dentry);
228         spin_unlock(&dentry->d_lock);
229         spin_unlock(&dcache_lock);
230         .....
```

drivers/infiniband/hw/ipath/ipath_fs.c

```
456 struct dentry *tmp;
457
458 tmp = lookup_one_len(name, parent, strlen(name));
459
460 spin_lock(&dcache_lock);
461 spin_lock(&tmp->d_lock);
462 if (!(d_unhashed(tmp) && tmp->d_inode)) {
463     dget_locked(tmp);
464     __d_drop(tmp);
465     spin_unlock(&tmp->d_lock);
466     spin_unlock(&dcache_lock);
467     .....
```

Type-2 クローン

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

```
219 struct dentry * dentry = sd->s_dentry;
220
221 if (dentry) {
222     /* the following parts are detected as clones */
223     spin_lock(&dcache_lock);
224     spin_lock(&dentry->d_lock);
225     if (!(d_unhashed(dentry) && dentry->d_inode)) {
226         dget_locked(dentry);
227         __d_drop(dentry);
228         spin_unlock(&dentry->d_lock);
229         spin_unlock(&dcache_lock);
230         .....
```

drivers/infiniband/hw/ipath/ipath_fs.c

```
456 struct dentry *tmp;
457
458 tmp = lookup_one_len(name, parent, strlen(name));
459
460 spin_lock(&dcache_lock);
461 spin_lock(&tmp->d_lock);
462 if (!(d_unhashed(tmp) && tmp->d_inode)) {
463     dget_locked(tmp);
464     __d_drop(tmp);
465     spin_unlock(&tmp->d_lock);
466     spin_unlock(&dcache_lock);
467     .....
```

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;
```

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;
```

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;
```

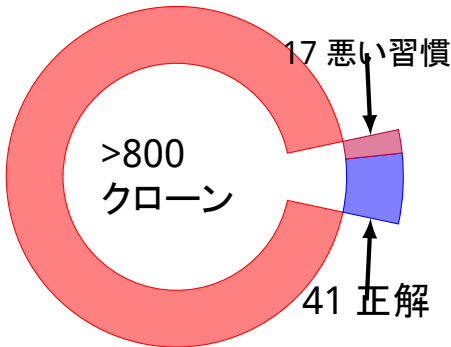
drivers/hwmon/g1520sm.c

```
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;
```

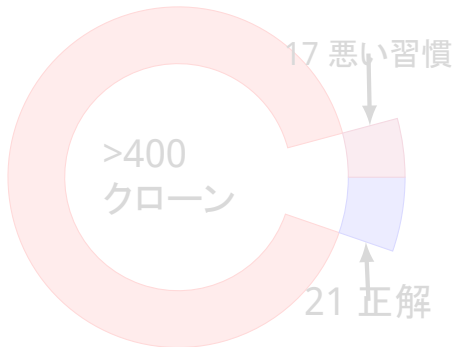
不具合があるクローンの誤検出率

False Positives in Anomaly Clones

Linux Kernel



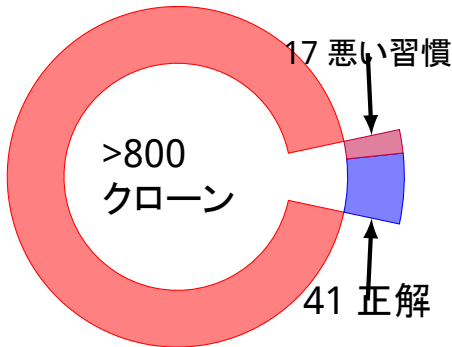
Eclipse



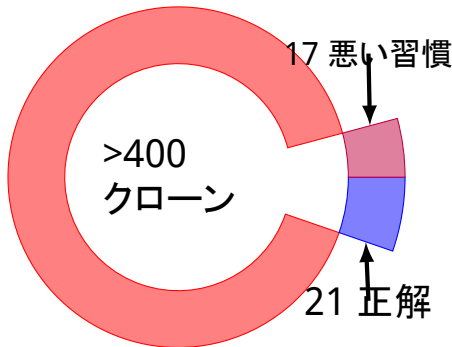
不具合があるクローンの誤検出率

False Positives in Anomaly Clones

Linux Kernel



Eclipse



不具合があるクローンの誤検出率 II

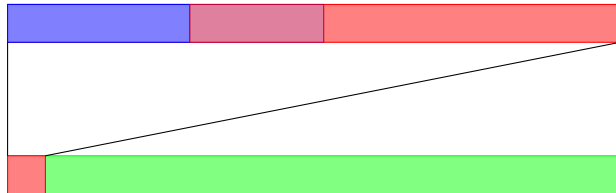
False Positives in Anomaly Clones II

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.

コードクロンの四つの象限

4 Quadrants of Code Clone Group

大阪大学大学院 CS 専攻 楊 嘉晨

出典 Publication

出典 Publication

背景 Introduction

背景 Introduction

既存研究 Related Researches

正解の例 Example of True Positive

誤検出の例 Example of False Positive

不具合があるクローンの誤検出率

コードクロンの四つの象限

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

動的洗練法 Dynamic Refinement

従来のクローンレポートの静的洗練法

クローンレポートの動的洗練法

動的洗練法の流れ

洗練エンジン Refinement Engine

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

特徴抽出: 構文木を構築

特徴抽出: 五つの特徴

特徴抽出: 五つの特徴の例

前处理: 特徵選取

前处理: Data Re-balancing

非ネスト汎化された最も近隣分類法

分類法: 類似度の計算
最良法を決定する毎に

具体的な洗練の振舞い

実験評価 Experiment and Evaluation

評価方法: 平均正解発見率 APPF

実験の対象と設定 Settings of Empirical Evaluation

実験の結果 Results of the Empirical Evaluation

Top-3 情報利得 Top-3 Information Gain
選定文脈数: 1, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50

