A STUDY OF APPLICATION OF NEURAL NETWORK TECHNIQUE ON SOFTWARE REPOSITORIES

Ana Maria Bautista, Tomas San Feliu

Abstract: *Context: The use of software repositories is recent in software engineering. Different techniques have been applied to software engineering problems. We want to know the degree of application of neural networks as data mining technique. The results have allowed the preliminary state of the art of the application of neural network techniques on software repositories.*

Keywords: *neural networks, repository, data mining.*

ACM Classification Keywords: I.2.6 Artificial Intelligence –Connectionism and neural nets, H.2.7 Database Administration – Data warehouse and repository.

Introduction

The main goal of a software repository is to maintain the project data for later use. The data in mature engineering are used to check and compare different techniques.

In software engineering there are now a wide variety of software repositories. Software engineering as an emerging discipline is beginning to use the software repositories to compare different techniques.

The systematic review can be considered as a secondary study that reviews articles related to a specific search topic. As a secondary study provides an overview of an area of research to evaluate existing evidence [Kitchenham, 2009] and can provide identifying gaps in primary studies, where they may require new or enhanced studies.

These studies require a rigorous search and inclusion criteria and exclusion that are clearly defined in the research protocol and are presented in the report results.

The purpose of this study is to determine which use is made of the repositories and techniques have been tested and compared with them.

A software engineering (SE) data repository is defined as a set of well-defined, useful, and pertinent real-world data related to software projects, called datasets, which include quantitative and descriptive information about resources, products, processes, techniques, management, etc. Such data are being collected for various purposes by recognized organizations, as well as by individual software organizations and researchers. In most scientific and engineering disciplines, these data are useful for conducting benchmarking, experimental, and empirical studies. While highly varied and widely available in mature disciplines, data repositories are much less frequently found in emerging disciplines, including software engineering, as illustrated by the Guide to the Software Engineering Body of Knowledge [SWEBOK, 2004].

Mining software repositories (MSR) has become a fundamental area of research for the Software Engineering community, and of vital importance in the case of empirical studies. Software repositories contain a large amount of valuable information that includes source control systems storing all the history of the source code, defect tracking systems that host defects, enhancements and other issues, and other communication means such as mailing lists or forums.

To extract information from the Software Repositories different techniques are used. Mohanty et al. classify intelligent techniques in the following [Mohanty, 2010]:

1. Different neural network (NN) architecture including multilayer perception (MLP) and cascade correlation NN;
2. Fuzzy logic;
3. Genetic algorithm (GA);
4. Decision tree;
5. Case-based reasoning (CBR);
6. Soft computing (hybrid intelligent systems).

The other techniques:

1. Analogy based;
2. Support vector machine;
3. Self organizing maps (SOM).

Specifically, this work will focus on studying the application of neural networks in existing repositories.

Neural networks are used broadly in the studies we have selected. Therefore consider your extension, we will set the yields obtained in the literature and discuss the tendency in recent years.

This paper will be organized as follow. Research methodology section will describe the systematic review. Next section title Data Collection, reports the most relevant information gathered. Results section will report the review results analyzing collected date ordered by research questions. Discussion section summarizes the main findings. Study limitation section will discuss the assumptions and considerations of the study. Finally, conclusion and future work will outline the main conclusions obtained in future research works

Research methodology

This section provides an overview of the steps involved in the process systematic review, including the formulation of research question, the search strategy, the inclusion and exclusion criteria, and finally the data collection process.

Systematic mapping studies are a type of systematic literature review that aims to collect and classify research papers related to a specific topic [Peticrew, 2006; Kitchenham, 2007; Petersen, 2008].

This section provides an overview of the steps involved in the process of mapping review following Petersen et al. [Petersen, 2008] including the formulation of the research questions, the search strategy for primary studies, the inclusion and exclusion criteria, and the data collection process.

- Research questions

The main goal addressed by this study is to analyze the use of repositories and techniques by the research community and to consider its weakness to undertake the appropriate scientific research.

In this study the following research questions were considered:

- ✓ *Research question number 1 (RQ1):* Which and how many journals and conferences include techniques for mining software repositories research papers? To identify what are the main literature sources where the software repository analysis are published.
- ✓ *Research question number 2 (RQ2):* How comprehensive is the use of neural networks in the analysis of software repositories? Is necessary to know the use of neural networks for the analysis of repositories and is required to characterize the types of networks are used.

- ✓ *Research question number 3 (RQ3):* In which years have conducted studies with neural networks? How have been evolving the use of neural networks over time. The aim is to establish whether there is a trend over the last years.

- Search for primary studies strategy

The following search engine Google Scholar was used to make a general search for relevant papers in journals and conference proceedings. This search engine was selected because it is major search engine and it has a good usability.

Search is based in the two most used open software repositories: ISBSG and PROMISE. Search terms Data Mining ISBSG Repository and “Data Mining” “PROMISE Repository” were used. Only have been considered papers published from year 2010. Search was completed in January 2014. There were 304 papers, 94 corresponding to the first search term and 210 to the second.

Some studies use both repositories and therefore they are being selected for both search terms. Also it was noted that some papers have been published in different journal and conference proceedings or in different years. Duplicate references have been eliminated, overall 46 papers.

Finally, 258 references remained. The overall primary study selection is summarized in Figure 1.

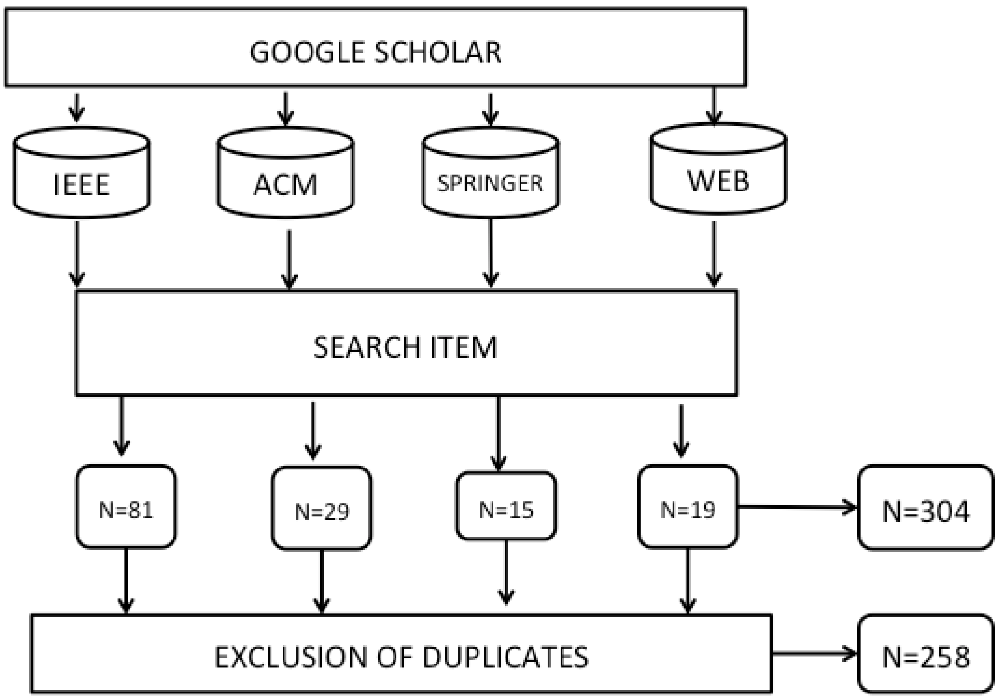

Figure 1. Search process for the selection of studies

- Inclusion and exclusion criteria

Inclusion and exclusion criteria are required to evaluate each primary study. In order to improve its reliability, only one author performed the screening process.

