Systematic Literature Review:
Pengantar, Tahapan dan Studi Kasus

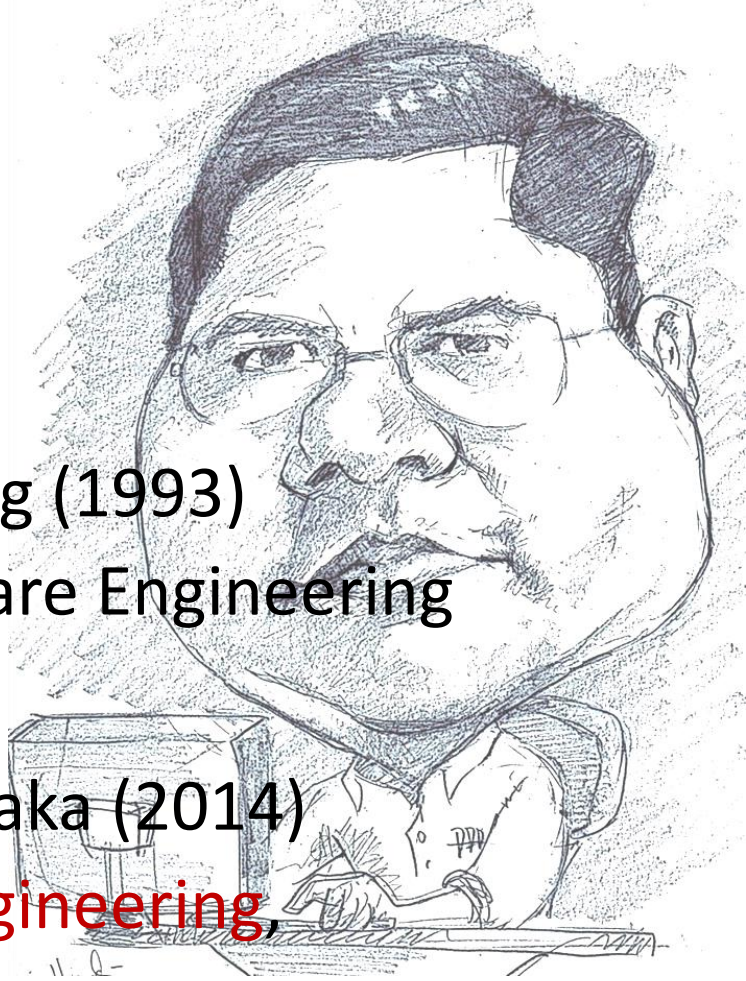
Romi Satria Wahono

romi@romisatriawahono.net

http://romisatriawahono.net

+6281586220090

Romi Satria Wahono



- SD Sompok Semarang (1987)
- SMPN 8 Semarang (1990)
- SMA Taruna Nusantara Magelang (1993)
- B.Eng, M.Eng and Ph.D in Software Engineering from
  Saitama University Japan (2004)
  Universiti Teknikal Malaysia Melaka (2014)
- Research Interests: Software Engineering, Machine Learning
- Founder dan Koordinator IlmuKomputer.Com
- Peneliti LIPI (2004-2007)
- Founder dan CEO PT Brainmatics Cipta Informatika

RomiSatriaWahono.Net

Lecture Notes in Software Engineering, Computing Research and Technopreneurship

HOME ABOUT ME IN THE NEWS LECTURES PUBLICATIONS RESEARCH

BAGAIMANA MELAKUKAN PENELITIAN YANG BAIK?

Pada artikel ini, saya mencoba merangkumkan tahapan melakukan penelitian yang ditulis oleh Prof Bochman. Tulisan pendek berjudul "How to do Good Research" ini, sebenarnya tidak terlalu jauh berbeda dengan artikel yang saya tulis di blog ini tentang, Tahapan Memulai Penelitian

TAHAPAN PENELITIAN DENGAN FOKUS PERBAIKAN METODE

Di artikel sebelumnya, saya sudah menjelaskan secara komprehensif tentang Tahapan Memulai Penelitian untuk

MIND MAP UNTUK MEMAHAMI TOPIK PENELITIAN

Satu hal penting yang biasanya dilupakan mahasiswa ketika melakukan penelitian adalah, memahami secara

KONTRIBUSI PENELITIAN DAN PERBAIKAN METODE

MENGAPA KONTRIBUSI PENTING DALAM PENELITIAN? Banyak mahasiswa, yang sedang melakukan penelitian untuk

METODE MENGELOLA PENELITIAN TESIS MAHASISWA

PROBLEMS AND REQUIREMENTS Semakin banyaknya jumlah mahasiswa bimbingan, membuat saya harus sedikit

TAHAPAN MEMULAI PENELITIAN UNTUK MAHASISWA GALAU

Jujur, secara umum saya agak kecewa dengan pertanyaan mahasiswa tingkat akhir yang masuk lewat email, inbox FB dan

CIYUS, CUMPAH, NGEBLOG ITU WOW BANGET!

27 Oktober, hari blogger! Lha kok sepi? Ya sudah banyak blogger yang lupa dengan hari jadinya, termasuk saya

MENUJU KEBEBASAN YANG MEMBEBASKAN

Sebuah essay kecil yang saya susun untuk para mahasiswa dan generasi muda, khususnya yang bergerak di bidang

KIAT MENYUSUN KERANGKA PEMIKIRAN PENELITIAN

Kerangka pemikiran adalah suatu diagram yang menjelaskan secara garis besar alur logika berjalannya sebuah

5 KARAKTER PARA INOVATOR

Menarik membaca buku yang ditulis oleh Carmine Gallo berjudul Rahasia Inovasi Steve Jobs (The Innovation Secrets of

KIAT MENYUSUN ALUR LATAR BELAKANG MASALAH PENELITIAN

Latar belakang masalah penelitian (research background) adalah bagian pertama dan sangat penting

SHARE THIS

LECTURES

Mata kuliah yang saya ajar di berbagai universitas di Indonesia. Seluruh materi kuliah bisa diunduh dan digunakan dengan bebas. Setiap halaman mata kuliah memuat course description, standard competency, slide, software requirements, dan textbook yang digunakan.

Computing Courses	• Research Methodology (updated January 2015) • Data Mining (updated January 2015) • Theory of Computation (updated March 2015)
Programming Courses	• Java Fundamentals (updated October 2013) • Java Enterprise Edition
Software Engineering Courses	• Systems Analysis and Design (updated January 2015) • Business Process Model and Notation (updated January 2015) • Software Engineering • Software Testing • Software Quality Assurance • Project Management
Enterprise Architecture Courses	• TOGAF 9.1 Fundamental • TOGAF 9.1 Foundation • TOGAF 9.1 Certified

YouTube ID

Upload

189 subscribers | 4,535 views | Video Manager | View as public

Romi Satria Wahono

Home | **Videos** | Playlists | Channels | Discussion | About

Uploads

Date added (newest - oldest)

Menjadi Programmer Technopreneur

116 views • 3 weeks ago

Kuliah 10 Menit tentang Enterprise Architecture

816 views • 3 weeks ago

Kuliah 20 Menit tentang Metodologi Penelitian

1,756 views • 4 weeks ago

Kuliah 10 Menit tentang Data Mining

1,898 views • 1 month ago

Forum Diskusi (Intelligent System Research Center)

http://facebook.com/groups/intelligentsystems/

Contents

1. Pengantar Systematic Literature Review (SLR)

Literature Review

- Literature Review is a critical and in depth evaluation of previous research (Shuttleworth, 2009) (*https://explorable.com/what-is-a-literature-review*)
- A summary and synopsis of a particular area of research, allowing anybody reading the paper to establish the reasons for pursuing a particular research
- A good Literature Review evaluates quality and findings of previous research

Manfaat Mereview Literatur

- Memperdalam pengetahuan tentang bidang yang diteliti (*Textbooks*)
- Mengetahui hasil penelitian yang berhubungan dan yang sudah pernah dilaksanakan (*Related Research*) (*Paper*)
- Mengetahui perkembangan ilmu pada bidang yang kita pilih (state-of-the-art) (*Paper*)
- Memperjelas masalah penelitian (*Paper*)

Metode Literature Review

- Types and Methods of Literature Review:
  1. Traditional Review
  2. Systematic Mapping Study (Scoping Study)
  3. Systematic Literature Review or Systematic Review
  4. Tertiary Study

- SLR is now well established review method in the field of software engineering

(Kitchenham & Charters, Guidelines in performing Systematic Literature Reviews in Software Engineering, EBSE Technical Report version 2.3, 2007)

1. Traditional Review

- Provides an overview of the research findings on particular topics
- Advantages: produce insightful, valid syntheses of the research literature if conducted by the expert
- Disadvantages: vulnerable to unintentional and intentional bias in the selection, interpretation and organization of content
- Examples:
  - Liao et al., Intrusion Detection System: A Comprehensive Review, Journal of Network and Computer Applications, 36(2013)
  - Galar et al., A Review on Ensembles for the Class Imbalance Problem: Bagging-, Boosting-, and Hybrid-Based Approaches, IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), Vol. 42, No. 4, July 2012
  - Cagatay Catal, Software fault prediction: A literature review and current trends, Expert Systems with Applications 38 (2011)

2. Systematic Mapping Study

- Suitable for a very broad topic
- Identify clusters of evidence (making classification)
- Direct the focus of future SLRs
- To identify areas for future primary studies
- Examples:
  - Neto et al., A systematic mapping study of software product lines testing, Information and Software Technology Vol. 53, Issue 5, May 2011
  - Elberzhager et al., Reducing test effort: A systematic mapping study on existing approaches, Information and Software Technology 54 (2012)

3. Systematic Literature Review (SLR)

- The purpose of a systematic literature reviews is to provide as complete a list as possible of all the published studies relating to a particular subject area
- A process of identifying, assessing, and interpreting all available research evidence, to provide answers for a particular research question
- A form of secondary study that uses a well-defined methodology
- SLRs are well established in other disciplines, particularly medicine. They integrate an individual clinical expertise and facilitate access to the outcomes of the research

(Kitchenham & Charters, Guidelines in performing Systematic Literature Reviews in Software Engineering, EBSE Technical Report version 2.3, 2007)

Examples of SLR

- Hall et al., A Systematic Literature Review on Fault Prediction Performance in Software Engineering, IEEE Transaction on Software Engineering, Vol. 38, No. 6, 2012
- Romi Satria Wahono, A Systematic Literature Review of Software Defect Prediction: Research Trends, Datasets, Methods and Frameworks, Journal of Software Engineering, Vol. 1, No. 1, April 2015
- Jianfeng Wen, Shixian Li, Zhiyong Lin, Yong Hu, Changqin Huang, Systematic literature review of machine learning based software development effort estimation models, Information and Software Technology 54 (2012) 41–59

4. Tertiary study

- Is a SLR of SLRs
- To answer a more wider question
- Uses the same method as in SLR
- Potentially less resource intensive
- Examples:
  - Kitchenham et al., Systematic literature reviews in software engineering – A tertiary study, Information and Software Technology 52 (2010)
  - Cruzes et al., Research synthesis in software engineering: A tertiary study, Information and Software Technology 53 (2011)

2. Tahapan Systematic Literature Review (SLR)

Tahapan SLR

1. Formulate the Review's Research Question
2. Develop the Review's Protocol

2.1 PLANNING

1. Identify the Relevant Literature
2. Perform Selection of Primary Studies
3. Perform Data Extraction
4. Assess Studies' Quality
5. Conduct Synthesis of Evidence

2.2 CONDUCTING

1. Write Up the SLR Paper
2. Choose the Right Journal

2.3 REPORTING

2.1 Tahapan Planning

1. Formulate the Review's Research Question
2. Develop the Review's Protocol

1. Formulate the Review's Research Question

- Features of good question:
  - The RQ is meaningful and important to practitioners and researchers
  - The RQ will lead to changes in current practice or to increase confidence in the value of current practice
  - The RQ will identify discrepancies between commonly held beliefs and the reality

- RQ can be derived primarily based on researcher's interest
  - An SLR for PhD thesis should identify existing basis for the research work and where it fits in the current body of knowledge

The Research Question (RQ)

- Is the most important part in any SLR
- Is not necessarily the same as questions addressed in your research
- Is used to guide the search process
- Is used to guide the extraction process
- Data analysis (synthesis of evidence) is expected to answer your SLR's RQ

RQ and PICOC

The formulation of RQs about effectiveness of a treatment should focus on 5 elements known as PICOC:

1. **Population (P)** - the target group for the investigation (e.g. people, software etc.)
2. **Intervention (I)** - specifies the investigation aspects or issues of interest to the researchers
3. **Comparison (C)**– aspect of the investigation with which the intervention is being compared to
4. **Outcomes (O)**– the effect of the intervention
5. **Context (C)**– the setting or environment of the investigation

(Petticrew et al., Systematic Reviews in the Social Sciences: A Practical Guide, Blackwell Publishing, 2006)

Example of PICOC (Kitchenham et al., 2007)

Kitchenham et al., A Systematic Review of Cross- vs. Within-Company Cost Estimation Studies, *IEEE Transactions on Software Engineering*, 33 (5), 2007

Population:	Software or web project
Intervention:	Cross-company project effort estimation model
Comparison:	Single-company project effort estimation model
Outcomes:	Prediction or estimate accuracy
Context:	None

Example of PICOC (Wahono, 2015)

Romi Satria Wahono, A Systematic Literature Review of Software Defect Prediction: Research Trends, Datasets, Methods and Frameworks, *Journal of Software Engineering, Vol. 1, No. 1*, pp. 1-16, April 2015

Population	Software, software application, software system, information system
Intervention	Software defect prediction, fault prediction, error-prone, detection, classification, estimation, models, methods, techniques, datasets
Comparison	n/a
Outcomes	Prediction accuracy of software defect, successful defect prediction methods
Context	Studies in industry and academia, small and large data sets

Example of RQs (Kitchenham, 2007)

Kitchenham et al., A Systematic Review of Cross- vs. Within-Company Cost Estimation Studies, *IEEE Transactions on Software Engineering*, 33 (5), 2007

- RQ1: What evidence is there that cross-company estimation models are not significantly different from within-company estimation models for predicting effort for software/Web projects?
- RQ2: What characteristics of the study data sets and the data analysis methods used in the study affect the outcome of within- and cross-company effort estimation accuracy studies?
- RQ3: Which experimental procedure is most appropriate for studies comparing within- and cross-company estimation models?

Example of RQs (Davis et al., 2006)

Davis et al., Effectiveness of Requirements Elicitation Techniques: Empirical Results Derived from a Systematic Review, *14th IEEE Requirements Engineering Conference*, 2006

- RQ: What elicitation technique is most efficient in a particular setting?

Example of RQs (Radjenovic et al., 2013)

Radjenovic et al., Software fault prediction metrics: A systematic literature review, *Information and Software Technology,* Vol. 8, No. 55, pp. 1397-1418, 2013

- RQ1: Which software metrics for fault prediction exist in literature?
- RQ2: What data sets are used for evaluating metrics?

Example of RQ (Wahono, 2015)

Romi Satria Wahono, A Systematic Literature Review of Software Defect Prediction: Research Trends, Datasets, Methods and Frameworks, *Journal of Software Engineering, Vol. 1, No. 1*, pp. 1-16, April 2015

ID	Research Question
RQ1	Which journal is the most significant software defect prediction journal?
RQ2	Who are the most active and influential researchers in the software defect prediction field?
RQ3	What kind of research topics are selected by researchers in the software defect prediction field?
RQ4	What kind of datasets are the most used for software defect prediction?
RQ5	What kind of methods are used for software defect prediction?
RQ6	What kind of methods are used most often for software defect prediction?
RQ7	Which method performs best when used for software defect prediction?
RQ8	What kind of method improvements are proposed for software defect prediction?
RQ9	What kind of frameworks are proposed for software defect prediction?

2. Develop the Review's Protocol

- A plan that specifies the basic review procedures (method)
- Components of a protocol:
  1. Background
  2. Research Questions
  3. Search terms
  4. Selection criteria
  5. Quality checklist and procedures
  6. Data extraction strategy
  7. Data synthesis strategy

2.2 Tahapan Conducting

1. Identify the Relevant Literature
2. Perform Selection of Primary Studies
3. Perform Data Extraction
4. Assess Studies' Quality
5. Conduct Synthesis of Evidence

1. Identifying Relevant Literature

- Involves a comprehensive and exhaustive searching of studies to be included in the review
- Define a search strategy
- Search strategies are usually iterative and benefit from:
  - Preliminary searches (to identify existing review and volume of studies)
  - Trial searches (combination of terms from RQ)
  - Check the search results against list of known studies
  - Consult the experts in the field

Approach to Construct Search String

- Derive major terms used in the review questions based on the PICOC
- List the keywords mentioned in the article
- Search for synonyms and alternative words
- Use the boolean OR to incorporate alternative synonyms
- Use the boolean AND to link major terms

Example of Search String (Kitchenham et al., 2007)

- Kitchenham et al. (2007) used their structured questions to construct search strings for use with electronic databases:
  - *Population*: software OR application OR product OR Web OR WWW OR Internet OR World-Wide Web OR project OR development
  - *Intervention*: cross company OR cross organisation OR cross organization OR multiple-organizational OR multiple-organisational model OR modeling OR modelling effort OR cost OR resource estimation OR prediction OR assessment
  - *Contrast*: within-organisation OR within-organization OR within-organizational OR within-organisational OR single company OR single organisation
  - *Outcome*: Accuracy OR Mean Magnitude Relative Error
- The search strings were constructed by linking the four OR lists using the Boolean AND

Example of Search String (Wahono, 2015)

Romi Satria Wahono, A Systematic Literature Review of Software Defect Prediction: Research Trends, Datasets, Methods and Frameworks, *Journal of Software Engineering, Vol. 1, No. 1*, pp. 1-16, April 2015

Search String:

(software OR applicati OR systems) AND (fault* OR defect* OR quality OR error-prone) AND (predict* OR prone* OR probability OR assess* OR detect* OR estimat* OR classificat*)*

Example of Search String (Salleh et al., 2011)

- The complete search term initially used :

  (student OR undergraduate*) AND (pair programming OR pair-programming) AND ((experiment* OR measurement OR evaluation OR assessment) AND (effective* OR efficient OR successful)*

- A very limited number of results retrieved when using the complete string, thus a much simpler string was derived.
- Subject librarian suggested to revise the search string:

"pair programming" OR "pair-programming"