順序を調整した例 Example of Re-ordering

考察, 結論及び今後の課題

妥当性の考察 Threats to Validity

結論と今後の課題 Conclusion and Future Work

		一貫性	
		Inconsistent	Consistant
可變性	嚴格 (Rigid)	✓	
	柔軟 (Flexible)	✓	

出典 Publication

出典 Publication

背景 Introduction

背景 Introduction

既存研究 Related Researches

正解の例 Example of True Positive

誤検出の例 Example of False Positive

不具合があるクローンの誤検出率

コードクローンの四つの象限

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

動的洗練法 Dynamic Refinement

従来のクローンレポートの静的洗練法

クローンレポートの動的洗練法

動的洗練法の流れ

洗練エンジン Refinement Engine

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

特徴抽出: 構文木を構築

特徴抽出: 五つの特徴

特徴抽出: 五つの特徴の例

前処理: 特徴選択

前処理: Data Re-balancing

非ネスト汎化された最も近隣分類法

分類法: 類似度の計算

具体的な洗練の振舞い

実験評価 Experiment and Evaluation

評価方法: 平均正解発見率 APPF

実験の対象と設定 Settings of Empirical Evaluation

実験の結果 Results of the Empirical Evaluation

Top-3 情報利得 Top-3 Information Gain

順番を調整した例 Example of Re-ordering

考察, 結論及び今後の課題

妥当性の考察 Threats to Validity

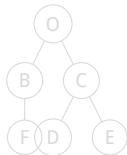
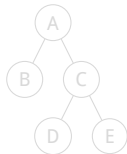
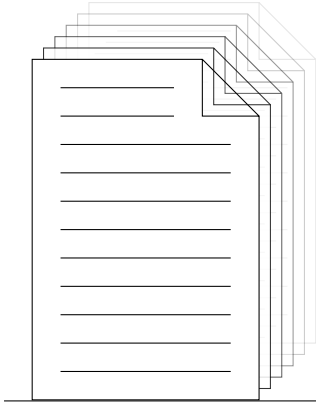
結論と今後の課題 Conclusion and Future Work

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)

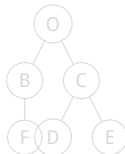
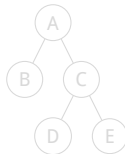
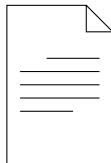
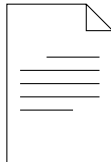
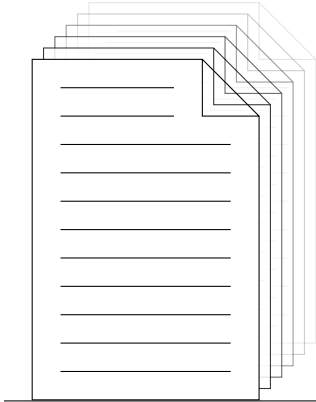


L. Jiang, G. Misherghi, Z. Su, and S. Glondu, ``Deckard: Scalable and accurate tree-based detection of code clones," in Proceedings of the 29th international conference on Software Engineering. IEEE Computer Society, 2007, pp. 96--105.

クローンに基づく不具合検出

Clone-based Anomaly Detection

Deckard(Jiang et al., 2007b)

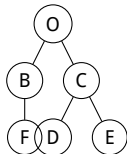
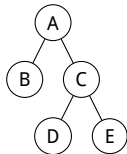
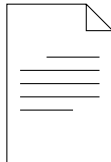
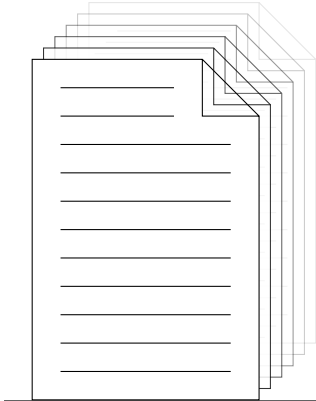


L. Jiang, G. Misherghi, Z. Su, and S. Glondu, ``Deckard: Scalable and accurate tree-based detection of code clones," in Proceedings of the 29th international conference on Software Engineering. IEEE Computer Society, 2007, pp. 96--105.

クローンに基づく不具合検出

Clone-based Anomaly Detection

Deckard(Jiang et al., 2007b)

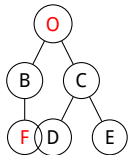
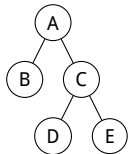
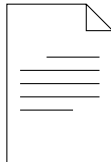
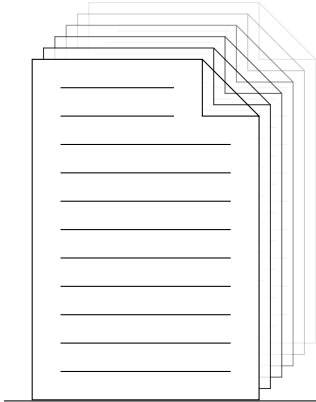


L. Jiang, G. Misherghi, Z. Su, and S. Glondu, "Deckard: Scalable and accurate tree-based detection of code clones," in Proceedings of the 29th international conference on Software Engineering. IEEE Computer Society, 2007, pp. 96--105.

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.

