(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 112988441 B
(45)授权公告日 2024.04.05

(21)申请号 202110236506.2
(22)申请日 2021.03.03
(65)同一申请的已公布的文献号
申请公布号 CN 112988441 A
(43)申请公布日 2021.06.18
(73)专利权人 北京京东乾石科技有限公司
地址 100176 北京市北京经济技术开发区科创十一街18号院2号楼19层A1905室
(72)发明人 韩金魁 岳晓敏
(74)专利代理机构 中原信达知识产权代理有限责任公司 11219
专利代理师 张效荣 王志远
(51)Int.Cl.
G06F 11/07(2006.01)
G06F 11/30(2006.01)
G06N 20/00(2019.01)

(56)对比文件
CN 102831513 A,2012.12.19
CN 111104242 A,2020.05.05
CN 112291258 A,2021.01.29
CN 112306722 A,2021.02.02
JP H0561697 A,1993.03.12
US 2018159871 A1,2018.06.07
US 2020218553 A1,2020.07.09
US 5907708 A,1999.05.25

审查员 郑岩

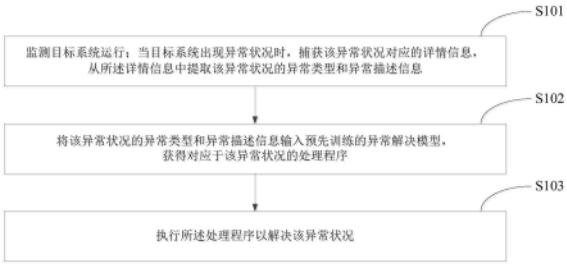
权利要求书2页 说明书9页 附图3页

(54)发明名称

异常处理方法和装置

(57)摘要

本发明公开了一种异常处理方法和装置,涉及计算机技术领域。该方法的一具体实施方式包括:监测目标系统运行;当目标系统出现异常状况时,捕获该异常状况对应的详情信息,从所述详情信息中提取该异常状况的异常类型和异常描述信息;将该异常状况的异常类型和异常描述信息输入预先训练的异常解决模型,获得对应于该异常状况的处理程序;执行所述处理程序以解决该异常状况。该实施方式能够实现目标系统异常状况的自动识别和解决。



CN 112988441 B

1.一种异常处理方法,其特征在于,包括:

监测目标系统运行;当目标系统出现异常状况时,捕获该异常状况对应的详情信息,从所述详情信息中提取该异常状况的异常类型和异常描述信息;

将该异常状况的异常类型和异常描述信息输入预先训练的异常解决模型,获得对应于该异常状况的处理程序;所述异常解决模型的训练样本中包括处理程序标识对应的处理策略;训练所述异常解决模型时,该处理策略与对应的处理程序标识一起作为标签数据;

执行所述处理程序以解决该异常状况;

所述方法进一步包括:在目标系统出现异常状况之后,确定引发该异常状况的特定对象;在执行获得的处理程序以解决该异常状况之前,将所述特定对象的标识传入所述处理程序;所述特定对象包括:与目标系统相关的类、Jar包或文件。

2.根据权利要求1所述的方法,其特征在于,所述异常解决模型通过以下步骤进行训练:

获取与目标系统相关的多个异常处理历史文本,提取每一异常处理历史文本中的异常类型、异常描述信息和处理策略;

将所述处理策略转换为可执行的处理程序;转换完成后,每一异常处理历史文本对应的异常类型、异常描述信息和处理程序标识形成一个训练样本;

依据所述训练样本训练基于机器学习算法的所述异常解决模型;其中,所述训练样本中的异常类型和异常描述信息作为训练输入数据,所述训练样本中的处理程序标识作为标签数据。

3.根据权利要求1或2所述的方法,其特征在于,所述处理程序包括:可执行脚本、可执行插件和/或可执行Jar包。

4.一种异常处理装置,其特征在于,包括:

异常跟踪单元,用于:监测目标系统运行;当目标系统出现异常状况时,捕获该异常状况对应的详情信息,从所述详情信息中提取该异常状况的异常类型和异常描述信息;

异常分析单元,用于:将该异常状况的异常类型和异常描述信息输入预先训练的异常解决模型,获得对应于该异常状况的处理程序;所述异常解决模型的训练样本中包括处理程序标识对应的处理策略;训练所述异常解决模型时,该处理策略与对应的处理程序标识一起作为标签数据;

自动解决单元,用于执行所述处理程序以解决该异常状况;

所述异常跟踪单元进一步用于:在目标系统出现异常状况之后,确定引发该异常状况的特定对象;在所述自动解决单元执行获得的处理程序以解决该异常状况之前,将所述特定对象的标识传入所述处理程序;所述特定对象包括:与目标系统相关的类、Jar包或文件。

5.根据权利要求4所述的装置,其特征在于,所述装置进一步包括模型训练单元,用于:

获取与目标系统相关的多个异常处理历史文本,提取每一异常处理历史文本中的异常类型、异常描述信息和处理策略;将所述处理策略转换为可执行的处理程序;转换完成后,每一异常处理历史文本对应的异常类型、异常描述信息和处理程序标识形成一个训练样本;依据所述训练样本训练基于机器学习算法的所述异常解决模型;其中,所述训练样本中的异常类型和异常描述信息作为训练输入数据,所述训练样本中的处理程序标识作为标签数据。

6.根据权利要求4所述的装置,其特征在于,所述处理程序包括:可执行脚本、可执行插件和/或可执行Jar包。

7.一种电子设备,其特征在于,包括:

一个或多个处理器;