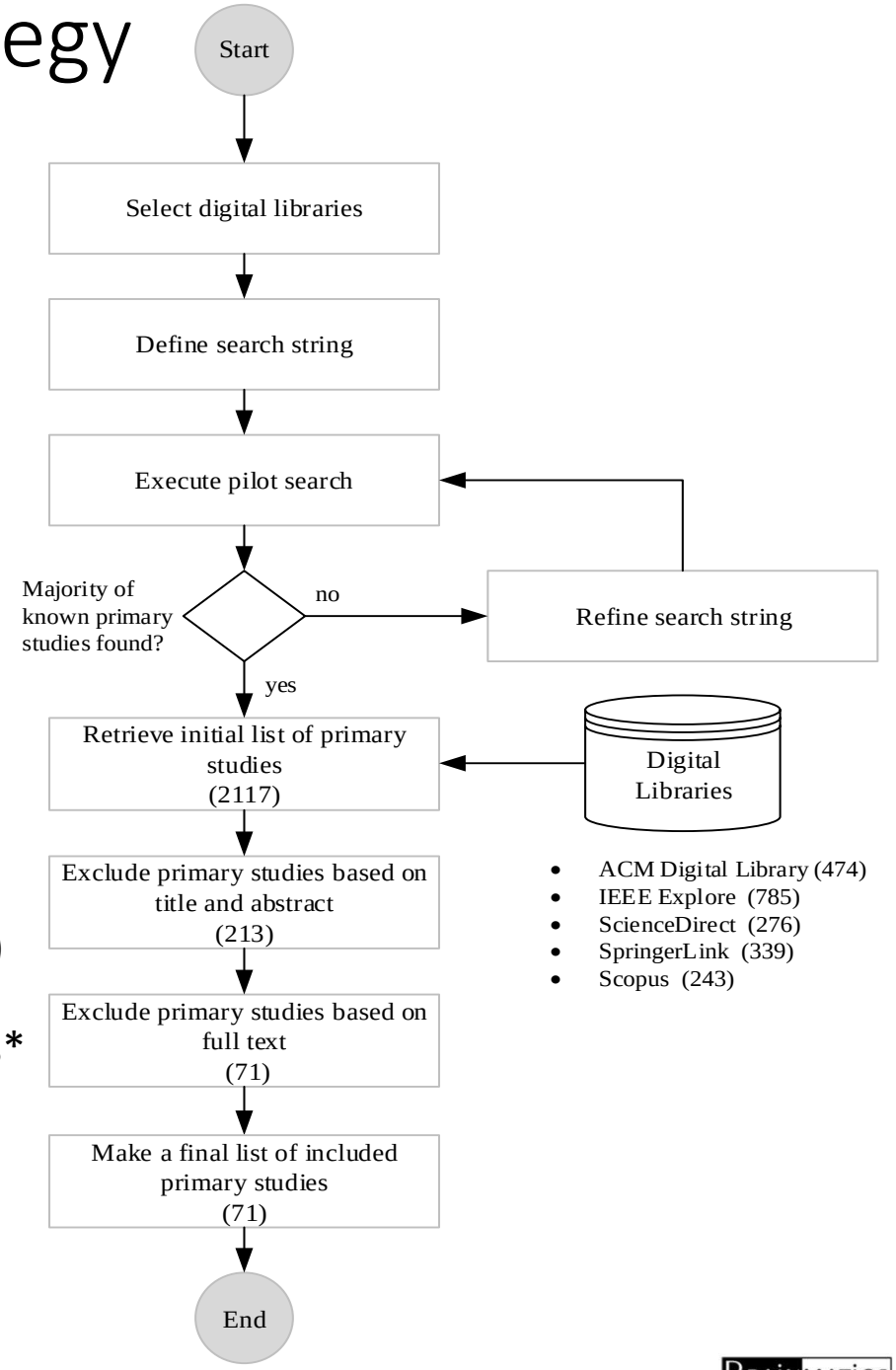
Sources of Evidence

- Digital libraries
- Reference lists from relevant primary studies and review articles
- Journals (including company journals such as the IBM Journal of Research and Development), grey literature (i.e. technical reports, work in progress)
- Conference proceedings
- Research registers
- The Internet (google)
- Direct contact specific researcher(s)

Studies Selection Strategy (Wahono, 2015)

- Publication Year:
  - ✓2000-2013
- Publication Type:
  - ✓Journal
  - ✓Conference Proceedings
- Search String:

  software
  AND
  (fault* OR defect* OR quality OR error-prone)
  AND
  (predict* OR prone* OR probability OR assess* OR detect* OR estimat* OR classificat*)
- Selected Studies:
  - ✓71



Sources of Evidence (Kitchenham et al., 2007)

- The search strings were used on 6 digital libraries:
  - INSPEC , El Compendex, Science Direct, Web of Science, IEEExplore, ACM Digital library
- Search specific journals and conf. proceedings:
  - Empirical Software Engineering (J)
  - Information and Software Technology (J)
  - Software Process Improvement and Practice (J)
  - Management Science (J)
  - International Software Metrics Symposium (C)
  - International Conference on Software Engineering (C)
- Manual search:
  - Evaluation and Assessment in Software Engineering (C)
- Check references of each relevant article
- Contact researchers

Managing Bibliography

- Use relevant Bibliographic package to manage large number of references

- E.g. Mendeley, EndNote, Zotero, JabRef Reference Manager etc.

Documenting the Search

- The process of conducting SLR must be transparent and replicable
- The review should be documented in sufficient detail
- The search should be documented and changes noted
- Unfiltered search results should be saved for possible reanalysis

Data Source	Documentation
Digital Library	Name of Database, Search strategy, Date of search, years covered by search
Journal Hand Searches	Name of journal, Years searched
Conference proceedings	Title of proceedings/Name of conference, Journal name (if published as part of a journal)

2. Selection of Studies

- Primary studies need to be assessed for their actual relevance
- Set the criteria for including or excluding studies (decided earlier during protocol development, can be refined later)
- Inclusion & exclusion criteria should be based on RQ
- Selection process should be piloted
- Study selection is a multistage process

Selection of Studies (Kitchenham et al., 2007)

- Kitchenham et al. (2007) used the following inclusion criteria:
  - Any study that compared predictions of cross-company models with within-company models based on analysis of single company project data.

- They used the following exclusion criteria:
  - Studies where projects were only collected from a small number of different sources (e.g. 2 or 3 companies)
  - Studies where models derived from a within-company data set were compared with predictions from a general cost estimation model.

Selection of Studies (Wahono, 2015)

Inclusion Criteria	Studies in academic and industry using large and small scale data sets
	Studies discussing and comparing modeling performance in the area of software defect prediction
	For studies that have both the conference and journal versions, only the journal version will be included
	For duplicate publications of the same study, only the most complete and newest one will be included
Exclusion Criteria	Studies without a strong validation or including experimental results of software defect prediction
	Studies discussing defect prediction datasets, methods, frameworks in a context other than software defect prediction
	Studies not written in English

Selection of Studies (*Salleh et al., 2011)*

- **Inclusion criteria**:
  - to include any empirical studies of PP that involved higher education students as the population of interest.
- **Exclusion criteria**:
  - Papers presenting unsubstantiated claims made by the author(s), for which no evidence was available.
  - Papers about Agile/XP describing development practices other than PP, such as test-first programming, refactoring etc.
  - Papers that only described tools (software or hardware) that could support the PP practice.
  - Papers not written in English.
  - Papers involving students but outside higher education

3. Assessing Studies' Quality

- To provide more detailed Inclusion/Exclusion criteria
- To check whether quality differences provide an explanation for differences in study results
- As a means of weighting the importance of individual studies when results are being synthesized
- To guide the interpretation of findings and determine the strength of inferences
- To guide recommendations for further research

Assessing Studies' Quality

- Quality relates to the extent to which the study minimizes bias and maximizes internal and external validity (Khan et al. 2001)
- Quality Concepts Definition (Kitchenham & Charter, 2007)

Terms	Synonyms	Definition
Bias	Systematic error	tendency to produce results that depart systematically from the 'true' results. Unbiased results are internally valid
Internal Validity	Validity	The extent to which the design and conduct of the study are likely to prevent systematic error. Internal validity is a prerequisite for external validity
External Validity	Generalizability, Applicability	The extent to which the effects observed in the study are applicable outside of the study

Assessing Studies' Quality

- Assessing quality of studies:
  - Methodology or design of the study
  - Analysis of studies' findings
- Quality checklist or instrument need to be designed to facilitate quality assessment
- Most quality checklists include questions aimed at assessing the extent to which articles have addressed bias and validity

Study Quality Assessment (Salleh et al., 2011)

Item	Answer
1. Was the article referred? [30]	Yes/No
2. Were the aim(s) of the study clearly stated? [16], [67]	Yes/No/Partially
3. Were the study participants or observational units adequately described? For example, students' programming experience, year of study etc. [44], [68]	Yes/No/Partially
4. Were the data collections carried out very well? For example, discussion of procedures used for collection, and how the study setting may have influenced the data collected [44], [48], [67], [68]	Yes/No/Partially
5. Were potential confounders adequately controlled for in the analysis? 67]	Yes/No/Partially
6. Were the approach to and formulation of the analysis well conveyed? For example, description of the form of the original data, rationale for choice of method/tool/package [48], [67], [68]	Yes/No/Partially
7. Were the findings credible? For example, the study was methodologically explained so that we can trust the findings; findings/conclusions are resonant with other knowledge and experience [48], [44], [68]	Yes/No/Partially

Study Quality Assessment (Kitchenham et al., 2007)

Kitchenham et al. (2007) constructed a quality questionnaire based on 5 issues affecting the quality of the study:

1. Is the data analysis process appropriate?
2. Did studies carry out a sensitivity or residual analysis?
3. Were accuracy statistics based on the raw data scale?
4. How good was the study comparison method?
5. The size of the within-company data set (e.g < 10 projects considered poor quality)

4. Data Extraction

- Involve reading the full text article
- Data extracted from primary studies should be recorded using *data extraction form*
- The form should be designed and piloted when the protocol is defined
- Collect all the information that can be used to answer the RQ and the study's quality criteria
- Both quality checklist and review data can be included in the same form
- In case of duplicates publications (reporting the same data), refer the most complete one
- For validation, a set of papers should be reviewed by 2 or more researchers. Compare results and resolve any conflicts

5. Synthesis of Evidence

- Involves collating and summarizing the results of the included primary studies
- Key objectives of data synthesis (Cruzes & Dyba, 2011):
  - to analyze and evaluate multiple studies
  - to select appropriate methods for integrating or providing new interpretive explanations about them
- Synthesis can be:
  - Descriptive (narrative/non-quantitative)
  - Quantitative (e.g. meta-analysis)

(Cruzes et al., Research Synthesis in Software Engineering: A tertiary study, *Information and Software Technology*, 53(5), 2011)

Descriptive Synthesis (Narrative)

"An approach to the synthesis of findings from multiple studies that relies primarily on the use of words and text to summarize and explain the findings of the synthesis. It adopts a textual approach to the process of synthesis to 'tell the story' of the findings from the included studies." (Popay et al. 2006)

- Use tables to tabulate information extracted from included studies (e.g. population, number of included studies, study quality etc.)
- Tables should be structured to highlight similarity or differences of study outcomes
- Were the findings consistent (homogeneous) or inconsistent?

Quantitative Synthesis (Meta-Analysis)

- Meta-analysis can be used to aggregate results or to pool data from different studies
- The outcome of a meta-analysis is an average effect size with an indication of how variable that effect size is between studies
- Meta-analysis involves three main steps:
  1. Decide which studies to be included in the meta-analysis
  2. Estimate an effect size for each individual study
  3. Combine the effect sizes from the individual studies to estimate and test the combined effect
- Results of the meta-analysis can be presented in a forest plot

2.3 Tahapan Reporting

1. Write Up the SLR Paper
2. Choose the Right Journal

1. Write Up the SLR Paper

1. Introduction
   - General introduction about the research. State the purpose of the review
   - Emphasize the reason(s) why the RQ is important
   - State the significance of the review work and how the project contributes to the body of knowledge of the field
2. Main Body
   1. Review method – briefly describe steps taken to conduct the review
   2. Results – findings from the review
   3. Discussion – implication of review for research & practice
3. Conclusions

2. Choose the Right Journal

- Some journals and conferences include a specific topic on SLR:
  - Information & Software Technology has an editor specializing in systematic reviews
  - Journal of Systems and Software
  - Expert Systems with Applications
  - IEEE Transactions on Software Engineering
  - International Symposium on Empirical Software Engineering & Measurement (ESEM)
  - International Conference on Evaluation & Assessment in Software Engineering (EASE)
  - International Workshop on Evidential Assessment of Software Technologies (EAST)

Listing Jurnal Tujuan dan Nilai SJR/JIF

- Lakukan pendataan journal-journal yang ada di topik SLR yang kita tulis, urutkan berdasarkan rangking SJR atau JIF
- Publikasikan paper SLR kita ke journal yang sesuai dengan kualitas SLR yang kita lakukan
- A paper is an organized description of hypotheses, data and conclusions, intended to instruct the reader. If your research does not generate papers, it might just as well not have been done (Whitesides 2004)

No	Journal Publications	SJR	Q Category
1	IEEE Transactions on Software Engineering	3.39	Q1 in Software
2	Information Sciences	2.96	Q1 in Information Systems
3	IEEE Transactions on Systems, Man, and Cybernetics	2.76	Q1 in Artificial Intelligence
4	IEEE Transactions on Knowledge and Data Engineering	2.68	Q1 in Information Systems
5	Empirical Software Engineering	2.32	Q1 in Software
6	Information and Software Technology	1.95	Q1 in Information Systems
7	Automated Software Engineering	1.78	Q1 in Software
8	IEEE Transactions on Reliability	1.43	Q1 in Software
9	Expert Systems with Applications	1.36	Q2 in Computer Science
10	Journal of Systems and Software	1.09	Q2 in Software
11	Software Quality Journal	0.83	Q2 in Software
12	IET Software	0.55	Q2 in Software
13	Advanced Science Letters	0.24	Q3 in Computer Science
14	Journal of Software	0.23	Q3 in Software
15	International Journal of Software Engineering and Its Application	0.14	Q4 in Software

3. Studi Kasus Systematic Literature Review

Romi Satria Wahono, A Systematic Literature Review of Software Defect Prediction: Research Trends, Datasets, Methods and Frameworks, *Journal of Software Engineering, Vol. 1, No. 1*, pp. 1-16, April 2015

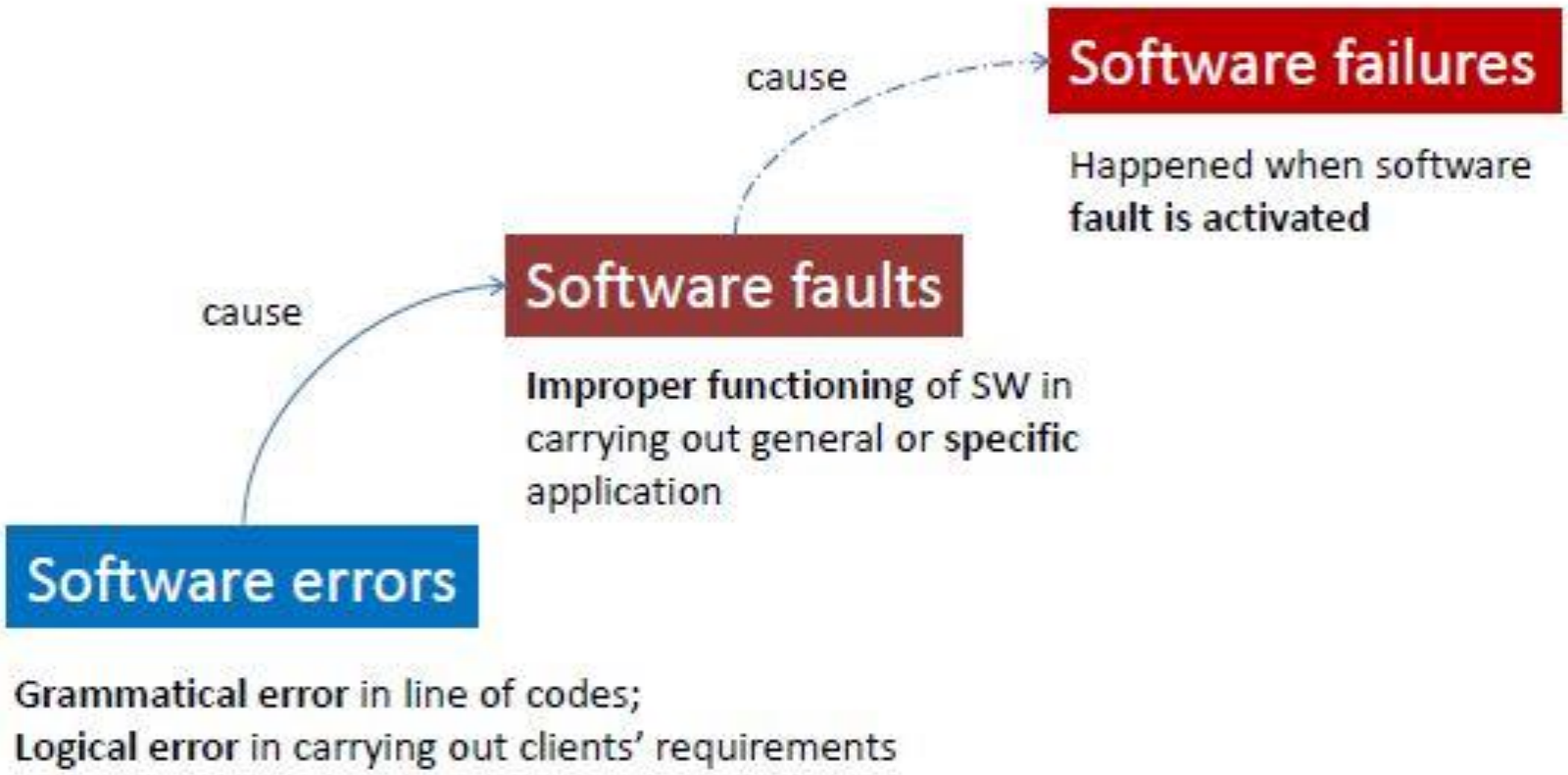
http://journal.ilmukomputer.org/index.php/jse/article/view/47

3.1 Introduction

Keunikan dari Software

Karakteristik	Software	Hardware
Kompleksitas	Tingkat kompleksitas dari produk software tinggi, dengan kemungkinan perubahan parameter dan fungsi yang sangat beragam	Tingkat kompleksitas produk lain rendah, dengan kemungkinan perubahan parameter dan fungsi tidak beragam
Visibilitas Produk	Produk tidak terlihat dengan kasat mata, termasuk bila ada cacat (defect) dari produk	Produk terlihat dengan kasat mata, termasuk bila ada cacat (defect) dari produk

Software Errors, Faults, Failures



Analisis Kasus

- Suatu perusahaan PT ABC memproduksi software yang akan ditanam ke dalam suatu device
- Salah satu fungsi yang terdapat pada software adalah akan mematikan device secara otomatis apabila suhu ruangan lebih besar daripada 30^o celcius
- Programmer salah menuliskan logika menjadi:

```
...
if (suhu > 3) shutdownDevice();
...
```

- Error ini tidak pernah menyebabkan failure pada software, dan perusahaan PT ABC sampai saat ini terkenal sebagai perusahaan yang memproduksi software tanpa bug
- Jelaskan mengapa bisa terjadi demikian!

Warranty Lawsuits

- Mortenson vs. Timeberline Software (TS) (≈1993)
  - Mortenson menggunakan software yang diproduksi TS untuk membuka tender pembangunan rumah sakit
  - Software memiliki bug sehingga memenangkan perusahaan yang mengajukan proposal paling mahal (kerugian 2 miliar USD)
  - TS tahu tentang bug itu, tapi tidak mengirimkan update ke Mortenson
  - Pengadilan di Amerika Serikat memenangkan perusahaan TS

- Uniform Computer Information Transaction Act (UCITA) allows software manufacturers to:
  - disclaim all liability for defects
  - prevent the transfer of software from person to person

Disclaimer of Warranties

DISCLAIMER OF WARRANTIES. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, MICROSOFT AND ITS SUPPLIERS PROVIDE TO YOU THE SOFTWARE COMPONENT, AND ANY (IF ANY) SUPPORT SERVICES RELATED TO THE SOFTWARE COMPONENT ("SUPPORT SERVICES") **AS IS AND WITH ALL FAULTS**; AND MICROSOFT AND ITS SUPPLIERS HEREBY DISCLAIM WITH RESPECT TO THE SOFTWARE COMPONENT AND SUPPORT SERVICES ALL WARRANTIES AND CONDITIONS, WHETHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING, BUT NOT LIMITED TO, ANY (IF ANY) WARRANTIES OR CONDITIONS OF OR RELATED TO: TITLE, NON-INFRINGEMENT, MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, LACK OF VIRUSES, ACCURACY OR COMPLETENESS OF RESPONSES, RESULTS, LACK OF NEGLIGENCE OR LACK OF WORKMANLIKE EFFORT, QUIET ENJOYMENT, QUIET POSSESSION, AND CORRESPONDENCE TO DESCRIPTION. **THE ENTIRE RISK ARISING OUT OF USE OR PERFORMANCE OF THE SOFTWARE COMPONENT AND ANY SUPPORT SERVICES REMAINS WITH YOU**.