出典 Publication

出典 Publication

背景 Introduction

背景 Introduction

既存研究 Related Researches

正解の例 Example of True Positive

誤検出の例 Example of False Positive

不具合があるクローンの誤検出率

コードクローンの四つの象限

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

動的洗練法 Dynamic Refinement

従来のクローンレポートの静的洗練法

クローンレポートの動的洗練法

動的洗練法の流れ

洗練エンジン Refinement Engine

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

特徴抽出: 構文木を構築

特徴抽出: 五つの特徴

特徴抽出: 五つの特徴の例

前処理: 特徴選択

前処理: Data Re-balancing

非ネスト汎化された最も近隣分類法

分類法: 類似度の計算

具体的な洗練の振舞い

実験評価 Experiment and Evaluation

評価方法: 平均正解発見率 APPF

実験の対象と設定 Settings of Empirical Evaluation

実験の結果 Results of the Empirical Evaluation

Top-3 情報利得 Top-3 Information Gain

順番を調整した例 Example of Re-ordering

考察, 結論及び今後の課題

妥当性の考察 Threats to Validity

結論と今後の課題 Conclusion and Future Work

動的洗練法 Dynamic Refinement

出典 Publication

出典 Publication

背景 Introduction

背景 Introduction

既存研究 Related Researches

正解の例 Example of True Positive

誤検出の例 Example of False Positive

不具合があるクローンの誤検出率

コードクローンの四つの象限

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

動的洗練法 Dynamic Refinement

従来のクローンレポートの静的洗練法

クローンレポートの動的洗練法

動的洗練法の流れ

洗練エンジン Refinement Engine

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

特徴抽出: 構文木を構築

特徴抽出: 五つの特徴

特徴抽出: 五つの特徴の例

前処理: 特徴選択

前処理: Data Re-balancing

非ネスト汎化された最も近隣分類法

分類法: 類似度の計算

具体的な洗練の振舞い

実験評価 Experiment and Evaluation

評価方法: 平均正解発見率 APPF

実験の対象と設定 Settings of Empirical Evaluation

実験の結果 Results of the Empirical Evaluation

Top-3 情報利得 Top-3 Information Gain

順番を調整した例 Example of Re-ordering

考察, 結論及び今後の課題

妥当性の考察 Threats to Validity

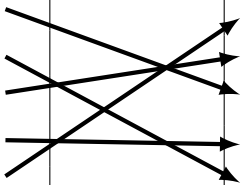
結論と今後の課題 Conclusion and Future Work

クローンレポートの動的洗練法

Dynamic Refinement of Clone Report

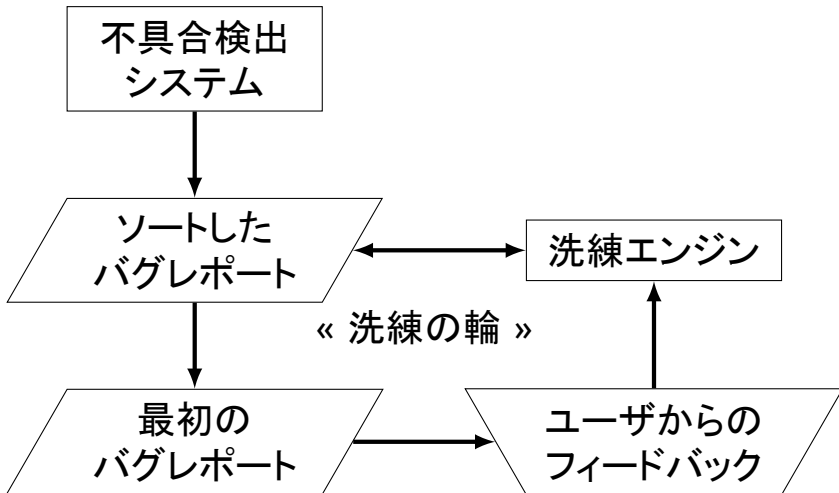
ID	内容	Bug?
1	AAA	
2	BBB	
3	CCC	
4	DDD	
5	EEE	
6	FFF	
7	III	
...

