ISSN 1673-9418 CODEN JKYTA8
Journal of Frontiers of Computer Science and Technology
1673-9418/2000/00(00)-0000-00
doi: 10.3778/j.issn.1673-9418.

E-mail: fcst@vip.163.com
http://www.ceaj.org
Tel: +86-10-89056056

一种基于文本挖掘的软件失效模式自动生成方法

孟令中[1], 王 航[2], 薛云志[1+], 武 斌[1], 马 兰[3]

1. 中国科学院软件研究所, 北京 100190
2. 中国科学院大学, 北京 100190
3. 清华大学附属小学, 北京 100084

An Automatic Generation Method of Software Failure Mode Based on Text Mining

MENG Lingzhong[1], WANG Hang[2], XUE Yunzhi[1+], WU Bin[1], MA Lan[3]

1. Institute of Software, Chinese Academy of Sciences, Beijing 100190, China
2. University of Chinese Academy of Sciences, Beijing 100190, China
3. Tsinghua University Primary School, Beijing 100084, China
+ Corresponding author: E-mail: yunzhi@iscas.ac.cn

MENG Lingzhong, WANG Hang, XUE Yunzhi, WU Bin, MA Lan. An Automatic Generation Method of Software Failure Mode Based on Text Mining. Journal of Frontiers of Computer Science and Technology, 2000, 0(0): 1-000.

Abstract: The use of software failure mode can help developers and testers to carry out the work of software failure prevention and discovery, and thus improve the quality of software. The usability of the existing software failure mode makes it difficult to play a greater role in the actual work, and the failure mode of the target software is difficult to manually extract. Therefore, the software failure mode is presented in the form of failure cause and failure effect combination, and the software failure mode automatic generation method for software failure text description is proposed by using text mining method. This method uses the software anomaly classification dictionary to construct the classifier to classify the description of the history software failure text. After the pretreatment of the invalid text, the k-means clustering algorithm is used to carry out the text clustering. And then use the text matching method to achieve the automatic generation of failure mode. Finally, the textual extraction of the failure cause and the failure effect is carried out and the automatic generation of the failure mode is realized by the text matching method.

Received 2000-00, Accepted 2000-00.

Key words: Software failure mode; Test mining; Clustering Analysis; Text similarity; Automatic extraction

摘 要:软件失效模式的使用可以帮助开发人员与测试人员能够高效的开展失效的预防和发现等工作,进而提高软件质量。现有软件失效模式的通用性使得在实际工作中难以发挥更大作用,且目标软件的失效模式难以人工提炼,因此提出利用失效原因和失效影响组合的形式表示软件失效模式,并利用文本挖掘方法,提出面向失效文本描述的软件失效模式自动生成方法。该方法通过利用软件异常分类字典构建分类器,用以对历史软件失效文本的描述进行分类;之后对经过预处理后的失效文本,利用 k-means 聚类算法开展文本聚类;进而从聚类后的类簇中选取有代表性的失效文本作为类簇标签;最后开展失效原因和失效影响的文本抽取并通过文本匹配方法实现失效模式的自动生成。

关键词: 软件失效模式; 文本挖掘; 聚类分析; 文本相似度; 自动抽取

文献标志码:A **中图分类号**:TP 311.5

1 引言

当今社会,软件应用十分广泛,已经成为影响国民经济、政治乃至社会生活的重要因素。随着软件应用领域的不断扩大,软件的开发和运行环境更加多元化,信息化需求的增加导致了软件的规模和复杂性都有了巨大的增长。影响软件质量的因素很多,软件本身不可避免的存在缺陷,由于系统对软件有着强烈依赖,这种已有的或者潜在的缺陷一旦触发势必造成软件的失效,将导致各领域依赖的软件业务陷入瘫痪,有可能造成严重的后果。

软件失效的机理是:软件产品中存在软件缺陷,引起软件在运行阶段发生故障,当软件故障没有被系统所容忍,并引发一个或多个失效行为,最终表现出系统失效。根据 IEEE 标准中的定义,软件缺陷指计算机程序中一个不正确的步骤、过程或数据定义[1];软件故障是系统在运行时由于缺陷造成的非正常状态的表现和反映;软件失效是指程序的运行偏离了需求,是动态运行的结果[2],当软件故障没有被系统所容忍,并引发一个或多个失效行为,最终表现出系统失效。