Software Engineering Problem

Building software will always be hard. There is inherently no silver bullet *(Brooks, 1987)*

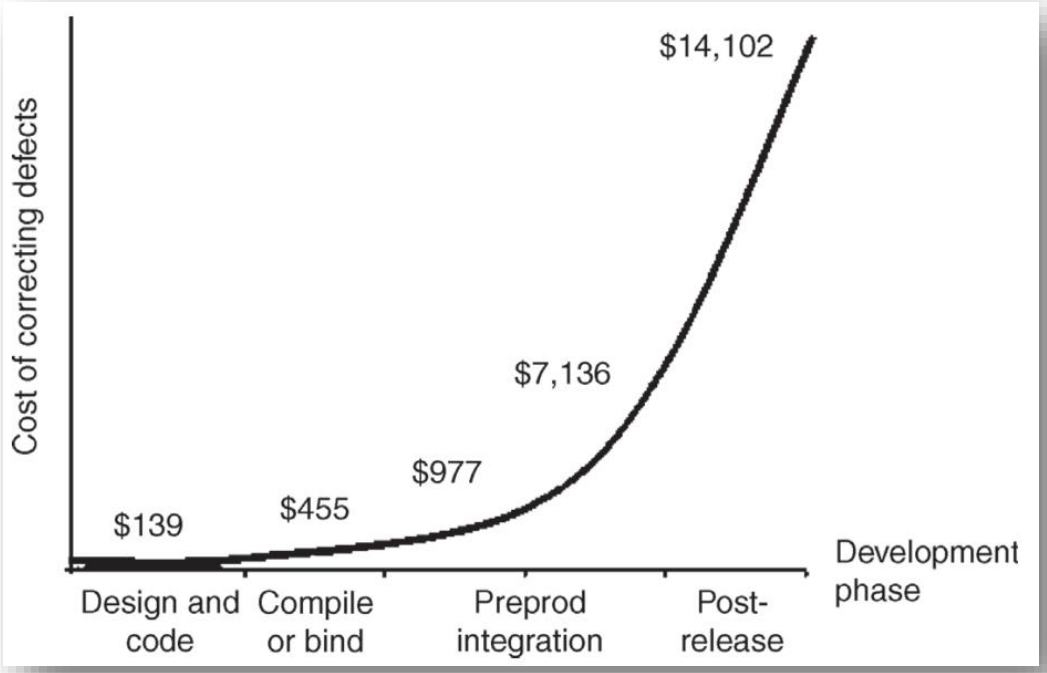
Software Defect?

Software defect is an error, failure, or fault in a software (Naik & Tripathy 2008) that produces an unexpected result (Hambling et al. 2008), and decreases the quality of the software (Lewis 2009)

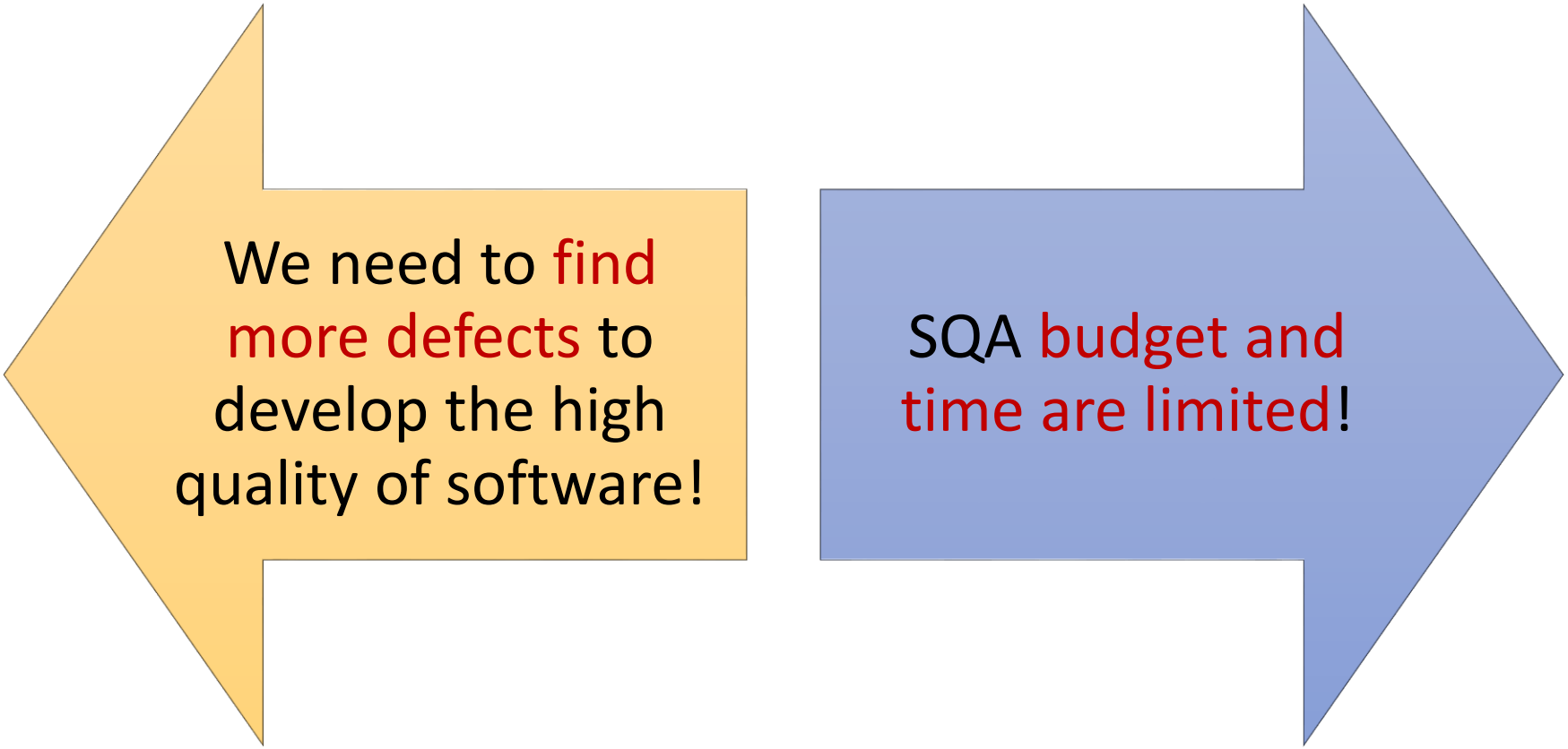
Why Software Defect Prediction?

- The cost of capturing and correcting defects is expensive
  - ✓ The most expensive activities (Jones 2012)
  - ✓ $14,102 per defect in post-release phase (Boehm & Basili 2008)
  - ✓ $60 billion per year (NIST 2002)

- Industrial methods of manual software reviews activities can find only 60% of defects (Shull et al. 2002)



Why Software Defect Prediction?



Why Software Defect Prediction?

- When budget and time do not allow for complete testing of an entire system

  → prediction models can be used to focus the testing on parts of the system that seem defect-prone

- The probability of detection of software fault prediction models is higher (71%) than software reviews (60%) (Menzies et al. 2010)

  → more cost-effective

- The accurate prediction of defect-prone modules can:

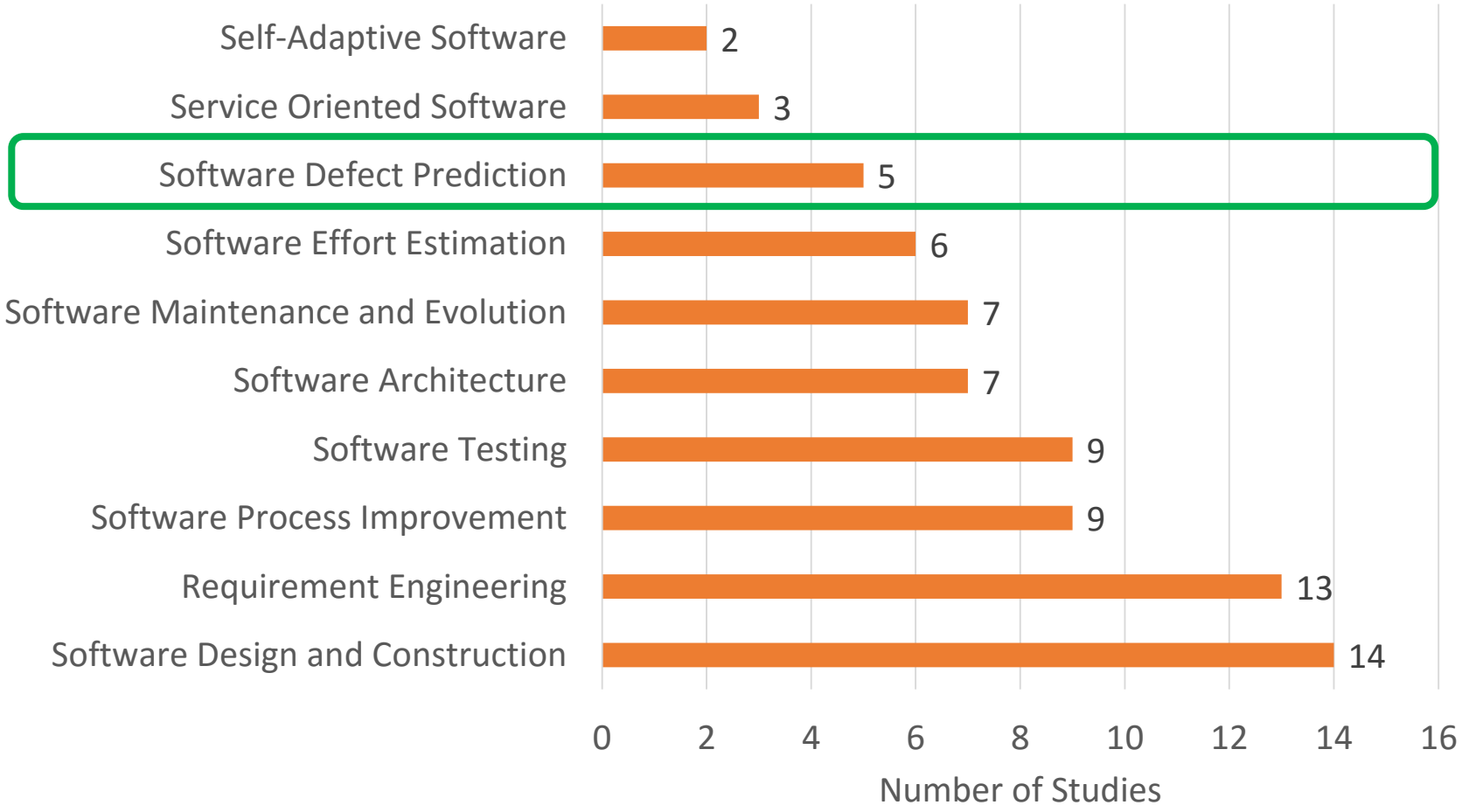
  ✓Reduce cost and improve the test effort by focusing on fault-prone modules that are predicted as fault-prone (Catal 2011)

  ✓Improve the quality of software (Hall et al. 2012)

Why Software Defect Prediction?

Software defect prediction has been an important research topic in the software engineering field (Hall et al. 2012) (Song et al. 2011)

Software Engineering Research Trends



** Resources: - Survey Papers from ScienceDirect, SpringerLink, and IEEE Explore*
- Publication Year: 2011-2014

3.2. Literature Review Methods

SLR Protocol

- A plan that specifies the basic review procedures (method)
- Components of a protocol:
  1. Background
  2. Research Questions
  3. Search terms
  4. Selection criteria
  5. Quality checklist and procedures
  6. Data extraction strategy
  7. Data synthesis strategy

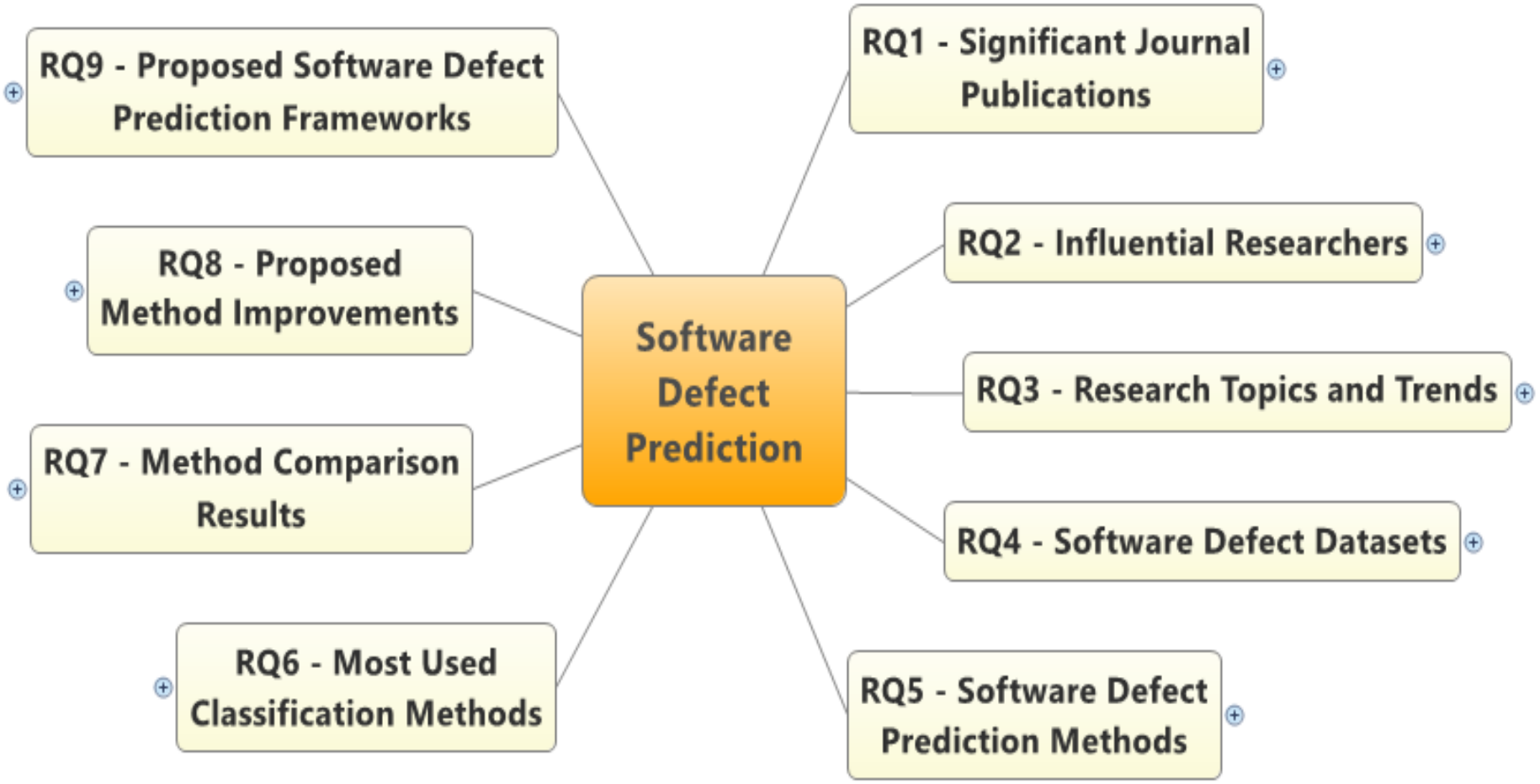
PICOC

Population	Software, software application, software system, information system
Intervention	Software defect prediction, fault prediction, error-prone, detection, classification, estimation, models, methods, techniques, datasets
Comparison	n/a
Outcomes	Prediction accuracy of software defect, successful defect prediction methods
Context	Studies in industry and academia, small and large data sets

Research Question (RQ)

ID	Research Question
RQ1	Which journal is the most significant software defect prediction journal?
RQ2	Who are the most active and influential researchers in the software defect prediction field?
RQ3	What kind of research topics are selected by researchers in the software defect prediction field?
RQ4	What kind of datasets are the most used for software defect prediction?
RQ5	What kind of methods are used for software defect prediction?
RQ6	What kind of methods are used most often for software defect prediction?
RQ7	Which method performs best when used for software defect prediction?
RQ8	What kind of method improvements are proposed for software defect prediction?
RQ9	What kind of frameworks are proposed for software defect prediction?

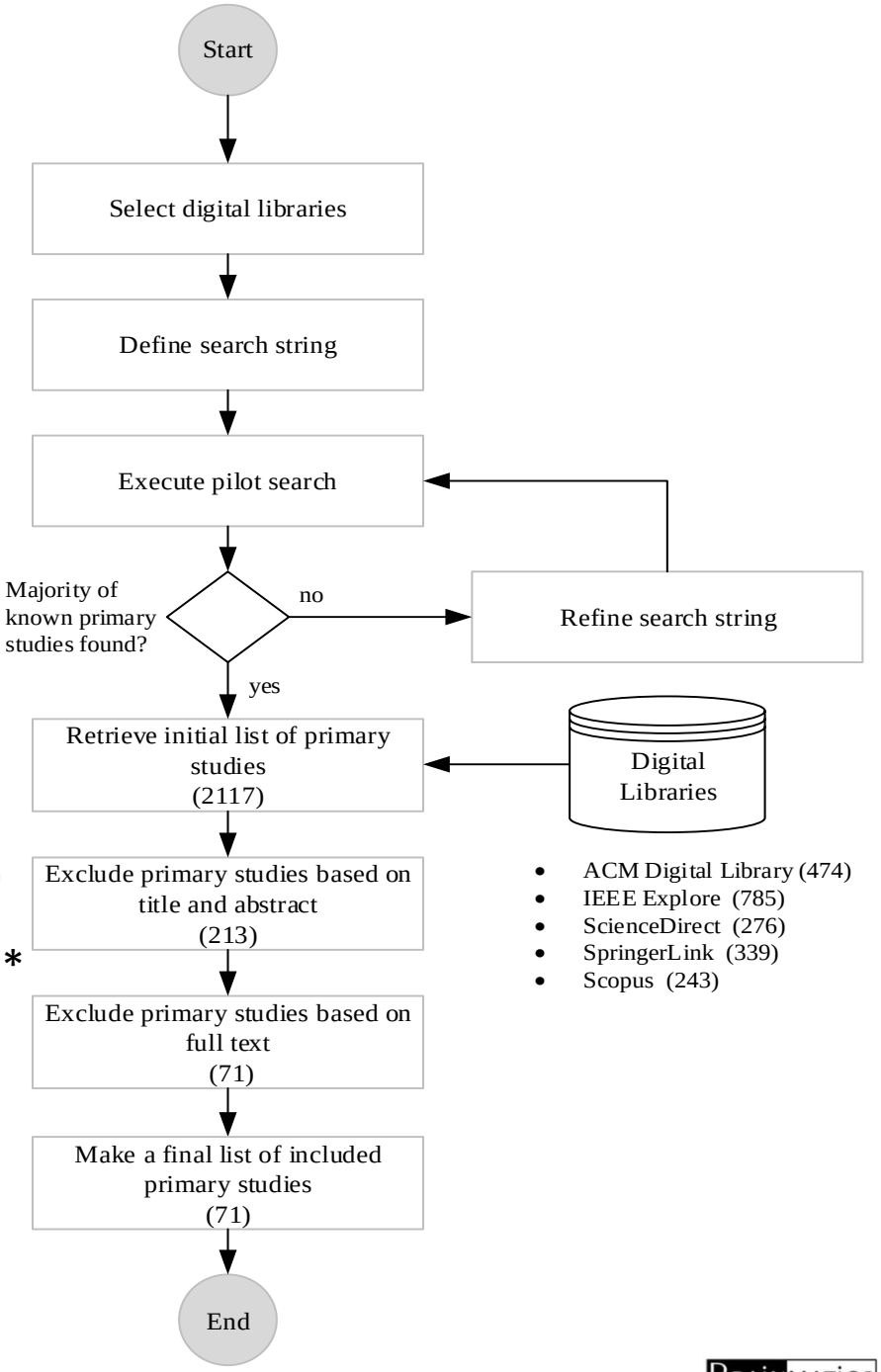
Research Question (RQ)