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



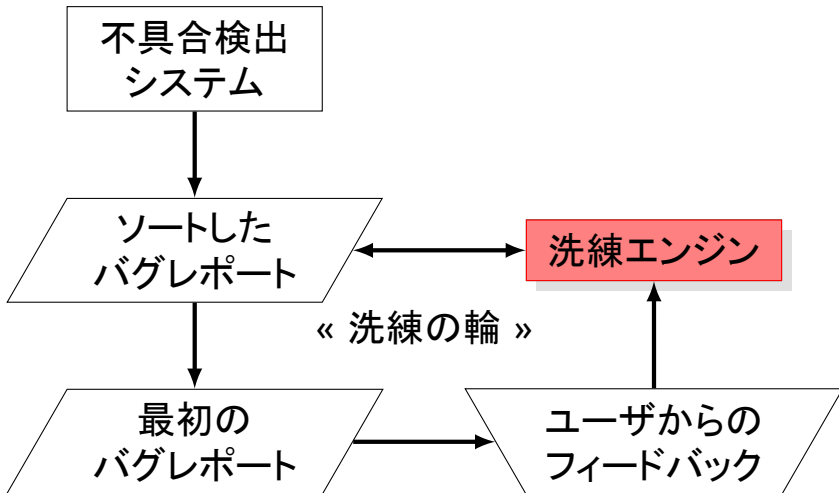
動的洗練法の流れ

Active Refinement Process



動的洗練法の流れ

Active Refinement Process



出典 Publication

出典 Publication

背景 Introduction

背景 Introduction

既存研究 Related Researches

正解の例 Example of True Positive

誤検出の例 Example of False Positive

不具合があるクローンの誤検出率

コードクローンの四つの象限

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

動的洗練法 Dynamic Refinement

従来のクローンレポートの静的洗練法

クローンレポートの動的洗練法

動的洗練法の流れ

洗練エンジン Refinement Engine

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

特徴抽出: 構文木を構築

特徴抽出: 五つの特徴

特徴抽出: 五つの特徴の例

前処理: 特徴選択

前処理: Data Re-balancing

非ネスト汎化された最も近隣分類法

分類法: 類似度の計算

具体的な洗練の振舞い

実験評価 Experiment and Evaluation

評価方法: 平均正解発見率 APF

実験の対象と設定 Settings of Empirical Evaluation

実験の結果 Results of the Empirical Evaluation

Top-3 情報利得 Top-3 Information Gain

順番を調整した例 Example of Re-ordering

考察, 結論及び今後の課題

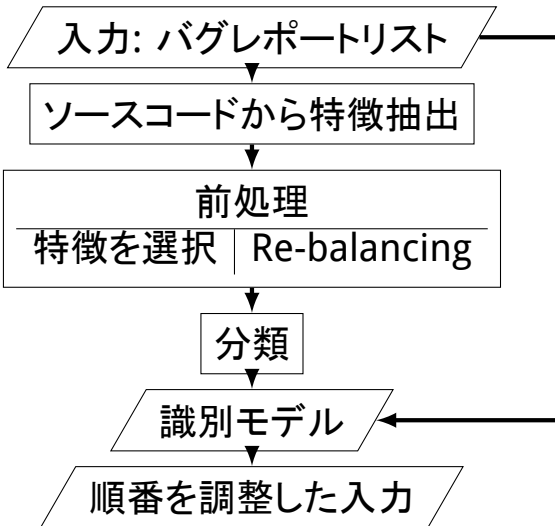
妥当性の考察 Threats to Validity

結論と今後の課題 Conclusion and Future Work

洗練エンジン Refinement Engine

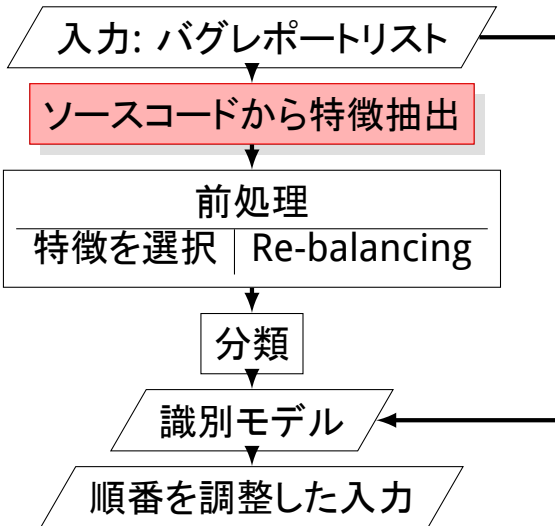
洗練エンジンの流れ

Process of Refinement Engine



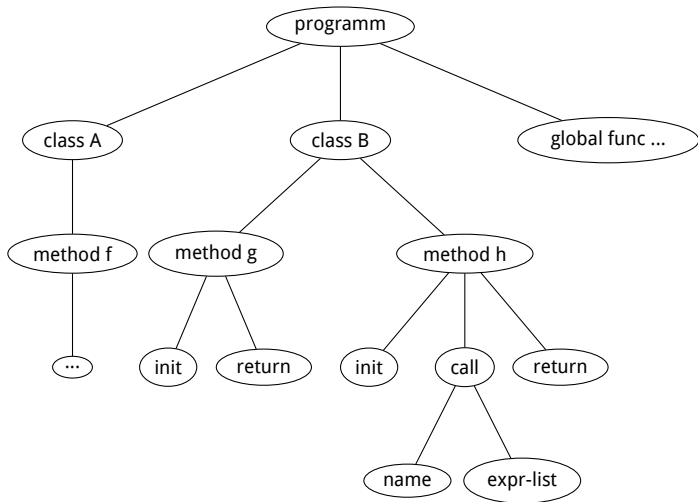
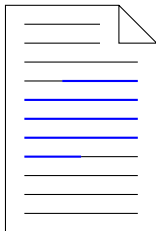
洗練エンジンの流れ

Process of Refinement Engine



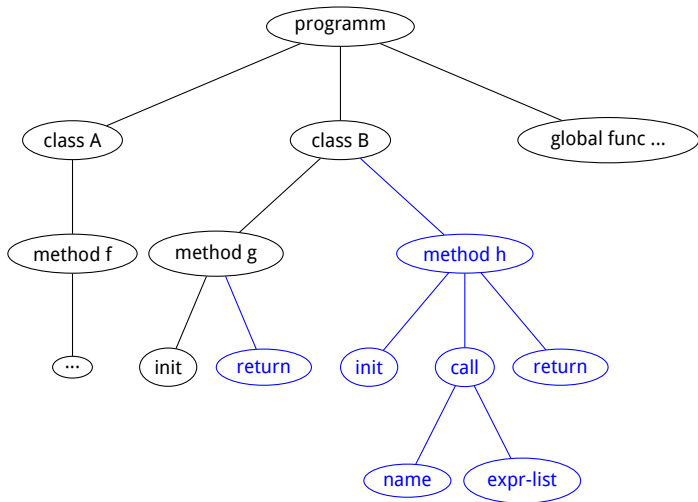
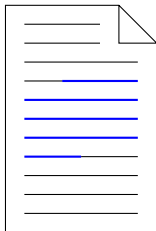
特徴抽出: 構文木を構築

Feature Extraction: Tree Construction



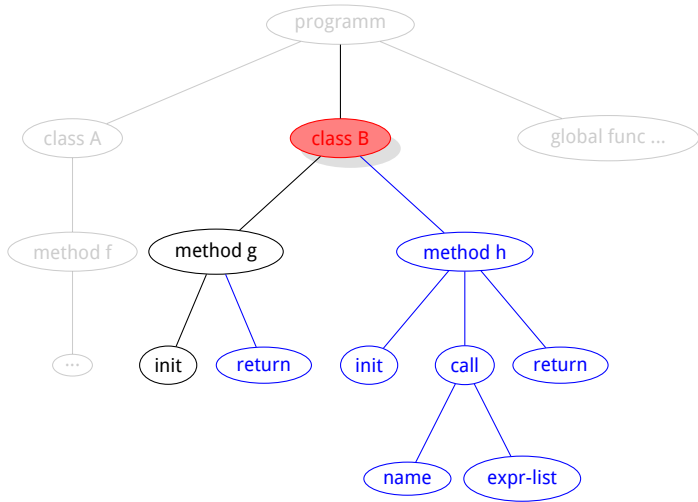
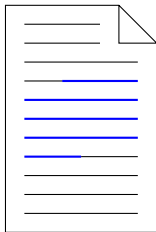
特徴抽出: 構文木を構築