存储装置,用于存储一个或多个程序,

当所述一个或多个程序被所述一个或多个处理器执行,使得所述一个或多个处理器实现如权利要求1-3中任一所述的方法。

8.一种计算机可读存储介质,其上存储有计算机程序,其特征在于,所述程序被处理器执行时实现如权利要求1-3中任一所述的方法。

异常处理方法和装置

技术领域

[0001] 本发明涉及计算机技术领域,尤其涉及一种异常处理方法和装置。

背景技术

[0002] 应用系统在运行过程中,不可避免地会出现各种异常状况,如CPU(Central Processing Unit,中央处理器)占用过高、内存占用过高、磁盘不足、服务器死机等,目前,处理异常状况的方法为:使用监控装置捕捉异常,向相关人员发送告警信息,主要通过人工方式予以解决。

[0003] 在实现本发明的过程中,发明人发现现有技术至少存在以下问题:人工处理方式响应较慢,效率较低,容易因处理不及时影响业务执行甚至导致应用系统崩溃。

发明内容

[0004] 有鉴于此,本发明实施例提供一种异常处理方法和装置,能够实现目标系统异常状况的自动识别和解决。

[0005] 为实现上述目的,根据本发明的一个方面,提供了一种异常处理方法。

[0006] 本发明实施例的异常处理方法包括:监测目标系统运行;当目标系统出现异常状况时,捕获该异常状况对应的详情信息,从所述详情信息中提取该异常状况的异常类型和异常描述信息;将该异常状况的异常类型和异常描述信息输入预先训练的异常解决模型,获得对应于该异常状况的处理程序;执行所述处理程序以解决该异常状况。

[0007] 可选地,所述异常解决模型通过以下步骤进行训练:获取与目标系统相关的多个异常处理历史文本,提取每一异常处理历史文本中的异常类型、异常描述信息和处理策略;将所述处理策略转换为可执行的处理程序;转换完成后,每一异常处理历史文本对应的异常类型、异常描述信息和处理程序标识形成一个训练样本;依据所述训练样本训练基于机器学习算法的所述异常解决模型;其中,所述训练样本中的异常类型和异常描述信息作为训练输入数据,所述训练样本中的处理程序标识作为标签数据。

[0008] 可选地,所述训练样本中进一步包括处理程序标识对应的处理策略;训练所述异常解决模型时,该处理策略与对应的处理程序标识一起作为标签数据。

[0009] 可选地,所述方法进一步包括:在目标系统出现异常状况之后,确定引发该异常状况的特定对象;在执行获得的处理程序以解决该异常状况之前,将所述特定对象的标识传入所述处理程序。

[0010] 可选地,所述处理程序包括:可执行脚本、可执行插件和/或可执行Jar包。

[0011] 为实现上述目的,根据本发明的另一方面,提供了一种异常处理装置。

[0012] 本发明实施例的异常处理装置包括:异常跟踪单元,用于:监测目标系统运行;当目标系统出现异常状况时,捕获该异常状况对应的详情信息,从所述详情信息中提取该异常状况的异常类型和异常描述信息;异常分析单元,用于:将该异常状况的异常类型和异常描述信息输入预先训练的异常解决模型,获得对应于该异常状况的处理程序;自动解决单

元,用于执行所述处理程序以解决该异常状况。

[0013] 可选地,所述装置进一步包括模型训练单元,用于:获取与目标系统相关的多个异常处理历史文本,提取每一异常处理历史文本中的异常类型、异常描述信息和处理策略;将所述处理策略转换为可执行的处理程序;转换完成后,每一异常处理历史文本对应的异常类型、异常描述信息和处理程序标识形成一个训练样本;依据所述训练样本训练基于机器学习算法的所述异常解决模型;其中,所述训练样本中的异常类型和异常描述信息作为训练输入数据,所述训练样本中的处理程序标识作为标签数据。

[0014] 可选地,所述训练样本中进一步包括处理程序标识对应的处理策略;训练所述异常解决模型时,该处理策略与对应的处理程序标识一起作为标签数据;所述异常跟踪单元进一步用于:在目标系统出现异常状况之后,确定引发该异常状况的特定对象;在所述自动解决单元执行获得的处理程序以解决该异常状况之前,将所述特定对象的标识传入所述处理程序;以及,所述处理程序包括:可执行脚本、可执行插件和/或可执行Jar包。

[0015] 为实现上述目的,根据本发明的又一方面,提供了一种电子设备。

[0016] 本发明的一种电子设备包括:一个或多个处理器;存储装置,用于存储一个或多个程序,当所述一个或多个程序被所述一个或多个处理器执行,使得所述一个或多个处理器实现本发明所提供的异常处理方法。

[0017] 为实现上述目的,根据本发明的再一方面,提供了一种计算机可读存储介质。

[0018] 本发明的一种计算机可读存储介质,其上存储有计算机程序,所述程序被处理器执行时实现本发明所提供的异常处理方法。

[0019] 根据本发明的技术方案,上述发明中的实施例具有如下优点或有益效果:

[0020] 预先获取与目标系统相关的多个异常处理历史文本,并从中提取异常类型、异常描述信息和处理策略,在将处理策略转换为可执行的处理程序之后,形成训练样本,利用该训练样本训练基于机器学习算法的异常解决模型;此后,监测目标系统运行,当目标系统出现异常状况时,获取该异常状况的异常类型和异常描述信息并输入训练完成的异常解决模型,从而得到对应于该异常状况的处理程序;最后,执行该处理程序即可解决该异常状况。通过以上方式,能够基于机器学习模型及时、高效地识别、解决异常状况,克服了现有的人工处理方式固有的响应慢、效率低等缺陷。

[0021] 上述的非惯用的可选方式所具有的进一步效果将在下文中结合具体实施方式加以说明。

附图说明

[0022] 附图用于更好地理解本发明,不构成对本发明的不当限定。其中:

[0023] 图1是本发明实施例中异常处理方法的主要步骤示意图;