Studies Selection Strategy

- Publication Year:
  - ✓2000-2013
- Publication Type:
  - ✓Journal
  - ✓Conference Proceedings
- Search String:

  software
  AND
  (fault* OR defect* OR quality OR error-prone)
  AND
  (predict* OR prone* OR probability OR assess* OR detect* OR estimat* OR classificat*)
- Selected Studies:
  - ✓71



Inclusion and Exclusion Criteria

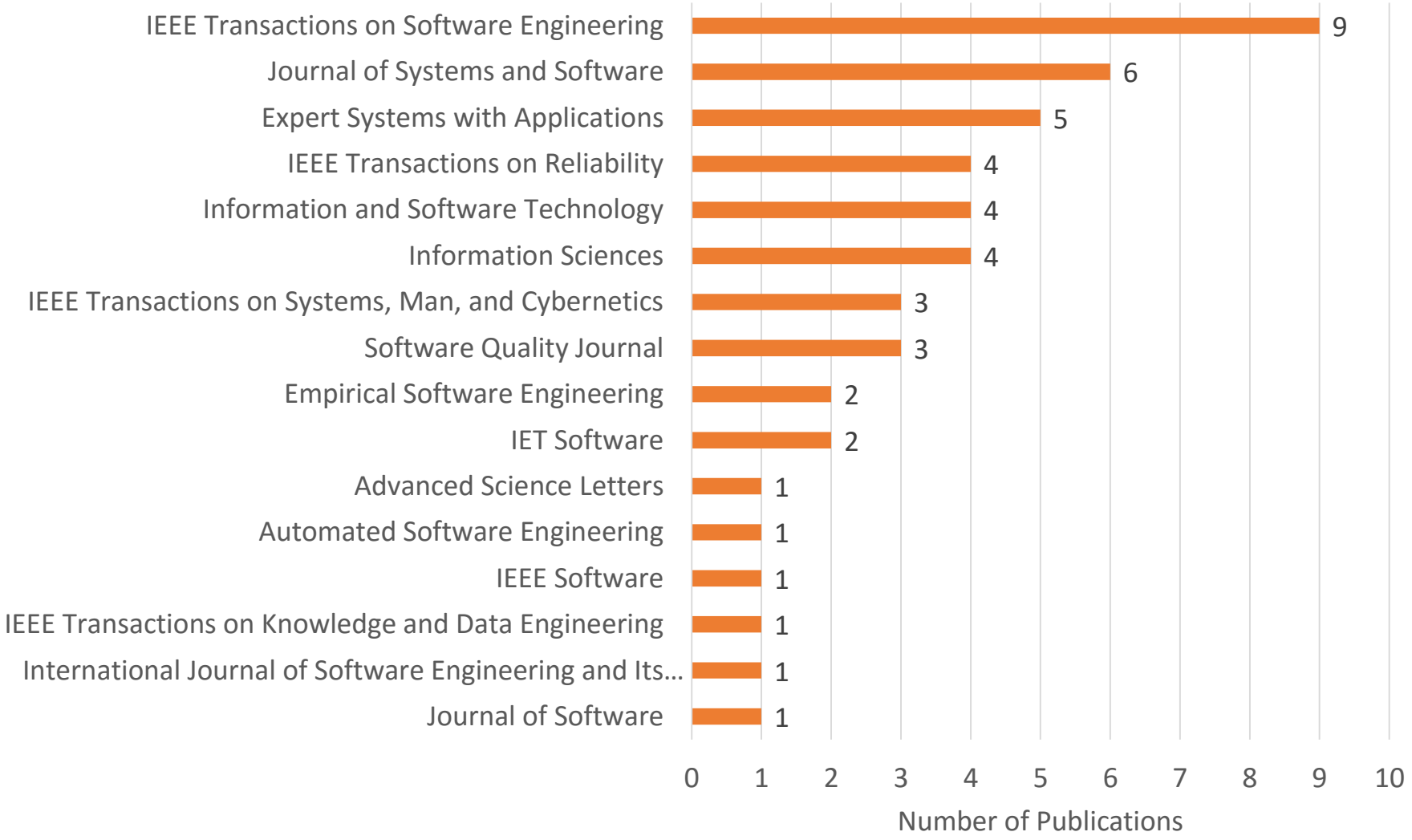
Inclusion Criteria	Studies in academic and industry using large and small scale data sets
	Studies discussing and comparing modeling performance in the area of software defect prediction
	For studies that have both the conference and journal versions, only the journal version will be included
	For duplicate publications of the same study, only the most complete and newest one will be included
Exclusion Criteria	Studies without a strong validation or including experimental results of software defect prediction
	Studies discussing defect prediction datasets, methods, frameworks in a context other than software defect prediction
	Studies not written in English

Data Extraction Properties Mapped to Research Questions

Property	Research Questions
Researchers and Publications	RQ1, RQ2
Research Trends and Topics	RQ3
Software Defect Datasets	RQ4
Software Metrics	RQ4
Software Defect Prediction Methods	RQ5, RQ6, RQ7, RQ8
Software Defect Prediction Frameworks	RQ9

3.3 Literature Review Results

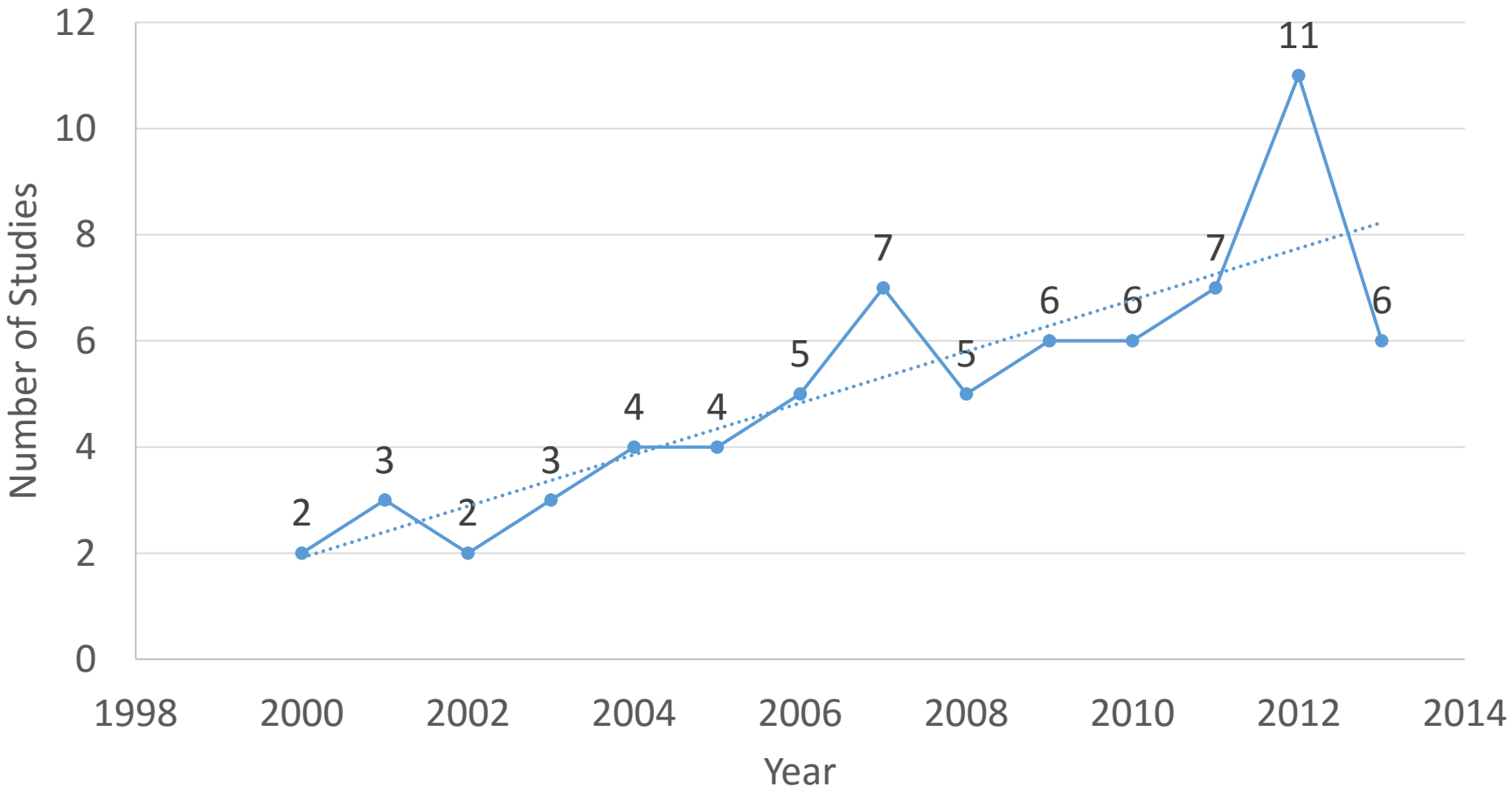
RQ1: Significant Journal Publications



Journal Quality Level of Selected Studies

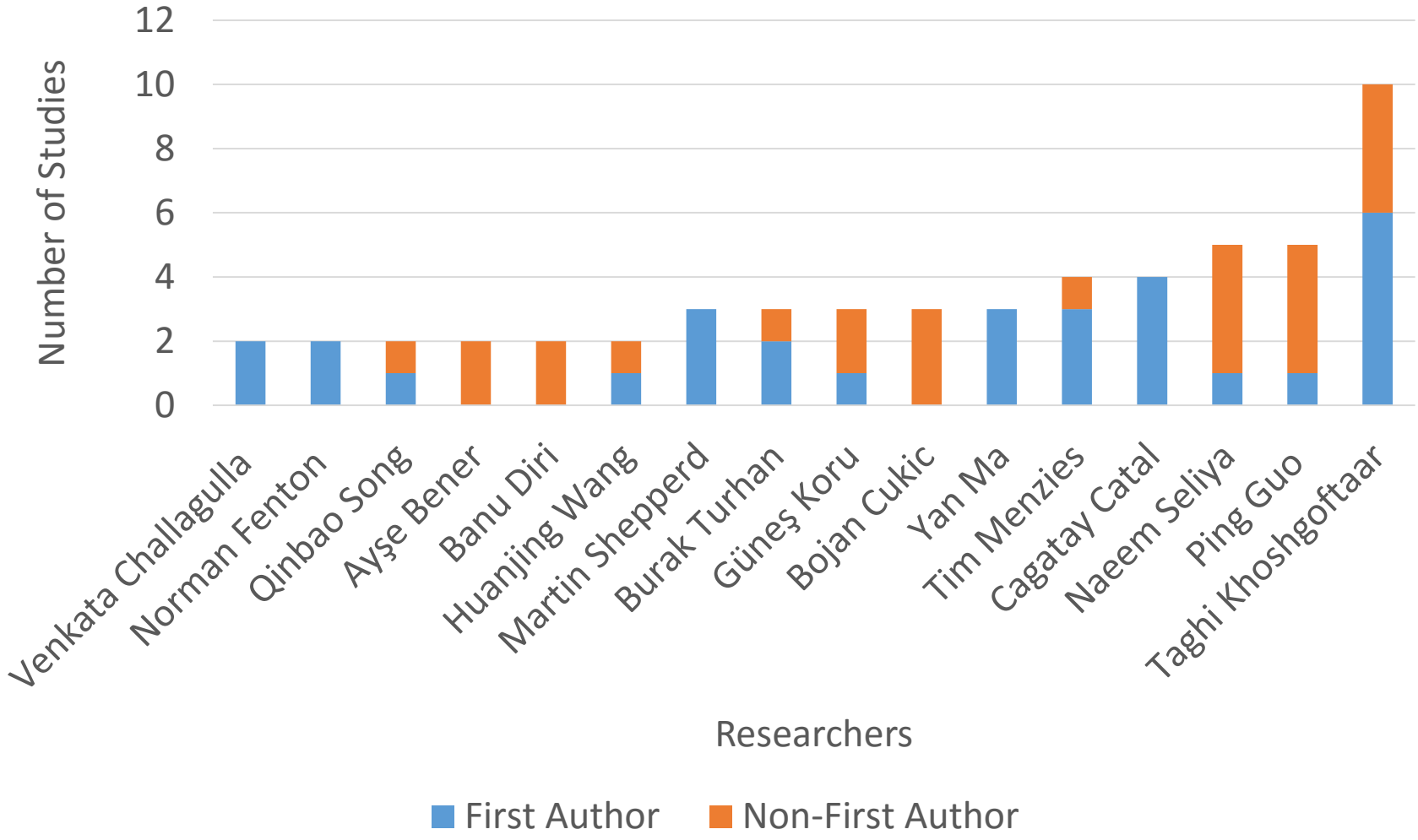
No	Journal Publications	SJR	Q Category
1	IEEE Transactions on Software Engineering	3.39	Q1 in Software
2	Information Sciences	2.96	Q1 in Information Systems
3	IEEE Transactions on Systems, Man, and Cybernetics	2.76	Q1 in Artificial Intelligence
4	IEEE Transactions on Knowledge and Data Engineering	2.68	Q1 in Information Systems
5	Empirical Software Engineering	2.32	Q1 in Software
6	Information and Software Technology	1.95	Q1 in Information Systems
7	Automated Software Engineering	1.78	Q1 in Software
8	IEEE Transactions on Reliability	1.43	Q1 in Software
9	Expert Systems with Applications	1.36	Q2 in Computer Science
10	Journal of Systems and Software	1.09	Q2 in Software
11	Software Quality Journal	0.83	Q2 in Software
12	IET Software	0.55	Q2 in Software
13	Advanced Science Letters	0.24	Q3 in Computer Science
14	Journal of Software	0.23	Q3 in Software
15	International Journal of Software Engineering and Its Application	0.14	Q4 in Software

Distribution of Selected Studies by Year



- The interest in software defect prediction has changed over time
- Software defect prediction research is still very much relevant to this day

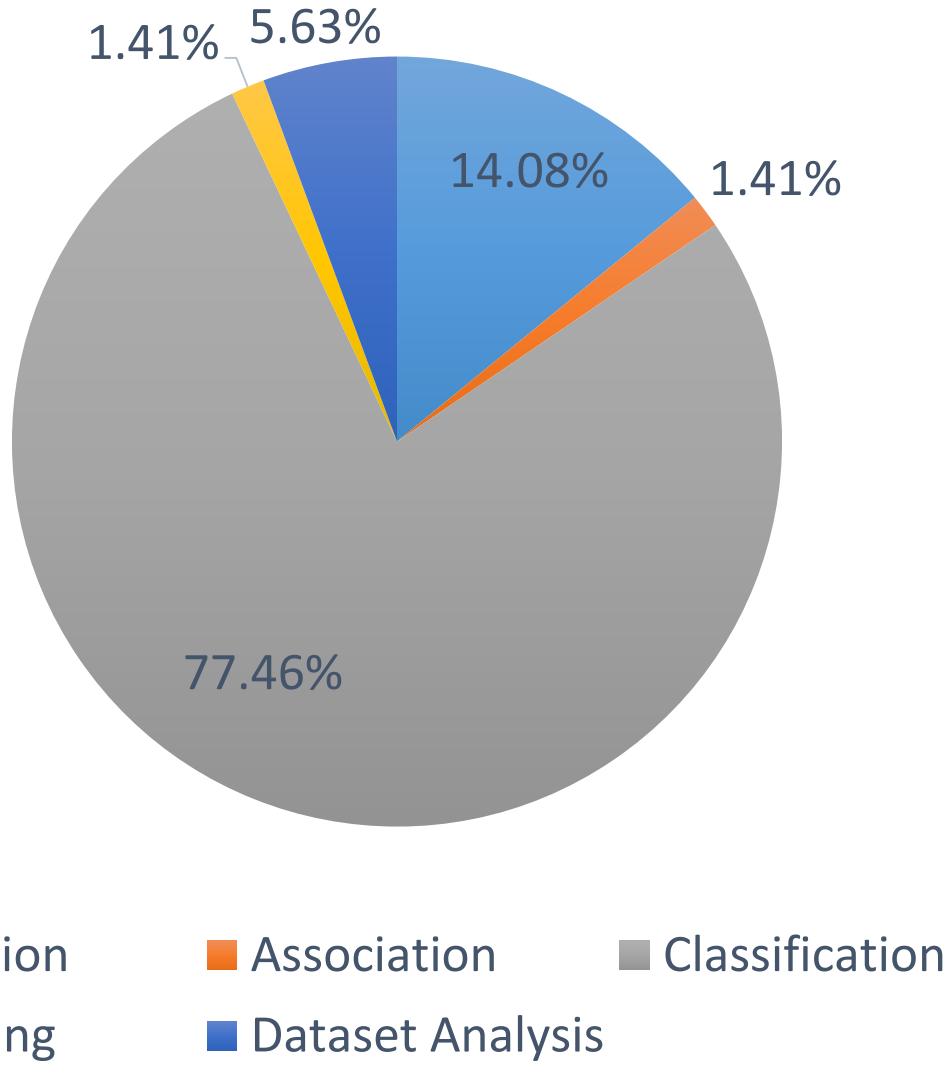
RQ2: Influential Researchers



RQ3: Research Topics and Trends

1. Estimating the number of defects remaining in software systems using estimation algorithm (Estimation)
2. Discovering defect associations using association rule algorithm (Association)
3. Classifying the defect-proneness of software modules, typically into two classes, defect-prone and not defect-prone, using classification algorithm (Classification)
4. Clustering the software defect based on object using clustering algorithm (Clustering)
5. Analyzing and pre-processing the software defect datasets (Dataset Analysis)

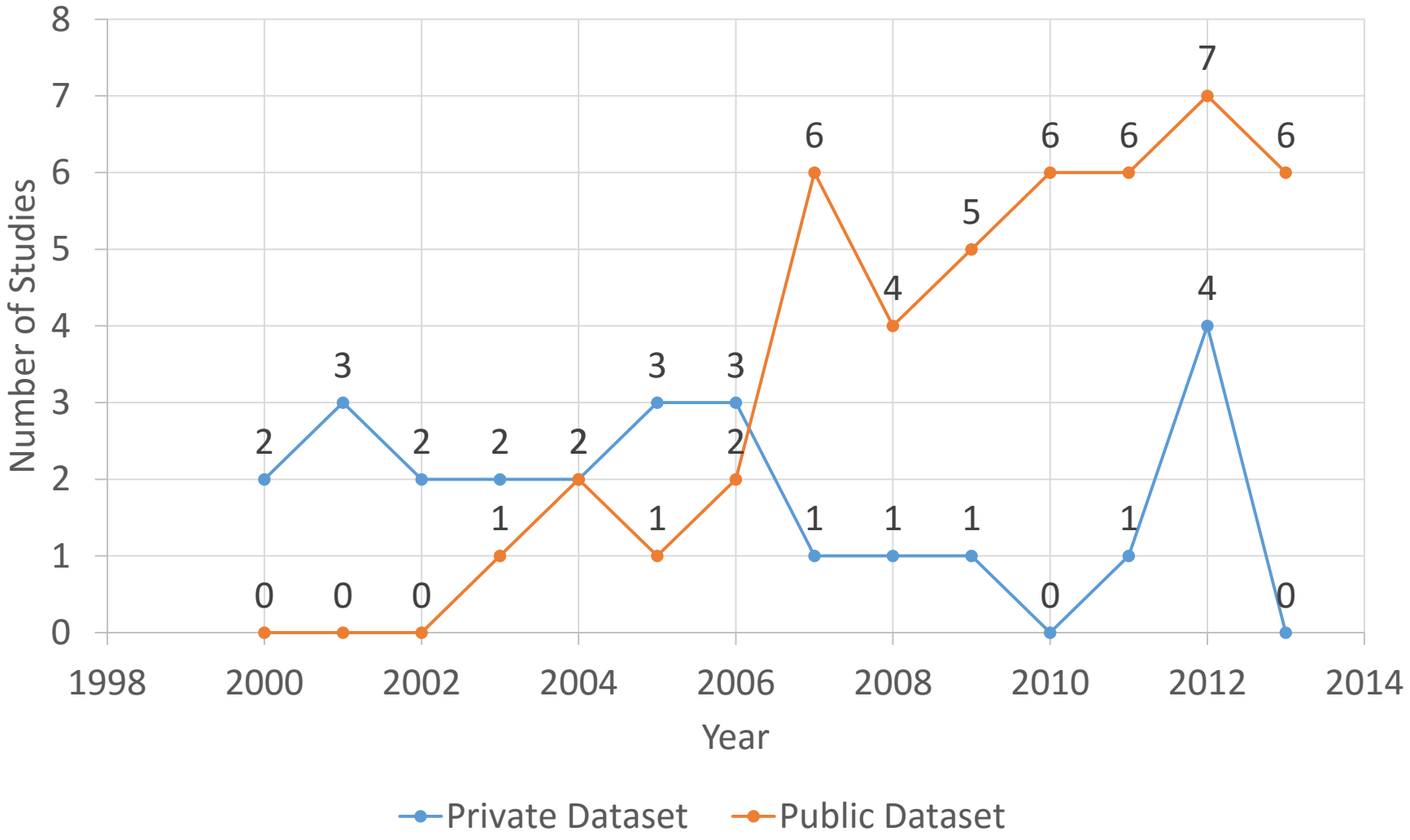
Distribution of Research Topics and Trends