Feature Extraction: Tree Construction



特徴抽出: 構文木を構築

Feature Extraction: Tree Construction



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

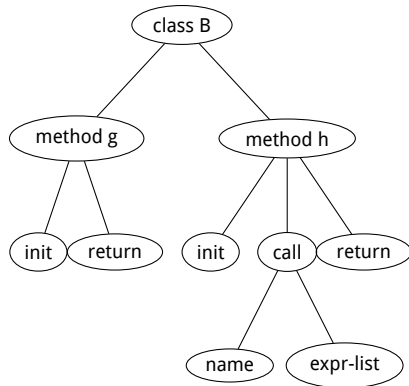
Proportional Features—Basic

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

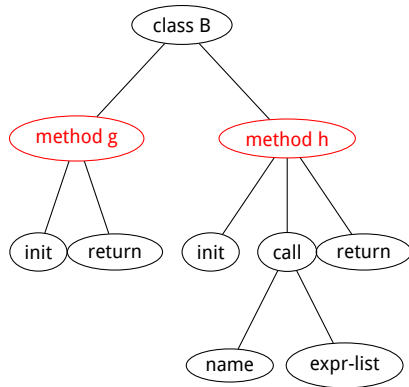
Proportional Features—Basic

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

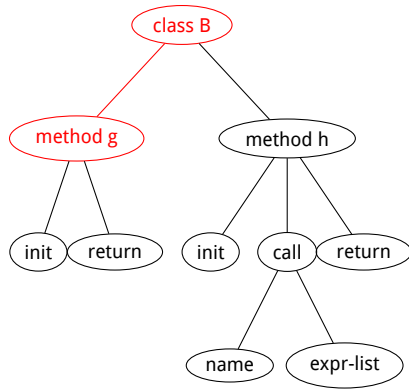
Proportional Features—Basic

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

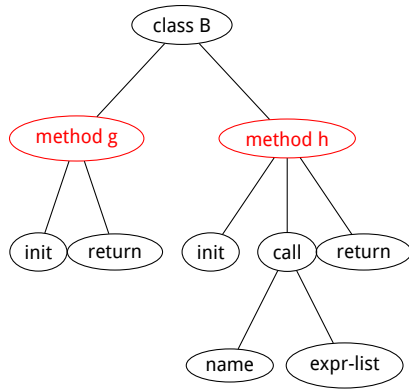
Proportional Features—Basic

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

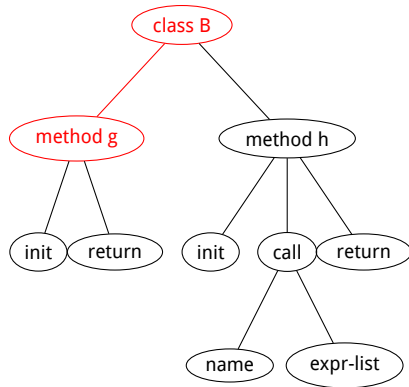
Proportional Features—Basic

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

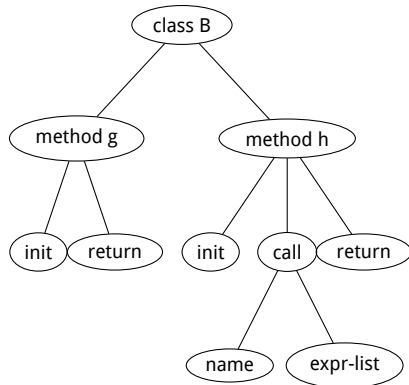
Proportional Features—Basic

対の特徴の割合

Proportional Features—Pair

他の特徴

Other Features

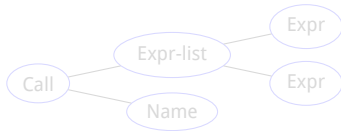


クローンの数 $|CG|$, クローンの
の平均サイズ $\frac{\sum_{c \in CG} |c|}{|CG|}$

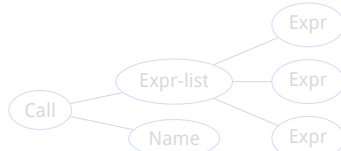
特徴抽出: 五つの特徴の例

Feature Extraction: Example of 5 features

1 decode_satr3(p, &args->attrs)



1 decode_filename(p, &args->tname, &args->tlen)

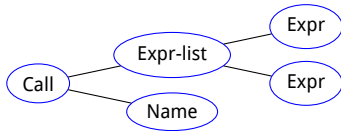


基本		対		基本割合		対割合		他	
タイプ	数	タイプ	数	タイプ	割合	タイプ	割合	タイプ	値
Call	2	Call/Name	2	Call	100%	Call/Name	100%	Num	2
Name	2	Call/ Expr-list	2	Name	100%	Call/ Expr-list	100%	Avg	5.5
Expr-list	2	Expr-list/ Expr	2	Expr-list	100%	Expr-list/ Expr	100%		
Expr	2			Expr	100%				

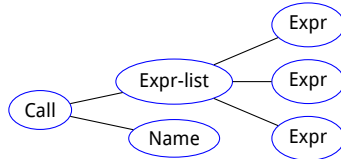
特徴抽出: 五つの特徴の例

Feature Extraction: Example of 5 features

1 decode_satr3(p, &args->attrs)



1 decode_filename(p, &args->tname, &args->tlen)

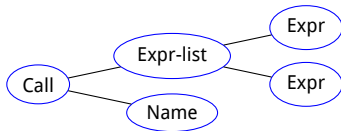


基本		対		基本割合		対割合		他	
タイプ	数	タイプ	数	タイプ	割合	タイプ	割合	タイプ	値
Call	2	Call/Name	2	Call	100%	Call/Name	100%	Num	2
Name	2	Call/ Expr-list	2	Name	100%	Call/ Expr-list	100%	Avg	5.5
Expr-list	2	Expr-list/ Expr	2	Expr-list	100%	Expr-list/ Expr	100%		
Expr	2			Expr	100%				

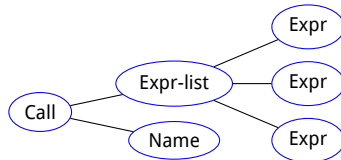
特徴抽出: 五つの特徴の例

Feature Extraction: Example of 5 features

1 decode_satr3(p, &args->attrs)



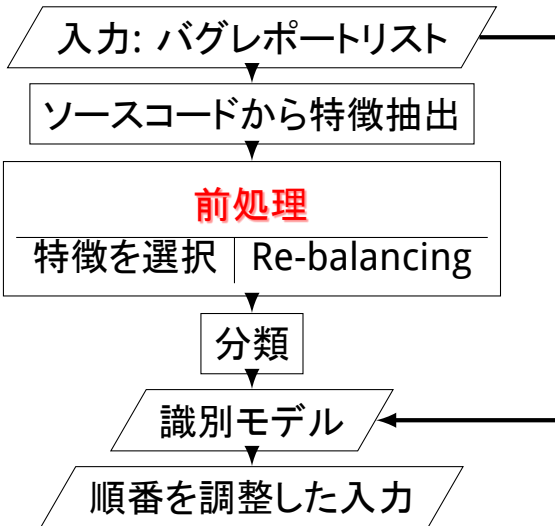
1 decode_filename(p, &args->tname, &args->tlen)



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)



- 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:

$$\text{IG}(\mathbf{c}|\mathbf{f}) = H(\mathbf{c}) - H(\mathbf{c}|\mathbf{f}) \quad (1)$$

$$H(c) = - \sum_{c_i \in \{\pm ve\}} P(c_i) \log P(c_i) \quad (2)$$

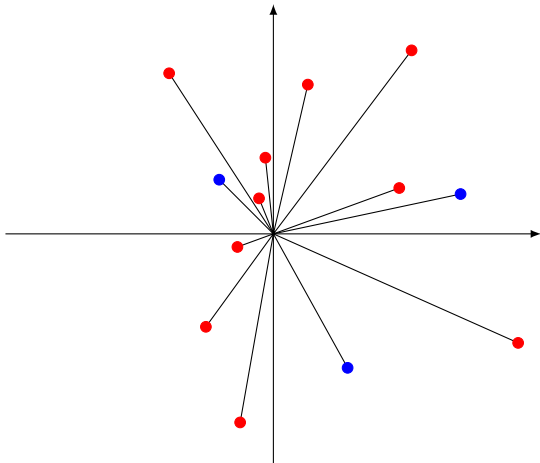
$$H(c|f) = - \sum P(f) \sum_{c_i \in \{\pm ve\}} P(c_i|f) \log P(c_i|f) \quad (3)$$

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)

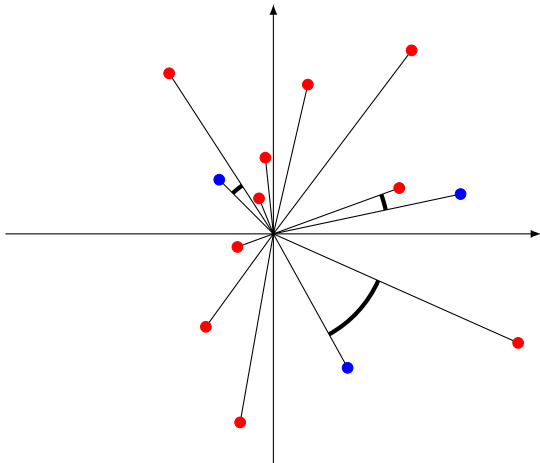
M. Kantardzic, Data mining: concepts, models, methods, and algorithms. Wiley-IEEE Press, 2011.

