Improved Defect Prediction using Code Cleaning

DAVID BOWES[§], STEVE COUNSELL[*],
TRACY HALL[*], THOMAS SHIPPEY[*]

[*]BRUNEL UNIVERSITY LONDON
[§]UNIVERSITY OF HERTFORDSHIRE

THIS WORK WAS PARTLY FUNDED BY A GRANT FROM THE UK'S ENGINEERING AND PHYSICAL SCIENCES RESEARCH COUNCIL UNDER GRANT NUMBER: EP/L011751/1

Agenda

- Introduction
- Code Cleaning
- Research Questions
- Methods
- Results
- Conclusions

Introduction

- Hundreds of published defect prediction models
- Generic machine learning approaches used
- Defect prediction is a bit ‘special’
- Code cleaning is our new simple code-specific technique…

Code Cleaning

- Data cleaning is a good practice in defect prediction
- Code cleaning goes further
  - DP models are trained on fix data
  - Fix data is not clean for many reasons
  - Noisy fix data impairs the performance of DP models
- Code cleaning tries to:
  1. Identify *methods* most likely to contain true fix data
  2. Clean out methods most likely not to contain noisy fixes
  3. Establish a more reliable set of cleaned methods for DP
- A taxonomy of method types…

A Taxonomy of Method Types

Method Type	Description
Abstract	A method that is labelled as abstract and is declared without implementation.
Anonymous	A method within an anonymous class.
Constructor	This is a constructor that holds executable code.
Empty Constructor	A constructor that holds no executable code.
Empty Method	A method that contains no executable code.
Getter	A method that returns a class field.
Interface	A method that is located within an Interface class and is declared without implementation.
Normal	A method which contains executable code. These methods can return a variable or be void. These methods are ones that contain the logic code that make the system operate.
Setter	A method that sets a class field.
Test	A unit test method.

Cleaned Code Used in Training and Testing

Method Type	Non Cleaned	Cleaned
Abstract	✓	×
Anonymous	✓	×
Constructor	✓	×
Empty Constructor	✓	×
Empty Method	✓	×
Getter	✓	×
Interface	✓	×
Normal	✓	✓
Setter	✓	×
Test	×	×

Research Questions

RQ1: Does code cleaning have a significant effect on the performance of a basic defect prediction model?

RQ2 Is the improvement of our code cleaning due to the reduction of the data imbalance?

Methods

1. Three systems were analysed…

The Three Systems Analysed

.

System	Release	Release Date	KLOC	Methods
EJDT	3.0	25/04/2004	292	13,571
T1	2.38	22/03/2013	52	4,552
T2	2.38	22/03/2013	36	4,996

Methods

2. We used the SZZ algorithm to collect fix data…

Faulty Methods in Each System

System	Version	Total Methods	Faulty Methods	% Faulty
T2	2.38	4,996	612	11.75
T1	2.38	4,552	322	7.07
EJDT	3.0	13,571	573	4.22

Methods

3. We created the IdentifierELFF tool to label each method according to our taxonomy…

Method Types in each System

Method Type	EJDT No.	EJDT %	T1 No.	T1 %	T2 No.	T2 %
Normal	9,478	69.84	2,765	60.74	2,096	41.95
Getter	905	6.67	448	9.84	959	19.2
Constructor	848	6.25	532	11.69	616	12.33
Interface	1131	8.33	520	11.42	185	3.7
Anonymous	331	2.44	113	2.48	751	15.03
Empty Method	569	4.19	78	1.71	4	0.08
Setter	125	0.92	79	1.74	239	4.78
Abstract	141	1.04	8	0.18	35	0.7
Empty Constructor	43	0.32	9	0.2	111	2.22
Total	13,571		4,552		4,996	

Methods

4. We created two different datasets containing:
   1. All methods
   2. Cleaned methods…

Impact on Faults when Cleaning Applied

System	Version	Total Methods	Faulty Methods	% Faulty
T2	2.38	2,096	441	21.04
T1	2.38	2,765	267	9.66
EJDT	3.0	9,478	522	5.51

Methods

5. We built basic DP models using:
   - Standard source code analysis metrics collected using JHawk
   - Naıve Bayes, J48 and Random Forest
   - Ten stratified folds of the data with each experiment repeated 100 times.
6. We compared our results to generic data balancing:
   - SMOTE and random under-sampling
   - Manual under-sampling

Results

RQ1 Does code cleaning have a significant effect on the performance of a basic defect prediction model?