Example Distribution of Research Topics and Trends

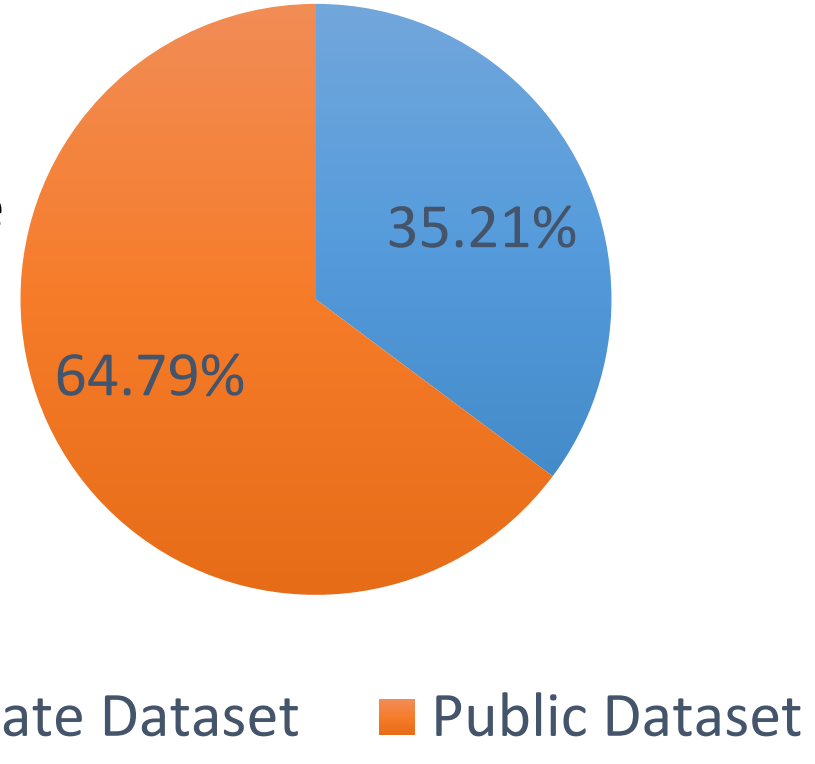
Year	Primary Studies	Publications	Datasets	Topics
2008	(Lessmann et al., 2008)	IEEE Transactions on Software Engineering	Public	Classification
	(Bibi et al., 2008)	Expert Systems with Applications	Private	Estimation
	(Gondra, 2008)	Journal of Systems and Software	Public	Classification
	(Vandecruys et al., 2008)	Journal of Systems and Software	Public	Classification
	(Elish and Elish 2008)	Journal of Systems and Software	Public	Classification
2012	(Gray et al., 2012)	IET Software	Public	Dataset Analysis
	(Ying Ma, Luo, Zeng, & Chen, 2012)	Information and Software Technology	Public	Classification
	(Benaddy and Wakrim 2012)	International Journal of Software Engineering	Private	Estimation
	(Y. Peng, Wang, & Wang, 2012)	Information Sciences	Public	Classification
	(Zhang and Chang 2012)	International Conference on Natural Computation	Private	Estimation
	(Bishnu and Bhattacherjee 2012)	IEEE Transactions on Knowledge and Data Engineering	Private	Clustering
	(Sun, Song, & Zhu, 2012)	IEEE Transactions on Systems, Man, and Cybernetics	Public	Classification
	(Pelayo and Dick 2012)	IEEE Transactions on Reliability	Public	Classification
	(Jin, Jin, & Ye, 2012)	IET Software	Public	Classification
	(Cao, Qin, & Feng, 2012)	Advanced Science Letters	Public	Classification
2013	(Park et al., 2013)	Information Sciences	Public	Classification
	(Dejaeger, Verbraken, & Baesens, 2013)	IEEE Transactions on Software Engineering	Public	Classification
	(Shepperd, Song, Sun, & Mair, 2013)	IEEE Transactions on Software Engineering	Public	Dataset Analysis
	(Wang and Yao 2013)	IEEE Transactions on Reliability	Public	Classification
	(Peters, Menzies, Gong, & Zhang, 2013)	IEEE Transactions on Software Engineering	Public	Dataset Analysis
	(Radjenović et al., 2013)	Information and Software Technology	Public	Dataset Analysis

RQ4: Software Defect Datasets



Distribution of Software Defect Datasets

- The use of public data sets makes the research repeatable, refutable, and verifiable (Catal & Diri 2009a)
- Since 2005 more public datasets were used
- NASA MDP repository have been developed in 2005 and researchers started to be aware regarding the use of public datasets



NASA MDP Dataset

Dataset	Project Description	Language	Number of Modules	Number of *fp* Modules	Faulty Percentage
CM1	Spacecraft instrument	C	505	48	12.21%
KC1	Storage management for ground data	C++	1571	319	15.51%
KC3	Storage management for ground data	Java	458	42	18%
MC2	Video guidance system	C	127	44	34.65%
MW1	Zero gravity experiment related to combustion	C	403	31	10.23%
PC1	Flight software from an earth orbiting satellite	C	1059	76	8.04%
PC2	Dynamic simulator for attitude control systems	C	4505	23	1.01%
PC3	Flight software for earth orbiting satellite	C	1511	160	12.44%
PC4	Flight software for earth orbiting satellite	C	1347	178	12.72%

Code Attributes		Symbols	Description
LOC counts	LOC_total		The total number of lines for a given module
	LOC_blank		The number of blank lines in a module
	LOC_code_and_comment	NCSLOC	The number of lines which contain both code and comment in a module
	LOC_comments		The number of lines of comments in a module
	LOC_executable		The number of lines of executable code for a module
	number_of_lines		Number of lines in a module
Halstead	content	μ	The halstead length content of a module $\mu = \mu_1 + \mu_2$
	difficulty	D	The halstead difficulty metric of a module D = 1/L
	effort	E	The halstead effort metric of a module E = V/L
	error_est	B	The halstead error estimate metric of a module $B = E^{2/3}/1000$
	length	N	The halstead length metric of a module $N = N_1+N_2$
	level	L	The halstead level metric of a module $L = (2^* \mu_2)/ \mu_1^*N_2$
	prog_time	T	The halstead programming time metric of a module T = E/18
	volume	V	The halstead volume metric of a module $V = N^*\log_2(\mu_1+ \mu_2)$
	num_operands	N_1	The number of operands contained in a module
	num_operators	N_2	The number of operators contained in a module
	num_unique_operands	μ_1	The number of unique operands contained in a module
	num_unique_operators	μ_2	The number of unique operators contained in a module
McCabe	cyclomatic_complexity	v(G)	The cyclomatic complexity of a module v(G) = e – n +2
	cyclomatic_density		v(G) / NCSLOC
	design_complexity	iv(G)	The design complexity of a module
	essential_complexity	ev(G)	The essential complexity of a module
Misc.	branch_count		Branch count metrics
	call_pairs		Number of calls to functions in a module
	condition_count		Number of conditionals in a given module
	decision_count		Number of decision points in a module
	decision_density		condition_count / decision_count
	edge_count		Number of edges found in a given module from one module to another
	essential_density		Essential density is calculated as: (ev(G)-1)/(v(G)-1)
	parameter_count		Number of parameters to a given module
	maintenance_severity		Maintenance Severity is calculated as: ev(G)/v(G)
	modified_condition_count		The effect of a condition affect a decision outcome by varying that condition only
	multiple_condition_count		Number of multiple conditions within a module
	global_data_complexity	gdv(G)	the ratio of cyclomatic complexity of a module's structure to its parameter count
	global_data_density		Global Data density is calculated as: gdv(G)/v(G)
	normalized_cyclo_cmplx		v(G) / numbe_of_lines
	percent_comments		Percentage of the code that is comments
	node_count		Number of nodes found in a given module

Code Attributes		NASA MDP Dataset								
		CM1	KC1	KC3	MC2	MW1	PC1	PC2	PC3	PC4
LOC counts	LOC_total	√	√	√	√	√	√	√	√	√
	LOC_blank	√	√	√	√	√	√		√	√
	LOC_code_and_comment	√	√	√	√	√	√	√	√	√
	LOC_comments	√	√	√	√	√	√	√	√	√
	LOC_executable	√	√	√	√	√	√	√	√	√
	number_of_lines	√		√	√	√	√	√	√	√
Halstead	content	√	√	√	√	√	√	√	√	√
	difficulty	√	√	√	√	√	√	√	√	√
	effort	√	√	√	√	√	√	√	√	√
	error_est	√	√	√	√	√	√	√	√	√
	length	√	√	√	√	√	√	√	√	√
	level	√	√	√	√	√	√	√	√	√
	prog_time	√	√	√	√	√	√	√	√	√
	volume	√	√	√	√	√	√	√	√	√
	num_operands	√	√	√	√	√	√	√	√	√
	num_operators	√	√	√	√	√	√	√	√	√
	num_unique_operands	√	√	√	√	√	√	√	√	√
	num_unique_operators	√	√	√	√	√	√	√	√	√
McCabe	cyclomatic_complexity	√	√	√	√	√	√	√	√	√
	cyclomatic_density	√		√	√	√	√	√	√	√
	design_complexity	√	√	√	√	√	√	√	√	√
	essential_complexity	√	√	√	√	√	√	√	√	√
Misc.	branch_count	√	√	√	√	√	√	√	√	√
	call_pairs	√		√	√	√	√	√	√	√
	condition_count	√		√	√	√	√	√	√	√
	decision_count	√		√	√	√	√	√	√	√
	decision_density	√		√	√	√	√	√	√	√
	edge_count	√		√	√	√	√	√	√	√
	essential_density	√		√	√	√	√	√	√	√
	parameter_count	√		√	√	√	√	√	√	√
	maintenance_severity	√		√	√	√	√	√	√	√
	modified_condition_count	√		√	√	√	√	√	√	√
	multiple_condition_count	√		√	√	√	√	√	√	√
	global_data_complexity			√	√					
	global_data_density			√	√					
	normalized_cyclo_complx	√		√	√	√	√	√	√	√
	percent_comments	√		√	√	√	√	√	√	√
	node_count	√		√	√	√	√	√	√	√
Programming Language		C	C++	Java	C	C	C	C	C	C
Number of Code Attributes		37	21	39	39	37	37	36	37	37
Number of Modules		344	2096	200	127	264	759	1585	1125	1399
Number of fp Modules		42	325	36	44	27	61	16	140	178

Code Attribute

```
1.  void main()
2.  {
3.          //This is a sample code

4.          //Declare variables
5.          int a, b, c;

6.          // Initialize variables
7.          a=2;
8.          b=5;

9.          //Find the sum and display c if greater
    than zero
10.         c=sum(a,b);
11.         if c < 0
12.           printf("%d\n", a);
13.         return;
14. }


15. int sum(int a, int b)
16. {
17.         // Returns the sum of two numbers
18.         return a+b;
19. }
```

c > 0

c

Module	LOC	LOCC	V	CC	**Error**
main()	16	4	5	2	**2**
sum()	5	1	3	1	**0**

LOC: Line of Code
LOCC: Line of commented Code
V: Number of unique operands&operators
CC: Cyclometric Complexity

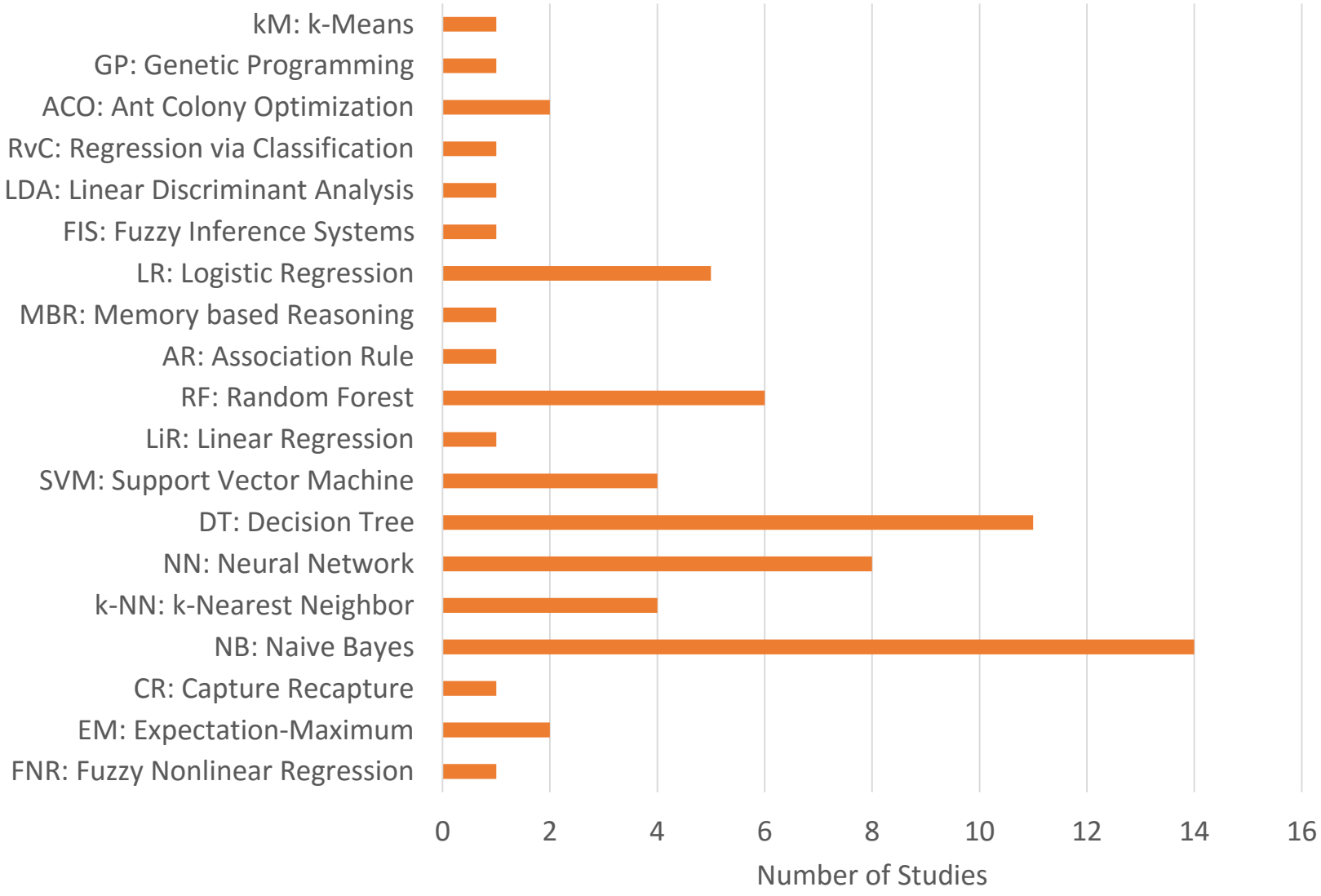
Code Complexity Measurement

1. Source Lines of Codes
2. Operator and Operand Numbers
   - Halstead
3. Coupling
4. Flow
   - McCabe

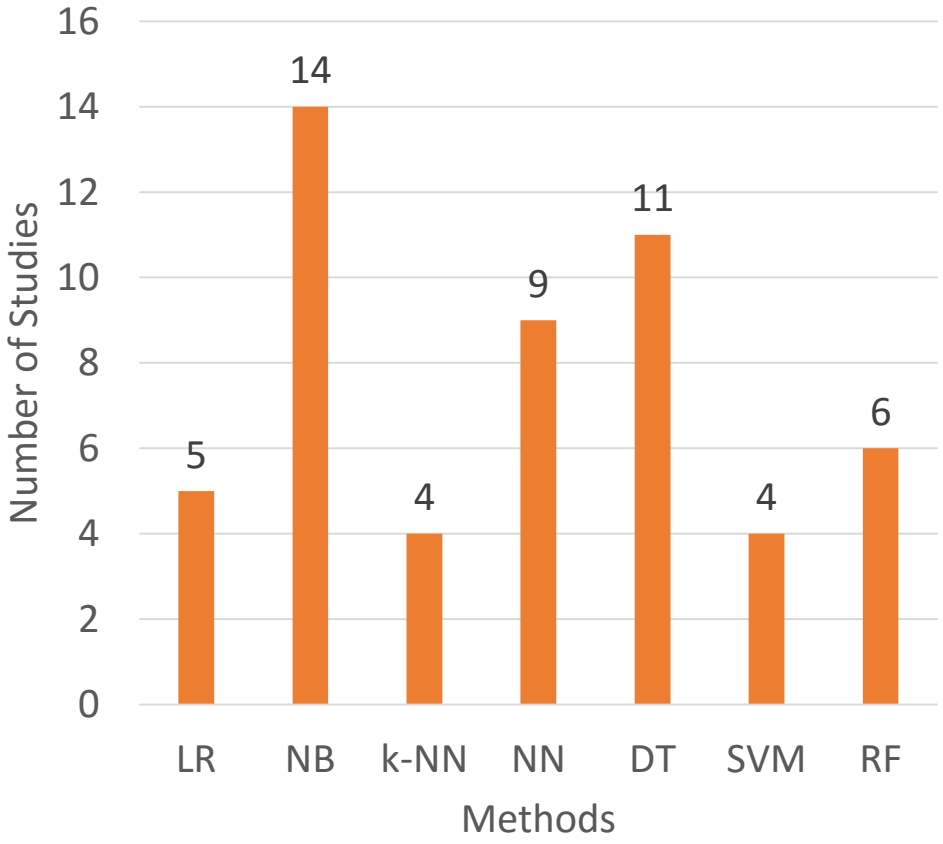
McCabe and Halstead

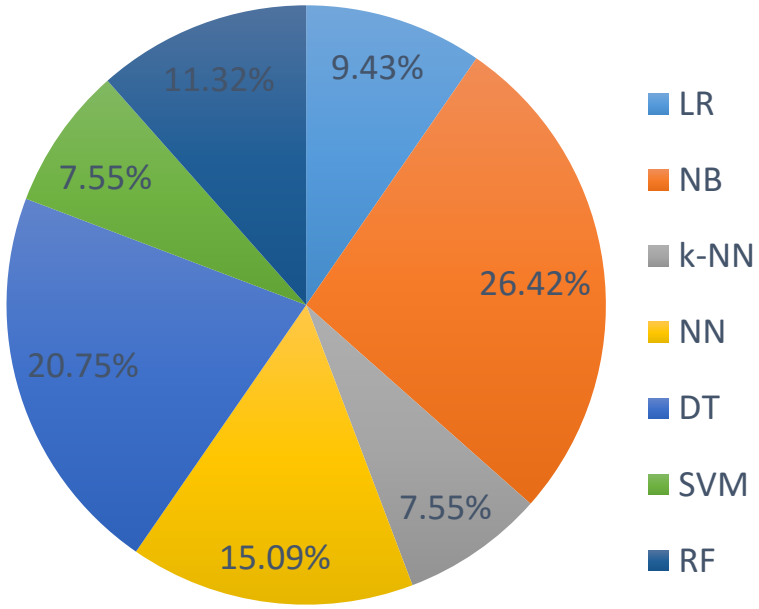
- There are 2 main types of software reliability models:
  1. the deterministic
  2. the probabilistic
- Two well known models of the deterministic type are the Halstead's software metric and the McCabe's cyclomatic complexity metric
  1. Halstead's software metric is used to estimate the number of errors in a program (based on the number of operands and operators in programs)
  2. McCabe's cyclomatic complexity metric (McCabe 1976) is used to determine an upper bound on the model for estimating the number of remaining defects (based on the number of decision points)

