Ecological Inference in Empirical Software Engineering
经验软件工程中的生态推理

作者： Daryl Posnett
Vladimir Filkov
Premkumar Devanbu

汇报人：SY1806214 陈鸿超

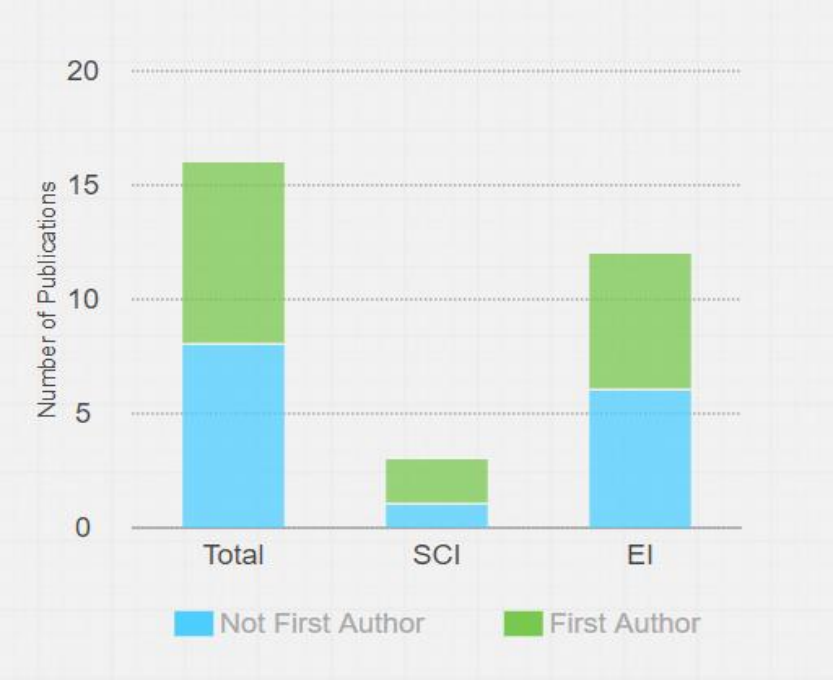
目录

CONTENTS

作者简介

Daryl Posnett

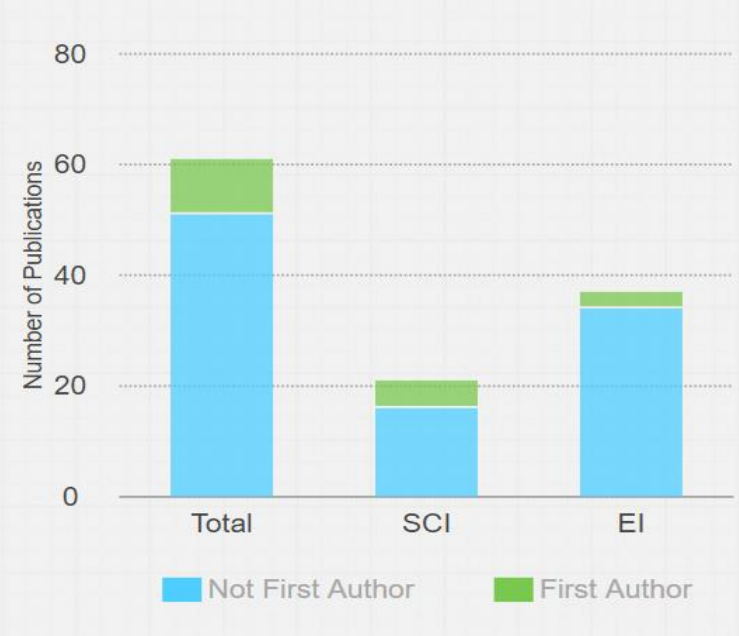
加利福尼亚大学戴维斯分校计算机科学系教授，共发表16篇论文，被引用100余次，主要研究方向包括计算机科学、软件工程、经验过程、软件度量与验证等。