根据美国国家技术标准研究院的一项统计,平均每年美国花费在减少各个领域软件出现失效的经费在GDP的大约0.6%。所以对软件失效的预测、发现及预防是及其重要的。但是由于软件缺陷的产生会因编程人员的不同而不同、因开发语言的不同而不同、因开发对象的领域不同而具有特定的缺陷。同时由于受到各类资源与软件交付时间的限制,很难对设计的每一个模块每一个操作都做到完美的测试;即使有了充足的时间与人力,也并不是每个开发人员或测试人员都具备发现所有软件失效的能力。但是随着软件项目中开发和测试工作的不断开展与深入,软件开发团队和测试团队会积累越来越多的有关软件失效的数据,这就为预测、发现及预防软件失效提供了可能。

模式是针对复杂系统中重复出现的问题而提出的一个概念,用以描述不断重复发生的问题。在实际工程实际中,有经验的专家或工作人员在分析问题和解决问题时,通常先考虑以前发生过的类似问题,并重用该问题的解决方案以解决问题,这个不断被使用的解决方法通常称之为模式。软件失效模式(software failure mode)指软件失效发生的不同方式,是可以从不断重复出现的或类似的软件失效中发现和抽象出的规律描述。

在国军标 GJB/Z 1391-2006 中,给出了嵌入式软件系统的失效模式,并分为输入失效、输出失效、程序失效、未满足功能及性能要求失效和其他类型等[3]。但是该标准给出的失效模式为通用的失效模式,仅包含软件失效影响,以输入失效为例,包括:1)未收到输入;2)收到错误输入;3)收到数据轻微超差;4)收到数据中度超差;等。因此在实际工作存在以下问题: 1)仅是对软件失效影响的模式

描述，缺少相关的失效原因的模式描述，因此对于软件开发人员与测试人员来说，还需要更多的工作经验与项目背景去进行分析以发现软件失效；2）难以满足特定领域的软件失效的预测与测试工作。同时随着软件规模与软件复杂度的增加，单个项目的软件失效数目剧增，传统的利用人工方式总结失效模式存在工作效率低、过程繁琐及受到专业技能限制等问题。

因此，为了能够高效及准确的预测和发现实际软件项目中的软件失效，本文利用“失效原因”和“失效影响”的组合来表示失效模式，增强对软件失效模式的描述。并提出了一种软件失效模式自动生成方法。该方法面向目标软件在软件开发和测试阶段发现的大量软件失效的文本描述，利用建立的异常分类器对软件失效进行分类，并对分类后的文本描述进行聚类分析，自动抽取出失效影响和失效原因，最终构建成更为完善的面向该目标软件领域的软件失效模式库。

本文组织结构如下：第 2 章介绍相关工作；第 3 章介绍核心方法与相关算法；第 4 章介绍实验与分析；最后进行工作总结及未来工作的改进方向。

2 相关工作

本文涉及的相关工作包括文本信息挖掘在信息领域的应用及软件失效模式分析研究。

在计算机信息领域，文献[4]综述了文本挖掘技术中文本分类、摘要提取、主题检测、搜索及文档聚类的方法，并介绍了文本挖掘技术在电信、推荐系统、信息检索、媒体等领域的应用；文献[5]综述了在现阶段拥有大量新的数据来源的情况下，可以通过文本挖掘方法开展自动化的数据汇总和分析工作，有利于知识库的扩展；文献[6]针对网络新闻的原始文本进行预处理，使用文本挖掘技术自动提取半结构化信息，提出了欺诈和盗窃者的行为模式与属性统计，建立了危险身份行为模型，用于阻止未来的危险事件发生；文献[7]应用文本挖掘技术和潜在语义分析算法自动提取海量历史数据中的场景描述，并利用模糊关联规则进行场景分析；文献[8]通过面向相关文献的开展文本挖掘工作自动获取了数据集用以提供分类器，以解决蛋白质-蛋白质相互作用的数据繁多和嘈杂的问题。

在软件失效研究领域，为解决软件可靠性建模过程中所需要的不同历史数据库中的软件的准确失效时间，文献[9]提出了一种文本挖掘方法构建相应的关键词字典及建立分类器，用以自动提取来自详细描述系统维护活动的工作单和停机数据记录，进而自动识别工作单中的故障；文献[10]针对测试人员发现的无效失效降低开发效率的问题，利用文本挖掘方法建立了无效失效分类器，并利用决策树方法对新的无效失效进行分类，以提高项目组的生产力；文献[11]使用文本挖掘技术检测恶意软件的动态行为并进行量化分析，进而建立检测决策模型用以提高恶意软件的检测率；文献[12]利用演进树的神经网络模型，将失效模式进行聚类并可视化为树形结构，并对失效模式进行分组排序，更好的进行失效模式影响分析；文献[13]通过分析真实的操作系统失效，提出失效模式协议方法，用于发现不同工作场景下的呈现一致性的失效，以分析操作系统失效背后的机制；文献[14]针对航空电子系统创建一个数据驱动的反馈回路，分析来自飞行器的传感器和仪器的时空数据流，进而发现失效模型用于在可能的情况下纠正错误的数据。