RQ5: Software Defect Prediction Methods



RQ6: Most Used Software Defect Prediction Methods





RQ7: Method Comparison Results

- The comparisons and benchmarking result of the defect prediction using machine learning classifiers indicate that:
  - ✓Poor accuracy level is dominant (Lessmann et al. 2008)
  - ✓No significant performance differences could be detected (Lessmann et al. 2008)
  - ✓No particular classifiers that performs the best for all the data sets (Song et al. 2011) (Hall et al. 2012)

- The accurate and reliable classification algorithms to build a better prediction model is an open issue in software defect prediction

RQ8: Method Improvement Efforts

- Researchers proposed some techniques for improving the accuracy of classifiers for software defect prediction
- Recent proposed techniques try to increase the prediction accuracy of a generated model:
  - ✓ By modifying and ensembling some machine learning methods (Mısırlı et al. 2011) (Tosun et al. 2008)
  - ✓ By using boosting algorithm (Zheng 2010) (Jiang et al. 2011)
  - ✓ by adding feature selection (Gayatri et al. 2010) (Khoshgoftaar & Gao, 2009) (Song et al. 2011)
  - ✓ By using parameter selection for some classifiers (Peng & Wang 2010) (Lin et al. 2008) (Guo et al. 2008)
- While considerable works have been done separately, limited research can be found on investigating them all together

RQ9: Existing Frameworks

Three frameworks have been highly cited and influential in software defect prediction field

Menzies Framework

(Menzies et al. 2007)

Lessmann Framework

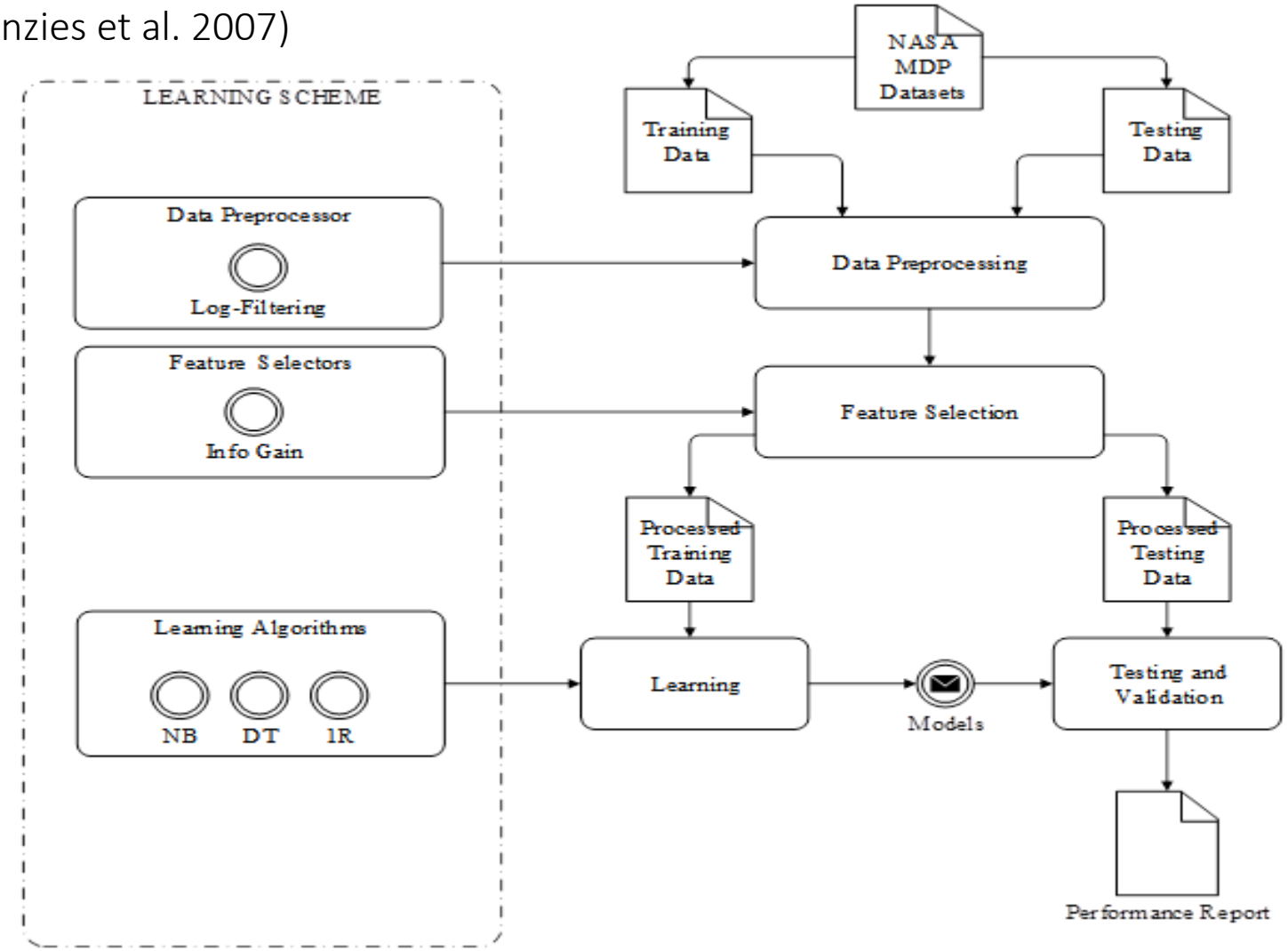
(Lessmann et al. 2008)

Song Framework

(Song et al. 2011)

Menzies Framework

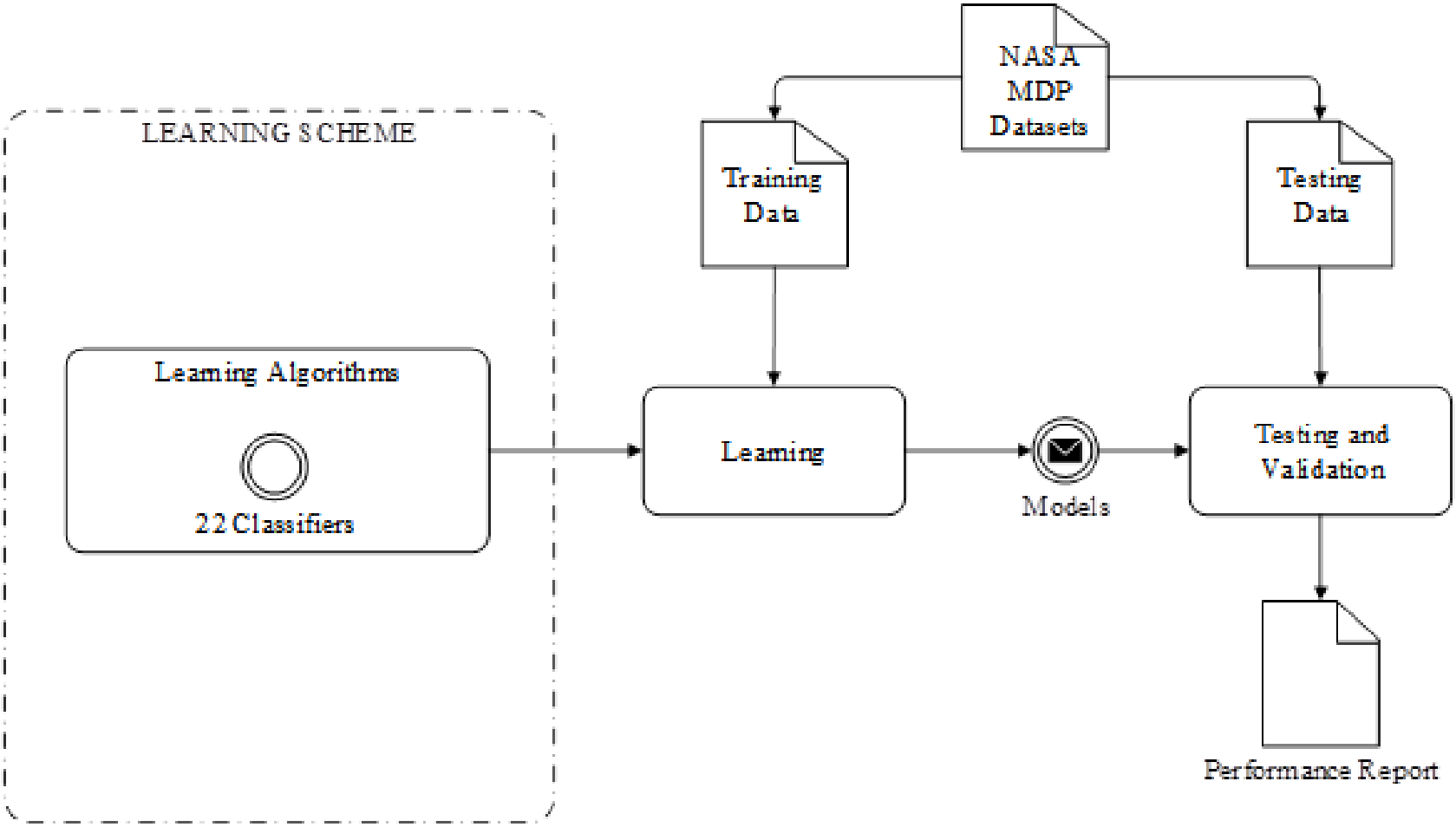
(Menzies et al. 2007)



Framework	Dataset	Data Preprocessor	Feature Selectors	Meta-learning	Classifiers	Parameter Selectors	Validation Methods	Evaluation Methods
(Menzies et al. 2007)	NASA MDP	**Log Filtering**	**Info Gain**	-	**3 algorithms (DT, 1R, NB)**	-	10-Fold X Validation	ROC Curve (AUC)

Lessmann Framework

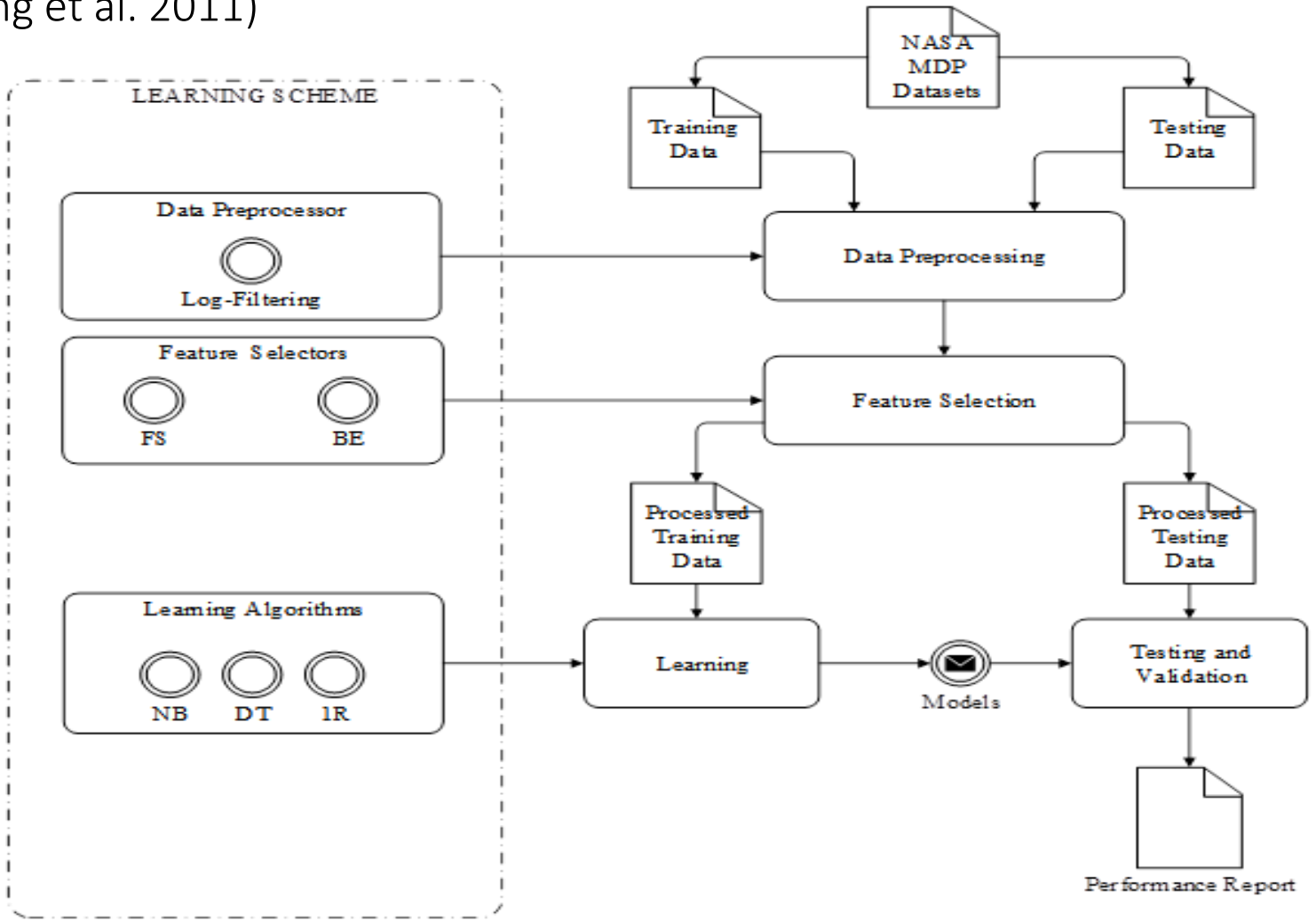
(Lessmann et al. 2008)



Framework	Dataset	Data Preprocessor	Feature Selectors	Meta-learning	Classifiers	Parameter Selectors	Validation Methods	Evaluation Methods
(Lessman et al. 2008)	NASA MDP	-	-	-	**22 algorithms**	-	10-Fold X Validation	ROC Curve (AUC)

Song Framework

(Song et al. 2011)



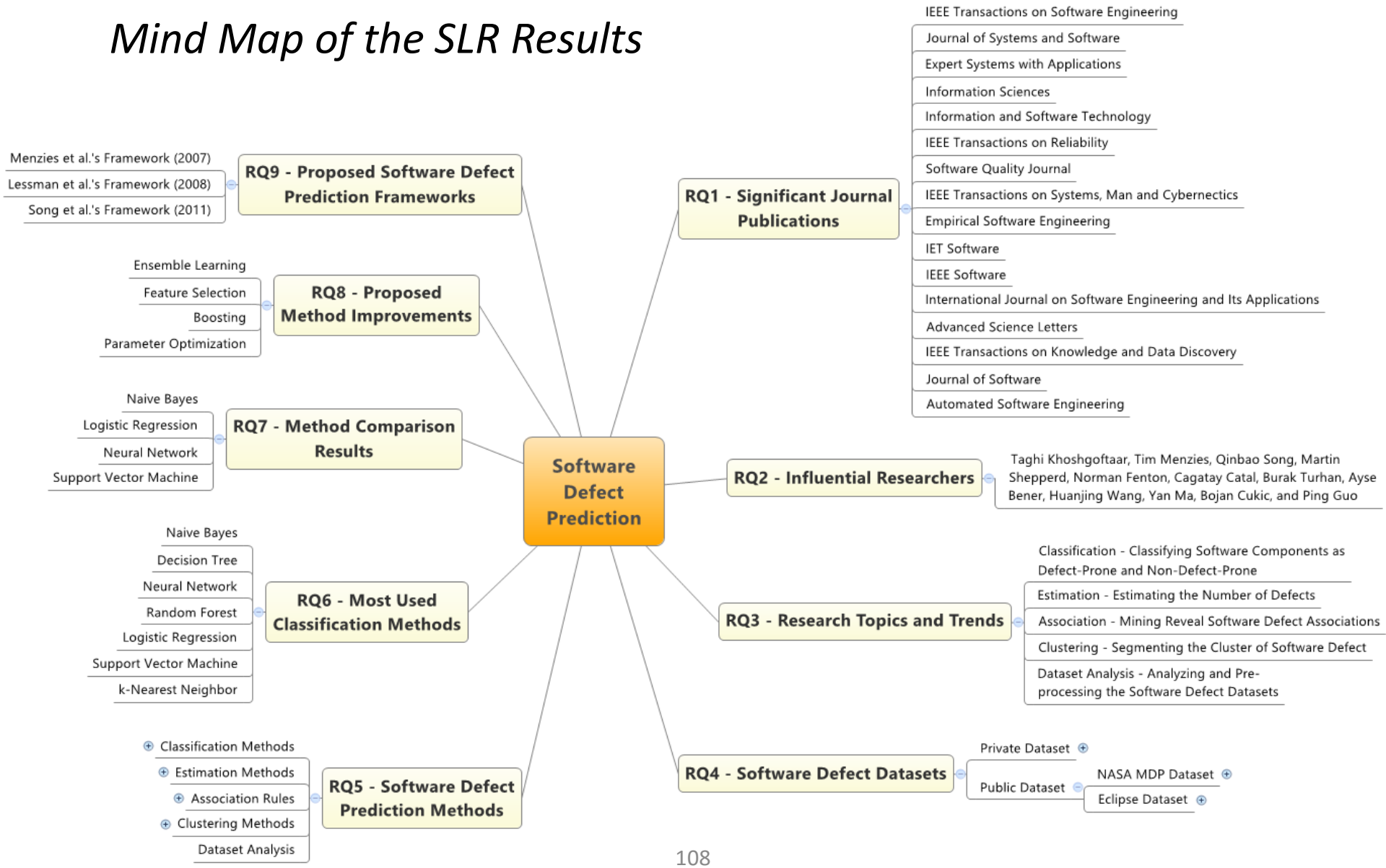
Framework	Dataset	Data Preprocessor	Feature Selectors	Meta-learning	Classifiers	Parameter Selectors	Validation Methods	Evaluation Methods
(Song et al. 2011)	NASA MDP	**Log Filtering**	**FS, BE**	-	**3 algorithms (DT, 1R, NB)**	-	10-Fold X Validation	ROC Curve (AUC)

Gap Analysis of Methods and Frameworks

- Noisy attribute predictors and imbalanced class distribution of software defect datasets result in inaccuracy of classification models
- Neural network and support vector machine have strong fault tolerance and strong ability of nonlinear dynamic processing of software fault data, but practicability of neural network and support vector machine are limited due to difficulty of selecting appropriate parameters

Conclusion

Mind Map of the SLR Results



4. Pengembangan ke Arah Penelitian Baru dari Analisis Gap

Gap Analisis di SLR dan Arah Penelitian Baru

- Dengan SLR, kita bisa memahami state-of-the-art research dan methods, yang selama ini telah dilakukan oleh para peneliti
- *State-of-the-art methods* ini akan membawa kita ke pemahaman terhadap gap penelitian yang ada, yang mungkin bisa kita angkat menjadi arah penelitian yang baru
- Berikut akan saya berikan satu contoh bagaimana dari analisis gap yang kita lakukan, kita bisa membentuk research problem (RP), research objective (RO) dan research contributions (RC) baru