作者简介

Vladimir Filkov

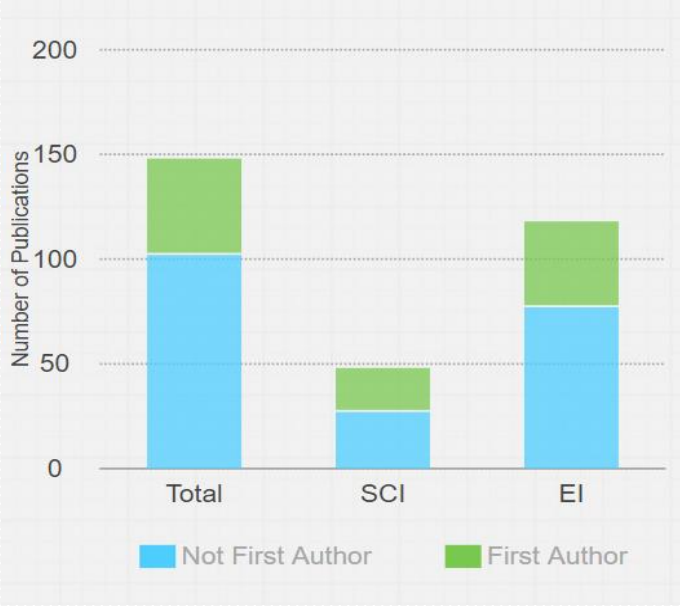
加利福尼亚大学戴维斯分校计算机科学系教授，共发表61篇论文，被引用600余次，主要研究方向包括计算机科学、经验软件工程、计算机生物学、数据分析、复杂应用网络等。



作者简介

Premkumar Devanbu

加利福尼亚大学戴维斯分校计算机科学系教授，共发表148篇论文，被引用2000余次，主要研究方向包括计算机科学、软件工程、数据分析、信息系统、软件验证、软件质量等。



重要术语介绍

在很多的问题研究中，都可以对研究对象进行层次分解。比如研究社会问题，可以把研究对象分解为国家层次、地区层次、国民层次等；研究软件问题，就可以把研究对象分为模块层次、包层次、文件层次等。

这里，聚合程度较高的一些层次叫做**聚合层次**，比如国家层次、模块层次；聚合程度较低的一些层次叫做**分解层次**，比如国民层次、文件层次。

当然，这两个概念本身是相对的，比如地区层次相对国家层次而言是分解层次，相对国民层次而言又是聚合层次。

重要术语介绍

之前介绍了在进行问题研究时，会对研究对象进行层次分解。而在分解之后，很多研究都会先针对聚合层次进行分析，得到成果后再将其应用在分解层次上。

这种在聚合层次上研究得到某种现象或规律后，推断它在分解层次上的现象或规律的行为称为**生态推理(EI)**。但在进行生态推理的过程中，可能会发现聚合层次与分解层次的现象和规律并不相同，这叫做**生态谬论(EF)**。社会学中普遍认为，生态推理是具有一定风险的，在进行问题研究中一定要考虑到这种风险的可能性。

论文目标

对于大型的系统和软件产品而言，模块化分解(或者说是层次分解)对软件开发、团队协调、软件扩展等都是非常重要的部分。比如在Eclipse的代码结构中，就至少有3个分层级别：文件(files)、包(packages)、模块(modules)。

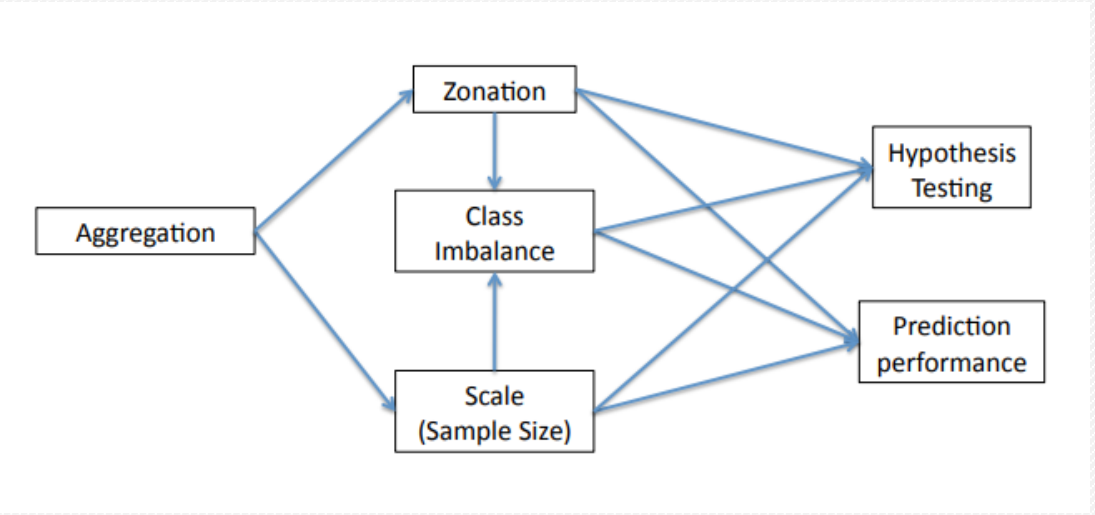
在经验软件工程中，通常会在聚合层次建立模型，进行现象研究，但是其他学科已经证明，生态推理是具有生态谬论的风险，聚合层次的研究成果不一定适合分解层次；分解层次的研究成果不一定适合聚合层次。因此，研究人员需要考虑在什么层次上对软件进行研究？在不同层次上进行研究对结果有什么影响？等等的各种相关问题。