[0024] 图2是本发明实施例中异常解决模型的训练步骤、使用步骤示意图;

[0025] 图3是本发明实施例中异常处理装置的组成部分示意图;

[0026] 图4是根据本发明实施例可以应用于其中的示例性系统架构图;

[0027] 图5是用来实现本发明实施例中异常处理方法的电子设备结构示意图。

具体实施方式

[0028] 以下结合附图对本发明的示范性实施例做出说明,其中包括本发明实施例的各种细节以助于理解,应当将它们认为仅仅是示范性的。因此,本领域普通技术人员应当认识到,可以对这里描述的实施例做出各种改变和修改,而不会背离本发明的范围和精神。同样,为了清楚和简明,以下的描述中省略了对公知功能和结构的描述。

[0029] 需要指出的是,在不冲突的情况下,本发明的实施例以及实施例中的技术特征可以相互结合。

[0030] 图1是根据本发明实施例中异常处理方法的主要步骤示意图。

[0031] 如图1所示,本发明实施例的异常处理方法可具体按照如下步骤执行:

[0032] 步骤S101:监测目标系统运行;当目标系统出现异常状况时,捕获该异常状况对应的详情信息,从详情信息中提取该异常状况的异常类型和异常描述信息。

[0033] 在本发明实施例中,目标系统可以是软件系统或者软硬件系统。实际应用中,可以针对目标系统可能出现的不同异常状况采取不同的跟踪策略进行监测和异常捕获,从而得到异常状况对应的详情信息。其中,以上异常状况是对CPU占用过高、内存占用过高、磁盘不足、服务器死机等非正常现象的泛称,异常状况的详情信息指的是目标系统用于记录异常状况的相关数据,如异常状况发生时相关的堆栈信息、通过跟踪策略采集到的服务器负载率、硬盘使用率、Java虚拟机GC(Garbage Collection,垃圾回收)次数等指标。

[0034] 得到异常状况对应的详情信息之后,可以从中自动提取该异常状况的异常类型和异常描述信息。具体地,异常类型用于从宏观角度表征异常状况所属的预设类别,异常描述信息则可以包括能够具体反映异常状况发生细节和特点的提示、指标等信息。在详情信息中,异常类型和异常描述信息往往处在特定的位置,利用这种规律性可以实现异常类型和异常描述信息的自动提取。可以理解,结合异常类型和异常描述信息能够准确描述和定位任意一个异常状况。以下通过几个示例对上述跟踪策略及异常类型和异常描述信息提取进行说明。

[0035] 针对基于Java(一种面向对象的编程语言)的目标系统,可以在目标系统中引入跟踪插件,当目标系统启动后,跟踪插件启动,向Java虚拟机注册全局异常捕获器,当目标系统发生异常状况时,异常状况对应的异常堆栈信息(属于以上详情信息)会进入异常捕获器。该异常堆栈信息具有以下规律:第一行代码中,“Caused by:”之后的字符串为异常类型,异常类型后冒号之后的字符串为异常描述信息。利用以上规律可以从上述异常堆栈信息中提取异常类型“NoSuchMethodError”(表示未找到所需方法)和异常描述信息“com.xxx.sketch.compile.Compiler.generateClass”(表示Compiler类中未找到所需方法)。

[0036] 针对“服务器过载”异常,可以开发相应的插件引入目标系统,目标系统启动后,采用守护线程技术监控服务器的CPU、内存等负载情况。当目标系统发生过载时,该插件获取服务器状态监控信息(属于以上详情信息),并从中确定异常类型为“服务器过载”、异常描述信息为“92%”(表示当前服务器负载率为92%)。

[0037] 针对“磁盘空间不足”异常,可以开发相应的插件引入目标系统,目标系统启动后,采用守护线程技术监控磁盘使用率情况。当磁盘空间不足时,该插件获取磁盘监控信息(属于以上详情信息),并从中确定异常类型为“磁盘空间不足”、异常描述信息为“96%”(表示当前磁盘占用96%)。

[0038] 针对“GC次数偏大”异常,可以开发相应的插件引入目标系统,目标系统启动后,采用守护线程技术监控GC次数。当GC次数超过阈值时,该插件获取GC情况监控信息(属于以上详情信息),并从中确定异常类型为“GC次数偏大”、异常描述信息为“6”(表示当前GC次数为6)。

[0039] 特别地,目标系统出现某种异常状况时,在执行前述详情信息捕获以及异常类型和异常描述信息提取之前、之后或同时,基于上述跟踪策略还可以确定引发该异常状况的特定对象。以上特定对象可以是与目标系统相关的类、Jar(Java Archive,Java归档)包、文件等,实际应用中,可以通过检测日志等方式确定引发异常状况的特定对象,这有助于后续流程的执行,具体内容将在下文说明。

[0040] 步骤S102:将该异常状况的异常类型和异常描述信息输入预先训练的异常解决模型,获得对应于该异常状况的处理程序。

[0041] 在执行本步骤之前,需要预先训练基于机器学习算法的异常解决模型,异常解决模型的训练步骤和使用步骤如图2所示。

[0042] 较佳地,异常解决模型的训练步骤如下:首先,获取与目标系统相关的多个异常处理历史文本,提取每一异常处理历史文本中的异常类型、异常描述信息和针对异常的处理策略。例如,如果目标系统为Java系统,可以利用爬虫技术从互联网相关平台抓取与Java异常处理相关的文本,并利用文本的标题、关键词等确定每一文本中的异常类型、异常描述信息和处理策略,如,某文本的标题为“遇到NoSuchMethodError的通用解决思路”,则可根据预设策略将“NoSuchMethodError”确定为异常类型;某文本正文中如果存在“具体”、“描述”、“详情”等关键词时,可以对关键词前后的相关内容进行分析进而确定异常描述信息;某文本正文中如果存在“方法”、“解决”、“策略”等关键词时,可以对关键词前后的相关内容进行分析进而确定处理策略。