Inclusion and exclusion criteria are required to assess each potential primary study. In order to improve its reliability, the filtering process based on inclusion and exclusion criteria. All conflicts were resolved via discussion.

The following list describes the filtering process:

- ✓ *F1:* The first filter (F1) was used to identify papers that didn´t speak about data mining techniques and for this reason they didn´t answer our search questions, 82 papers were filtered;
- ✓ *F2:* The second filter (F2) was used to identify reviews that didn´t anything new to the rest of papers, 5 papers were filtered.
- ✓ *F3:* The third filter (F3) was used to locate the papers that were not written in English, 2 papers were filtered.

169 papers satisfied the logical condition (F1 AND F2 AND F3). Filtering process is summarized in Figure 2.

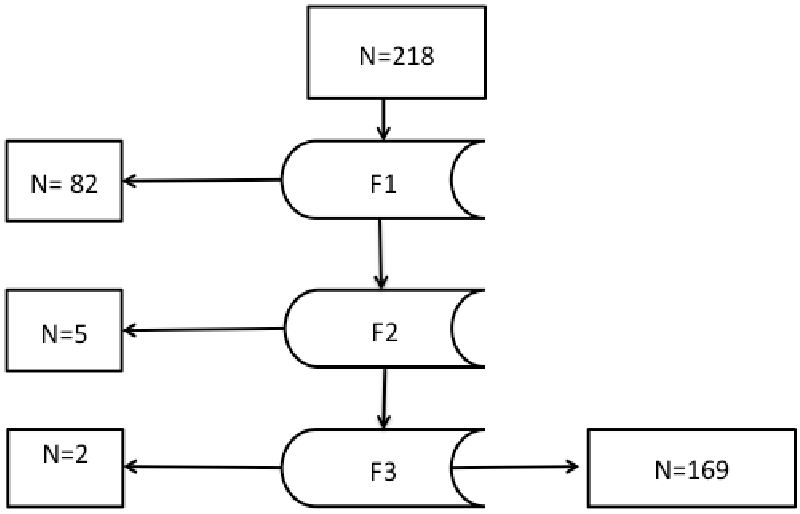

Figure 2. Filtering process

Data Collection

After the filtering process, the most relevant information was obtained from remained studies.

After the filtering process, the most relevant information was obtained from each of the 169 remained studies. This includes both general information and data addressing the five research questions. After reading each paper, the data was extracted and stored in a spreadsheet.

Papers classified by sources are shown in Table 1. Table 1 shows the evolution along the filtering process.

Table 1. Papers by sources

	Initial dataset	Duplicates	Filtered	Final dataset
IEEE	81	12	20	49
ACM	29	9	8	12
Springer	15	2	5	8
Others	179	23	56	100
TOTAL	304	46	89	169

It was analyzed the filtering process from the point of view of publication year. This point of view is shown in Table 2.

Table 2. Papers by published year

	Initial dataset	Duplicates	Filtered	Final dataset
2010	60	5	24	31
2011	51	6	13	32
2012	70	9	24	37
2013	108	25	27	56
2014	15	1	1	13
TOTAL	304	46	89	169

Additionally, the different sources was analyzed the distribution of type of papers. The most important source is the journals. The conferences are the second source of information. The Table 3 and Table 4 present the distribution of studies across different journals and conferences

Table 3. Papers by Journal

Journal	Total
Information and Software Technology	11
IEEE Transaction on Software Engineering	9
International Journal of Software Engineering and Knowledge Engineering	7
Empirical Software Engineering	6
Journal of Systems and Software	4
Information Sciences	4
Software Quality Journal	3
IET Software	2
IEEE Transaction on Reliability	2
International Journal of Software Engineering and Computing	2
International Journal of Reliability, Quality and Safety Engineering	2
Others	36
TOTAL	88

Table 4. Papers by Conference

Conference	Items
International Conference on Predictive Models in Software Engineering	8
International Conference on Software Process and Product Measurement	6
CSI International Conference on Software Engineering	3
International Joint Conference on Neural Networks	3
IEEE/ACM International Conference on Automated Software Engineering	2
International Conference on Computer, Information and Telecommunication Systems	2

International Symposium on Empirical Software Engineering and Measurement	2
IEEE International Conference on Tools with Artificial Intelligence	2
International Workshop on Realizing Artificial Intelligence Synergies in Software Engineering	2
Others	35
TOTAL	65

One import question is related with the occurrence of neural networks in data mining software repository. It was gathered information about the techniques used in selected studies. Sometimes one study contains several techniques. The distribution of techniques used is shown in Table 5.

Table 5. Distribution of techniques

Technique	Percentage
Neural Networks	14,29
Fuzzy Logic	2,77
Genetic algorithm	0,85
Decision trees	17,48
Case-base Reasoning	2,13
Hybrid intelligent systems	5,97
Analogy based	5,12
Support vector machine	2,77
Self organizing maps	2,77
Statistical techniques	44,78
Others	1,07

Results

The main results of systematic review are presented following the researched questions

- RQ1: Which and how many journals and conferences include techniques for mining software repositories research papers?

The publications of data mining techniques in software repositories have been captured of IEEE and ACM. The publications are concentrated in the domains of Software Engineering and Knowledge Engineering. In Table 4 is shown the ranking of journals. The three main journals are Information and Software Technology, IEEE Transaction on Software Engineering and International Journal of Software Engineering and Knowledge Engineering. The most important conference in this field of study are International Conference on Predictive Models in Software Engineering, International Conference on Software Process and Product Measurement, CSI International Conference on Software Engineering and International Joint Conference on Neural Networks. The tendency established by the journals is confirmed by the conference. It is denote IJCNN is specialized in neural networks and listed more conference publications.

- RQ2: How comprehensive is the use of neural networks in the analysis of software repositories?

This question is required to identify the use of neural networks in the studio of repositories and it requires characterize the types of networks are used. In the 169 papers selected in our study all kinds of data mining techniques are used. In many cases hybrid techniques are used with the original techniques or algorithms that improve their performance complement. In all studies the performance of at least two techniques are compared and in many numbers much higher. Most frequently statistical techniques are used. Neural Networks are ranked in third position below of decision trees as shown in Table 5. The multilayer perceptron (MLP) and Radial Basis function Network (RBFN) are some of the popular Neural Network architecture.

- RQ3: What years there have been conducted studies with neural networks?

How have been evolving the use of neural networks over time? It intends to establish whether there is a trend over the last years. The figure 3 shows that there is a significant increase over 2013.