文本聚类是指将文本聚合为由若干个文本簇组成的集合的过程，因为同一类的文本具有很大的相似性，不同类文本具有一定的相异性。文本聚类主要包括两个步骤：1）是对文本进行预处理，将文本表示为计算机可以识别及处理的形式；2）采用机器学习中的相关算法对形式化的文本进行聚类分析。K-means 是典型的基于距离的聚类算法，采用距离作为相似性的评价指标，即两个对象的距离越近其相似度就越大。

基于知网的词语语义相似度计算是利用知网中义原之间的关系进行语义计算及推理[15]。该方法可以将两个语义表达式的整体相似度分解成一

些义原对的相似度的组合，进而采用根据上下位关系得到语义距离并进行转换。

3 失效模式自动生成方法

为了更好的表示软件失效模式，本文提出利用失效原因（Failure Cause，FC）和失效影响（Failure Effect，FE）一起描述软件失效模式。其中失效原因指整个系统环境中导致产生某个特定失效模式的基本原因，包括外部设备环境、软件系统的输入和输出、系统逻辑设计及程序运行等多个方面；失效影响指某个失效模式对一个系统中操作、功能或状态所造成的一系列结果，失效影响可以分为局部影响、对上一层系统的影响以及对整个系统的影响[16]。

3.1 软件失效文本描述特点

领域内的软件失效数据更多的是来源于多年来软件开发团队和测试团队在实际工作中所积累的丰富的失效信息。

这些软件失效信息不仅包含一些数字信息，如报告日期、被测版本信息等，更重要的是有关失效的文本描述。这些失效文本描述具有以下特点：

1. 失效描述短小。通常有关失效的文本描述具有较少的字数，一般都在 150 字以内，是典型的短文本；

2. 有较少的语法错误和错别字。与微博或微信朋友圈随意发布不同，失效描述一般都经过测试人员认真编写，且有的失效描述经过开发人员与测试人员多次探讨，因此软件失效的描述一般相对准确，不存在语法错误；

3. 语义专业化。软件失效都是针对特定被测系统的描述，一般使用的词汇都是软件测试领域的专业词汇，且由于软件失效描述的专业性，其描述的软件失效一般无歧义性；

4. 语义完整性。软件失效描述与其他短文本具有语义碎片化不用，一般一条软件失效的描述是针对一个软件失效的情况进行详细说明。虽然字数较少，但通常包含一个完整的主题即一个软件失效，且清晰明确的表达了失效的情况；

5. 结构统一性。在软件失效描述文本的前部一般是对软件操作的描述，即失效原因的描述，后部一般是失效影响的描述，同时文本中间会有明确的标点作为分隔符。

正是由于软件失效的文本描述具有以上特点，使得利用文本挖掘技术与机器学习算法用于软件失效模式的自动提取存在可行性。

3.2 软件失效模式自动生成方法

软件失效模式自动生成是将短文本聚类分析技术应用于软件失效的描述分析中。主要的关键核心工作包括领域字典的构建和面向失效文本描述特性的聚类算法改进。其工作流程如图 1 所示。

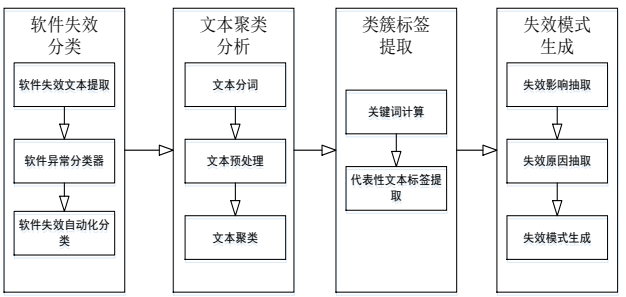

Fig. 1 Software failure mode automatic generation method

图 1 软件失效模式自动生成方法

3.2.1 软件失效分类方法

不同类型的软件在发生失效的时候，所表现出的异常情况也是不同的，其失效的文本描述内容也必然有所区别，因此为了提高后续失效模式聚类的准确性及减少聚类过程中的离群点，需要首先将软件失效按照发生异常的情况进行分类。