[0043] 实际应用中,还可以在相关界面手动输入某异常状况的异常类型、异常描述信息和采用的处理策略。通过以上两种方式获取的四条示例数据如下表所示。

[0044]

序号	异常类型	异常描述信息	处理策略
1	NoSuchMethodError	com.xxx.sketch.compile.	排除

[0045]

		Compiler.generateClass	core-3.1.1.jar包
2	磁盘空间不足	大于 95%	删除无用日志
3	服务器过载	大于 90%	系统拒绝请求
4	GC次数偏大	大于 5 次	扩展虚拟机内存

[0046] 此后,根据预设规则将每条数据中的处理策略转换为可执行的处理程序,示例性地,以上处理程序可以是可执行脚本、可执行插件和/或可执行Jar包,以上转换可以通过人工执行,也可以通过程序自动执行。

[0047] 例如,对于“排除core-3.1.1.jar包”,可以首先执行Maven(一种Java项目构建系统)命令标记引发异常状况的Jar包(该Jar包即为前述特定对象,其标识——如名称可以在处理程序执行前传入),接着得到正确Jar包的版本依赖,之后修改pom(Project Object

Model,项目对象模型)文件进行Jar包排除,从而生成“排除core-3.1.1.jar包”对应的Shell(一种用C语言编写的程序)脚本(名称例如a.sh)。可以理解,在Shell脚本中,使用固定参数表示引发异常状况的Jar包(即特定对象),在执行Shell脚本之前,执行前述跟踪逻辑的模块会将目标系统中特定对象的标识(如名称)传入Shell脚本,以定位实际需要排除的Jar包。

[0048] 再如,对于“删除无用日志”,可以生成相应的Shell脚本(名称例如b.sh)作为处理程序来删除数天前的日志。对于“系统拒绝请求”,可以生成相应的可执行Jar包(名称例如c.jar),其中采用字节码增强技术实现针对用户请求的全局拦截。对于“扩展虚拟机内存”,可以生成用于加大内存的Shell脚本(名称例如d.sh)。这样,即可得到下表。

[0049]

序号	异常类型	异常描述信息	处理策略	处理程序标识
1	NoSuchMethodError	com.xxx.sketch.compile.Compiler.generateClass	排除core-3.1.1.jar包	a.sh
2	磁盘空间不足	大于 95%	删除无用日志	b.sh
3	服务器过载	大于 90%	系统拒绝请求	c.jar
4	GC次数偏大	大于 5 次	扩展虚拟机	d.sh

[0050]

			内存	

[0051] 在执行以上转换后,上表中的每一条记录即形成一个训练样本。实际应用中,训练样本也可以没有“处理策略”字段,仅有“异常类型”、“异常描述信息”和“处理程序标识”字段。

[0052] 此后,可以依据上述训练样本训练异常解决模型,训练时,使用有监督学习方法,训练样本中的异常类型和异常描述信息作为训练输入数据,训练样本中的处理程序标识作为标签数据。异常解决模型可以基于朴素贝叶斯、决策树等分类算法构建,以下以决策树为例对训练流程进行说明。

[0053] 首先将所有训练样本组成的训练集作为决策树的根节点,根据预设的分裂规则逐一确定每次分裂采用的分裂属性(如异常类型或异常描述信息),此后基于确定的分裂属性从根节点进行至少一次分裂处理,直到分裂处理形成的节点满足终止条件,满足终止条件的节点即为终止节点;当所有终止节点确定后,即形成决策树。

[0054] 在步骤S102中,得到训练完成的异常解决模型之后,可以将根据步骤S101得到的异常状况的异常类型和异常描述信息输入该异常解决模型,即可得到对应于该异常状况的处理程序标识(如处理程序名称),进而确定相应的处理程序,该处理程序即为能够解决相应异常状况的程序。可以理解,当训练样本中包含处理策略时,输入该异常解决模型还可得到对应于该异常状况的处理策略,该处理策略可以在相应界面向相关人员展示。

[0055] 通过以上设置,异常解决模型会借助不断更新的训练集不断优化,从而具有不断

提高的异常识别与处理能力。

[0056] 步骤S103:执行处理程序以解决该异常状况。

[0057] 在本步骤中,可以执行经异常解决模型确定的处理程序以自动解决异常状况,从而实现及时、高效的异常状况识别和解决。可以理解,对于处理程序a.sh,由于该处理程序执行前已包含经跟踪逻辑确定、并传入处理程序的特定对象标识,因此执行该处理程序能够准确排除目标系统中的特定对象。

[0058] 在本发明实施例的技术方案中,预先获取与目标系统相关的多个异常处理历史文本,并从中提取异常类型、异常描述信息和处理策略,在将处理策略转换为可执行的处理程序之后,形成训练样本,利用该训练样本训练基于机器学习算法的异常解决模型;此后,监测目标系统运行,当目标系统出现异常状况时,获取该异常状况的异常类型和异常描述信息并输入训练完成的异常解决模型,从而得到对应于该异常状况的处理程序;最后,执行该处理程序即可解决该异常状况。通过以上方式,能够基于机器学习模型及时、高效地识别、解决异常状况,克服了现有的人工处理方式固有的响应慢、效率低等缺陷。

[0059] 需要说明的是,对于前述的各方法实施例,为了便于描述,将其表述为一系列的动作组合,但是本领域技术人员应该知悉,本发明并不受所描述的动作顺序的限制,某些步骤事实上可以采用其它顺序进行或者同时进行。此外,本领域技术人员也应该知悉,说明书中所描述的实施例均属于优选实施例,所涉及的动作和模块并不一定是实现本发明所必须的。