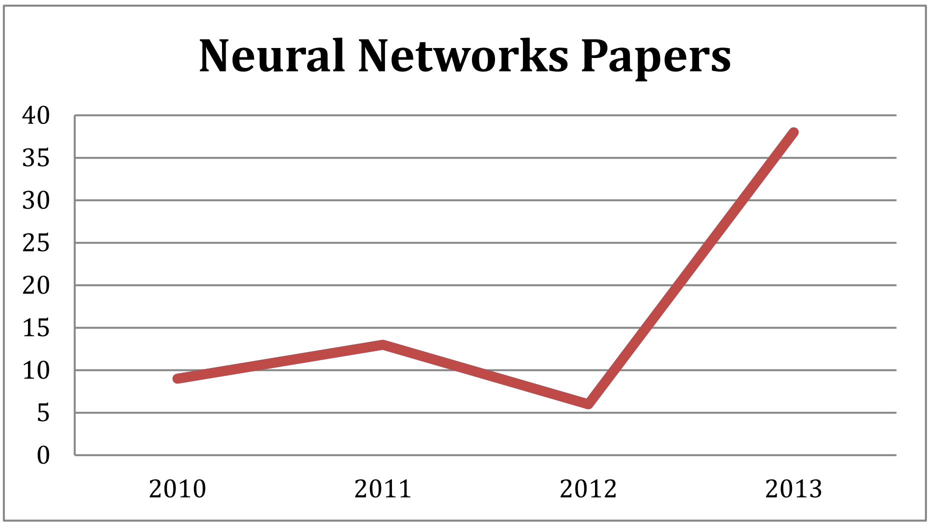

Figure 3. Evolution of Neural Networks publications

Discussion

This section summarizes the main findings of the systematic review. It also includes the limitations of the study and discusses the implications for researchers.

This study shows the extent to which and how software engineering researchers have used ISBSG and PROMISE until January of 2014. Thereby, the papers that have worked with these datasets have been identified and classified by answering a set of research questions. This systematic mapping review conveys a picture of application of neural networks on software repositories.

The search terms Data Mining ISBSG Repository and "Data Mining" "PROMISE Repository" were the input for Google Scholar. This resulted 94 y 210 results respectively. The search was completed in January 2014. After the elimination of duplicates and the filtering process, the most relevant information was obtained from each of the 169 remaining studies.

The first research has been a consolidation of journals and conferences where articles are published on this subject. It is noteworthy that the conference that has emerged in recent years the subject of analysis software repository is maintained.

A second research question shows that the neural networks maintain their presence as data mining technique. Furthermore, from the viewpoint of performance which is a consolidated sample technique. In fact neural networks are consolidated as reference technique in comparisons to characterize its performance.

Regarding the third question, we see that during the years 2010 to 2012 remained stable presence. In the years 2013-2014 a remarkable growth over the previous three years is provided.

Study limitations

It is important to consider that the results obtained form a systematic review could be affected by researchers conducting the review, by the selected search terms, and by the chosen time frame [Elberzhaer, 2012].

The first limitation concerns the search strategy employed, to use ISBSG and PROMISE in search terms, results can be affected.

It is important to indicate that some more recent studies may be missing because the search engines may not have indexed them.

Finally, exclusion of papers written in a language other than English may have lead to bias in the selection process. This could not be avoided due to impossibility of the revision team to address these languages.

The second limitation concerns bias in the data collection. The first information collected was about the identification and general details of the paper such as title, its authors, source where it was published, abstract, year of publication. Full texts have been read when abstract of papers not provide enough information.

Conclusion and future work

This paper presents the results of a systematic review about the usage of techniques on software repositories until January of 2014. After the searching and filtering process, 169 papers were analyzed.

They have cataloged the papers selected based repository using the techniques of data mining that use storing spreadsheet data needed to answer questions research.

Analyzing results of this study note that interest in software repositories usage and trend to employ neural networks techniques to data mining is increasing.

In summary, this paper presents a comprehensive snapshot of actual use of neural network to analyze software repositories.

In the future, the authors intend to explain the period of gathering of papers to keep the work up-to-date and the answering other interesting questions concerning the trend of usage of neural networks. Other future work will delve into the use of neural networks as a tool for mining software repositories to facilitate the work of developers and project managers.

Bibliography

[Cheikhi, 2013] Cheikhi, L., & Abran, A. Promise and ISBSG Software Engineering Data Repositories: A Survey. In Software Measurement and the 2013 Eighth International Conference on Software Process and Product Measurement (IWSM-MENSURA), 2013, Joint Conference of the 23rd International Workshop on IEEE, pp. 17-24

[Elberzhaer, 2012] Elberzhager, F., Münch, J., & Nha, V. T. N. A systematic mapping study on the combination of static and dynamic quality assurance techniques.Information and Software Technology, 2012, 54(1), 1-15

[Fernández-Diego, 2014] Fernández-Diego, M., & González-Ladrón-de-Guevara, F. Potential and Limitations of the ISBSG Dataset in Enhancing Software Engineering Research: A Mapping Review. Information and Software Technology, 2014

[Kitchenham, 2007] B. Kitchenham, S. Charters, Guidelines for Performing Systematic Literature Reviews in Software Engineering, Software Engineering Group, School of Computer Science and Mathematics, Keele University, 2007.

[Kitchenham, 2009] Kitchenham, B., Pearl Brereton, O., Budgen, D., Turner, M., Bailey, J., & Linkman, S. Systematic literature reviews in software engineering–A systematic literature review. Information and software technology, 2009, 51(1), 7-15.

[Mohanty, 2010] Mohanty, R., Ravi, V., & Patra, M. R. The application of intelligent and soft-computing techniques to software engineering problems: a review. International Journal of Information and Decision Sciences, 2010, 2(3), pp. 233-272.

[Petersen, 2008] Petersen K., Feldt R., Mujtaba S., Mattsson M., Systematic mapping studies in software engineering, in: 12th Int. Conf. Eval. Assess. Softw. Eng., 2008, p. 1.

[Petticrew, 2006] M. Petticrew, H. Roberts, Systematic Reviews in the Social Sciences: A Practical Guide, John Wiley & Sons, Limited, 2006.

[Robles, 2010] Robles, G. Replicating MSR: A study of the potential replicability of papers published in the Mining Software Repositories proceedings. In Mining Software Repositories (MSR), 2010 7th IEEE Working Conference on pp. 171-180, May 2010

[SWEBOK, 2004] IEEE. 2004. “Guide to the Software Engineering Body Of knowledge- SWEBOK.” Los Alamitos, California: IEEE Computer Society, 204 p. (last accessed on 30/01/2013). http://www.computer.org/portal/web/swebok

Appendix A

This section provides the primary studies selected from the systematic review

Abaei, G., & Selamat, A. (2013). A survey on software fault detection based on different prediction approaches. Vietnam Journal of Computer Science, 1-17.

Abaei, G., Rezaei, Z., & Selamat, A. (2013, November). Fault prediction by utilizing self-organizing Map and Threshold. In Control System, Computing and Engineering (ICCSCE), 2013 IEEE International Conference on (pp. 465-470). IEEE.

Afzal, W., Torkar, R., & Feldt, R. (2012). Resampling Methods in Software Quality Classification. International Journal of Software Engineering and Knowledge Engineering, 22(02), 203-223.