Gap Analysis of Methods and Frameworks

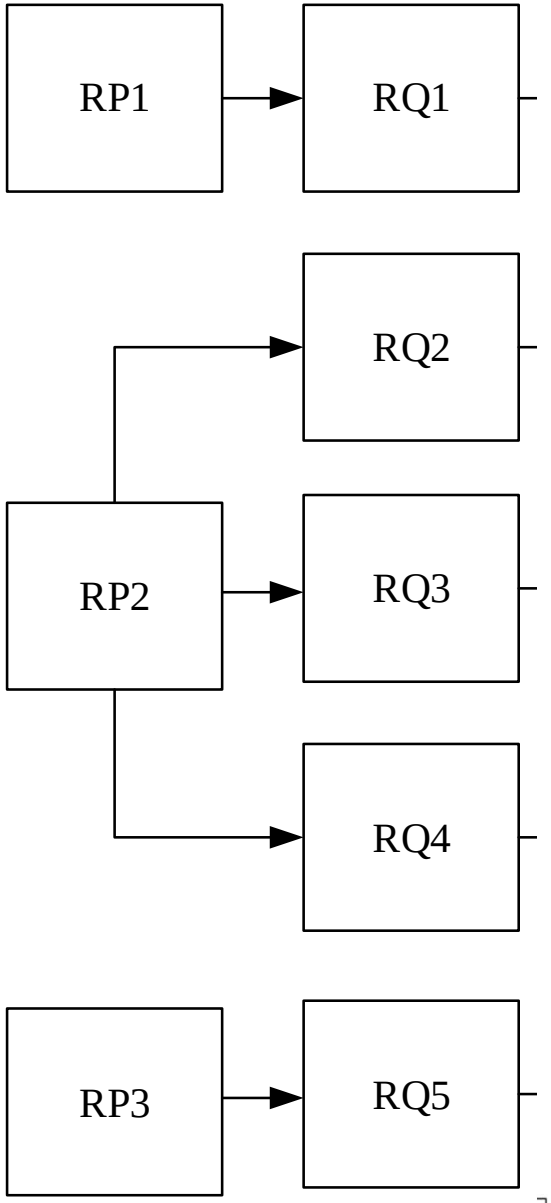
- Noisy attribute predictors and imbalanced class distribution of software defect datasets result in inaccuracy of classification models
- Neural network and support vector machine have strong fault tolerance and strong ability of nonlinear dynamic processing of software fault data, but practicability of neural network and support vector machine are limited due to difficulty of selecting appropriate parameters

New Research Problems (RP)

RP1 While many studies on software defect prediction report the comparative performance of the classification algorithms used, but there is no strong consensus on which classifiers perform best when individual studies are looked separately

RP2 Noisy attribute predictors and imbalanced class distribution of software defect datasets result in inaccuracy of classification models

RP3 Neural network has strong fault tolerance and strong ability of nonlinear dynamic processing of software fault data, but practicability of neural network is limited due to difficulty of selecting appropriate parameters



New Research Questions 1 (RQ1)

Research Problems (RP)		Research Questions (RQ)		Research Objectives (RO)	
RP1	While many studies on software defect prediction report the comparative performance of the modelling techniques they have used, no clear consensus on which classifier perform best emerges when individual studies are looked at separately	RQ1	**Which machine learning classification algorithms perform best when used in software defect prediction?**	RO1	To identify and determine the best machine learning classification algorithms when used in software defect prediction

New Research Questions 2-4 (RQ2-RQ4)

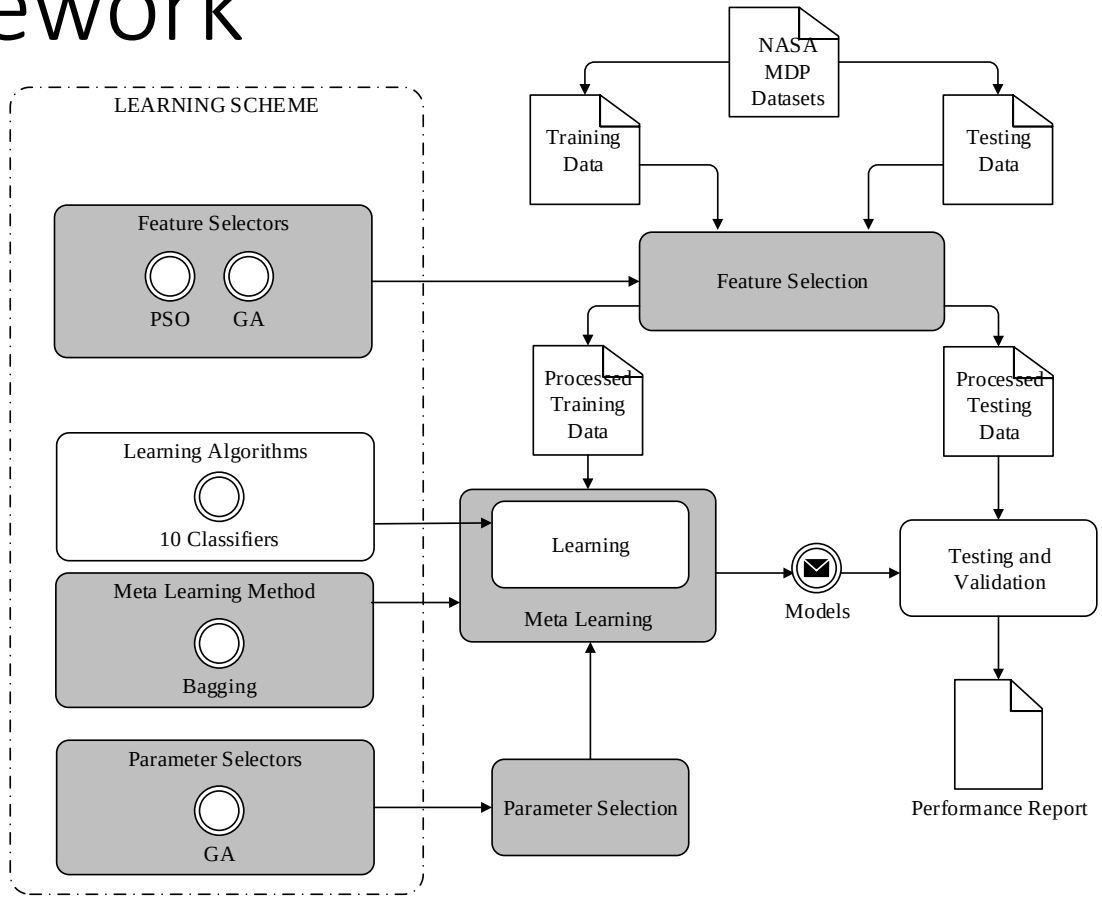
Research Problems (RP)		Research Questions (RQ)		Research Objectives (RO)	
RP2	Noisy attribute predictors and imbalanced class distribution of software defect datasets result in inaccuracy of classification models	RQ2	**How does the integration between genetic algorithm based feature selection and bagging technique affect the accuracy of software defect prediction?**	RO2	To develop a hybrid genetic algorithm based feature selection and bagging technique for improving the accuracy of software defect prediction
		RQ3	**How does the integration between particle swarm optimization based feature selection and bagging technique affect the accuracy of software defect prediction?**	RO3	To develop a hybrid particle swarm optimization based feature selection and bagging technique for improving the accuracy of software defect prediction
		RQ4	**Which metaheuristic optimization techniques perform best when used in feature selection of software defect prediction?**	RO4	To identify the best metaheuristic optimization techniques when used in feature selection of software defect prediction

New Research Questions 5 (RQ5)

Research Problems (RP)		Research Questions (RQ)		Research Objectives (RO)	
RP3	Neural network has strong fault tolerance and strong ability of nonlinear dynamic processing of software fault data, but practicability of neural network is limited due to difficulty of selecting appropriate parameters	RQ5	**How does the integration between genetic algorithm based neural network parameter selection and bagging technique affect the accuracy of software defect prediction?**	RO5	To develop a hybrid genetic algorithm based neural network parameter selection and bagging technique for improving the accuracy of software defect prediction

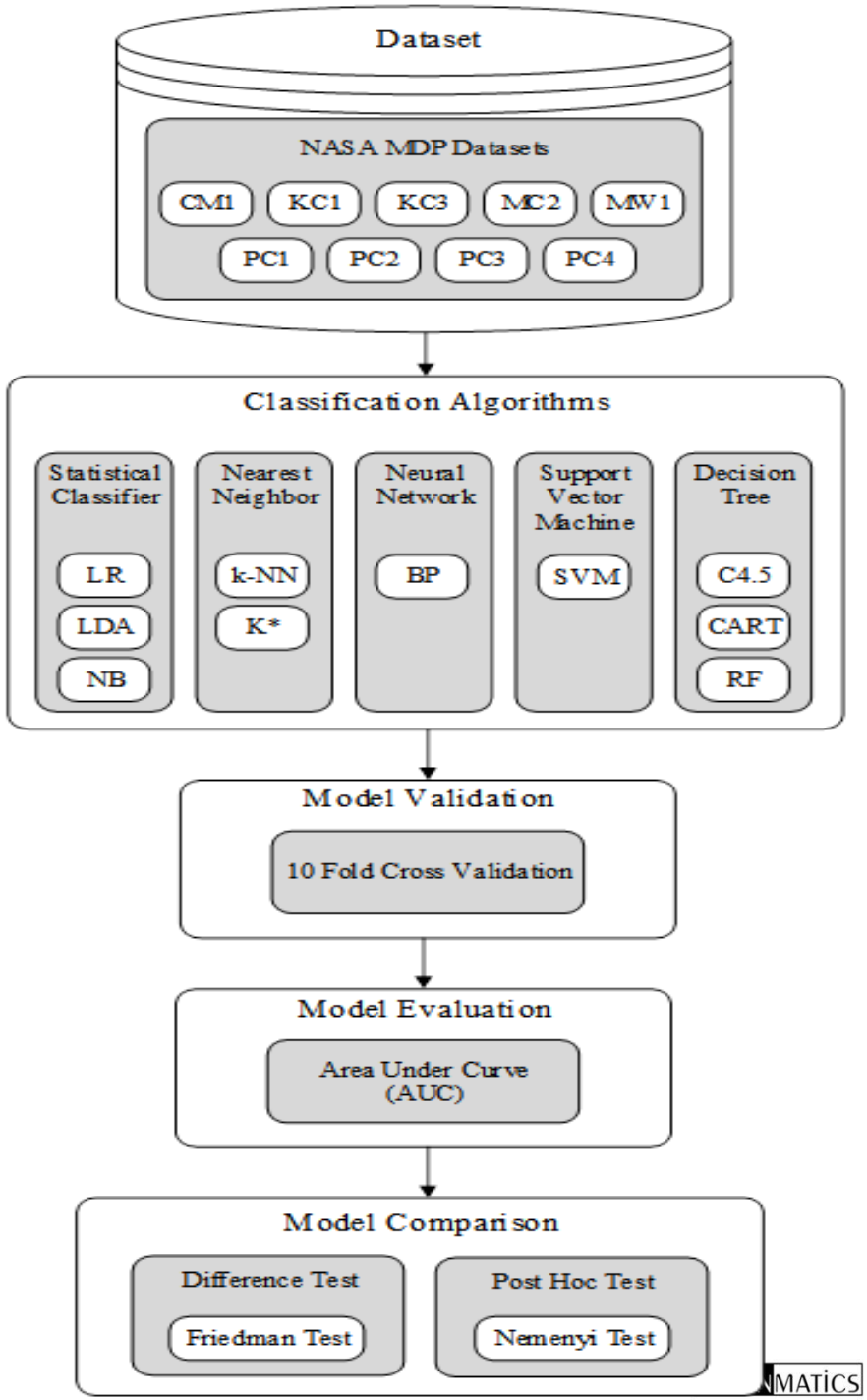
Wahono Framework

(Wahono et al., 2013)
(Wahono et al., 2014)



Framework	Dataset	Data Preprocessor	Feature Selectors	Meta-Learning	Classifiers	Parameter Selectors	Validation Methods	Evaluation Methods
(Menzies et al. 2007)	NASA MDP	Log Filtering	Info Gain		3 algorithm (DT, 1R, NB)	-	10-Fold X Validation	ROC Curve (AUC)
(Lessman et al. 2008)	NASA MDP	-	-		22 algorithm	-	10-Fold X Validation	ROC Curve (AUC)
(Song et al. 2011)	NASA MDP	Log Filtering	FS, BE		3 algorithm (DT, 1R, NB)	-	10-Fold X Validation	ROC Curve (AUC)
Proposed Framework	NASA MDP	-	**PSO, GA**	**Bagging**	**10 algorithms**	**GA**	10-Fold X Validation	ROC Curve (AUC)

A Comparison Framework of Classification Models for Software Defect Prediction (CF SDP)



(Romi Satria Wahono, Nanna Suryana Herman and Sabrina Ahmad, A Comparison Framework of Classification Models for Software Defect Prediction, Advanced Science Letters, Vol. 20, No. 10-12, October 2014)

CF-SDP: AUC and Friedman Test Results

	CM1	KC1	KC3	MC2	MW1	PC1	PC2	PC3	PC4	*M*	*R*
LR	**0.763**	**0.801**	**0.713**	**0.766**	0.726	**0.852**	0.849	**0.81**	0.894	**0.797**	**1.44**
LDA	0.471	0.536	0.447	0.503	0.58	0.454	0.577	0.524	0.61	0.522	8.33
NB	0.734	0.786	0.67	0.739	0.732	0.781	0.811	0.756	0.838	0.761	3
k-NN	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	8.778
K*	0.6	0.678	0.562	0.585	0.63	0.652	0.754	0.697	0.76	0.658	5.33
BP	0.713	0.791	0.647	0.71	0.625	0.784	0.918	0.79	0.883	0.762	3.22
SVM	0.753	0.752	0.642	0.761	0.714	0.79	0.534	0.75	**0.899**	0.733	3.33
C4.5	0.565	0.515	0.497	0.455	0.543	0.601	0.493	0.715	0.723	0.567	7.78
CART	0.604	0.648	0.637	0.482	0.656	0.574	0.491	0.68	0.623	0.599	6.89
RF	0.573	0.485	0.477	0.525	**0.74**	0.618	0.649	0.678	0.2	0.549	6.89

- LR is dominant in most datasets
- *R* rank: LR has the highest rank, followed by NB, BP, and SVM
- *M* results: no excellent or good models, and a few fair models

AUC	Meaning	Symbol
0.90 - 1.00	excellent classification	
0.80 - 0.90	good classification	
0.70 - 0.80	fair classification	
0.60 - 0.70	poor classification	
< 0.60	failure	

CF-SDP: *P-value* of Nemenyi Post Hoc Test

	LR	LDA	NB	k-NN	K*	BP	SVM	C4.5	CART	RF
LR	1	**0.0001**	0.986	**0.0001**	0.164	0.965	0.949	**0.000**	**0.005**	**0.005**
LDA	**0.0001**	1	**0.007**	1.000	0.526	**0.013**	**0.017**	1.000	0.992	0.992
NB	0.986	**0.007**	1	**0.002**	0.831	1.000	1.000	**0.028**	0.164	0.164
k-NN	**0.0001**	1.000	**0.002**	1	0.318	**0.004**	**0.005**	1.000	0.949	0.949
K*	0.164	0.526	0.831	0.318	1	0.901	0.927	0.789	0.986	0.986
BP	0.965	**0.013**	1.000	**0.004**	0.901	1	1.000	**0.046**	0.232	0.232
SVM	0.949	**0.017**	1.000	**0.005**	0.927	1.000	1	0.058	0.273	0.273
C4.5	**0.000**	1.000	**0.028**	1.000	0.789	**0.046**	0.058	1	1.000	1.000
CART	**0.005**	0.992	0.164	0.949	0.986	0.232	0.273	1.000	1	1.000
RF	**0.005**	0.992	0.164	0.949	0.986	0.232	0.273	1.000	1.000	1

- If *P value < 0.05* (boldfaced print), it indicate that there is significant different between two classifiers
- Based on significant difference results, there is no significant difference between LR, NB, BP, and SVM models

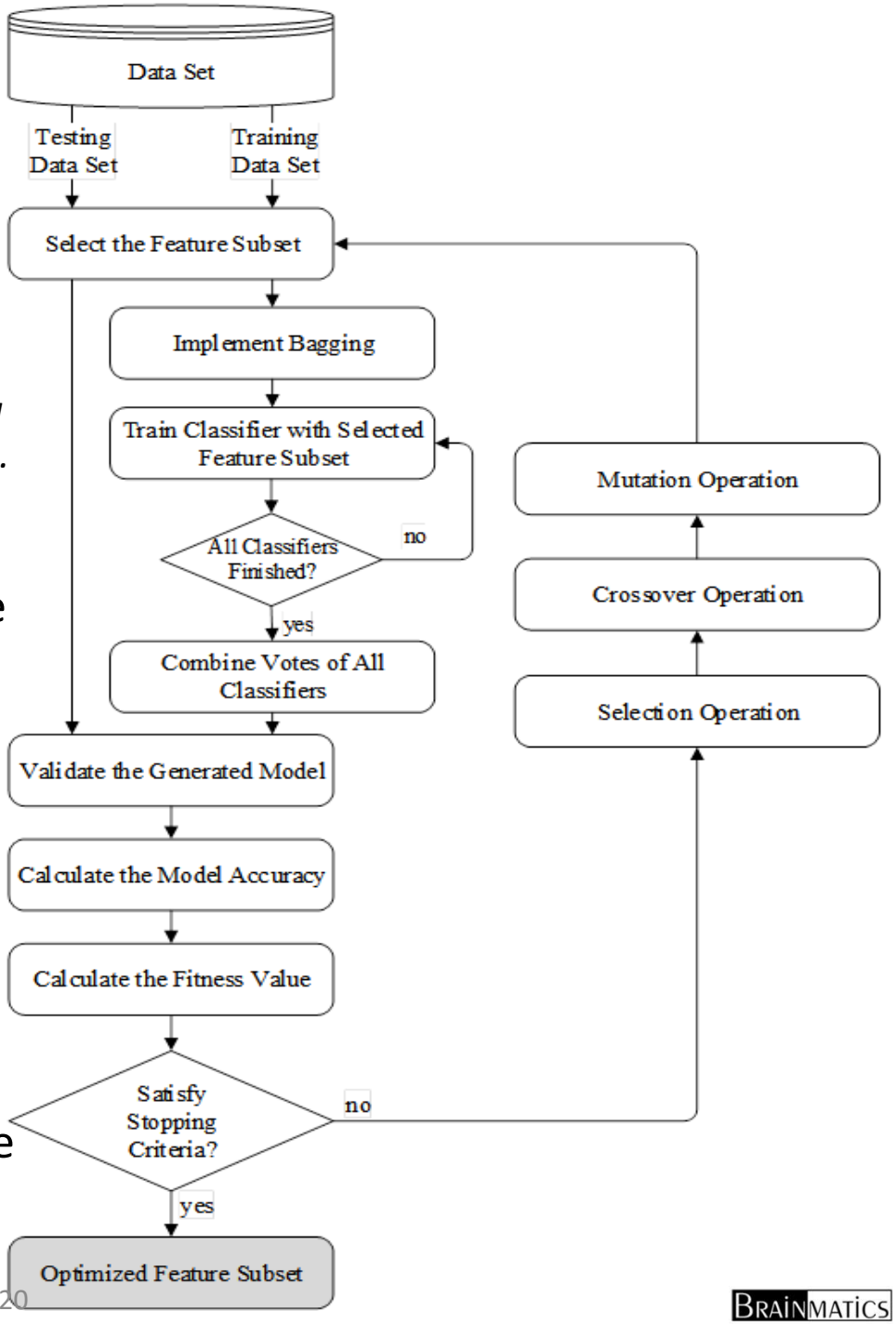
A Hybrid Genetic Algorithm based Feature Selection and Bagging Technique (GAFS+B)

(Romi Satria Wahono and Nanna Suryana, Combining Particle Swarm Optimization based Feature Selection and Bagging Technique for Software Defect Prediction, International Journal of Software Engineering and Its Applications, Vol. 7, No. 5, pp. 153-166, October 2013)

- Every chromosome is evaluated by the fitness function Equation

$$fitness = W_A \times A + W_F \times \left(P + \left(\sum_{i=1}^{n_f} C_i \times F_i \right) \right)^{-1}$$

