SMURF: A SVM-based Incremental Anti-pattern Detection Approach

Abdou Maiga, Nasir Ali, Neelesh Bhattacharya, Aminata Sabané, Yann-Gaël Guéhéneuc, Esma Aïmeur

DIRO, Université de Montréal
DGIGL, École Polytechnique, Québec, Canada
E-mails: {abdou.maiga, nasir.ali, n.bhattacharya, a.sabane}@polymtl.ca,
{guehene, aimeur}@iro.umontreal.ca

WCRE 2012
October 18, 2012

Outline

Introduction

Motivation

- Anti-patterns: **"poor" solutions** to recurring design and implementation problems.
- **Impact** program comprehension, software evolution and maintenance activities [1].
- Important to **detect them early** in software development process, to reduce the maintenance costs

Related Work

Smell/Anti-pattern Detection

Many researchers studied anti-patterns detection.

- Alikacem et al. [2] used **meta-model** for representing the source code and **fuzzy thresholds**;
- Langelier et al. [3] used a **visual approach**;
- Marinescu [4] used quantifiable **expression of rules**;
- Sahraoui et al. [5] used **search-based techniques**;
- Moha et al. [6] proposed an approach based on a **set of rules that describes** each anti-pattern;
- Khomh et al. [7] present BDTEX a **probabilistic** anti-patterns detection approach.

Related Work

Limitations

The works carried out so far suffered from some limitations:

- they have **limited** precision and recall (if reported at all);
- had not been **adopted by practitioners** yet;
- cannot be applied on **subsets of systems**;
- required **extensive knowledge** of anti-patterns;
- are not **iterative** and **incremental**.

To the best of our knowledge, no previous approach used SVM for anti-pattern detection.

Related Work

Contributions: SMURF

We propose

- SMURF approach to detect anti-patterns using SVM and practitioners' feedback;
- Use of precision and recall to **compare SMURF** to **DETEX** [6] and **BDTEX** [7];
- the accuracy of SMURF is greater than that of DETEX and BDTEX on subsets and whole system;
- SMURF can be applied in both intra-system and inter-system configurations;
- SMURF accuracy improves when using practitioners' feedback.