Al Khalidi, N., Saifan, A. A., & Alsmadi, I. M. (2012, May). Selecting a standard set of attributes for cost estimation of software projects. In Computer, Information and Telecommunication Systems (CITS), 2012 International Conference on (pp. 1-5). IEEE.

Al Khalidi, N., Saifan, A. A., & Alsmadi, I. M. (2012, May). Selecting a standard set of attributes for cost estimation of software projects. In Computer, Information and Telecommunication Systems (CITS), 2012 International Conference on (pp. 1-5). IEEE.

Alan, O., & Catal, C. (2011). Thresholds based outlier detection approach for mining class outliers: An empirical case study on software measurement datasets. Expert Systems with Applications, 38(4), 3440-3445.

Al-Jamimi, H. A., & Ghouti, L. (2011, December). Efficient prediction of software fault proneness modules using support vector machines and probabilistic neural networks. In Software Engineering (MySEC), 2011 5th Malaysian Conference in (pp. 251-256). IEEE.

Alsmadi, I., & Najadat, H. (2011). Evaluating the change of software fault behavior with dataset attributes based on categorical correlation. Advances in Engineering Software, 42(8), 535-546.

ALTIDOR, W., KHOSHGOFTAAR, T. M., & GAO, K. (2010). Wrapper-based feature ranking techniques for determining relevance of software engineering metrics. International Journal of Reliability, Quality and Safety Engineering, 17(05), 425-464.

Amasaki, S., & Yokogawa, T. (2013, October). The Effects of Variable Selection Methods on Linear Regression-Based Effort Estimation Models. In Software Measurement and the 2013 Eighth International Conference on Software Process and Product Measurement (IWSM-MENSURA), 2013 Joint Conference of the 23rd International Workshop on (pp. 98-103). IEEE.

Amasaki, S., & Yokogawa, T. (2013, October). The Effects of Variable Selection Methods on Linear Regression-Based Effort Estimation Models. In Software Measurement and the 2013 Eighth International Conference on Software Process and Product Measurement (IWSM-MENSURA), 2013 Joint Conference of the 23rd International Workshop on (pp. 98-103). IEEE.

Amasaki, S., Takahara, Y., & Yokogawa, T. (2011, November). Performance Evaluation of Windowing Approach on Effort Estimation by Analogy. In Software Measurement, 2011 Joint Conference of the 21st Int'l Workshop on and 6th Int'l Conference on Software Process and Product Measurement (IWSM-MENSURA) (pp. 188-195). IEEE.

Amasaki, S., Takahara, Y., & Yokogawa, T. (2011, November). Performance Evaluation of Windowing Approach on Effort Estimation by Analogy. In Software Measurement, 2011 Joint Conference of the 21st Int'l Workshop on and 6th Int'l Conference on Software Process and Product Measurement (IWSM-MENSURA) (pp. 188-195). IEEE.

Anwar, S., Rana, Z. A., Shamail, S., & Awais, M. M. (2012, September). Using association rules to identify similarities between software datasets. In Quality of Information and Communications Technology (QUATIC), 2012 Eighth International Conference on the (pp. 114-119). IEEE.

Armah, G. K., Luo, G., & Qin, K. (2013, November). Multi_level data pre_processing for software defect prediction. In Information Management, Innovation Management and Industrial Engineering (ICIII), 2013 6th International Conference on (Vol. 2, pp. 170-174). IEEE.

Azzeh, M. (2011, September). Software effort estimation based on optimized model tree. In Proceedings of the 7th International Conference on Predictive Models in Software Engineering (p. 6). ACM.

Azzeh, M., & Alseid, M. (2013). Value of ranked voting methods for estimation by analogy. Software, IET, 7(4).

Azzeh, M., Cowling, P. I., & Neagu, D. (2010, June). Software Stage-Effort Estimation Based on Association Rule Mining and Fuzzy Set Theory. In Computer and Information Technology (CIT), 2010 IEEE 10th International Conference on (pp. 249-256). IEEE.

Bakır, A., Turhan, B., & Bener, A. (2011). A comparative study for estimating software development effort intervals. Software Quality Journal, 19(3), 537-552.

Bakır, A., Turhan, B., & Bener, A. B. (2010). A new perspective on data homogeneity in software cost estimation: a study in the embedded systems domain. Software Quality Journal, 18(1), 57-80.

Balsera, J. V., Montequin, V. R., Fernandez, F. O., & González-Fanjul, C. A. (2012). Data Mining Applied to the Improvement of Project Management.

Baojun, M., Dejaeger, K., Vanthienen, J., & Baesens, B. (2011). Software defect prediction based on association rule classification. Available at SSRN 1785381.

Bardsiri, V. K., Jawawi, D. N. A., Bardsiri, A. K., & Khatibi, E. (2013). LMES: A localized multi-estimator model to estimate software development effort. Engineering Applications of Artificial Intelligence, 26(10), 2624-2640.

Bardsiri, V. K., Jawawi, D. N. A., Hashim, S. Z. M., & Khatibi, E. (2013). A flexible method to estimate the software development effort based on the classification of projects and localization of comparisons. Empirical Software Engineering, 1-28.

Barros, R. C., Basgalupp, M. P., Cerri, R., da Silva, T. S., & de Carvalho, A. C. (2013, July). A grammatical evolution approach for software effort estimation. In Proceeding of the fifteenth annual conference on Genetic and evolutionary computation conference (pp. 1413-1420). ACM.

Benala, T. R., Mall, R., Srikavya, P., & HariPriya, M. V. (2014, January). Software Effort Estimation Using Data Mining Techniques. In ICT and Critical Infrastructure: Proceedings of the 48th Annual Convention of Computer Society of India-Vol I (pp. 85-92). Springer International Publishing.

Bettenburg, N., Nagappan, M., & Hassan, A. E. (2014). Towards improving statistical modeling of software engineering data: think locally, act globally!. Empirical Software Engineering, 1-42.

Borges, K. C. A. D., de Barcelos Tronto, I. F., de Aquino Lopes, R., & da Silva, J. D. S. (2013). A Data Pre-Processing Method for Software Effort Estimation Using Case-Based Reasoning. CLEI ELECTRONIC JOURNAL, 16(3).

Borges, R., & Menzies, T. (2012, September). Learning to change projects. In Proceedings of the 8th International Conference on Predictive Models in Software Engineering (pp. 11-18). ACM.

Brady, A., & Menzies, T. (2010, September). Case-based reasoning vs parametric models for software quality optimization. In Proceedings of the 6th International Conference on Predictive Models in Software Engineering (p. 3). ACM.

Bruntink, M. (2013). Towards base rates in software analytics: Early results and challenges from studying Ohloh. Science of Computer Programming.

Castejón-Limas, M., Ordieres-Meré, J., González-Marcos, A., & González-Castro, V. (2011). Effort estimates through project complexity. Annals of Operations Research, 186(1), 395-406.

Catal, C. (2014). A Comparison of Semi-Supervised Classification Approaches for Software Defect Prediction. Journal of Intelligent Systems, 23(1), 75-82.

Catal, C., & Diri, B. (2013). A fault detection strategy for software projects. Tehnički vjesnik, 20(1), 1-7.