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)

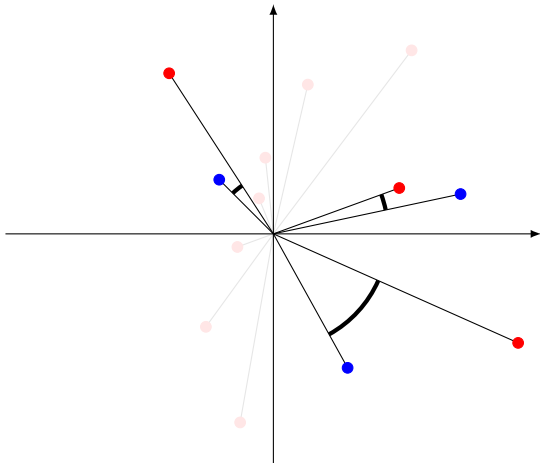
M. Kantardzic, Data mining: concepts, models, methods, and algorithms. Wiley-IEEE Press, 2011.

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)

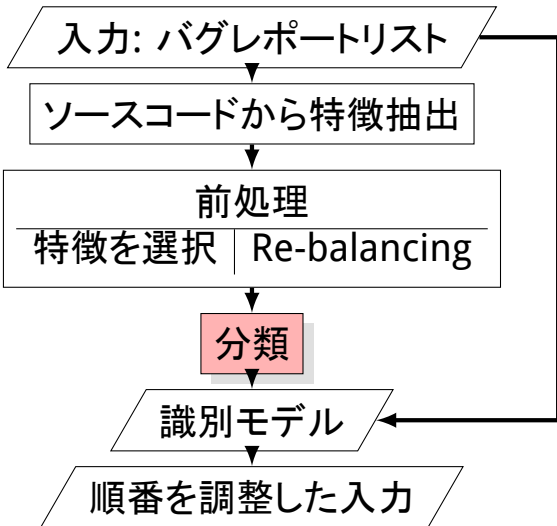
M. Kantardzic, Data mining: concepts, models, methods, and algorithms. Wiley-IEEE Press, 2011.

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)



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)

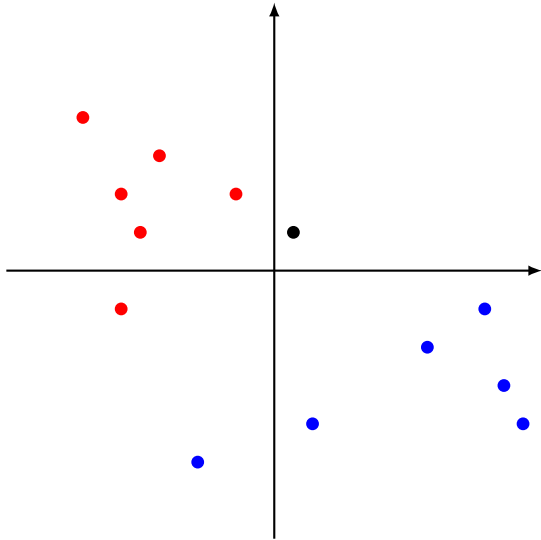
分類法 (NNGe) (Martin, 1995)

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)

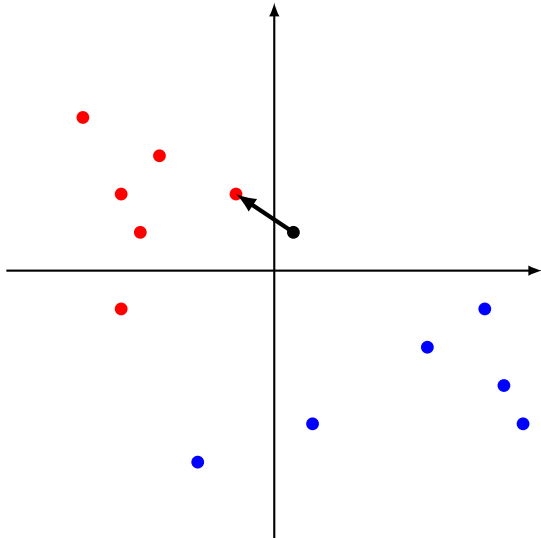
分類法 (NNGe) (Martin, 1995)

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)

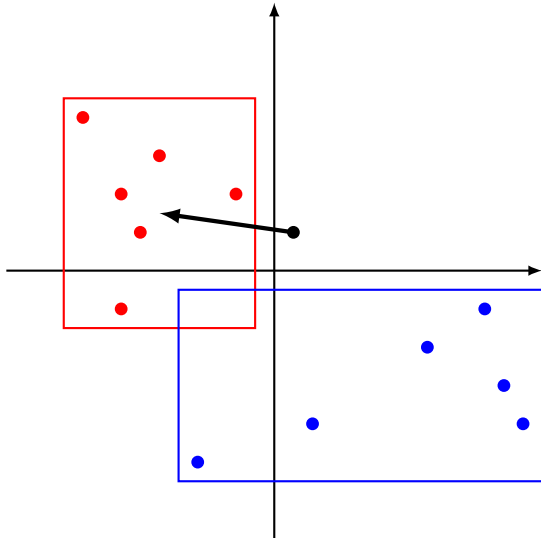
分類法 (NNGe) (Martin, 1995)

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)

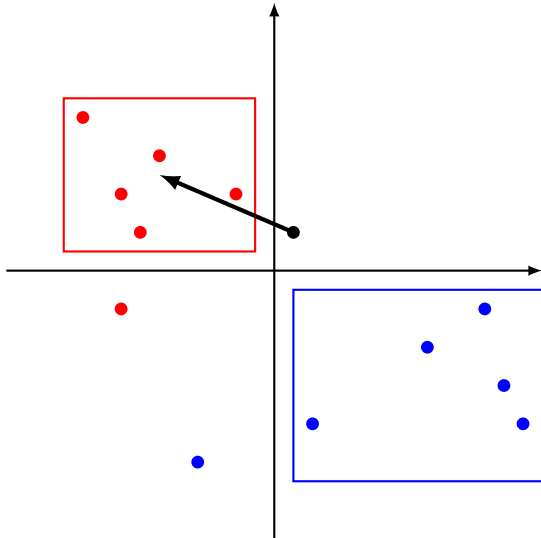
分類法 (NNGe) (Martin, 1995)

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.



