计算机科学
COMPUTER SCIENCE

基于配置语句树的网络设备配置异常检测算法

沈袁程, 班瑞, 陈昕, 华润多, 汪云海

引用本文

沈袁程, 班瑞, 陈昕, 华润多, 汪云海. 基于配置语句树的网络设备配置异常检测算法[J]. 计算机科学, 2023, 50(11A): 230200128-10.

SHEN Yuancheng, BAN Rui, CHEN Xin, HUA Runduo, WANG Yunhai. Anomaly Detection Algorithm for Network Device Configuration Based on Configuration Statement Tree [J]. Computer Science, 2023, 50(11A): 230200128-10.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.230200128

基于配置语句树的网络设备配置异常检测算法

沈袁程[1]　班　瑞[2]　陈　昕[1]　华润多[2]　汪云海[1]
1 山东大学计算机科学与技术学院　山东 青岛 266200
2 中讯邮电咨询设计院有限公司　北京 100000
(202135158@mail.sdu.edu.cn)

摘　要　随着网络通信设备的发展,设备配置异常引发的问题日益显著。传统的检测工具通常只针对拼写、格式等进行检测,无法检测逻辑问题。因此,目前的配置异常检测工作高度依赖工程师经验。为了提高网络服务质量并减少工程师的重复工作,以及解决传统工具检测速度慢、检测能力弱、通用性差等问题,文中借鉴了抽象语法树的设计理念,创新性地提出了一种基于“配置语句树”的无监督异常检测算法。通过统计分析,该算法可以确定7种可检测异常类型,并支持异常定位和异常修改方案的推荐。文中采用运营商现网运营中的配置,根据算法可检测种类、运行时间、准确率和召回率这几个指标进行量化评估和对比分析。实验结果表明,该算法具有良好的鲁棒性,完全能够有效应对网络设备配置异常引发的网络通信问题。

关键词:异常检测;聚类分析;设备自动巡检;抽象语法树;共现语料分析;无监督学习;关联分析
中图法分类号　TP301

Anomaly Detection Algorithm for Network Device Configuration Based on Configuration Statement Tree

SHEN Yuancheng[1],BAN Rui[2],CHEN Xin[1],HUA Runduo[2] and WANG Yunhai[1]
1 School of Computer Science and Technology,Shandong University,Qingdao,Shandong 266200,China
2 China Information Technology Designing & Consulting Institute,Beijing 100000,China

Abstract　The problem of device configuration anomalies is becoming increasingly significant with the development of network communication equipment. Traditional detection tools usually only detect spelling,formatting and other issues,and cannot identify logic problems. Consequently,engineers' experience plays a critical role in detecting such anomalies. To improve network service quality,reduce repetitive work,and address issues like slow detection speed,weak detection capabilities,and poor versatility of traditional tools,this paper draws on the design concept of abstract syntax trees and proposes an innovative unsupervised anomaly detection algorithm based on "configuration statement trees." It can identify seven types of detectable anomalies and provides recommendations for anomaly localization and modification plans. The paper evaluates and compares the algorithm based on indicators such as detectable types,runtime,accuracy,and recall using configurations from the operator's current network operation. The results demonstrate that the algorithm has good robustness and can effectively address network communication issues resulting from configuration anomalies in network communication equipment.

Keywords　Anomaly detection,Cluster analysis,Automatic inspection of equipment,Abstract syntax tree,Co-occurrence corpus analysis,Unsupervised learning,Association analysis

1　引言

近年来,随着5G技术的发展和终端用户的爆炸式增长,通信网络的规模和网络运营商的业务量也随之飞速增长,越来越多的厂家加入网络设备的供应环节和软硬件解决方案的服务提供中[1]。运营商们为了降低成本、提升网络服务质量,往往会选择和不同的厂家合作,这些OEM厂家通常会提供专有的配套软硬件及其协议和规范,网络结构也因此复杂多样。同时,以自动驾驶等为代表的实时处理技术[2-3]的发展对通信网络的高效性和稳定性提出了更高的需求。用户需求的提升和网络结构的升级优化使得网络基础设施的维护工作面临着史无前例的挑战[4]。

除了网络设备本身的物理性能外,设备配置的异常而引发的网络通信问题日益显著。目前,运营商们都采用与之合作的网络基础设施供应商(OEM厂家)提供的整体解决方案,包括硬件及其配套软件。为了占据市场份额,主导行业生态,这些供应商们通常会提供专用的协议、规范以及闭源的系统,其中就包括网络通信设备、配套的配置文件书写规范、设备异常排查与检测的工具包等。不同厂家设备的规格以及配置文件的书写逻辑具有较大的差异,甚至同一个厂家不同版本的

基金项目:国家重点研发计划(2022ZD0160805);面向泛在计算环境大数据可视分析的人机交互理论与方法(62141217)
This work was supported by the National Key R & D Program of China(2022ZD0160805) and Human Computer Interaction Theory and Methods for Visual Analysis of Big Data in Ubiquitous Computing Environment(62141217).
通信作者:汪云海(cloudseawang@gmail.com)