Catal, C., Alan, O., & Balkan, K. (2011). Class noise detection based on software metrics and ROC curves. Information Sciences, 181(21), 4867-4877.

Catal, C., Sevim, U., & Diri, B. (2010). Metrics-driven software quality prediction without prior fault data. In Electronic Engineering and Computing Technology (pp. 189-199). Springer Netherlands.

Chang, R., Mu, X., & Zhang, L. (2011). Software Defect Prediction Using Non-Negative Matrix Factorization. Journal of Software (1796217X), 6(11).

Chaturvedi, K. K., & Singh, V. B. (2012, September). Determining bug severity using machine learning techniques. In Software Engineering (CONSEG), 2012 CSI Sixth International Conference on (pp. 1-6). IEEE.

Chatzikonstantinou, G., Kontogiannis, K., & Attarian, I. M. (2013, January). A goal driven framework for software project data analytics. In Advanced Information Systems Engineering (pp. 546-561). Springer Berlin Heidelberg.

Cheikhi, L., & Abran, A. (2013, October). Promise and ISBSG Software Engineering Data Repositories: A Survey. In Software Measurement and the 2013 Eighth International Conference on Software Process and Product Measurement (IWSM-MENSURA), 2013 Joint Conference of the 23rd International Workshop on (pp. 17-24). IEEE.

Cheikhi, L., & Abran, A. (2013, October). Promise and ISBSG Software Engineering Data Repositories: A Survey. In Software Measurement and the 2013 Eighth International Conference on Software Process and Product Measurement (IWSM-MENSURA), 2013 Joint Conference of the 23rd International Workshop on (pp. 17-24). IEEE.

Corazza, A., Di Martino, S., Ferrucci, F., Gravino, C., Sarro, F., & Mendes, E. (2013). Using tabu search to configure support vector regression for effort estimation. Empirical Software Engineering, 18(3), 506-546.

Crespo, D., & Ruiz, M. (2012, December). Decision making support in CMMI process areas using multiparadigm simulation modeling. In Simulation Conference (WSC), Proceedings of the 2012 Winter (pp. 1-12). IEEE.

Czibula, G., Marian, Z., & Czibula, I. G. (2014). Software defect prediction using relational association rule mining. Information Sciences.

Dejaeger, K., Verbraken, T., & Baesens, B. (2013). Toward Comprehensible Software Fault Prediction Models Using Bayesian Network Classifiers. Software Engineering, IEEE Transactions on, 39(2), 237-257.

del Águila, I. M., & Del Sagrado, J. (2011). Requirement risk level forecast using bayesian networks classifiers. International Journal of Software Engineering and Knowledge Engineering, 21(02), 167-190.

Di Martino, S., Ferrucci, F., Gravino, C., & Sarro, F. (2011). A genetic algorithm to configure support vector machines for predicting fault-prone components. In Product-Focused Software Process Improvement (pp. 247-261). Springer Berlin Heidelberg.

Elyassami, S., & Idri, A. (2013). Evaluating software cost estimation models using fuzzy decision trees. Recent Advances in Knowledge Engineering and Systems Science, WSEAS Press, 243-248.

Falessi, D., Cantone, G., & Canfora, G. (2013). Empirical principles and an industrial case study in retrieving equivalent requirements via natural language processing techniques. Software Engineering, IEEE Transactions on, 39(1), 18-44.

Fernandes, P., Lopes, L., Normey, S., & Ruiz, D. (2013, May). Stochastic aware random forests-a variation less impacted by randomness. In Proc. of the Twenty Sixth Int. FLAIRS Conf.(FLAIRS 2013), C. Boonthum-Denecke and GM Youngblood, Eds. AAAI Press (pp. 146-149).

Fernández-Diego, M., & González-Ladrón-de-Guevara, F. (2014). Potential and Limitations of the ISBSG Dataset in Enhancing Software Engineering Research: A Mapping Review. Information and Software Technology.

Gao, K., & Khoshgoftaar, T. M. (2011). Software Defect Prediction for High-Dimensional and Class-Imbalanced Data. In SEKE (pp. 89-94).

GAO, K., KHOSHGOFTAAR, T. M., & WALD, R. (2014). THE USE OF UNDER-AND OVERSAMPLING WITHIN ENSEMBLE FEATURE SELECTION AND CLASSIFICATION FOR SOFTWARE QUALITY PREDICTION. International Journal of Reliability, Quality and Safety Engineering.

Garcia, A., Gonzalez, I., Colomo-Palacios, R., Lopez, J. L., & Ruiz, B. (2011). Methodology for software development estimation optimization based on neural networks. Latin America Transactions, IEEE (Revista IEEE America Latina), 9(3), 384-398.

Gashler, M. S., Smith, M. R., Morris, R., & Martinez, T. (2013). Missing Value Imputation With Unsupervised Backpropagation. arXiv preprint arXiv:1312.5394.

Gay, G. (2010, September). A baseline method for search-based software engineering. In Proceedings of the 6th International Conference on Predictive Models in Software Engineering (p. 2). ACM.

Gayatri, N., Nickolas, S., & Reddy, A. V. (2010). Feature selection using decision tree induction in class level metrics dataset for software defect predictions. In Proceedings of the World Congress on Engineering and Computer Science (Vol. 1, pp. 124-129).

Gayatri, N., Nickolas, S., & Reddy, A. V. (2012). ANOVA Discriminant Analysis for Features Selected through Decision Tree Induction Method. In Global Trends in Computing and Communication Systems (pp. 61-70). Springer Berlin Heidelberg.

Gray, D. P. H. (2013). Software Defect Prediction Using Static Code Metrics: Formulating a Methodology. PhD

Gray, D., Bowes, D., Davey, N., Sun, Y., & Christianson, B. (2011). The misuse of the nasa metrics data program data sets for automated software defect prediction. In Proceeding of EASE 2011

Gray, D., Bowes, D., Davey, N., Sun, Y., & Christianson, B. (2012). Reflections on the NASA MDP data sets. Software, IET, 6(6), 549-558.

Hazrati, N. (2011). Geometric Approaches to Statistical Defect Prediction and Learning (Doctoral dissertation, Concordia University).

He, P., Li, B., Liu, X., Chen, J., & Ma, Y. (2014). An Empirical Study on Software Defect Prediction with Simplified Metric Set. arXiv preprint arXiv:1402.3873.

He, Z., Shu, F., Yang, Y., Li, M., & Wang, Q. (2012). An investigation on the feasibility of cross-project defect prediction. Automated Software Engineering, 19(2), 167-199.

Herbold, S. (2013, October). Training data selection for cross-project defect prediction. In Proceedings of the 9th International Conference on Predictive Models in Software Engineering (p. 6). ACM.

Idri, A., & AMAZAL, F. A. (2012, August). Software cost estimation by fuzzy analogy for ISBSG repository. In Proceedings of the 10th International FLINS Conference on Uncertainty Modeling in Knowledge Engineering and Decision Making, Istanbul.

Jin, C., & Jin, S. W. (2014). Applications of fuzzy integrals for predicting software fault-prone. Journal of Intelligent and Fuzzy Systems.