出典 Publication

背景 Introduction

背景 Introduction

既存研究 Related Researches

正解の例 Example of True Positive

誤検出の例 Example of False Positive

不具合があるクローンの誤検出率

コードクロンの四つの象限

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

動的洗練法 Dynamic Refinement

従来のクローンレポートの静的洗練法

クローンレポートの動的洗練法

動的洗練法の流れ

洗練エンジン Refinement Engine

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

特徴抽出: 構文木を構築

特徴抽出: 五つの特徴

特徴抽出: 五つの特徴の例

前处理: 特徵選択

前处理: Data Re-balancing

非ネスト汎化された最も近隣分類法
の計算 類似度の計算

分類法: 類似度の計算
 学習法: 教師-学習

具体的な洗練の振舞い

実験評価 Experiment and Evaluation

評價方法：平均正解発見率 APF

実験の対象と設定 Settings of Empirical Evaluation

実験の結果 Results of the Empirical Evaluation

緊要な情報: 4. 例: Examples of Re-ordering

順序を調整した例 Example of Re-ordering

考察、結論及び今後の課題

結論と今後の課題 Conclusion and

結論と今後の課題 Conclusion and Future Work

[illegible]

出典 Publication

出典 Publication

背景 Introduction

背景 Introduction

既存研究 Related Researches

正解の例 Example of True Positive

誤検出の例 Example of False Positive

不具合があるクローンの誤検出率

コードクローンの四つの象限

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

動的洗練法 Dynamic Refinement

従来のクローンレポートの静的洗練法

クローンレポートの動的洗練法

動的洗練法の流れ

洗練エンジン Refinement Engine

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

特徴抽出: 構文木を構築

特徴抽出: 五つの特徴

特徴抽出: 五つの特徴の例

前処理: 特徴選択

前処理: Data Re-balancing

非ネスト汎化された最も近隣分類法

分類法: 類似度の計算

具体的な洗練の振舞い

実験評価 Experiment and Evaluation

評価方法: 平均正解発見率 APPF

実験の対象と設定 Settings of Empirical Evaluation

実験の結果 Results of the Empirical Evaluation

Top-3 情報利得 Top-3 Information Gain

順番を調整した例 Example of Re-ordering

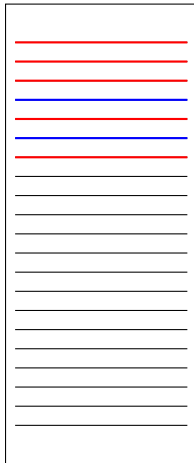
考察, 結論及び今後の課題

妥当性の考察 Threats to Validity

結論と今後の課題 Conclusion and Future Work

具体的な洗練の振舞い

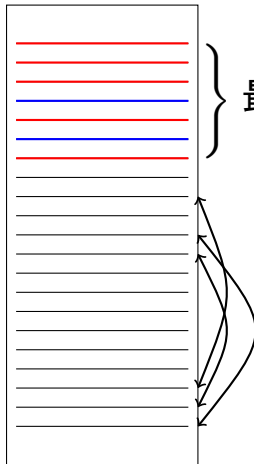
Concrete Refinement Process



} 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

出典 Publication

出典 Publication

背景 Introduction

背景 Introduction

既存研究 Related Researches

正解の例 Example of True Positive

誤検出の例 Example of False Positive

不具合があるクローンの誤検出率

コードクローンの四つの象限

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

動的洗練法 Dynamic Refinement

従来のクローンレポートの静的洗練法

クローンレポートの動的洗練法

動的洗練法の流れ

洗練エンジン Refinement Engine

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

特徴抽出: 構文木を構築

特徴抽出: 五つの特徴

特徴抽出: 五つの特徴の例

前処理: 特徴選択

前処理: Data Re-balancing

非ネスト汎化された最も近隣分類法

分類法: 類似度の計算

具体的な洗練の振舞い

実験評価 Experiment and Evaluation

評価方法: 平均正解発見率 APPF

実験の対象と設定 Settings of Empirical Evaluation

実験の結果 Results of the Empirical Evaluation

Top-3 情報利得 Top-3 Information Gain

順番を調整した例 Example of Re-ordering

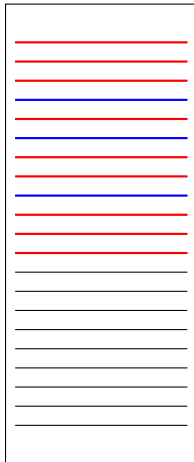
考察, 結論及び今後の課題

妥当性の考察 Threats to Validity

結論と今後の課題 Conclusion and Future Work

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

出典 Publication

出典 Publication

背景 Introduction

背景 Introduction

既存研究 Related Researches

正解の例 Example of True Positive

誤検出の例 Example of False Positive

不具合があるクローンの誤検出率

コードクローンの四つの象限

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

動的洗練法 Dynamic Refinement

従来のクローンレポートの静的洗練法

クローンレポートの動的洗練法

動的洗練法の流れ

洗練エンジン Refinement Engine

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

特徴抽出: 構文木を構築

特徴抽出: 五つの特徴

特徴抽出: 五つの特徴の例

前処理: 特徴選択

前処理: Data Re-balancing

非ネスト汎化された最も近隣分類法

分類法: 類似度の計算

具体的な洗練の振舞い

実験評価 Experiment and Evaluation

評価方法: 平均正解発見率 APF

実験の対象と設定 Settings of Empirical Evaluation

実験の結果 Results of the Empirical Evaluation

Top-3 情報利得 Top-3 Information Gain

順番を調整した例 Example of Re-ordering

考察, 結論及び今後の課題

妥当性の考察 Threats to Validity

結論と今後の課題 Conclusion and Future Work

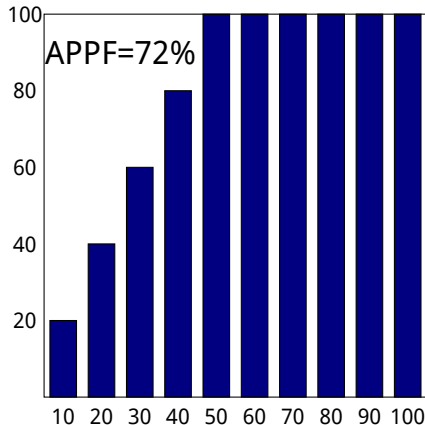
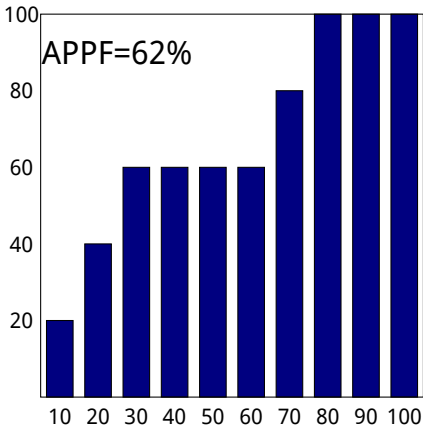
実験評価 Experiment and Evaluation

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).