论文目标

不幸的是，这些问题在经验软件工程中还不曾有人进行过明确的研究。因此，本文参考了社会学和流行病学中的生态推理(ecological inference)和生态谬论(ecological fallacy)两个概念，将其与软件工程结合起来，对会产生生态推理风险的因素进行理论分析。并设计了一个验证实验，去证明了经验软件工程中确实存在生态推理的风险，研究人员在进行研究中一定要考虑到这种风险的可能性。

理论分析

本文从理论上分析了三个会产生生态推理风险(也就是生态谬论)的因素：分区、样本量、分类失衡



分区

分区也就是聚合，将较小单位的数据聚合成较大单位的数据，比如将文件聚合成包。在分区的过程中，如果聚合的方式不合适，即聚合在一起的数据过于混杂，会导致模型混乱，无法学习到适合的规律，有效性降低。

理论分析

样本量

在经验软件工程中，通常采用的是统计模型，这就需要足够的数据量去学习有效的规律。但对于聚合层次的数据而已，聚合的程度越高，数据就会越少，因此很多底层的数据才能聚合成一个新数据。而很多经验软件工程中的变量必须在很高的聚合程度上使用，这就导致最终适合模型的数据可能会比较少，无法充分学习到有效的规律。

分类失衡

以缺陷预测为例，对于聚合层次的数据进行标记时，包含至少一个缺陷的实体被标记为有缺陷，不包含缺陷的实体被标记为无缺陷。这种情况下，不同类的标签很少是平衡的，因为大多数实体都是无缺陷的。此时，数据集就会失衡，导致模型的结果更偏向于占比较大的一类数据。

实验验证

因为经验软件工程中研究的方向非常多，无法一一进行分析，因此本文主要针对经验软件工程中最常见的软件缺陷预测模型进行分析验证。

当然，选择该模型还有另一个原因，当前在该领域方面的研究中，很多论文都支持在聚合层次进行分析。他们认为在聚合层次进行分析能够提高模型的效果，克服缺陷分布偏差，获得更有意义的统计结果。

但是，考虑到生态谬论的可能性，在聚合层次进行分析的效果并不一定比分解层次要好，聚合层次的模型也不一定适用于分解层次。这正是本实验希望看到的结果。

实验验证

数据采集

本文从JIRA中收集了18个ASF(Apache Software Foundation)项目的87个不同版本的代码数据，并通过git日志，将每一版项目中修复的缺陷问题与其对应的文件关联起来，标记该文件在该版本中有缺陷。