- Where
  - A: classification accuracy
  - F_i: feature value
  - W_A: weight of classification accuracy
  - W_F: feature weight
  - C_i: feature cost
- When ending condition is satisfied, the operation ends, otherwise, continue with the next genetic operation



Results: Without GAFS+B

Classifiers		CM1	KC1	KC3	MC2	MW1	PC1	PC2	PC3	PC4
Statistical Classifier	LR	0.763	0.801	0.713	0.766	0.726	0.852	0.849	0.81	0.894
	LDA	0.471	0.536	0.447	0.503	0.58	0.454	0.577	0.524	0.61
	NB	0.734	0.786	0.67	0.739	0.732	0.781	0.811	0.756	0.838
Nearest Neighbor	k-NN	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
	K*	0.6	0.678	0.562	0.585	0.63	0.652	0.754	0.697	0.76
Neural Network	BP	0.713	0.791	0.647	0.71	0.625	0.784	0.918	0.79	0.883
Support Vector Machine	SVM	0.753	0.752	0.642	0.761	0.714	0.79	0.534	0.75	0.899
Decision Tree	C4.5	0.565	0.515	0.497	0.455	0.543	0.601	0.493	0.715	0.723
	CART	0.604	0.648	0.637	0.482	0.656	0.574	0.491	0.68	0.623
	RF	0.573	0.485	0.477	0.525	0.74	0.618	0.649	0.678	0.2

Results: With GAFS+B

Classifiers		CM1	KC1	KC3	MC2	MW1	PC1	PC2	PC3	PC4
Statistical Classifier	LR	0.753	0.795	0.691	0.761	**0.742**	0.852	0.822	0.813	**0.901**
	LDA	**0.592**	**0.627**	**0.635**	**0.64**	**0.674**	**0.637**	**0.607**	**0.635**	**0.715**
	NB	0.702	**0.79**	**0.677**	0.739	0.724	**0.799**	0.805	**0.78**	**0.861**
Nearest Neighbor	k-NN	**0.666**	**0.689**	**0.67**	**0.783**	**0.656**	**0.734**	**0.554**	**0.649**	**0.732**
	K*	**0.71**	**0.822**	0.503	**0.718**	**0.68**	**0.876**	**0.877**	**0.816**	**0.893**
Neural Network	BP	**0.744**	**0.797**	**0.707**	**0.835**	**0.689**	**0.829**	0.905	**0.799**	**0.921**
Support Vector Machine	SVM	0.667	**0.767**	0.572	**0.747**	**0.659**	0.774	0.139	0.476	**0.879**
Decision Tree	C4.5	**0.64**	**0.618**	**0.658**	**0.732**	**0.695**	**0.758**	**0.642**	**0.73**	**0.844**
	CART	**0.674**	**0.818**	**0.754**	**0.709**	**0.703**	**0.819**	**0.832**	**0.842**	**0.9**
	RF	**0.706**	**0.584**	**0.605**	0.483	0.735	**0.696**	**0.901**	**0.734**	**0.601**

- Almost all classifiers that implemented GAFS+B method outperform the original method
- GAFS+B affected significantly on the performance of the class imbalance suffered classifiers

Without GAFS+B vs With GAFS+B

Classifiers		*P* value of t-Test	Result
Statistical Classifier	LR	0.156	Not Sig. ($\alpha > 0.05$)
	LDA	**0.00004**	**Sig. ($\alpha < 0.05$)**
	NB	0.294	Not Sig. ($\alpha > 0.05$)
Nearest Neighbor	k-NN	**0.00002**	**Sig. ($\alpha < 0.05$)**
	K*	**0.001**	**Sig. ($\alpha < 0.05$)**
Neural Network	BP	**0.008**	**Sig. ($\alpha < 0.05$)**
Support Vector Machine	SVM	**0.03**	**Sig. ($\alpha < 0.05$)**
Decision Tree	C4.5	**0.0002**	**Sig. ($\alpha < 0.05$)**
	CART	**0.0002**	**Sig. ($\alpha < 0.05$)**
	RF	**0.01**	**Sig. ($\alpha < 0.05$)**

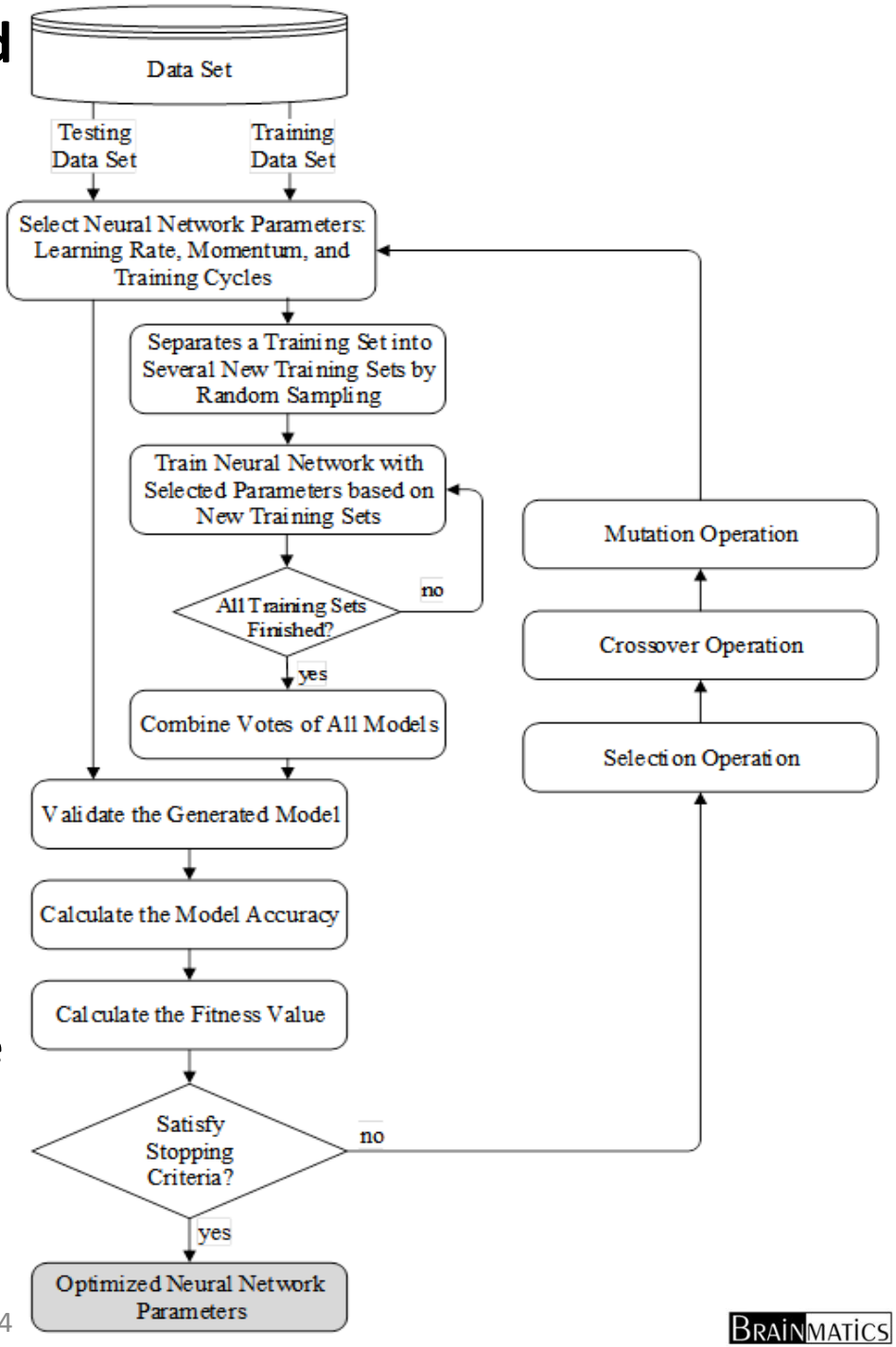
- Although there are two classifiers (LR and NB) that have no significant difference (*P value* > 0.05), the remaining eight classifiers (LDA, k-NN, K*, BP, SVM, C4.5, CART and RF) have significant difference (*P value* < 0.05)
- The proposed GAFS+B method makes an improvement in prediction performance for most classifiers

A Hybrid Genetic Algorithm based Neural Network Parameter Optimization and Bagging Technique for Software Defect Prediction (NN GAPO+B)

- Every chromosome is evaluated by the fitness function Equation

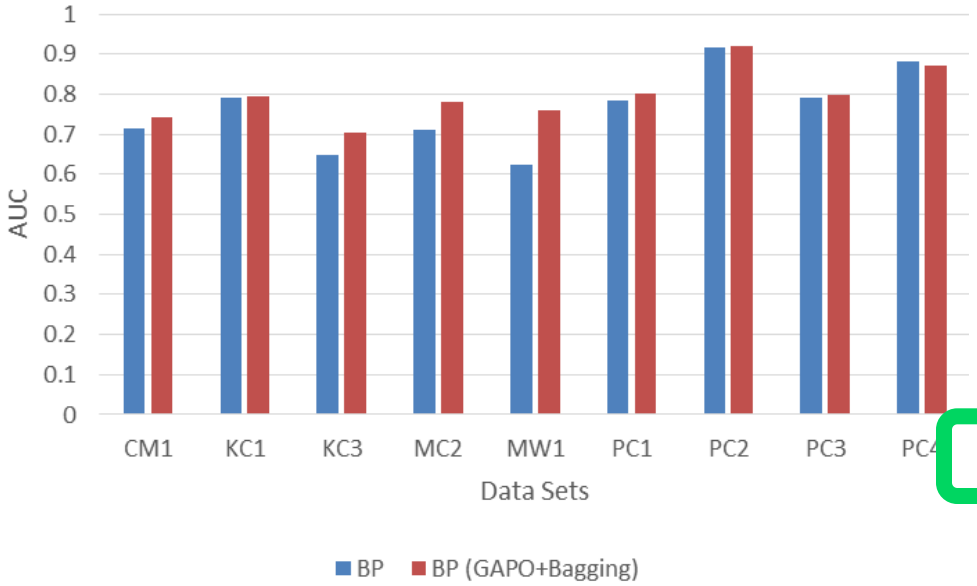
$$fitness = W_A \times A + W_P \times \left(S + \left(\sum_{i=1}^{n} C_i \times P_i \right) \right)^{-1}$$

- Where
  - A: classification accuracy
  - P_i: parameter value
  - W_A: weight of classification accuracy
  - W_p: parameter weight
  - C_i: feature cost
  - S: setting constant
- When ending condition is satisfied, the operation ends and the optimized NN parameters are produced. Otherwise, the process will continue with the next generation operation



Results: NN GAPO+B

Classifiers	CM1	KC1	KC3	MC2	MW1	PC1	PC2	PC3	PC4
NN	0.713	0.791	0.647	0.71	0.625	0.784	0.918	0.79	0.883
NN GAPO+B	**0.744**	**0.794**	**0.703**	**0.779**	**0.76**	**0.801**	**0.92**	**0.798**	0.871

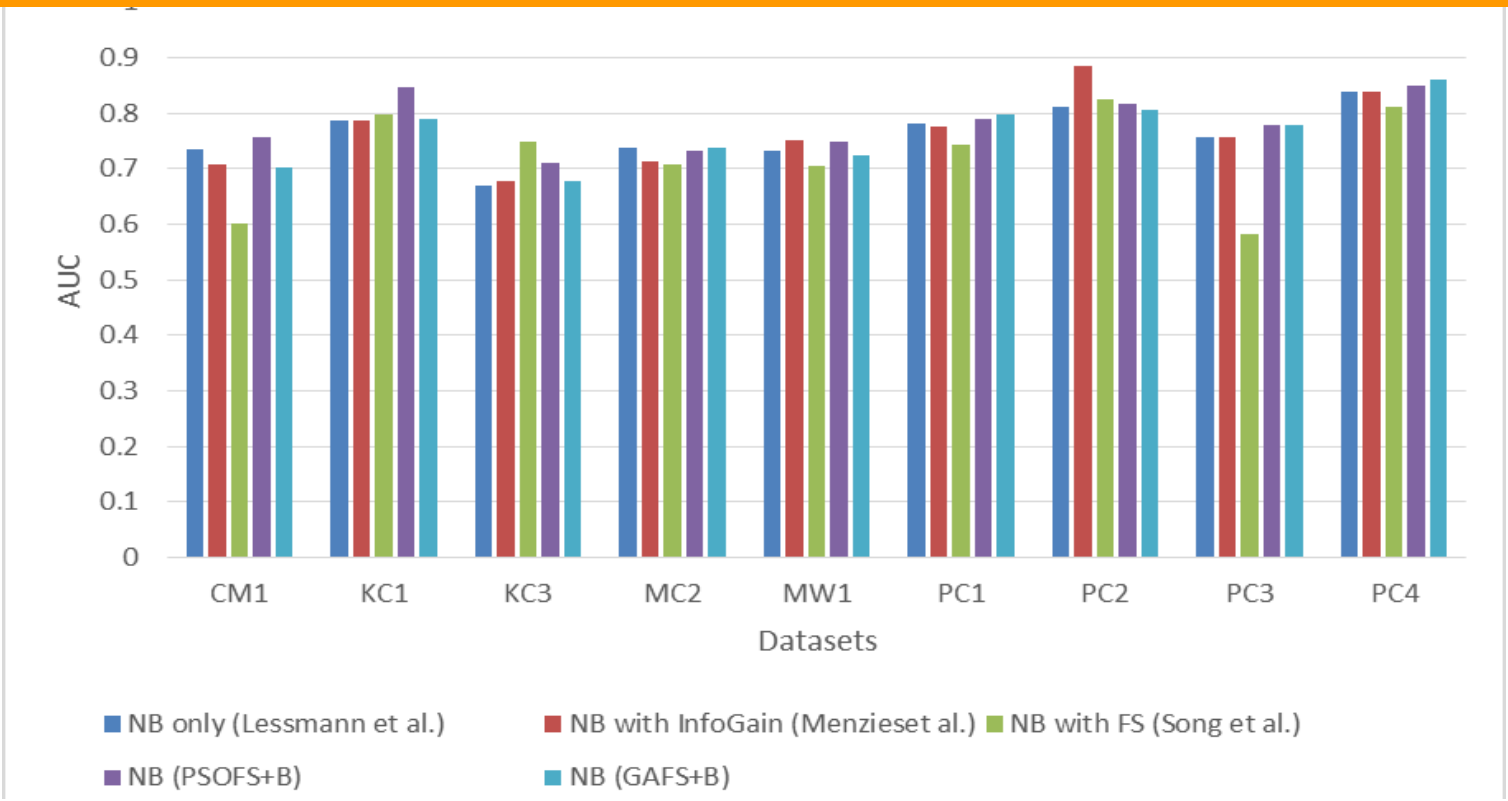


	Variable 1	Variable 2
Mean	0.762333333	0.796666667
Variance	0.009773	0.004246
Observations	9	9
Pearson Correlation	0.923351408	
Hypothesized Mean Difference	0	
df	8	
t Stat	-2.235435933	
P	**0.02791077**	

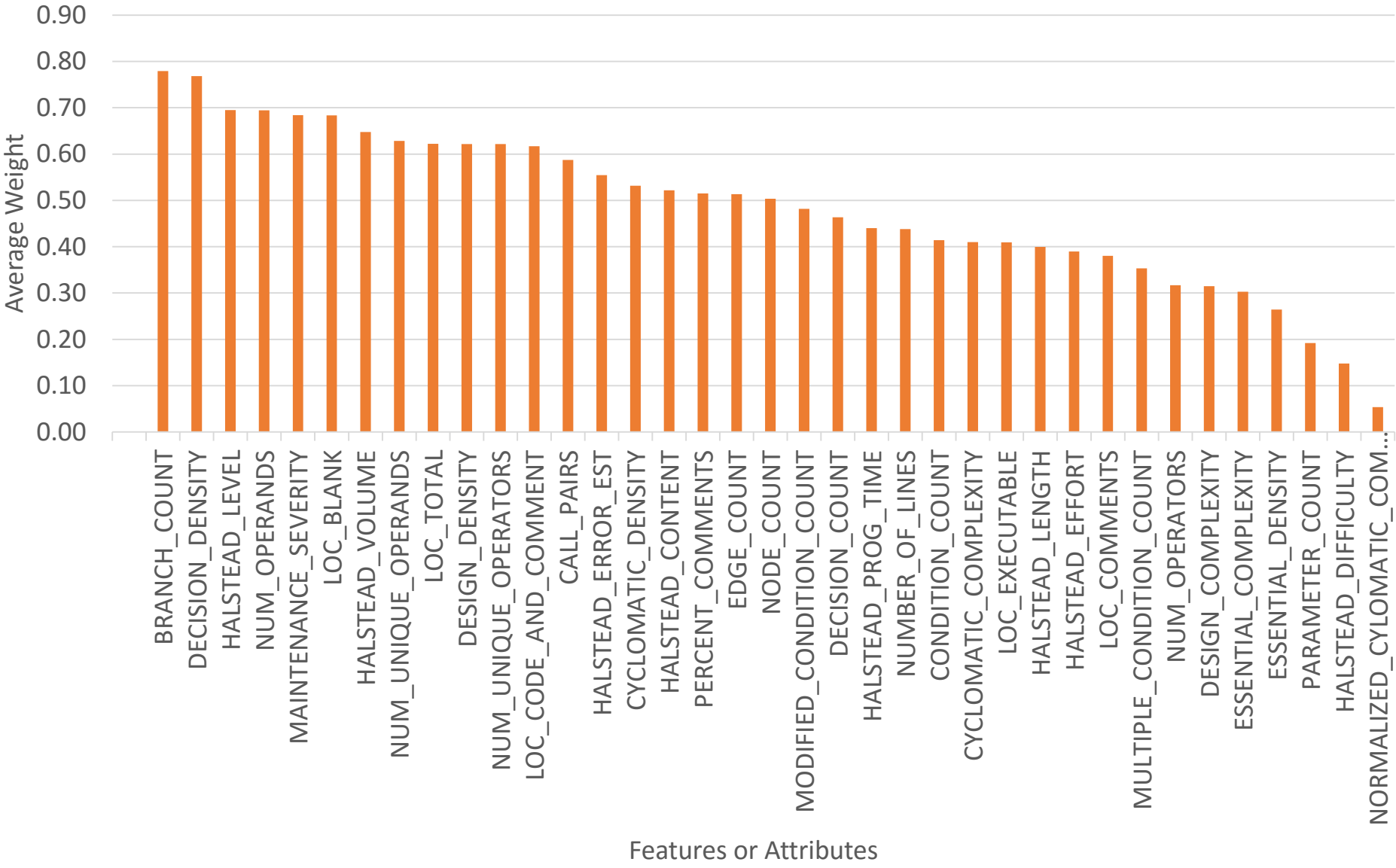
- NN GAPO+B outperforms the original method in almost all datasets
- The proposed (NN GAPO+B) method makes an improvement in prediction performance for back propagation neural network ($P<0.05$)

Framework Comparison

	CM1	KC1	KC3	MC2	MW1	PC1	PC2	PC3	PC4
NB only (Lessmann et al.)	0.734	0.786	0.67	**0.739**	0.732	0.781	0.811	0.756	0.838
NB with InfoGain (Menzies et al.)	0.708	0.786	0.677	0.712	**0.752**	0.775	**0.885**	0.756	0.84
NB with FS (Song et al.)	0.601	0.799	**0.749**	0.707	0.704	0.742	0.824	0.583	0.812
NB (PSOFS+B)	**0.756**	**0.847**	0.71	0.732	0.748	0.79	0.818	**0.78**	0.85
NB (GAFS+B)	0.702	0.79	0.677	**0.739**	0.724	**0.799**	0.805	**0.78**	**0.861**



Relevant Attributes of Software Defect Prediction



Dari Pengembangan Arah Penelitian Baru Menuju ke Kontribusi ke Pengetahuan