Notes on *Ecological inference in empirical software engineering* [1]

Akond Rahman Manish Singh Bennett Narron

August 24, 2015

Abstract

AbstractSoftware systems are decomposed hierarchically, for example, into modules, packages and files. This hierarchical decomposition has a profound influence on evolvability, main- tainability and work assignment. Hierarchical decomposition is thus clearly of central concern for empirical software engineering researchers; but it also poses a quandary. At what level do we study phenomena, such as quality, distribution, collaboration and productivity? At the level of files? packages? or modules? How does the level of study affect the truth, meaning, and relevance of the findings? In other fields it has been found that choosing the wrong level might lead to misleading or fallacious results. Choosing a proper level, for study, is thus vitally important for empirical software engineering research; but this issue hasnt thus far been explicitly investigated. We describe the related idea of ecological inference and ecological fallacy from sociology and epidemiology, and explore its relevance to empirical software engineering; we also present some case studies, using defect and process data from 18 open source projects to illustrate the risks of modeling at an aggregation level in the context of defect prediction, as well as in hypothesis testing.

1 Thesis

Summary of the thesis being investigated along with research questions, goals, and hypotheses. This paper focuses on the importance of ecological inference in software engineering and the risk of ecological fallacy as a

result of mistaken ecological inference. Given that large systems consist of complex software resulting in hierarchical organization of the systems, the teams and the processes to develop the systems, it becomes important to study the ecological inference done at different levels of aggregation and how good the findings hold at other aggregated/dis-aggregated levels. The author emphasizes the importance of selecting the right level of aggregation to conduct empirical studies to measure observable outcomes such as quality and productivity. The paper discusses the various risks that accompany the ecological inference done at the aggregated levels and which result in ecological fallacy when there is discrepancy between the findings at the aggregated and disaggregated levels. The paper also discusses the various factors - sample size, zonation and class imbalance , that contribute to the risk of ecological fallacy.

The goal of this paper is to have a conceptual framework of ecological inference risk in software engineering and empirically demonstrate the existence of this risk , by building prediction models at different aggregation levels and comparing inferences drawn from these models. While pursuing this goal,this paper tries to answer the following research questions:

1. What is the right level of study?
2. How does the level of study affect the truth, meaning and relevance of the findings?
3. Are prediction models subject to ecological inference risk?
4. Is hypothesis testing subject to ecological inference risk?
5. What are the effects of aggregation on model quality?
6. Do inferences drawn from models built at aggregated levels transfer to disaggregated levels used to build the aggregations?

2 Contributions

Summary of the authors' contributions, either implicit or explicit, to ASE

The authors explore the relevance of ecological inference and ecological fallacy in software engineering. The paper discusses their importance in software engineering and what are the risks in using ecological inference in software engineering. The authors further discuss the various factors that contribute to the risk of ecological fallacy - sample size, zonation and class imbalance. The authors conduct experiment , involving 18 open source projects in order to study and understand the incidence of ecological inference in these projects. They further construct models at various aggregation levels and check if the inference derived at an aggregation level holds true for the disaggregated levels that build those levels. They find that there exists a risk of ecological fallacy if the inference is transferred from one level to another. Their findings support the claims that - "Prediction models are subject to ecological inference risk" and "Hypothesis testing is subject to ecological risk".The paper lays out a conceptual framework of ecological risk in software engineering and concludes that ecological inference is unavoidable in software engineering research and coming up with ways to manage and mitigate the risks resulting from ecological fallacy is the way to go ahead.

3 Keywords

We identify the following keywords to be the most significant ones. Each of the keywords are accompanied with definitions.

- ii1. Aggregation Levels:
  A software system that has a hierarchical organization has different layers, where each higher layer is comprised of sub-layer components. The paper defines each of these higher layers as an aggregated level. The paper uses Eclipse as a real-life example: Eclipse si consists of modules, which is a collection of packages. Each package is a collection of files. Here, each package, and file is an aggregated level.

- ii2. Ecological Inference:
  Ecological inference is the empirical finding that is evident at aggregated level of software, as well as, dis-aggregated level of software. For example, in case of ecological inference, if an empirical finding that is evident at package level, which is collection of files, will also be evident at dis-aggregated levels such as files.

- ii3. Ecological Fallacy:
  Ecological fallacy is that particular empirical finding that is evident at aggregated level of software is *not evident* at dis-aggregated level of software. In case of ecological fallacy, if an empirical finding that is evident at package level, will not be evident at dis-aggregated levels such as files.

- ii4. Ecological Inference Risk:
  The paper defines empirical inference risk as generalizing an empirical inference that is evident at aggregated level, to a dis-aggregated level without running the same model with the factors existent at the dis-aggregated level.

4 Brief Notes

We identify the following keywords to be the most significant ones. Each of the keywords are accompanied with definitions.

- iii1. Motivational Statement:
  Researchers have mined large scale software repositories at different levels of the software hierarchy and presented their findings. However, what remains unknown is, how strongly can a hypotheses that was achieved at one level agreegation is applicable to a dis-aggregated level of the software. This paper presents a conceptual framework that investigates the *ideal* level to look for empirical findings, and whether those findings hold for both: aggregated and dis-aggregated levels of the software.

- iii2. Study Instruments:
  The paper used data extracted from the JIRA tracking system abg Github repositories which contained 18 different Apache Software Foundation projects including Cassandra, Lucene, and OpenEJB.

- iii3. Statistical Tests:
  The paper uses hypotheses testing to determine if a certain hypotheses holds at aggregated level and disaggregated level. The authors state that they found a number of cases where the null hypotheses is rejected for aggregated level, which however, was not rejected at dis-aggregated

level. The opposite observation was also observed and reported in the paper.

- iii4. Future Work:
  One possible future direction that can work on top of this paper is replicating the study for other software repositories which follow different hierarchical architectures, and are implemented in different languages. The authors have reported two levels of the hierarchy: files and packages. Another potenial extension of this study can look at the effects of looking at modules and observe if the empirical findings hold.
  The paper has observed and reported the hypotheses derived from an empirical invstigation cannot be generalized. One interesting future work wouldbe coming up with a model that can map these differences in empirical findings using the same factors.

5 Investigation Methods

Summary of the authors' investigation methods or experimental design.

6 Results

Summary of the authors' results

6.1 "Power"

Notes on the impact of the authors' results

6.2 Applicability

Notes on the applicability of the authors' results

7 Technical Development

Summary of the authors' technical development

7.1 Examples

Details of any examples to clarify technical developments

References

[1] D. Posnett, V. Filkov, and P. Devanbu. Ecological inference in empirical software engineering. In *Proceedings of the 2011 26th IEEE/ACM International Conference on Automated Software Engineering*, pages 362–371. IEEE Computer Society, 2011.