[0060] 为便于更好的实施本发明实施例的上述方案,下面还提供用于实施上述方案的相关装置。

[0061] 请参阅图3所示,本发明实施例提供的异常处理装置300可以包括:异常跟踪单元301、异常分析单元302和自动解决单元303。

[0062] 其中,异常跟踪单元301用于:监测目标系统运行;当目标系统出现异常状况时,捕获该异常状况对应的详情信息,从所述详情信息中提取该异常状况的异常类型和异常描述信息;异常分析单元302用于:将该异常状况的异常类型和异常描述信息输入预先训练的异常解决模型,获得对应于该异常状况的处理程序;自动解决单元303用于执行所述处理程序以解决该异常状况。

[0063] 在本发明实施例中,所述装置300可进一步包括模型训练单元,用于:获取与目标系统相关的多个异常处理历史文本,提取每一异常处理历史文本中的异常类型、异常描述信息和处理策略;将所述处理策略转换为可执行的处理程序;转换完成后,每一异常处理历史文本对应的异常类型、异常描述信息和处理程序标识形成一个训练样本;依据所述训练样本训练基于机器学习算法的所述异常解决模型;其中,所述训练样本中的异常类型和异常描述信息作为训练输入数据,所述训练样本中的处理程序标识作为标签数据。

[0064] 作为一个优选方案,所述训练样本中进一步包括处理程序标识对应的处理策略;训练所述异常解决模型时,该处理策略与对应的处理程序标识一起作为标签数据;所述异常跟踪单元301可进一步用于:在目标系统出现异常状况之后,确定引发该异常状况的特定对象;在所述自动解决单元303执行获得的处理程序以解决该异常状况之前,将所述特定对象的标识传入所述处理程序;以及,所述处理程序包括:可执行脚本、可执行插件和/或可执行Jar包。

[0065] 根据本发明实施例的技术方案,预先获取与目标系统相关的多个异常处理历史文本,并从中提取异常类型、异常描述信息和处理策略,在将处理策略转换为可执行的处理程序之后,形成训练样本,利用该训练样本训练基于机器学习算法的异常解决模型;此后,监测目标系统运行,当目标系统出现异常状况时,获取该异常状况的异常类型和异常描述信息并输入训练完成的异常解决模型,从而得到对应于该异常状况的处理程序;最后,执行该处理程序即可解决该异常状况。通过以上方式,能够基于机器学习模型及时、高效地识别、解决异常状况,克服了现有的人工处理方式固有的响应慢、效率低等缺陷。

[0066] 图4示出了可以应用本发明实施例的异常处理方法或异常处理装置的示例性系统架构400。

[0067] 如图4所示,系统架构400可以包括终端设备401、402、403,网络404和服务器405(此架构仅仅是示例,具体架构中包含的组件可以根据申请具体情况调整)。网络404用以在终端设备401、402、403和服务器405之间提供通信链路的介质。网络404可以包括各种连接类型,例如有线、无线通信链路或者光纤电缆等。

[0068] 用户可以使用终端设备401、402、403通过网络404与服务器405交互,以接收或发送消息等。终端设备401、402、403上可以安装有各种通讯客户端应用,例如异常处理应用(仅为示例)。

[0069] 终端设备401、402、403可以是具有显示屏并且支持网页浏览的各种电子设备,包括但不限于智能手机、平板电脑、膝上型便携计算机和台式计算机等等。

[0070] 服务器405可以是提供各种服务的服务器,例如对用户利用终端设备401、402、403所操作的异常处理应用提供支持的后台服务器(仅为示例)。服务器405可以对接收到的异常处理请求等进行处理,并将处理结果(例如是否解决了异常状况--仅为示例)反馈给终端设备401、402、403。

[0071] 需要说明的是,本发明实施例所提供的异常处理方法一般由服务器405执行,相应地,异常处理装置一般设置于服务器405中。

[0072] 应该理解,图4中的终端设备、网络和服务器的数目仅仅是示意性的。根据实现需要,可以具有任意数目的终端设备、网络和服务器。

[0073] 本发明还提供了一种电子设备。本发明实施例的电子设备包括:一个或多个处理器;存储装置,用于存储一个或多个程序,当所述一个或多个程序被所述一个或多个处理器执行,使得所述一个或多个处理器实现本发明所提供的异常处理方法。

[0074] 下面参考图5,其示出了适于用来实现本发明实施例的电子设备的计算机系统500的结构示意图。图5示出的电子设备仅仅是一个示例,不应对本发明实施例的功能和使用范围带来任何限制。

[0075] 如图5所示,计算机系统500包括中央处理单元(CPU)501,其可以根据存储在只读存储器(ROM)502中的程序或者从存储部分508加载到随机访问存储器(RAM)503中的程序而执行各种适当的动作和处理。在RAM503中,还存储有计算机系统500操作所需的各种程序和数据。CPU501、ROM 502以及RAM 503通过总线504彼此相连。输入/输出(I/O)接口505也连接至总线504。

[0076] 以下部件连接至I/O接口505:包括键盘、鼠标等的输入部分506;包括诸如阴极射线管(CRT)、液晶显示器(LCD)等以及扬声器等的输出部分507;包括硬盘等的存储部分508;

以及包括诸如LAN卡、调制解调器等的网络接口卡的通信部分509。通信部分509经由诸如因特网的网络执行通信处理。驱动器510也根据需要连接至I/O接口505。可拆卸介质511,诸如磁盘、光盘、磁光盘、半导体存储器等等,根据需要安装在驱动器510上,以便从其上读出的计算机程序根据需要被安装入存储部分508。