配置文件也会存在书写规范的差异。然而，目前配置文件的维护和配置异常的排查、检测工作高度依赖运维人员的经验和技术水平。随着网络结构复杂性和多样性的提高，运维人员的素质、经验以及操作规范导致的配置异常也越来越多。

目前的网络基础设施供应商通常会提供相应的配置文件检查工具包，也称“PTN 自动化网络巡检工具”[5]，这些工具包大多是基于配置文件的语法规则对配置语句进行逐行筛查，如中兴研发的 ZXTIM 和 NetNumen TM U31、华为的 SmartKit NSE 等。基于规则的配置异常检测工具可以较为准确地检测出配置文件中的异常，但能检测的异常都相对简单，通常只是支持拼写、命令格式、参数格式等异常的检测，无法对配置的逻辑（如漏配、错配、端口号和 ip 地址异常）进行检测。这些工具检测效率较低，无法承担日益增长的配置异常检测需求。并且，不同厂家与不同版本配置文件的差异导致这些工具的通用性较差。随着网络设备的更新迭代，配置语句的书写规则也在不断地升级调整，传统的巡检工具显然无法很好地适配网络基础设施的升级。

为了解决规则匹配的方法通用性差以及检测工具版本众多、缺乏灵活性的问题，许多专家学者以及相关领域的工程师们提出了“运维智能化”的想法[6-7]，将大数据分析和人工智能技术运用于配置文件的异常检测工作中。他们致力于为不同厂家、不同版本的配置文件建立统一的配置异常检测模型，充分发挥大数据技术在规模化异常检测中的作用与人工智能在自动化决策中的优势[8-9]，最终实现高效全面地对全网设备的配置进行筛查。届时，网络运维工程师们只需要考虑如何书写高质量的配置语句，而无需关注繁琐的配置异常排查与更正工作。比如最近，Liu 等提出将 AI 关联分析中的弱关联规则作为异常配置语句的特征来训练异常检测模型[10]，从而达到配置异常检测的目的，本文 2.3 节将对 AI 关联分析算法给出更详细的介绍。这些方法虽然能够批量对不同厂家、不同版本的配置文件进行异常检测，但是在系统运行初期仍然需要依赖人工标注一定量的异常，并且需要人工一一给出这些异常的类型、严重程度等信息。然而，在实际生产过程中，绝大多数现网运营中的配置文件的异常均是配置语句的结构、顺序问题导致的，这些异常很难被工程师们察觉，甚至有部分异常根本无法被人工发觉，采用关联规则训练的 AI 检测模型不具备检测这些异常的能力。

为了解决上述的问题，本文首先基于运营商在现网运行的部分配置文件，提出了一种将大数据分析[11]、频率分析和共现语料分析[12]相结合的方法，该方法能确定现网运营中的配置文件可能存在的异常类型，尽可能避免了因运维工程师的主观经验或技术水平的限制给异常检测工作带来问题；其次，借鉴了抽象语法树的设计思想[13]，创新性地提出了一种将每个配置文件构建成一棵结构化的语句树（下文称为“配置语句树”）的方法，通过该配置语句树提取配置文件的特征，对其使用无监督的 K-Means 聚类算法[14]即可给出异常配置文件；最后，提出了一套高效的异常检测方法，该方法能够准确地定位异常并且给出异常类型及推荐的修改方案，对于不在本文确定类型中的异常，则直接给出其位置。本文提出的配置异常检测方法能够快速高效地对不同厂家、不同版本的配置文件进行异常检测，检测过程无需人工干预，极大地减少了网络运维工程师们的重复性工作，提升了运营商网络服务的质量。

2 相关工作

近年来，配置异常检测一直是异常检测领域研究的热点之一。国内外研究者提出了很多异常检测的方法，这些方法大致可以分为三大类：基于规则判断的、基于统计学的和基于人工智能的。基于规则判断的异常检测指由行业专家预先定义好所有异常检测的规则，根据规则对配置文件进行检测；基于统计学的方法遵循“异常配置出现频率比正常配置低”的基本原则，通过统计配置文件各语料的频率进行异常检测；基于人工智能的方法致力于通过挖掘配置语句间的关联规则来进行异常检测。