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

出典 Publication

出典 Publication

背景 Introduction

背景 Introduction

既存研究 Related Researches

正解の例 Example of True Positive

誤検出の例 Example of False Positive

不具合があるクローンの誤検出率

コードクローンの四つの象限

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

動的洗練法 Dynamic Refinement

従来のクローンレポートの静的洗練法

クローンレポートの動的洗練法

動的洗練法の流れ

洗練エンジン Refinement Engine

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

特徴抽出: 構文木を構築

特徴抽出: 五つの特徴

特徴抽出: 五つの特徴の例

前処理: 特徴選択

前処理: Data Re-balancing

非ネスト汎化された最も近隣分類法

分類法: 類似度の計算

具体的な洗練の振舞い

実験評価 Experiment and Evaluation

評価方法: 平均正解発見率 APPF

実験の対象と設定 Settings of Empirical Evaluation

実験の結果 Results of the Empirical Evaluation

Top-3 情報利得 Top-3 Information Gain

順番を調整した例 Example of Re-ordering

考察, 結論及び今後の課題

妥当性の考察 Threats to Validity

結論と今後の課題 Conclusion and Future Work

実験の結果

Results of the Empirical Evaluation

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

出典 Publication

出典 Publication

背景 Introduction

背景 Introduction

既存研究 Related Researches

正解の例 Example of True Positive

誤検出の例 Example of False Positive

不具合があるクローンの誤検出率

コードクローンの四つの象限

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

動的洗練法 Dynamic Refinement

従来のクローンレポートの静的洗練法

クローンレポートの動的洗練法

動的洗練法の流れ

洗練エンジン Refinement Engine

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

特徴抽出: 構文木を構築

特徴抽出: 五つの特徴

特徴抽出: 五つの特徴の例

前処理: 特徴選択

前処理: Data Re-balancing

非ネスト汎化された最も近隣分類法

分類法: 類似度の計算

具体的な洗練の振舞い

実験評価 Experiment and Evaluation

評価方法: 平均正解発見率 APPF

実験の対象と設定 Settings of Empirical Evaluation

実験の結果 Results of the Empirical Evaluation

Top-3 情報利得 Top-3 Information Gain

順番を調整した例 Example of Re-ordering

考察, 結論及び今後の課題

妥当性の考察 Threats to Validity

結論と今後の課題 Conclusion and Future Work

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_ex- pression ^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 \mapsto 18

Example of Re-ordering: Linux 694 \mapsto 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 \mapsto 4

Example of Re-ordering: Eclipse 373 \mapsto 4

debug/internal/core/LaunchConfiguration.java

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

debug/internal/core/

LaunchConfigurationWorkingCopy.java

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

順番を調整した例: ArgoUML 40 \mapsto 12

Example of Re-ordering: ArgoUML 40 \mapsto 12

argouml/uml/diagram/UMLMutableGraphSupport.java

```
331 if (edge instanceof CommentEdge) {
332     ...
333 } else if (Model.getFacade().isARelationship(edge)
334           || Model.getFacade().isATransition(edge)
335           || Model.getFacade().isAAssociationEnd(edge))
336     return Model.getUmlHelper().getDestination(edge);
337 } else if (Model.getFacade().isALink(edge)) {
338     ...
339 }
```

argouml/uml/diagram/UMLMutableGraphSupport.java

```
360 if (edge instanceof CommentEdge) {
361     ...
362 } else if (Model.getFacade().isAAssociation(edge)) {
363     List conns = new
364         ArrayList(Model.getFacade().getConnections(edge)
365                 );
366     return conns.get(1);
367 } else if (Model.getFacade().isARelationship(edge)
368           || Model.getFacade().isATransition(edge)
369           || Model.getFacade().isAAssociationEnd(edge))
370     return Model.getUmlHelper().getDestination(edge);
371 } else if (Model.getFacade().isALink(edge)) {
372     ...
373 }
```

出典 Publication

出典 Publication

背景 Introduction

背景 Introduction

既存研究 Related Researches

正解の例 Example of True Positive

誤検出の例 Example of False Positive

不具合があるクローンの誤検出率

コードクローンの四つの象限

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

動的洗練法 Dynamic Refinement

従来のクローンレポートの静的洗練法

クローンレポートの動的洗練法

動的洗練法の流れ

洗練エンジン Refinement Engine

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

特徴抽出: 構文木を構築

特徴抽出: 五つの特徴

特徴抽出: 五つの特徴の例

前処理: 特徴選択

前処理: Data Re-balancing

非ネスト汎化された最も近隣分類法

分類法: 類似度の計算

具体的な洗練の振舞い

実験評価 Experiment and Evaluation

評価方法: 平均正解発見率 APF

実験の対象と設定 Settings of Empirical Evaluation

実験の結果 Results of the Empirical Evaluation

Top-3 情報利得 Top-3 Information Gain

順番を調整した例 Example of Re-ordering

考察, 結論及び今後の課題

妥当性の考察 Threats to Validity

結論と今後の課題 Conclusion and Future Work

考察, 結論及び今後の 課題

- 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%

- 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
クローン学習特徴	構文木 Type-3 木の Node, Edge の数と割合	トークン列 Type-2 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: Definition of 5 features

定義: 基本的な特徴 (Basic Features)

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

定義: 対の特徴 (Pair Features)

$(t_1, t_2, |CS|)$, 若し $CS = \{c \in CG \mid \exists n_1, n_2 \in c \text{ } n_1 \text{ と } n_2 \text{ は連結されている} \wedge n_1 \text{ のタイプは } t_1 \wedge n_2 \text{ のタイプは } t_2\} \wedge |CS| > 0$

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

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

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

$(t_1, t_2, \frac{|CS|}{|CG|})$, 若し $CS = \{c \in CG \mid \exists n_1, n_2 \in c \text{ } n_1 \text{ と } n_2 \text{ は連結されている} \wedge n_1 \text{ のタイプは } t_1 \wedge n_2 \text{ のタイプは } t_2\} \wedge |CS| > 0$

定義: 他の特徴

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

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.
- 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.

参考文献 II

Reference

- 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.