Our Approach: SMURF

SMURF - Steps

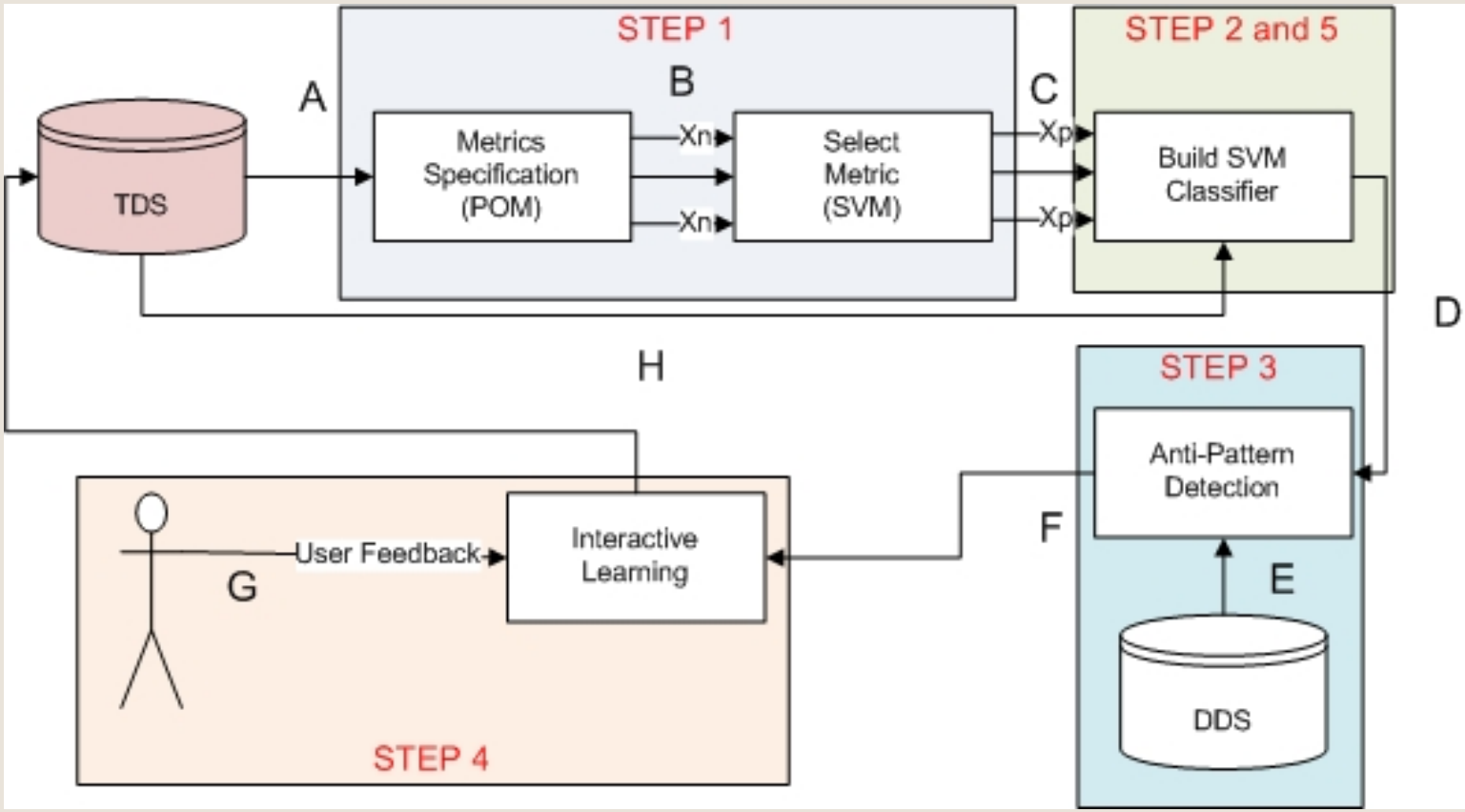

Figure : SMURF process overview

Empirical Study

Empirical Study

- *Goal*: **validate** that SMURF can **overcome** previous approaches' limitations.
- *Quality focus*: **accuracy** of SMURF, in terms of **precision** and **recall**.
- *Perspective*: researchers and practitioners interested in **verifying** if SMURF can be **effective** in detecting various kinds of anti-patterns, and in **overcoming** the previous limitations.

Empirical Study

Research Questions

- **RQ1** and **RQ2**: How does the accuracy of **SMURF** compare with that of **DETEX** and **BDTEX**, in terms of precision and recall?
- **RQ3**: How does the accuracy of SMURF change when trained/applied on the same system and trained/applied on different systems, in terms of precision and recall?
- **RQ4**: How does the accuracy of SMURF, with **feedback**, compare with that of SMURF without feedback, in terms of precision and recall?

Empirical Study

Objects and Subjects

- Use of 3 open source systems: ArgoUML 0.19.8, Azureus 2.3.0.6 and Xerces 2.7.0
- Use of 4 anti-patterns:Blob, Functional Decomposition (FD), Spaghetti Code (SC),Swiss Army Knife (SAK)

These 3 systems and 4 anti-patterns because **well known**, **commonly studied** in **previous** work and for **comparison**.

Study Results

Subsets of System: $RQ1_1$

Table : Precision of SMURF vs. DETEX in subsets (%)

		ArgoUML	Azureus	Xerces
Blob	DETEX	0.00	0.00	0.00
	SMURF	97.09	97.32	95.51
FD	DETEX	0.00	0.00	0.00
	SMURF	70.68	72.01	66.93
SC	DETEX	0.00	0.00	0.00
	SMURF	85.00	88.00	86.00
SAK	DETEX	10.00	10.00	0.00
	SMURF	75.46	84.54	80.76

Study Results

Subsets of System: $RQ1_1$

Table : Recall of SMURF vs. DETEX in subsets (%)

		ArgoUML	Azureus	Xerces
Blob	DETEX	0.00	0.00	0.00
	SMURF	84.09	91.33	95.29
FD	DETEX	0.00	0.00	0.00
	SMURF	57.50	84.28	70.00
SC	DETEX	0.00	0.00	0.00
	SMURF	71.00	89.00	86.00
SAK	DETEX	0.00	0.00	0.00
	SMURF	77.14	85.71	75.50

Study Results

Complete System: $RQ1_2$

Table : Total recovered occurrences of BLOB by DETEX and SMURF on whole system

	DETEX	**SMURF**
ArgoUML	25	40
Azureus	38	48
Xerces	39	55
Total	102	**143**

We answer RQ1: "How does the accuracy of SMURF compare with that of DETEX, in terms of precision and recall?" as follows:

- on subsets of systems, SMURF **dramatically outperforms** DETEX.
- on entire systems, SMURF **detects more occurrences** of Blob than DETEX.