步骤 1 软件失效文本描述的提取

软件失效的文本描述一般为结构化或半结构化信息，其中保存在管理工具（如 Bugzilla、Mantis 或其他缺陷系统）中的失效文本描述为结构化信息，可以通过数据库直接导出；以软件问题报告单（如第三方测评中心提交的成果）等文档形式保存的因其一般都根据文档模板进行编写故而为半结构化信息，这些软件失效的文本描述可以通过脚本语言进行批量获取。在提取过程中，需要对数据进行初步清洗，如去除失效文本描述为空、失效文本描述

小于 10 个汉字、非文本格式的文件、编码异常数据等。将获取的软件失效文本保存在数据库中，为后续的失效模式自动生成做准备。

步骤 2 软件异常分类器的建立

软件失效的异常分类可以参考 IEEE[1]和 GJB/Z 1391[3]提供的异常分类并结合目标软件的特性进行裁剪，建立适应于该目标软件的异常分类。

其中，IEEE 提供的 9 种异常分类如下：

1. 操作系统挂起；
2. 程序挂起；
3. 程序失败；
4. 输入问题；
5. 输出问题；
6. 未达到要求的性能；
7. 发现的整个产品失败；
8. 系统错误信息；
9. 其他。

GJB/Z 1391 提供的 5 种异常分类如下：

1. 输入失效；
2. 输出失效；
3. 程序失效；
4. 未满足功能及性能要求失效；
5. 其他。

依据以上分类的详细说明，针对每一种软件异常构建其对应的异常分类字典 $Dic.Abnormal_n$，其中 n 表示异常分类的数量。同时可以根据目标软件项目的历史失效，进行相应的补充完善工作，构建更加全面的异常分类字典 $Dic.Abnormal_n$。由于不同的失效原因可能会导致相同的失效影响，因此利用词语组合的形式构建异常字典，用以区分不同的异常分类情况。例如“输入”+“异常”为输入失效的异常分类字典 $Dic.Abnormal_1$；“程序”+“异常”为程序失效的异常分类字典 $Dic.Abnormal_3$。最后利用多个异常分类字典建立一个决策树分类器。

步骤 3 软件失效的异常分类

将每一个软件失效文本描述放入到异常分类器中进行软件异常分类。由于建立的分类器是基于失效影响的描述，因此在分类中需要加入失效文本描述中关键词位置的权重 a，可以得出文本后部的系数较高。通过加入该权重的计算，可以确保每一个失效文本描述最多属于一个软件异常。

相关算法如下：

Input：待分类的失效文本描述，期望阈值 M，字典 $Dic.Abnormal_n$，文本位置权重系数 a；

Step1：失效文本描述放入软件异常决策树分类器中，并考虑权重系数 a 进行计算并分类；

Step2：若低于 M，则认为该失效不属于任何异常情况，则丢弃该失效文本描述；若不低于 M，则将该失效划入到相应的软件异常中；

Output：失效文本描述所属失效类别。

3.2.2 失效文本聚类分析方法

在对软件失效文本进行异常分类后，对每一个异常分类下的失效文本描述进行聚类分析。

步骤 1 失效文本分词

软件失效文本描述具有语义专业化的特点，为减少文本分析过程中中文分词不准确的问题，本文利用搜狗输入法提供的计算机领域词包建立计算机领域关键词词典 Dic.Computer，并加入到公开的 NLPIR 分词系统中。利用该改进的 NLPIR 分析系统对失效文本描述进行分词，并保存到数据库中。

步骤 2 文本预处理

由于软件失效文本描述相对短小，内容精炼且有较少的语法错误和错别字，因此在预处理过程中不需要进行去掉低频词、替换奇异词等数据处理步骤。

停用词过滤是文本预处理的关键步骤，本文采用哈工大公开发表的停用词表作为基础。由于在失效文本描述中，时间副词（如总是、同时等）、范围副词（都、只、一起等）及否定副词（没有、无法、不、不能等）对失效原因或失效影响的含义有重要影响，因此对停用词表进行改进，将部分副词从停用词表中去除。之后利用改进后的停用词表对分词后失效文本进行处理。