The Impact of Code Cleaning on Prediction

Model	System	Cleaned	Precision	Recall	F-Measure	MCC
J48	T2	No	0.57	0.31	0.40	0.37
		Yes	0.63	0.44	0.52	0.43
	T1	No	0.18	0.68	0.28	0.32
		Yes	0.27	0.61	0.36	0.35
	EJDT	No	0.17	0.38	0.24	0.23
		Yes	0.24	0.42	0.30	0.29
Random Forest	T2	No	0.75	0.32	0.45	0.46
		Yes	0.78	0.42	0.55	0.50
	T1	No	0.26	0.65	0.37	0.38
		Yes	0.35	0.67	0.45	0.44
	EJDT	No	0.13	0.58	0.22	0.26
		Yes	0.17	0.61	0.27	0.31
Naïve Bayes	T2	No	0.25	0.21	0.23	0.13
		Yes	0.35	0.19	0.25	0.13
	T1	No	0.15	0.12	0.13	0.06
		Yes	0.13	0.16	0.14	0.05
	EJDT	No	0.36	0.27	0.31	0.28
		Yes	0.35	0.32	0.34	0.30

Effect Sizes of MCC using Code Cleaning

Classifier	System	P Value	Effect Size
J48	T2	0.00	0.15
	T1	0.00	0.35
	EJDT	0.00	0.78
Random Forest	T2	0.00	0.72
	T1	0.00	0.58
	EJDT	0.00	0.59
Naïve Bayes	T2	0.69	0.02
	T1	0.05	0.09
	EJDT	0.00	0.32

Results

RQ2 Is the improvement of our code cleaning due to the reduction of the data imbalance?

Applying SMOTE v Cleaning

Classifier	System	Recall	Precision	F-measure	MCC
J48	T2	0.06	0.07	0.06	0.02
	T1	-0.04	0.13	-0.01	0.01
	EJDT	-0.01	0.09	0.02	0.03
Random Forest	T2	0.07	0.09	0.08	0.05
	T1	0.03	0.04	0.03	0.02
	EJDT	-0.02	0.06	-0.02	-0.01
Naïve Bayes	T2	-0.04	0.12	0.01	-0.01
	T1	-0.04	0.04	0.00	-0.01
	EJDT	-0.04	0.06	0.02	0.01

Applying Random Undersampling v Cleaning

Classifier	System	Recall	Precision	F-measure	MCC
J48	T2	0.10	0.00	0.08	0.01
	T1	-0.14	-0.01	-0.12	-0.05
	EJDT	0.01	0.02	0.02	0.02
Random Forest	T2	0.11	0.02	0.10	0.05
	T1	-0.04	-0.05	-0.05	-0.01
	EJDT	-0.00	0.01	-0.00	0.00
Naïve Bayes	T2	-0.01	0.11	0.03	-0.00
	T1	-0.04	-0.16	-0.08	-0.04
	EJDT	-0.01	-0.01	-0.01	-0.01

Applying Manual Undersampling v Cleaning

Classifier	System	Recall	Precision	F-measure	MCC
J48	T2	0.03	0.01	0.03	0.03
	T1	-0.01	0.05	-0.00	0.01
	EJDT	0.01	0.02	0.02	0.02
Random Forest	T2	0.01	0.06	0.03	0.04
	T1	0.03	-0.05	0.02	-0.00
	EJDT	0.01	0.03	0.02	0.02
Naïve Bayes	T2	-0.02	-0.02	-0.02	-0.02
	T1	-0.02	-0.01	-0.02	-0.01
	EJDT	-0.03	-0.01	-0.02	-0.02

Conclusion

- Code cleaning can:
  - have a significant impact with large effect sizes at method level on DP performance
  - perform better than generic data balancing.
- Code cleaning a potentially important new defect-specific technique.
- Lots of potential uses of our approach.
- Much more work to do investigating the possibilities of code cleaning.

Any Questions?

David Bowes: d.h.bowes@herts.ac.uk

Steve Counsell: steve.counsell@brunel.ac.uk

Tracy Hall: tracy.hall@brunel.ac.uk

Thomas Shippey: thomas.shippey@brunel.ac.uk