Study Results

RQ2

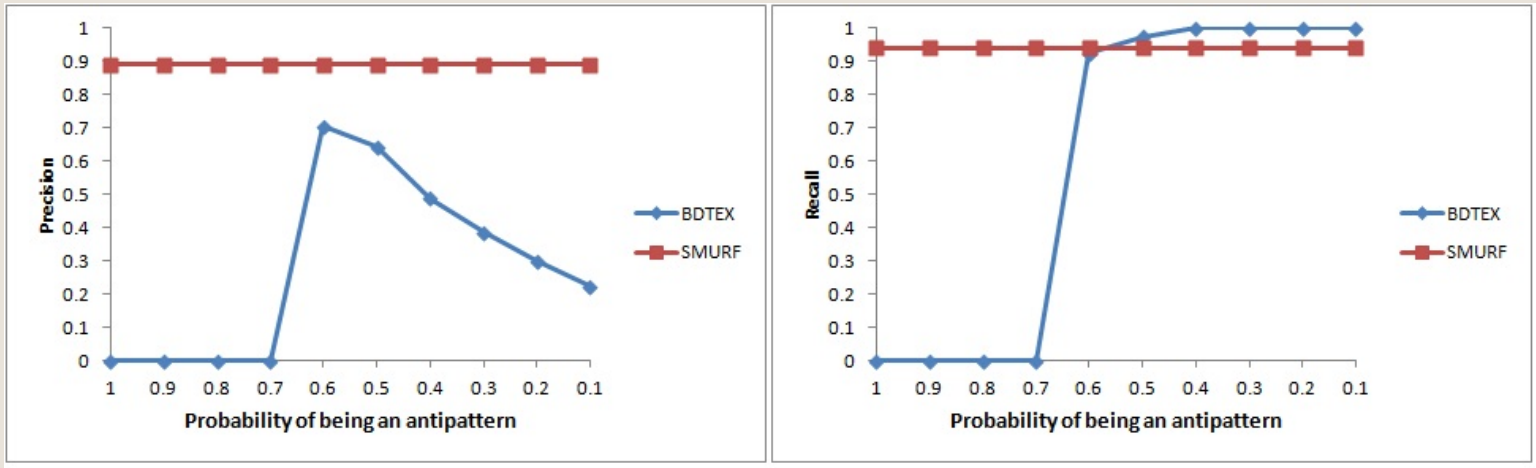

Figure : Trends in the increase of precision and recall when decreasing the probabilty of being an antipattern for Blob and Xerces