[0077] 特别地,根据本发明公开的实施例,上文的主要步骤图描述的过程可以被实现为计算机软件程序。例如,本发明实施例包括一种计算机程序产品,其包括承载在计算机可读介质上的计算机程序,该计算机程序包含用于执行主要步骤图所示的方法的程序代码。在上述实施例中,该计算机程序可以通过通信部分509从网络上被下载和安装,和/或从可拆卸介质511被安装。在该计算机程序被中央处理单元501执行时,执行本发明的系统中限定的上述功能。

[0078] 需要说明的是,本发明所示的计算机可读介质可以是计算机可读信号介质或者计算机可读存储介质或者是上述两者的任意组合。计算机可读存储介质例如可以是——但不限于——电、磁、光、电磁、红外线、或半导体的系统、装置或器件,或者任意以上的组合。计算机可读存储介质的更具体的例子可以包括但不限于:具有一个或多个导线的电连接、便携式计算机磁盘、硬盘、随机访问存储器(RAM)、只读存储器(ROM)、可擦式可编程只读存储器(EPROM或闪存)、光纤、便携式紧凑磁盘只读存储器(CD-ROM)、光存储器件、磁存储器件、或者上述的任意合适的组合。在本发明中,计算机可读存储介质可以是任何包含或存储程序的有形介质,该程序可以被指令执行系统、装置或者器件使用或者与其结合使用。在本发明中,计算机可读信号介质可以包括在基带中或者作为载波一部分传播的数据信号,其中承载了计算机可读的程序代码。这种传播的数据信号可以采用多种形式,包括但不限于电磁信号、光信号或上述任意合适的组合。计算机可读信号介质还可以是计算机可读存储介质以外的任何计算机可读介质,该计算机可读介质可以发送、传播或者传输用于由指令执行系统、装置或者器件使用或者与其结合使用的程序。计算机可读介质上包含的程序代码可以用任何适当的介质传输,包括但不限于:无线、电线、光缆、RF等等,或者上述的任意合适的组合。

[0079] 附图中的流程图和框图,图示了按照本发明各种实施例的系统、方法和计算机程序产品的可能实现的体系架构、功能和操作。在这点上,流程图或框图中的每个方框可以代表一个模块、程序段、或代码的一部分,上述模块、程序段、或代码的一部分包含一个或多个用于实现规定的逻辑功能的可执行指令。也应当注意,在有些作为替换的实现中,方框中所标注的功能也可以以不同于附图中所标注的顺序发生。例如,两个接连地表示的方框实际上可以基本并行地执行,它们有时也可以按相反的顺序执行,这根据所涉及的功能而定。也要注意的是,框图或流程图中的每个方框、以及框图或流程图中的方框的组合,可以用执行规定的功能或操作的专用的基于硬件的系统来实现,或者可以用专用硬件与计算机指令的组合来实现。

[0080] 描述于本发明实施例中所涉及到的单元可以通过软件的方式实现,也可以通过硬件的方式来实现。所描述的单元也可以设置在处理器中,例如,可以描述为:一种处理器包括异常跟踪单元、异常分析单元和自动解决单元。其中,这些单元的名称在某种情况下并不构成对该单元本身的限定,例如,异常跟踪单元还可以被描述为“向异常分析单元提供异常状况的异常类型和异常描述信息的单元”。

[0081] 作为另一方面,本发明还提供了一种计算机可读介质,该计算机可读介质可以是上述实施例中描述的设备中所包含的;也可以是单独存在,而未装配入该设备中的。上述计算机可读介质承载有一个或者多个程序,当上述一个或者多个程序被该设备执行时,使得该设备执行的步骤包括:监测目标系统运行;当目标系统出现异常状况时,捕获该异常状况对应的详情信息,从所述详情信息中提取该异常状况的异常类型和异常描述信息;将该异常状况的异常类型和异常描述信息输入预先训练的异常解决模型,获得对应于该异常状况的处理程序;执行所述处理程序以解决该异常状况。

[0082] 在本发明实施例的技术方案中,预先获取与目标系统相关的多个异常处理历史文本,并从中提取异常类型、异常描述信息和处理策略,在将处理策略转换为可执行的处理程序之后,形成训练样本,利用该训练样本训练基于机器学习算法的异常解决模型;此后,监测目标系统运行,当目标系统出现异常状况时,获取该异常状况的异常类型和异常描述信息并输入训练完成的异常解决模型,从而得到对应于该异常状况的处理程序;最后,执行该处理程序即可解决该异常状况。通过以上方式,能够基于机器学习模型及时、高效地识别、解决异常状况,克服了现有的人工处理方式固有的响应慢、效率低等缺陷。

[0083] 上述具体实施方式,并不构成对本发明保护范围的限制。本领域技术人员应该明白的是,取决于设计要求和其他因素,可以发生各种各样的修改、组合、子组合和替代。任何在本发明的精神和原则之内所作的修改、等同替换和改进等,均应包含在本发明保护范围之内。