步骤 3 文本聚类

针对每一种异常下的失效文本集合，对经过预处理后的失效文本进行 K-means 聚类。本文采用基

于知网的词语语义相似度计算方法得出文本相似度，用以表示聚类过程中的距离。

3.2.3 类簇标签提取方法

对每一个类簇进行标签提取工作，标签提取的目的是为了后续的软件失效原因的抽取。利用代表性样本作为每一个类簇的标签。

步骤 1 词计算

将一个类簇内所有样本的词语合并成一个集合，利用文档频率和词性等属性对集合内的每一个词语进行计算。其中文档频率指给定词汇的文本描述个数，文档频率较高的词一般可以很好的体现该类簇的特征；词性，名词和动词可以较好的表达主题，因此会设定较高的分值。

之后对每一个失效文本中的词语进行词语位置权重计算。根据对软件失效文本描述的分析，可以发现绝大部分描述失效原因的文本位于整个文本的前半部分，由于类簇标签提取的目的是为了软件失效原因的抽取，因此对于前面部分的词语设定较高的分值。

步骤 2 代表性文本提取

将步骤 1 计算得出的类簇内每个词语的分值作为得分初始值。利用前文建立的计算机领域关键词字典 Dic.Computer，为其建立相应的初始分值，用于对每个文本描述中出现 Dic.Computer 中的词语进行额外加分。之后，将每一个文本的特征词集合中所有词语得分的算数平均值进行排序，选取初始失效文本描述长度大于 20 个字且分值排序最高的前若干个文本作为该类簇的代表性文本。

3.2.4 软件失效模式自动生成方法

本文定义的失效模式是由“失效原因”和“失效影响”组合而成。同时软件失效文本描述具有语义完整与结构统一的特性，这为分别抽取“失效原因”和“失效影响”提供了可行途径。

步骤 1 软件失效影响的抽取

异常分类字典 $Dic.Abnormal_n$ 是失效影响分析的关键词组合的集合，由于集合中存在语义相近的关键词组合，因此需要进行语义相似度计算，设定相似度阈值 M 并合并相似度较高的关键词组合。合并后的关键词组合即作为失效影响集合并用 $Set(FE_m)_n$ 表示，即第 n 个异常分类中共有 m 个失效影响。

步骤 2 软件失效原因的抽取

利用 3.2.3 节中获取的类簇标签，即前若干个代表性文本开展软件失效原因的抽取。设定相似度阈值 M，对代表性文本之间进行相似度计算，相关算法依旧选择基于知网的词语语义相似度计算方法。在迭代完成相似度大于 M 的文本合并后，输出剩余的合并后的文本集合，作为该类簇的失效原因的文本集合。最后将该异常分类下所有类簇进行以上计算后的组合的文本进行汇总，则汇总后的集合即可作为软件失效原因集合并用 $Set(FC_i)_n$ 表示，即第 n 个异常分类中共有 i 个失效原因。

步骤 3 软件失效模式自动生成

计算 $Set(FE_m)_n$ 和 $Set(FC_i)_n$ 的笛卡尔积，并将得出的集合记为 $Set(FC_i*FE_m)_n$。给定相似度阈值 M，并依次将该类簇内的失效文本与 $Set(FC_i*FE_m)_n$ 进行相似度比较，当大于 M 时，即可得到第 n 个异常下的一个软件失效模式，其失效模式的文本描述由 FC_j 和 FE_K 组成，其中 $0 < j \leq i, 0 < k \leq m$。

4 实验分析

实验数据来源于国家“核高基”科技重大专项课题中实际测试工作中的软件失效描述。测试工作持续 3 年以上且测试内容很广，软件失效数据对该目标软件具有很强的代表性。

4.1 软件失效分类

从缺陷管理工具 Mantis 中导出所有的软件失效描述，并去除垃圾数据，得到 6739 条失效文本描述。

考虑该软件的特性和使用范围，建立 6 种软件异常用于软件失效的异常分类，分别是输入失效、输出失效、程序失效、未满足功能及性能要求、用户体验不佳及其他。同时相应的构建 6 个异常字典

$Dic.Abnormal_n$作为软件异常分类器，每个异常分类字典包括的关键词组合数目如表 1 所示。并以未满足功能及性能要求的异常关键字字典$Dic.Abnormal_4$为例进行说明，如表 2 所示。

Table 1 Abnormal classification dictionary statistics
表 1 异常分类字典的统计信息

编号	异常类型	关键词组个数
1.	输入异常	44
2.	输出异常	103
3.	程序故障	85
4.	未满足功能及性能要求	162
5.	用户体验不佳	48
6.	其他	55