Study Results

RQ2

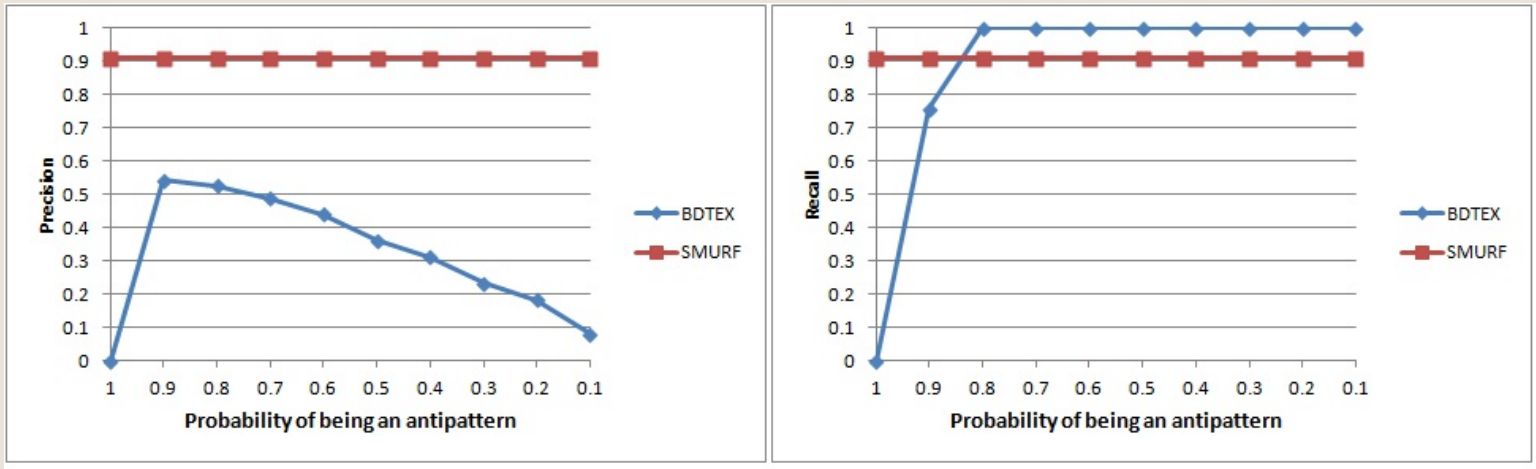

Figure : Trends in the increase of precision and recall when decreasing the probabilty of being an antipattern for Spaguetti Code and Xerces

Thus, we answer RQ2 as follows: SMURF has a better precision and recall than BDTEX.

Study Results

RQ3

Table : Precision of SMURF in inter-systems configuration

	ArgoUML (%)	Azureus (%)	Xerces (%)
Blob	92.00	96.00	89.00
FD	57.00	62.00	36.00
SC	77.00	74.00	91.00
SAK	56.00	73.00	90.00

Table : Recall of SMURF in inter-systems configuration

	ArgoUML (%)	Azureus (%)	Xerces (%)
Blob	62.00	48.00	94.00
FD	40.00	100.00	20.00
SC	96.00	88.00	91.00
SAK	68.00	84.00	56.00

Study Results

RQ3

Thus, we answer RQ3 as follows: SMURF has a better precision and recall than DETEX. Even in the inter-system configuration, its precision and recall are acceptable in the most of cases excepted for the functional decomposition in the programs ArgoUML (the recall is 40%) and Xerces (the precision is 36% and the recall 20%).