然后使用SLOCcount工具将文件进行聚合成包。最终将缺陷、文件、包、版本、开发人员都关联起来，借助JIRA跟踪系统进行分析。

同时，本实验将数据点数量不足或者每个类数据点数量不足的数据进行剔除，最终使用了68个版本的数据。

实验验证

Project	Releases	Description	# Releases	# Files	# Packages
Abdera	1.0,1, 1.1	Atom (XML Syndication) implementation	2	672-680	112-113
Cassandra	0.6.0 - 0.6.8	Distributed Database	9	314-332	31-33
Cayenne	3.0, 3.0.1	Java Object Relational Mapping framework	2	2763-2764	160-162
CXF	2.11-2.3.1	Services Framework	17	3086-4097	491-598
HttpCore	4.0.1,4.1	Http Core Library	1	451-451	29-29
Ivy	2.0.0 - 2.2.0	Agile Dependency Manager	3	481-498	65-67
IvyDE	2.0.0, 2.1.0	Eclipse plugin for Ivy	2	118-95	20-23
James	2.3.0 - 3.0	Java Apache Mail Enterprise Server	5	375-477	39-85
Lucene	1.9.1 - 3.0.3	Text search engine library	7	1010-957	102-85
Mahout	0.4	Machine Learning framework	1	1119-1119	147-147
Nutch	1.1, 1.2	Web search software	2	446-453	89-91
ODE	1.2-1.3.4	Business process executor	7	1034-954	122-99
OpenEJB	3.0 - 3.1.3	Enterprise Java Beans	9	2191-2949	124-191
Pluto	2.0.0, 2.0.1	Java Portlet reference implementation	2	370-371	44-44
Shindig	2.0.0, 2.0.1	OpenSocial application	2	811-812	75-75
Solr	1.3.0 -1.4.0	Lucene search server container	2	542-749	33-36
Wicket	1.2.7 - 1.3.7	Web Application Framework	9	1776-1947	240-249
XercesJ	2.7.1 - 2.11.0	Java XML parser	5	740-827	67-71

TABLE I: Apache projects and their description

实验验证

缺陷预测模型设计

Metric	description
LOC	Source lines of code
Lines	Total lines in file/package
# Developers	Number of developers who have edited this file/package
# Active Developers	Number of developers on this file/pkg in current release
Churn	Number of added changed lines
Commits	Count of commits to file/pkg
Features	Number of new features as identified by issue tracker
Improvements	Number of improvements as identified by issue tracker

TABLE II: Metrics gathered and their description.

本文参考经验软件工程现今已有的重多缺陷预测模型，设计了8个需要收集的指标，包括：源代码行数、总行数、文件/包的所有开发者、文件/包的当前版本的开发者、修改行数、文件/包的评论个数、新功能个数、改进功能个数。

然后使用自动选择模型技术对预测变量进行组合测试与模型训练，最终通过分析多种评价指标选择出最拟合的模型。

实验验证

实验设计

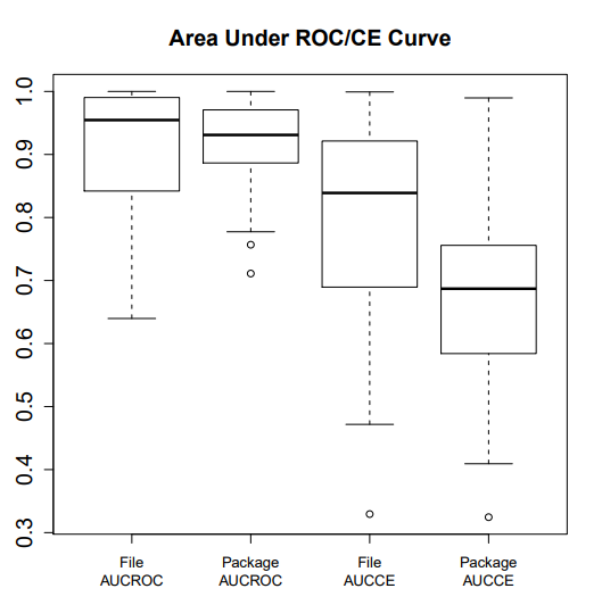
本文设计了两个实验，一个实验用于分析在不同层次上进行研究对模型质量的影响，一个用于研究经验软件工程中的生态推理风险。

对于第一个实验，本文使用之前介绍的方法设计了两个模型，一个是基于聚合层次，对包数据进行分析和预测；一个是基于分解层次，对文件数据进行分析。同时，两个模型采用相同的特征设计、模型选择与评估标准。

对于第二个实验，本文先使用之前介绍的方法得当一个最优的聚合模型，然后使用该聚合模型的变量在分解层次上建立一个分解模型，比较两个对应模型的效果。

结果分析

聚合对模型质量的影响



参数说明

这个图展示了在不同的类别比(正/负类数据的比例)的数据集中，两个模型的AUCROC和AUCCE值。

AUCROC值表示着模型的判别效果，越接近1说明模型对相应层次数据的判别效果越好。

AUCCE值表示着模型的成本收益，一个好的预测模型，不仅需要具有良好的判别效果，还需要在检查最少的代码行后识别最大数量的缺陷问题。AUCCE的值也是越接近1越好。