Johansson, U., Konig, R., Lofstrom, T., & Bostrom, H. (2013, June). Evolved decision trees as conformal predictors. In Evolutionary Computation (CEC), 2013 IEEE Congress on (pp. 1794-1801). IEEE.

Johansson, U., Lofstrom, T., & Bostrom, H. (2013, April). Overproduce-and-select: The grim reality. In Computational Intelligence and Ensemble Learning (CIEL), 2013 IEEE Symposium on (pp. 52-59). IEEE.

Keung, J., Kocaguneli, E., & Menzies, T. (2011). A ranking stability indicator for selecting the best effort estimator in software cost estimation. Automated Software Engineering (submitted).

Keung, J., Kocaguneli, E., & Menzies, T. (2013). Finding conclusion stability for selecting the best effort predictor in software effort estimation. Automated Software Engineering, 20(4), 543-567.

Khan, K. (2010). The Evaluation of Well-known Effort Estimation Models based on Predictive Accuracy Indicators.

Khoshgoftaar, T. M., Gao, K., & Napolitano, A. (2012). An empirical study of feature ranking techniques for software quality prediction. International Journal of Software Engineering and Knowledge Engineering, 22(02), 161-183.

Khoshgoftaar, T. M., Gao, K., & Seliya, N. (2010, October). Attribute selection and imbalanced data: Problems in software defect prediction. In Tools with Artificial Intelligence (ICTAI), 2010 22nd IEEE International Conference on (Vol. 1, pp. 137-144). IEEE.

Kocaguneli, E., & Menzies, T. (2013). Software effort models should be assessed via leave-one-out validation. Journal of Systems and Software, 86(7), 1879-1890.

Kocaguneli, E., Cukic, B., & Lu, H. (2013, May). Predicting more from less: synergies of learning. In Realizing Artificial Intelligence Synergies in Software Engineering (RAISE), 2013 2nd International Workshop on (pp. 42-48). IEEE.

Kocaguneli, E., Gay, G., Menzies, T., Yang, Y., & Keung, J. W. (2010, September). When to use data from other projects for effort estimation. In Proceedings of the IEEE/ACM international conference on Automated software engineering (pp. 321-324). ACM.

Kocaguneli, E., Menzies, T., & Keung, J. W. (2012). On the value of ensemble effort estimation. Software Engineering, IEEE Transactions on, 38(6), 1403-1416.

Koru, G., Liu, H., Zhang, D., & El Emam, K. (2010). Testing the theory of relative defect proneness for closed-source software. Empirical Software Engineering, 15(6), 577-598.

Krishnan, S. (2013). Evidence-based defect assessment and prediction for software product lines. PHDThesis

Krishnan, S., Strasburg, C., Lutz, R. R., Goseva-Popstojanova, K., & Dorman, K. S. (2013). Predicting failure-proneness in an evolving software product line. Information and Software Technology, 55(8), 1479-1495.

KUMAR PANDEY, A. J. E. E. T., & Goyal, N. K. (2012). A Fuzzy Model for Early Software Quality Prediction and Module Ranking. International Journal of Performability Engineering, 8(6).

Liparas, D., Angelis, L., & Feldt, R. (2012). Applying the Mahalanobis-Taguchi strategy for software defect diagnosis. Automated Software Engineering, 19(2), 141-165.

Litoriya, R., Sharma, N., & Kothari, A. (2012, September). Incorporating Cost driver substitution to improve the effort using Agile COCOMO II. In Software Engineering (CONSEG), 2012 CSI Sixth International Conference on (pp. 1-7). IEEE.

Liu, Y., Khoshgoftaar, T. M., & Seliya, N. (2010). Evolutionary optimization of software quality modeling with multiple repositories. Software Engineering, IEEE Transactions on, 36(6), 852-864.

Löfström, T., Johansson, U., & Boström, H. (2013). Effective Utilization of Data in Inductive Conformal Prediction. Neural Networks (IJCNN), The 2013 International Joint Conference on, Aug 2013

Lokan, C., & Mendes, E. (2014). Investigating the Use of Duration-based Moving Windows to Improve Software Effort Prediction: a Replicated Study. Information and Software Technology.

Lopez-Martin, C., Isaza, C., & Chavoya, A. (2012). Software development effort prediction of industrial projects applying a general regression neural network. Empirical Software Engineering, 17(6), 738-756.

Lumpe, M., Vasa, R., Menzies, T., Rush, R., & Turhan, B. (2012). Learning better inspection optimization policies. International Journal of Software Engineering and Knowledge Engineering, 22(05), 621-644.

Ma, Y., Luo, G., Zeng, X., & Chen, A. (2012). Transfer learning for cross-company software defect prediction. Information and Software Technology, 54(3), 248-256.

Mausa, G., Grbac, T. G., & Basic, B. D. (2012, May). Multivariate logistic regression prediction of fault-proneness in software modules. In MIPRO, 2012 Proceedings of the 35th International Convention (pp. 698-703). IEEE.

Mende, T., & Koschke, R. (2010, March). Effort-aware defect prediction models. In Software Maintenance and Reengineering (CSMR), 2010 14th European Conference on (pp. 107-116). IEEE.

Menzies, T., & Zimmermann, T. (2013). Software Analytics: So What?. Software, IEEE, 30(4), 31-37.

Menzies, T., Brady, A., Keung, J., Hihn, J., Williams, S., El-Rawas, O., ... & Boehm, B. (2013). Learning Project Management Decisions: A Case Study with Case-Based Reasoning Versus Data Farming. Software Engineering, IEEE Transactions on,

Menzies, T., Butcher, A., Marcus, A., Zimmermann, T., & Cok, D. (2011, November). Local vs. global models for effort estimation and defect prediction. In Proceedings of the 2011 26th IEEE/ACM International Conference on Automated Software Engineering (pp. 343-351). IEEE Computer Society.

Minku, L. L., & Yao, X. (2011, September). A principled evaluation of ensembles of learning machines for software effort estimation. In Proceedings of the 7th International Conference on Predictive Models in Software Engineering (p. 9). ACM.

Minku, L. L., & Yao, X. (2012, June). Using unreliable data for creating more reliable online learners. In Neural Networks (IJCNN), The 2012 International Joint Conference on (pp. 1-8). IEEE.

Minku, L. L., & Yao, X. (2013). Ensembles and locality: Insight on improving software effort estimation. Information and Software Technology, 55(8), 1512-1528.

Mısırlı, A. T., Bener, A. B., & Turhan, B. (2011). An industrial case study of classifier ensembles for locating software defects. Software Quality Journal, 19(3), 515-536.

Mittas, N., & Angelis, L. (2010). Visual comparison of software cost estimation models by regression error characteristic analysis. Journal of Systems and Software, 83(4), 621-637.

Mittas, N., & Angelis, L. (2013). Ranking and clustering software cost estimation models through a multiple comparisons algorithm. Software Engineering, IEEE Transactions on, 39(4), 537-551.

Mizuno, O., & Hata, H. (2010). An integrated approach to detect fault-prone modules using complexity and text feature metrics. In Advances in Computer Science and Information Technology (pp. 457-468). Springer Berlin Heidelberg.

Mizuno, O., & Hata, H. (2013). A metric to detect fault–prone software modules using text filtering. International Journal of Reliability and Safety, 7(1), 17-31.