Table 2 Category 4 exclusive category dictionary content (Partial)
表 2 第四类异常分类字典（部分）

关键词组合	关键词组合	关键词组合
内容-未汉化	无法-恢复	大于-基准值
启动-缓慢	无法-启动	显示-不全
菜单-未汉化	无法-解析	不是-最优
显示-异常	无法-显示	设置-未生效
功能-无效	无法-打开	用时-过长
添加-失败	配置-失败	修改-不成功

将 6739 条失效文本放入到软件异常分类器中，同时给定阈值 M=5，对于位置权重 a 进行如下设定：当关键词处于文本的前三分之一时，a=0.7；当关键词处于文本的后三分之一时，a=2；其余部分 a=1。进行分类计算后，共有 6187 条软件失效被分入到 6 种异常中，其余 545 条软件失效由于低于阈值，不属于任何异常。每类软件异常中的软件失效文本描述数量如表 3 所示。

Table 3 Abnormal classification statistics of software failure text
表 3 软件失效文本的异常分类统计

编号	异常类型	失效文本数	所在比例
1.	输入异常	378	6.1%
2.	输出异常	1515	24.5%
3.	程序故障	1177	19.1%
4.	未满足功能及性能要求	2419	39.1%
5.	用户体验不佳	448	7.2%
6.	其他	250	4.0%

4.2 失效文本聚类

首先利用改进的 NLPIR 分词系统对每类异常中的失效文本描述进行分词，之后利用改进的停用词表对分词后的文本进行预处理，将经过处理后的文本信息进行保存。

利用基于知网的词语语义对相似度进行计算，同时经过多次试验，将失效文本数/30 作为每类异常的类簇个数，开始对每类异常中的失效文本开展 K-means 聚类分析。

后续均以第四类异常为例，其中聚类个数设定为 80 个（2419/30 ≈ 80），统计每个类簇的样本数量如图 2 所示。其中失效文本的样本总和在 20 到 40 区间内的类簇个数最多，达到了 44 个，样本数小于 40 个类簇个数为 64 个，占总类簇个数的 80%。

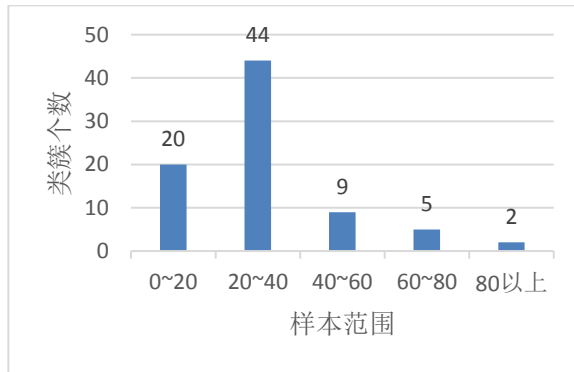

Fig. 2 The number of failure text in cluster
图 2 类簇中失效文本个数分布

4.3 类簇标签提取

对每个异常分类下的每个类簇内进行词计算。其中对于词语位置权重 a 进行如下设定：当关键词处于文本的前三分之一时，a=2；当关键词处于文本的后三分之一时，a=0.7；其余部分 a=1。

同时给定计算机领域关键词字典 Dic.Computer 内词语的额外值均为 1，计算得出每一个失效文本的词语的算数平均值。

以第四类异常为例，由于类簇内样本总数小于20的类簇占总类簇的25%，样本总数小于40的类簇个数占总类簇的80%，同时类簇内的文本有较高的相似度，因此选择每个类簇内排序最高的 10 个文本作为该类簇的代表性文本。以第四类簇内的类簇22的部分代表性文本为例，如表 4 所示。

Table 4 Class 22 cluster representative text (part)

表 4 第 22 类簇代表性文本（部分）

序号	代表性文本
1.	蓝牙、设备、窗口、菜单、设置、内容、未汉化
2.	图片、查看器、打印、对话框、存在、汉化、错误
3.	截图、帮助、页面、汉化、不完整、名称、不一致
4.	图像、查看器、菜单、内容、未汉化
5.	文件、管理器、帮助、页面、标题、汉化、错误

4.4 软件失效模式自动生成

利用基于知网的词语语义相似度计算对异常分类字典 $Dic.Abnormal_n$ 内的关键词组合进行合并，避免出现类似的失效影响。以表 2 为例，经过相似度计算合并后的关键词组合如表 5 所示。例如，表 2 中关键词词组“内容-未汉化”和 “菜单-未汉化” 在计算相似度得分时较高，被合并成一个关键词词组“内容-未汉化”。