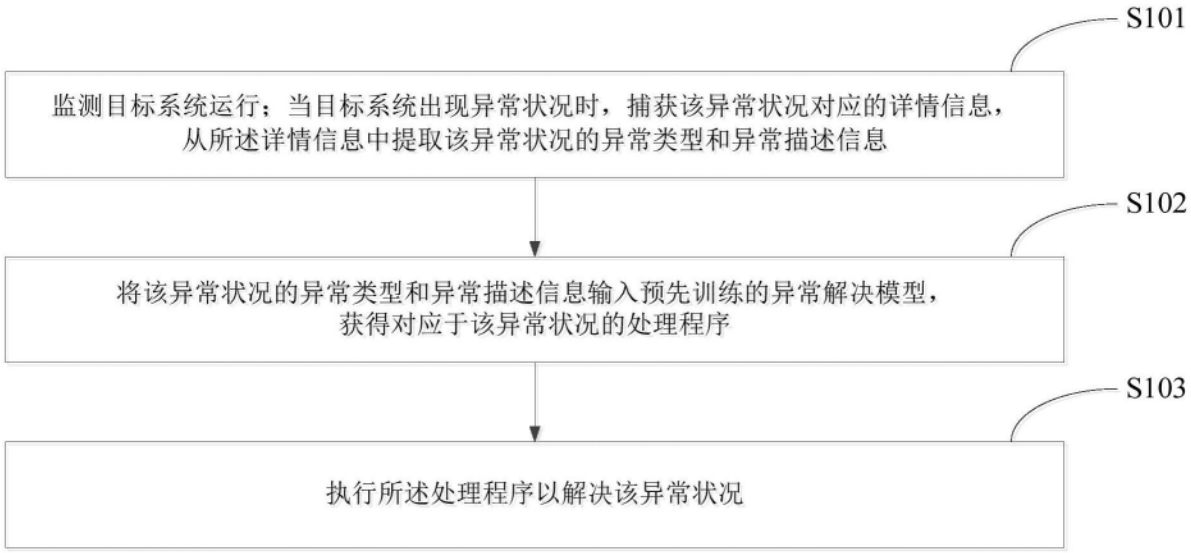

S101
监测目标系统运行；当目标系统出现异常状况时，捕获该异常状况对应的详情信息，从所述详情信息中提取该异常状况的异常类型和异常描述信息
S102
将该异常状况的异常类型和异常描述信息输入预先训练的异常解决模型，获得对应于该异常状况的处理程序
S103
执行所述处理程序以解决该异常状况

图1

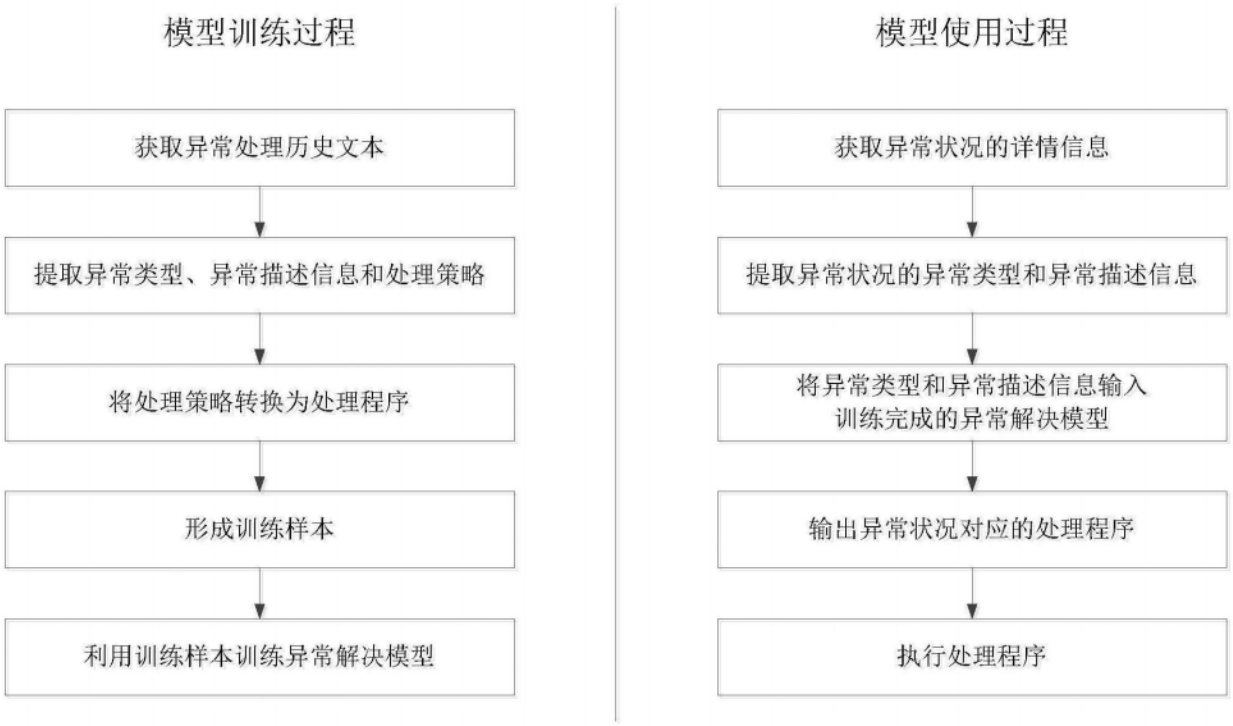

模型训练过程
获取异常处理历史文本
提取异常类型、异常描述信息和处理策略
将处理策略转换为处理程序
形成训练样本
利用训练样本训练异常解决模型
模型使用过程
获取异常状况的详情信息
提取异常状况的异常类型和异常描述信息
将异常类型和异常描述信息输入训练完成的异常解决模型
输出异常状况对应的处理程序
执行处理程序