结果分析

通过上图，我们可以看到，聚合模型判别效果要比分解模型稍微好一些(AUCROC值普遍更大)，但是判定一个模型的好坏不能仅仅依靠判别的效果，还需要考虑到模型的成本效益(AUCEE)。因此，针对本次实验，分解模型要比聚合模型更优秀。

但是在很多研究中，研究人员只专注于模型的判别效果，这就使得聚合模型看起来比它真实的情况更好。

结果分析

生态推理的风险

Type	Predictor	# Releases	# Significant Releases	# Projects	# Significant Projects	Projects
LS	commits	5	26	2	12	abdera, cxf
LS	activedevs	2	32	2	11	abdera, wicket
LS	improvements	5	15	2	6	cxf, openejb
LS	devs	2	16	2	7	cxf, openejb
LS	lines	3	1	2	1	cxf, wicket
LS	features	5	9	4	6	cxf, james, nutch, ode
LS	added	1	8	1	4	cxf
LS	loc	2	1	2	1	cxf, openejb
GS	commits	6	26	5	12	cassandra, ivy, nutch, openejb, wicket
GS	activedevs	4	32	4	11	cassandra, cxf, ivy, wicket
GS	improvements	4	15	3	6	cxf, openejb, wicket
GS	devs	2	16	2	7	ivy, openejb
GS	lines	5	1	4	1	cxf, wicket, abdera, mahout
GS	features	2	9	1	6	cxf
GS	added	1	8	1	4	wicket
GS	loc	4	1	4	1	lucene, cxf, ode, xercesj

参数说明

LS： 该变量在聚合层次中的影响较大，在分解层次中影响较小。

GS：该变量从分解层次到聚合层次，影响增加。

LS和GS都是在说明该变量在两个不同的层次中影响不同，即针对同一组数据，在不同层次上分析得到的模型的参数和统计推论都是不同的，即存在生态谬论。

结果分析

在本实验所得到的68组模型的108个变量中，有28个GS变量和25个LS变量，将近一半的比例。这说明在这68个对比实验中，大多数情况下，在聚合层次分析得到的模型参数和统计推论并不适用于分解层次。

因此，作者得出以下结论：对聚合数据(比如包、模块)进行训练得到的模型、变量等信息可能并不适合直接使用到分解数据(比如文件)上，即生态推理具有一定的风险。

结论

由于软件本质上是分层的，因此在进行经验软件研究的过程中，生态推理是难以避免的。我们需要做的并不是去避免进行生态推理，而是在进行生态推理充分考虑到发送生态谬论的可能性，去研究样本样本大小、分区、分类失衡对生态推理的影响。

贡献

本文的主要贡献有以下几点：

- 详细介绍了生态学中生态推理和生态谬论的概念以及他们与软件工程之间的关系
- 理论分析了几种会引起生态谬论的因素
- 通过实验证明了在经验软件工程研究中进行生态推理的风险

借鉴之处

知识迁移

生态推理本是社会学的一项研究内容，被本文作者应用到了经验软件工程的研究内容之中，并根据其在社会学中的研究成果来推测与证明经验软件工程研究过程中存在的问题。

巧妙的实验设计

在第二个实验中，作者先是在聚合层次上进行分析得到一个**最优**的聚合模型，然后使用该聚合模型的变量在分解层次上建立一个分解模型。巧妙的证明了在聚合层次分析得到的参数和统计推论并不一定适应于分解层次。

问题

本文想要说明的是，在经验软件工程的研究过程中，进行生态推理是具有风险的，即在某个层次上分析得到的规律并不一定适用于其他层次。

对于这个观点，我是支持的，但是，我觉得本文的理论分析和实验验证都不具有说服性。

个人认为想要证明生态推理存在风险，首先要保证前提是成立的，即在某个层次上分析得到的规律必须是正确的。

而本文所做的理论分析(分区、样本量、分类失衡)，更像是导致模型不正确(效果不好)的因素，而不是会产生生态推理风险的因素。

还有本文的第二个实验，如果假设模型都是有效的，那么确实可以得到生态推理的风险确实是存在的结论。但是本文并没有说明这些对比模型的效果，所以假设是否成立也无从得知，并不能让人信服。

THANKS!