Study Results

RQ4

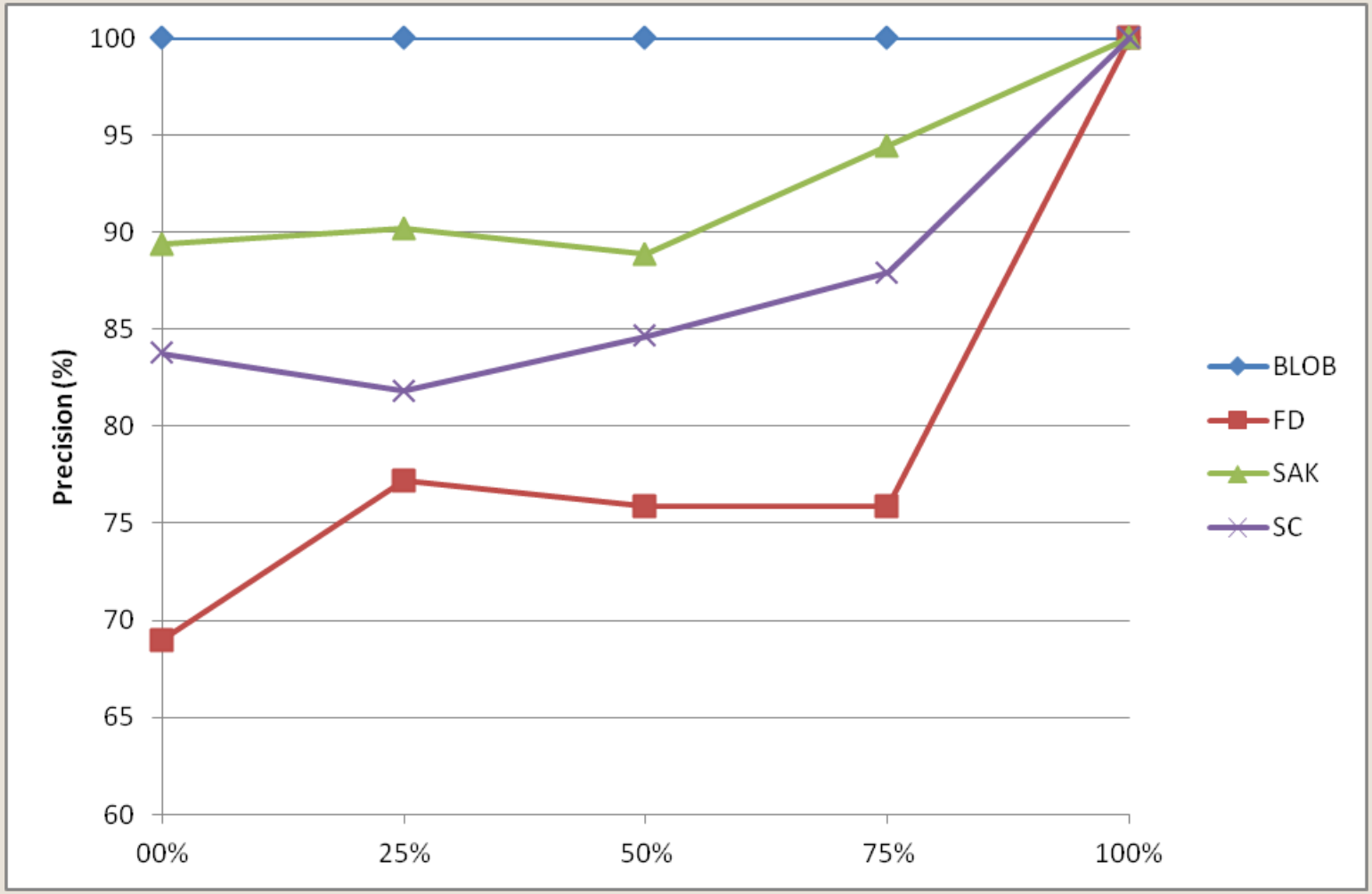

Figure : Trends in the increase of precision when integrating incremental feedback