Mizuno, O., & Hirata, Y. (2010). Fault-prone module prediction using contents of comment lines. In International Workshop on Empirical Software Engineering in Practice 2010 (IWESEP 2010) (p. 39).

Mohanty, R., Ravi, V., & Patra, M. R. (2010). The application of intelligent and soft-computing techniques to software engineering problems: a review. International Journal of Information and Decision Sciences, 2(3), 233-272.

Nagpal, G., Uddin, M., & Kaur, A. (2012). A Comparative Study of Estimation by Analogy using Data Mining Techniques. Journal of Information Processing Systems, 8(4).

Nagpal, G., Uddin, M., & Kaur, A. (2012). A Hybrid Technique using Grey Relational Analysis and Regression for Software Effort Estimation using Feature Selection. International Journal of Soft Computing and Engineering (IJSCE), 1(6), 20-27.

Nagpal, G., Uddin, M., & Kaur, A. (2014). Grey relational effort analysis technique using robust regression methods for individual projects. International Journal of Computational Intelligence Studies, 3(1), 40-73.

Nassif, A. B., Ho, D., & Capretz, L. F. (2013). Towards an early software estimation using log-linear regression and a multilayer perceptron model. Journal of Systems and Software, 86(1), 144-160.

Nguyen, V. H., & Tran, L. M. S. (2010, September). Predicting vulnerable software components with dependency graphs. In Proceedings of the 6th International Workshop on Security Measurements and Metrics (p. 3). ACM.

Okutan, A., & Yıldız, O. T. (2014). Software defect prediction using Bayesian networks. Empirical Software Engineering, 19(1), 154-181.

Paikari, E., Richter, M. M., & Ruhe, G. (2012). Defect prediction using case-based reasoning: An attribute weighting technique based upon sensitivity analysis in neural networks. International Journal of Software Engineering and Knowledge Engineering, 22(06), 747-768.

Paikari, E., Sun, B., Ruhe, G., & Livani, E. (2011, September). Customization support for CBR-based defect prediction. In Proceedings of the 7th International Conference on Predictive Models in Software Engineering (p. 16). ACM.

Pandey, A. K., & Goyal, N. K. (2010). Predicting Fault-prone Software Module Using Data Mining Technique and Fuzzy Logic. International Journal of Computer and Communication Technology (Special Issue), 2(2-4), 56-63.

Papakroni, V. (2013). Data Carving: Identifying and Removing Irrelevancies in the Data. West Virginia University.

Park, B. J., Oh, S. K., & Pedrycz, W. (2013). The design of polynomial function-based neural network predictors for detection of software defects. Information Sciences, 229, 40-57.

Pelayo, L., & Dick, S. (2012). Evaluating stratification alternatives to improve software defect prediction. Reliability, IEEE Transactions on, 61(2), 516-525.

Peters, F., & Menzies, T. (2012, June). Privacy and utility for defect prediction: Experiments with morph. In Proceedings of the 2012 International Conference on Software Engineering (pp. 189-199). IEEE Press.

Peters, F., Menzies, T., & Marcus, A. (2013, May). Better cross company defect prediction. In Mining Software Repositories (MSR), 2013 10th IEEE Working Conference on (pp. 409-418). IEEE.

Peters, F., Menzies, T., Gong, L., & Zhang, H. (2013). Balancing privacy and utility in cross-company defect prediction. Software Engineering, IEEE Transactions on,

Port, D., Nikora, A., Hayes, J. H., & Huang, L. (2011, January). Text mining support for software requirements: Traceability assurance. In System Sciences (HICSS), 2011 44th Hawaii International Conference on (pp. 1-11). IEEE.

Prakash, B. A., Ashoka, D. V., & Aradhya, V. M. (2013). An Evaluation of Neural Networks Approaches used for Software Effort Estimation. Proc of Int. Conference on Multimedia Processing

Premraj, R., & Herzig, K. (2011, September). Network versus code metrics to predict defects: A replication study. In Empirical Software Engineering and Measurement (ESEM), 2011 International Symposium on (pp. 215-224). IEEE.

Radlinski, L., & Hoffmann, W. (2010). On predicting software development effort using machine learning techniques and local data. International Journal of Software Engineering and Computing, 2(2), 123-136.

Radlinski, L., & Hoffmann, W. (2010). On predicting software development effort using machine learning techniques and local data. International Journal of Software Engineering and Computing, 2(2), 123-136.

Rana, Z. A., Malik, S. A., Shamail, S., & Awais, M. M. (2013, January). Identifying Association between Longer Itemsets and Software Defects. In Neural Information Processing (pp. 133-140). Springer Berlin Heidelberg.

Rodriguez, D., Herraiz Tabernero, I., & Harrison, R. (2012). On software engineering repositories and their open problems.

Rodríguez, D., Ruiz, R., Riquelme, J. C., & Aguilar–Ruiz, J. S. (2012). Searching for rules to detect defective modules: A subgroup discovery approach. Information Sciences, 191, 14-30.

Rodriguez, D., Ruiz, R., Riquelme, J. C., & Harrison, R. (2013). A study of subgroup discovery approaches for defect prediction. Information and Software Technology, 55(10), 1810-1822.

Rodríguez, D., Sicilia, M. A., García, E., & Harrison, R. (2012). Empirical findings on team size and productivity in software development. Journal of Systems and Software, 85(3), 562-570.

Sami, A., & Fakhrahmad, S. M. (2010, March). Design-level metrics estimation based on code metrics. In Proceedings of the 2010 ACM Symposium on Applied Computing (pp. 2531-2535). ACM.

Sankar, K., Kannan, S., & Jennifer, P. (2014). Prediction of Code Fault Using Naive Bayes and SVM Classifiers. Middle-East Journal of Scientific Research, 20(1), 108-113.

Sasidharan, R., & Sriram, P. (2014). Hyper-Quadtree-Based K-Means Algorithm for Software Fault Prediction. In Computational Intelligence, Cyber Security and Computational Models (pp. 107-118). Springer India.

Seliya, N., & Khoshgoftaar, T. M. (2011). The use of decision trees for costsensitive classification: an empirical study in software quality prediction. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 1(5), 448-459.

Seo, Y. S., Bae, D. H., & Jeffery, R. (2013). AREION: Software effort estimation based on multiple regressions with adaptive recursive data partitioning. Information and Software Technology, 55(10), 1710-1725.

Setiono, R., Dejaeger, K., Verbeke, W., Martens, D., & Baesens, B. (2010, October). Software effort prediction using regression rule extraction from neural networks. In Tools with Artificial Intelligence (ICTAI), 2010 22nd IEEE International Conference on (Vol. 2, pp. 45-52). IEEE.

Sharma, N., Bajpai, A., & Litoriya, M. R. (2012). Comparison the various clustering algorithms of weka tools. facilities, International Journal of Emerging Technology and Advanced Engineering, vol 2, Iss5, May 2012,4, 7.