图2

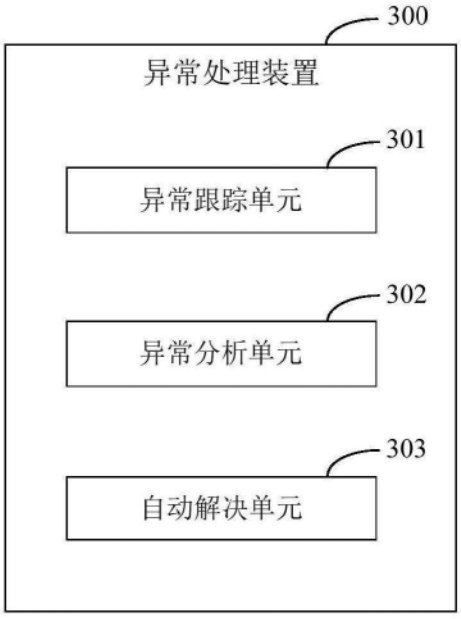

300
异常处理装置
301
异常跟踪单元
302
异常分析单元
303
自动解决单元

图3

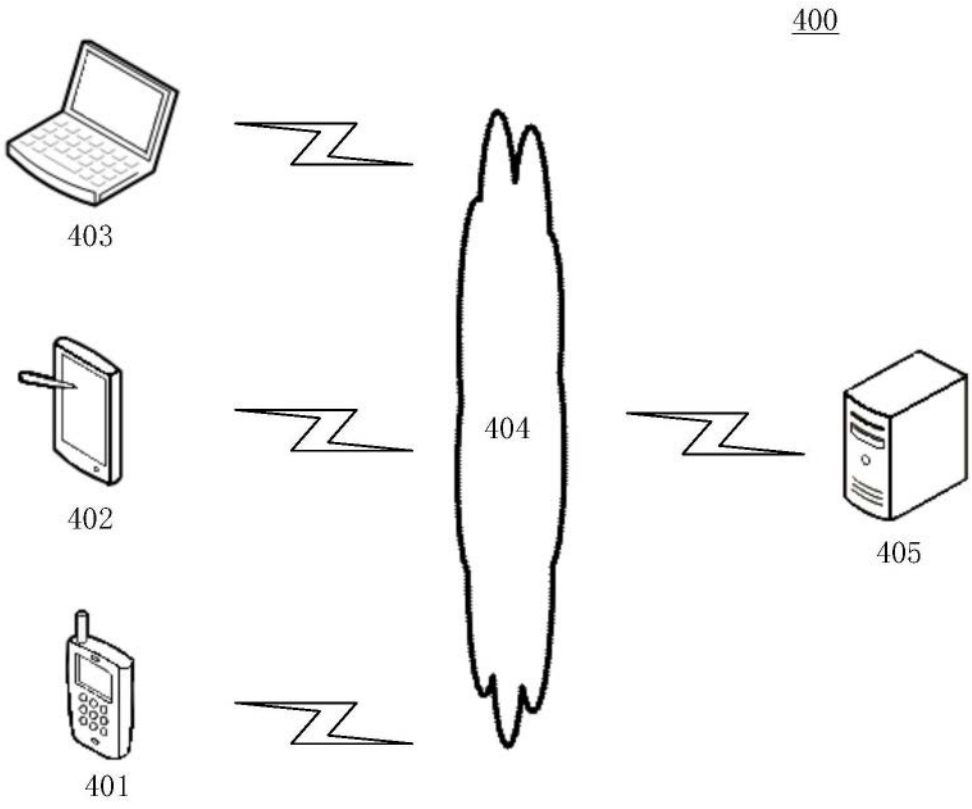

400
403
402
401
404
405

图4

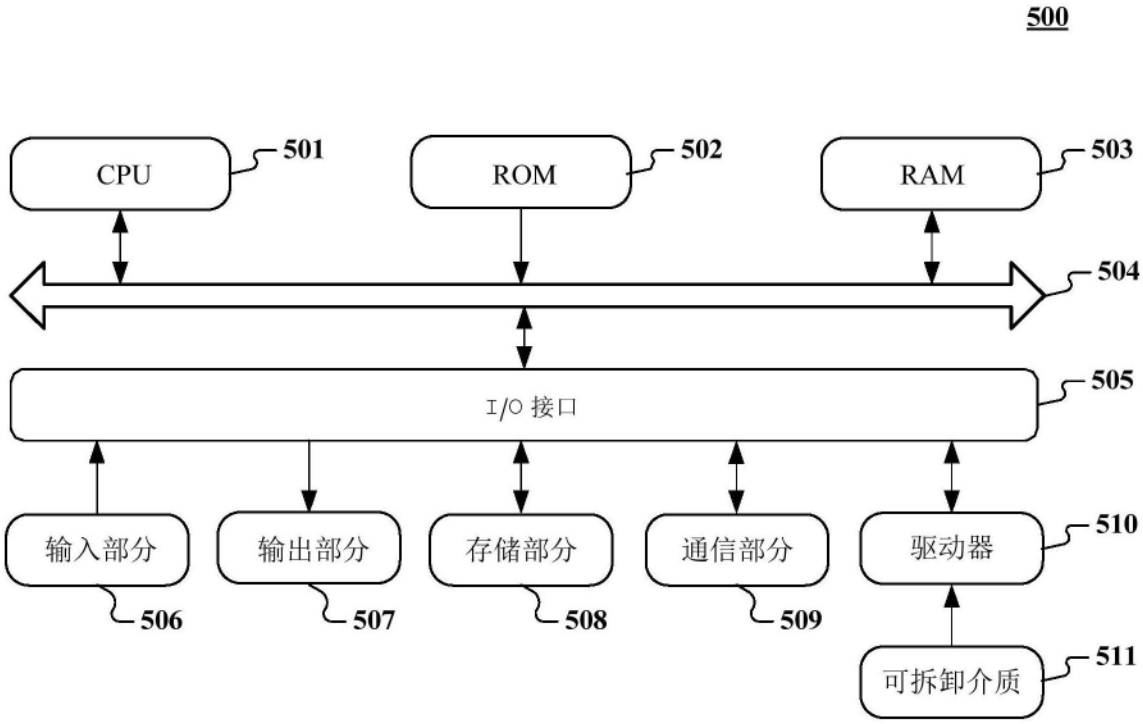

500
CPU
501
ROM
502
RAM
503
504
I/O 接口
505
输入部分
506
输出部分
507
存储部分
508
通信部分
509
驱动器
510
可拆卸介质
511

图5