Table 5 The fourth class of abnormal dictionary after the merger(part)

表 5 合并后的第四类异常字典（部分）

关键词组合	关键词组合	关键词组合
内容-未汉化	配置-失败	大于-基准值
启动-缓慢	无法-启动	不是-最优
显示-异常	无法-解析	功能-无效

经过计算合并后的第四类异常的失效影响集合为 $Set(FE_{38})_4$，由 162 个关键词词组合并为 38 个关键词词组，即 38 个失效影响。

给定相似度阈值 M 为=7，并对每个类簇内选择的排序前 10 的文本进行相似度计算并合并，将合并后的文本集合作为该类簇的失效原因。将第四类异常分类下的 80 个类簇均完成相应的类簇内文本合后的集合记为 $Set(FC_{287})_4$，即第四类异常抽取出 287 个失效原因。

计算 $Set(FE_{38})_4$ 和 $Set(FC_{287})_4$ 的笛卡尔积，并记为 $Set(FC_{287}*FE_{38})_4$。依次将第四类类簇的失效文本与 $Set(FC_{287}*FE_{38})_4$ 进行相似度计算，当大于给定的阈值 M=8 时，即可得到一个相应的软件失效，且利用失效原因和失效影响对失效模式进行描述，自动生成的部分第四类软件失效模式如表 6 所示。

Table 6 Automatically generated failure mode(part)

表 6 自动生成的失效模式（部分）

序号	软件失效模式
1.	手机助手导入信息——功能无效
2.	邮件客户端最大化——显示异常
3.	图片查看器菜单——未汉化
4.	杀毒软件设置——配置失败
5.	系统开机——启动缓慢

根据上述计算工作，可以得出第四类异常下的软件失效模式 172 条。利用同样的方法对其他异常下的文本进行失效模式自动生成工作，最后该实验自动生成失效模式共计 428 条。

5 结束语

本文提出了利用“失效原因”和“失效影响”组合的形式对失效模式进行详细描述，为此提出了基于文本挖掘的软件失效模式自动生成方法。该失效模式的描述方式相比于现有的国内外标准，可以更好的帮助技术人员开展目标软件的失效预防和发现等工作；同时可以解决传统人工分析和总结失效模式中效率低下、需要专业经验及过程繁琐的问题。但在实际工作中，对软件失效文本描述的异常分类强烈依赖于依据异常分类字典 $Dic.Abnormal_n$ 构建的分类器，因此为提高异常分类的准确率和召回率，后期需要分析更多的失效影响数据不断训练该分

类器；同时软件失效模式强烈依赖于发现的软件失效的数量，失效文本描述的样本越多，越能开展聚类工作，也就越能准确的抽取出失效模型，因此对于失效文本描述样本数较少的软件系统，使用该方法具有一定的局限性。

未来的研究将在本文的基础上，不仅对关键的失效文本进行了分析，而且拟将软件失效中包含的其他信息，如测试类型、测试对象和失效严酷度等均作为有关帮助信息加入到失效模式的分析中，提高失效文本聚类的准确性，并最终提高目标软件的失效模式自动生成的能力。

References:

[1]IEEE Std 610.12，IEEE standard glossary of software engineering terminology[S], New York, NY, USA:IEEE, 2002.

[2]LU minyan. Software reliability engineering [M]. Frist edition. Beijing：National Defence Industry Press, 2011.

[3]GJB/Z 1391-2006 Guide to failure mode, effects and criticality analysis[S]. Beijing: The General Armament Department of the people's Liberation Army Chinese, 2006.

[4]Hussein Hashimi, Alaaeldin Hafez, Hassan Mathkour. Selection criteria for text mining approaches [J]. Computers in Human Behavior. 2015, 51 (Part B): 729-733.

[5]Victoria Kayser, Knut Blind. Extending the knowledge base of foresight: The contribution of text mining [J]. Technological Forecasting and Social Change. 2017, 116: 208-215.

[6]Razieh Nokhbeh Zaeem, Monisha Manoharan, Yongpeng Yang, et al. Modeling and analysis of identity threat behaviors through text mining of identity theft stories [J]. Computers & Security. 2017, 65: 50-63.

[7]Jieun Kim, Mintak Han, Youngjo Lee, et al. Futuristic data-driven scenario building: Incorporating text mining and fuzzy association rule mining into fuzzy cognitive map [J]. Expert Systems with Applications. 2016, 57: 311-323.