Study Results

RQ4

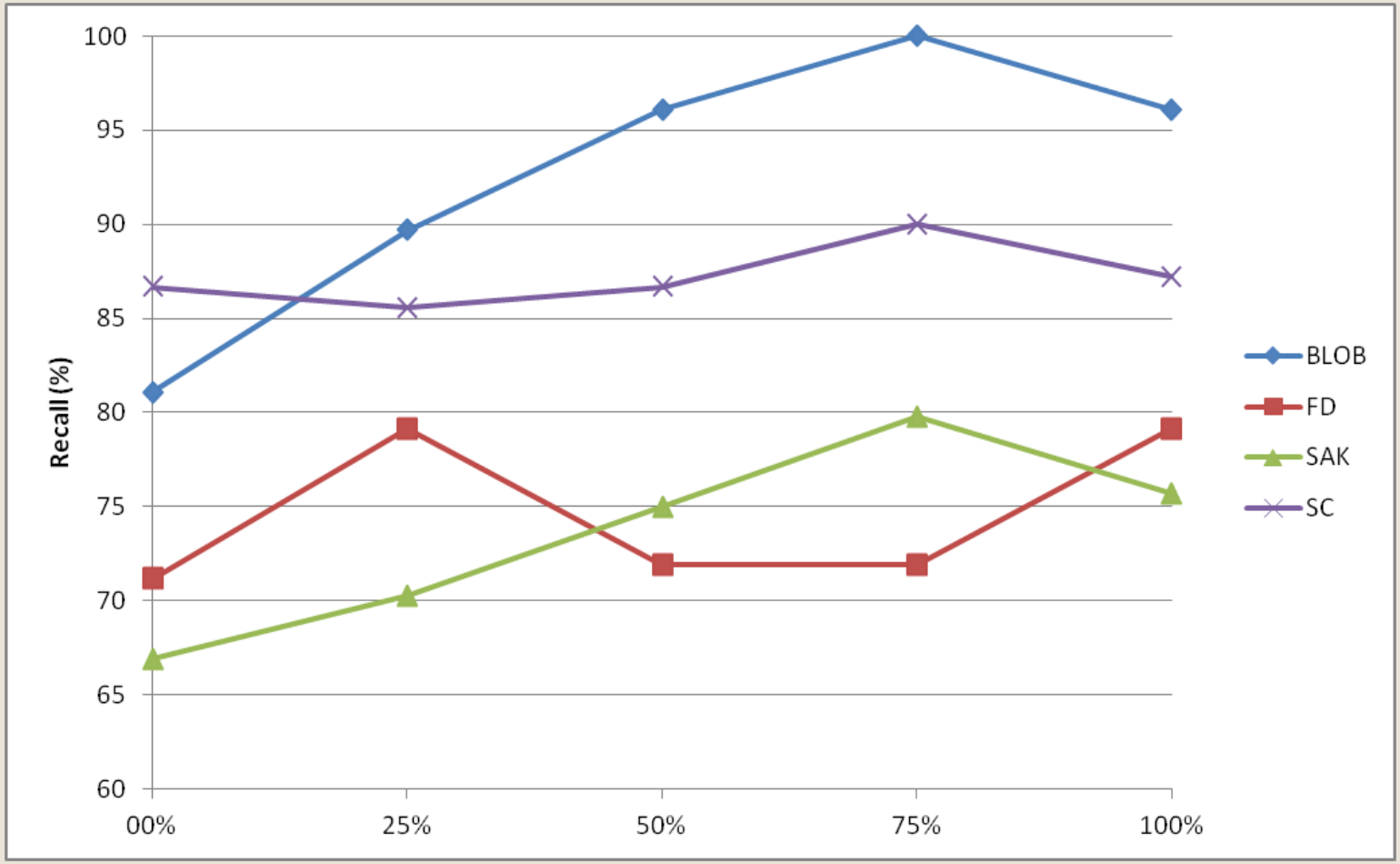

Figure : Trends in the increase of recall when integrating incremental feedback

Study Results

RQ4

We observe that the more feedback, the better the precision, up to 100%. For recall, the more feedback, the better the recall but with a slight decrease when we use 100% feedback. Thus, we answer RQ4 as follows: both precision and recall values increase when taking into account practitioners' feedback.

Conclusion and Future Work

Future Work

- use SMURF in real-world environments;
- integrate SMURF in eclipse;
- reproduce the study with other systems and anti-patterns to increase our confidence in the generalisability of our conclusions;
- evaluate the impact of the quality of training dataset and feedback set on SMURF results.

Conclusion and Future Work

Conclusion

- Introduced a novel approach to detect anti-patterns, SMURF, based on SVM;
- SMURF performs on 3 systems (ArgoUML v0.19.8, Azureus v2.3.0.6, and Xerces v2.7.0) and 4 anti-patterns (Blob, Functional Decomposition, Spaghetti Code, and Swiss Army Knife);
  - the accuracy of **SMURF** is **greater** than that of **DETEX**;
  - **SMURF** is more **stable** than the probabilistic approach **BDTEX**;
- SMURF can **overcome** the limitations of the previous approaches and could be more readily adopted by practitioners.
- SMURF is an **iterative** and **incremental** detection approach: could be applied in **continuous integration** context.

References

F. Khomh, M. D. Penta, and Y.-G. Guéhéneuc, "An exploratory study of the impact of antipatterns on class change-and fault-proneness," Journal of Empirical Software Engineering (EMSE), 2011.

E. H. Alikacem and H. A. Sahraoui, "Détection d'anomalies utilisant un langage de règle de qualité," in LMO. Hermes Science Publications, 2006, pp. 185–200.

G. Langelier, H. Sahraoui, and P. Poulin, "Visualization-based analysis of quality for large-scale software systems," in Proceedings of the 20th IEEE/ACM international Conference on Automated software engineering, ser. ASE '05. New York, NY, USA: ACM, 2005, pp. 214–223.

R. Marinescu, "Detection strategies: Metrics-based rules for detecting design flaws," in In Proceedings of the IEEE 20th International Conference on Software Maintenance. IEEE Computer Society Press, 2004, pp. 350–359.

M. Kessentini, S. Vaucher, and H. Sahraoui, "Deviance from perfection is a better criterion than closeness to evil when identifying risky code," in Proceedings of the IEEE/ACM international conference on Automated software engineering, ser. ASE '10. New York, NY, USA: ACM, 2010, pp. 113–122.

References

N. Moha, Y.-G. Guéhéneuc, L. Duchien, and A.-F. L. Meur, "DECOR: A method for the specification and detection of code and design smells," Transactions on Software Engineering (TSE), 2009.

F. Khomh, S. Vaucher, Y.-G. Guéhéneuc, and H. Sahraoui, "Bdtex: A gqm-based bayesian approach for the detection of antipatterns," J. Syst. Softw., vol. 84, no. 4, pp. 559–572, Apr. 2011.