Shepperd, M., Song, Q., Sun, Z., & Mair, C. (2013). Data quality: Some comments on the nasa software defect data sets. Software Engineering, IEEE Transactions on, vol 39, Iss 9

Shihab, E., Kamei, Y., Adams, B., & Hassan, A. E. (2013). Is lines of code a good measure of effort in effort-aware models?. Information and Software Technology, 55(11), 1981-1993.

Slankas, J., & Williams, L. (2013, May). Automated extraction of non-functional requirements in available documentation. In Natural Language Analysis in Software Engineering (NaturaLiSE), 2013 1st International Workshop on (pp. 9-16). IEEE.

Smith, M. R., & Martinez, T. (2011, July). Improving classification accuracy by identifying and removing instances that should be misclassified. In Neural Networks (IJCNN), The 2011 International Joint Conference on (pp. 2690-2697). IEEE.

Smith, M. R., & Martinez, T. (2013). An Extensive Evaluation of Filtering Misclassified Instances in Supervised Classification Tasks. arXiv preprint arXiv:1312.3970.

Smith, M. R., Martinez, T., & Giraud-Carrier, C. (2010). An empirical study of instance hardness (Doctoral dissertation, Brigham Young University. Department of Computer Science).

Smith, M. R., Martinez, T., & Giraud-Carrier, C. (2013). An instance level analysis of data complexity. Machine Learning, 1-32.

Song, Q., Jia, Z., Shepperd, M., Ying, S., & Liu, J. (2011). A general software defect-proneness prediction framework. Software Engineering, IEEE Transactions on, 37(3), 356-370.

Steff, M., & Russo, B. (2011, September). Measuring architectural change for defect estimation and localization. In Empirical Software Engineering and Measurement (ESEM), 2011 International Symposium on (pp. 225-234). IEEE.

Tierno, I. A., & Nunes, D. J. (2012, April). Assessment of Automatically Built Bayesian Networks in Software Effort Prediction. In CIbSE (pp. 196-209).

Tosun Misirli, A., Murphy, B., Zimmermann, T., & Basar Bener, A. (2011, September). An explanatory analysis on eclipse beta-release bugs through in-process metrics. In Proceedings of the 8th international workshop on Software quality (pp. 26-33). ACM.

Tosun, A., Bener, A. B., & Kale, R. (2010, July). AI-Based Software Defect Predictors: Applications and Benefits in a Case Study. In IAAI.

Tosun, A., Bener, A., Turhan, B., & Menzies, T. (2010). Practical considerations in deploying statistical methods for defect prediction: A case study within the Turkish telecommunications industry. Information and Software Technology, 52(11), 1242-1257.

Turhan, B., Tosun Mısırlı, A., & Bener, A. (2013). Empirical evaluation of the effects of mixed project data on learning defect predictors. Information and Software Technology, 55(6), 1101-1118.

Turhan, B., Tosun, A., & Bener, A. (2011, August). Empirical evaluation of mixed-project defect prediction models. In Software Engineering and Advanced Applications (SEAA), 2011 37th EUROMICRO Conference on (pp. 396-403). IEEE.

Twala, B., & Cartwright, M. (2010). Ensemble missing data techniques for software effort prediction. Intelligent Data Analysis, 14(3), 299-331.

Verma, R., & Gupta, A. (2012, September). An approach of attribute selection for reducing false alarms. In Software Engineering (CONSEG), 2012 CSI Sixth International Conference on (pp. 1-7). IEEE.

Vivanco, R., Kamei, Y., Monden, A., Matsumoto, K. I., & Jin, D. (2010, May). Using search-based metric selection and oversampling to predict fault prone modules. In Electrical and Computer Engineering (CCECE), 2010 23rd Canadian Conference on (pp. 1-6). IEEE.

Wang, H., Khoshgoftaar, T. M., & Liang, Q. (2013). A STUDY OF SOFTWARE METRIC SELECTION TECHNIQUES: STABILITY ANALYSIS AND DEFECT PREDICTION MODEL PERFORMANCE. International Journal on Artificial Intelligence Tools, 22(05).

Wang, H., Khoshgoftaar, T. M., Van Hulse, J., & Gao, K. (2011). Metric selection for software defect prediction. International Journal of Software Engineering and Knowledge Engineering, 21(02), 237-257.

Wang, H., Khoshgoftaar, T. M., Wald, R., & Napolitano, A. (2012, August). A novel dataset-similarity-aware approach for evaluating stability of software metric selection techniques. In Information Reuse and Integration (IRI), 2012 IEEE 13th International Conference on (pp. 1-8). IEEE.

Wang, S., & Yao, X. (2013). Using class imbalance learning for software defect prediction. Reliability, IEEE Transactions on,

WANG, S., MINKU, L. L., & YAO, X. (2013). ONLINE CLASS IMBALANCE LEARNING AND ITS APPLICATIONS IN FAULT DETECTION. International Journal of Computational Intelligence and Applications, 12(04).

Wang, T., Li, W., Shi, H., & Liu, Z. (2011). Software Defect Prediction Based on Classifiers Ensemble. Journal of Information & Computational Science, 8(16), 4241-4254.

Wang, X., Zhang, L., & Shi, Y. (2010, August). A Knowledge Discovery Case Study of Software Quality Prediction: ISBSG Database. In Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM International Conference on (Vol. 3, pp. 219-222). IEEE.

Wen, J., Li, S., Lin, Z., Hu, Y., & Huang, C. (2012). Systematic literature review of machine learning based software development effort estimation models. Information and Software Technology, 54(1), 41-59.

Wieloch, M., Amornborvornwong, S., & Cleland-Huang, J. (2013, May). Trace-by-classification: A machine learning approach to generate trace links for frequently occurring software artifacts. In Traceability in Emerging Forms of Software Engineering (TEFSE), 2013 International Workshop on (pp. 110-114). IEEE.

Xu, J., Ho, D., & Capretz, L. F. (2010). AN EMPIRICAL STUDY ON THE PROCEDURE TO DERIVE SOFTWARE QUALITY ESTIMATION MODELS. International Journal of Computer Science & Information Technology, 2(4).

Yu, L. (2012). Using Negative Binomial Regression Analysis to Predict Software Faults: A Study of Apache Ant. International Journal of Information Technology & Computer Science, 4(8).

Yu, L., & Mishra, A. (2012). Experience in predicting fault-prone software modules using complexity metrics. Quality Technology & Quantitative Management, 9(4), 421-433.

Zafar, H., Rana, Z., Shamail, S., & Awais, M. M. (2012, December). Finding focused itemsets from software defect data. In Multitopic Conference (INMIC), 2012 15th International (pp. 418-423). IEEE.

Zhang, H., Nelson, A., & Menzies, T. (2010, September). On the value of learning from defect dense components for software defect prediction. In Proceedings of the 6th International Conference on Predictive Models in Software Engineering (p. 14). ACM.

Authors' Information

Ana Maria Bautista *– E.T.S. Ingenieros Informáticos. Universidad Politécnica de Madrid, Madrid; e-mail: am.bautista@alumnos.upm.es*

Major Fields of Scientific Research: Artificial Intelligence

Tomás San Feliu *– E.T.S. Ingenieros Informáticos. Universidad Politécnica de Madrid, Madrid; e-mail: tomas.sanfeliu@upm.es*

Major Fields of Scientific Research: Software Engineering, Computer Science