[8]Renu Vyas, Sanket Bapat, Esha Jain, et al. Building and analysis of protein-protein interactions related to diabetes mellitus using support vector machine, biomedical text mining and network analysis [J]. Computational Biology and Chemistry. 2016, 65: 37-44.

[9]Kazi Arif-Uz-Zaman, Michael E. Cholette, Lin Ma, et al. Extracting failure time data from industrial maintenance records using text mining [J]. Advanced Engineering Informatics. 2016.

[10]Yihsiung Su, Pin Luarn, Yue-Shi Lee, et al. Creating an invalid defect classification model using text mining on server development[J]. Journal of Systems and Software. 2017, 125: 197-206.

[11]S. P. Choudhary, Deepti Vidyarthi. A Simple Method for Detection of Metamorphic Malware using Dynamic Analysis and Text Mining [J]. Procedia Computer Science. 2015, 54: 265-270.

[12]Wui Lee Chang, Kai Meng Tay, Chee Peng Lim. Clustering and visualization of failure modes using an evolving tree [J]. Expert Systems with Applications. 2015, 42（20）:7235-7244.

[13]Caio Augusto Rodrigues Dos Santos, Rivalino Matias Jr. Failure patterns in operating systems: An exploratory and observational study [J]. Journal of Systems and Software. 2017.

[14]Shigeru Imai, Alessandro Galli, Carlos A. Varela. Dynamic Data-driven Avionics Systems: Inferring Failure Modes from Data Streams [J]. Procedia Computer Science. 2015, 51: 1665-1674.

[15]Liu qun, Li sujian. Word similarity computing based on hownet. International Journal of Computational Linguistics & Chinese Language Processing, 2002, 7(2):59-76.

[16]NASA. Software Safety Guidebook [M], 2004.

附中文参考文献：

[2]陆民燕.软件可靠性工程[M]. 第一版. 北京:国防工业出版社, 2011.

[3] GJB/Z 1391-2006 故障模式、影响及危害性分析指南[S].北京：中国人民解放军总装备部，2006.

[15]刘群，李素建. 基于《知网》的词汇语义相似度计算[J]. 中文计算语言学期刊，2002，7（2）:59-76

MENG Lingzhong was born in 1981. He received the Ph.D. degree in system engineering from Beijing University of Aeronautics and Astronautics in 2012. He is a research assistant in Institute of Software, Chinese Academy of Sciences. His research interests include trustworthy software and artificial intelligence.

孟令中(1981–)，男，河北省石家庄市人，2012 年于北京航空航天大学获博士学位，现任中国科学院软件研究所助理研究员，主要研究领域为可信软件和人工智能。

WANG Hang was born in 1991. He received the M.S. degree in software engineering from University of Chinese Academy of Sciences in 2017. Her research interests include software engineering and natural language processing.

王航(1991–), 男, 福建省福州市人, 2017 年中国科学院软件研究所获硕士学位, 主要研究领域为软件工程, 自然语言处理。

XUE Yunzhi was born in 1979. He received the Ph.D. degree from Institute of Software, Chinese Academy of Sciences in 2009. He is a professor and M.S. supervisor at Institute of Software, Chinese Academy of Sciences. His research interests is artificial intelligence.

薛云志(1979 年–), 男, 山西省运城市人, 2009 年于中国科学院软件研究所获博士学位, 现任中国科学院软件研究所研究员, 硕士生导师, 主要研究领域为人工智能。

WU Bin was born in 1982. He received the Ph.D. degree from University of Science and Technology of China in 2010. He is a professor of engineering at Institute of Software, Chinese Academy of Sciences. His research interests include distributed calculating and artificial intelligence.

武斌(1982 年–), 男, 黑龙江省哈尔滨市人, 2010 年于中国科学技术大学获博士学位, 现任中国科学院软件研究所正高级工程师, 主要研究领域为分布式计算和人工智能。

MA Lan was born in 1978. She received the M.S. degree from Institute of Software, Chinese Academy of Sciences in 2004. She is an engineer at Tsinghua University Primary School. Her research interests include online education, artificial intelligence and education.

马兰(1978 年–), 女, 湖北省荆州人, 2004 年于中国科学院软件研究所获硕士学位, 曾任职清华大学公共管理学院, 现任职清华大学附小, 信息中心主任, 工程师职称, 主要研究领域为人工智能与教育、在线教育等。