2.1 基于规则判断的异常检测

基于规则判断的异常检测方法通常具有明确的检测模板，对异常的定义清晰且识别率较高，原则上能够实现 100% 准确地识别出规则中定义好的异常。这通常需要相关领域的专家预先制定规则，并且基于规则的方法的应用场景非常单一。目前，网络基础设施供应商再向运营商提供设备以及软硬件服务时，通常会提供相应的网络配置异常检测工具，这些工具大多都是基于规则判断的，如中兴的 ZXTIM 和 NetNumenTM U31、华为的 SmartKit NSE 等。但是这些工具或者算法所支持检测的异常种类较少，且较为死板，通常只能检测简单的拼写错误及 ip 地址、端口号格式的书写错误，对于配置语句逻辑上的错误，基于规则的工具无法很好地对其进行检测。尽管 Liu 等提出了一种将检测规则分为常规检测规则和特殊检测规则的方法[15]，用于提高这类工具检测的灵活性，但也只能达到拓展检测规则的目的，无法越过规则制定的环节。并且，基于规则的异常检测方法通常耗时较长，还容易造成设备故障，如采用中兴的 ZXTIM 对某个城市数千台设备，总计 200 多万行配置语句进行检测，耗时 40 h 后最终导致设备宕机。

2.2 基于统计学方法的异常检测

基于统计学的异常检测方法是一种比较传统的方法[16]，这种方法建立在一个合理的假设之上：正常情况是大多数的、高频的事件，而异常是少数的、低频的事件。这种方法完全基于统计学的理论对数据进行建模，一旦某个数据不符合这个模型的定义就被判定为异常。

Yu 等提出了一种基于语料库的高频汉字串互信息分布规律分析方法[17]，该方法基于文本的语料库，通过计算大规模高频二字串、三字串、四字串的互信息，发现了采用语料分析，基于统计学的方法研究词语、句子之间的一些关系对文本构建和异常分析都有不小的挑战。Pincombeb 提出的 ARIMA 算法[18]，采用基于预测的思想来进行异常检测，它们通常会为异常检测目标设定一个阈值，通过将检测任务的计算值与阈值进行匹配来判断是否存在异常，这个检测模型对数据要求非常高，需要提前对大量的数据进行训练，并且需要人为对结果进行分析，以增加模型的可靠性。

最近，许多专家学者都在研究通过共现词频分析[19]进行异常检测的方法，通过对前后相继出现的词语、句子或段落等语料进行统计学分析，来将低频语料划分为异常，从而进行文本异常的定位和检测。然而，对配置进行共现语料分析时，只考虑前后语料本身之间的关联而不考虑它们在整个配置语句中的结构信息的做法，导致采用这种方法进行异常检测所得

的结果不全面且不准确。

在自然语言环境中,许多学者尝试采用搭建神经网络或构建知识图谱等方式为自然语言的表达或异常检测建立人工智能的模型,通过学习大量的口头或书面的表达,来抽象提取出相应的网络结构或知识仓库,从而完成对自然语言表达的推荐或异常检测工作。然而,这一过程通常需要对大量的样本进行训练,需要花费大量的时间、人力以及算力,并且目前的方法仍无法完美适用于如此复杂灵活的自然语言。

虽然采用统计学相关方法来配置异常检测无法得到完整、准确的结果,但是由于配置文件是相对高度格式化的文本,相对于自然语言复杂的语法规则而言,配置文件包含的"语法成分"比较固定,通常包含设备基本信息、ip 地址、端口号、密码等固定格式的信息。采用统计学相关方法能够高效准确地帮助开发者找出配置文件中"大概率"存在的错误,研究人员据此可以归纳出配置文件中可能存在的错误类型。本文参考了共现频率分析的方法,对配置文件进行共现语料分析,找出了配置文件中可能存在的异常的类型。

2.3 基于人工智能的异常检测

人工智能算法凭借着其智能高效的算法分析与决策能力一度成为了异常检测领域的研究热点。目前,较为热门的异常检测算法几乎都是基于人工智能技术的,确切地来说是机器学习技术。常见的基于机器学习的文本异常检测方法主要分为有监督方法、半监督方法和无监督方法。

最近,Liu 等提出的 Opprentice[20] 作为一组经典的异常检测框架得到了良好的受众,该方法用工具标注 kpi 异常周期,同时提取大量的异常特征,然后用标签和特征来训练随机森林区分,该方法能够自动选择并调整检测器的阈值和相关参数,检测效率较高。但 Opprentice 是一种有监督的异常检测方法,在训练模型前需要依靠大量的人力进行数据标注的工作。Yang 等[21] 提出了一种基于 GMM 的异常检测方法,该方法通过训练高斯混合模型,计算数据通过该模型后"正常"的概率来进行异常检测,该方法的本质思想还是基于传统的统计学理论,在配置异常检测过程中忽略了配置的结构带来的影响。Rashidi 等[22] 采用贝叶斯网络(Bayesian Networks)对分类异常问题进行检测,贝叶斯网络是一种概率图模型,能够在一定程度上刻画语料之间的因果关系,通过建立合适的贝叶斯网络(Bayesian Networks)模型能够很好地挖掘配置文本语料间的关联规则,因此目前这种方法在文本异常检测上的效果较为显著。

目前,各大运营商都在面临 5G 网络新建和升级改造,网络规模以及业务量快速增长。在网络设备配置异常检测领域,运营商纷纷聚焦于基于 AI 关联分析的方法来进行网络设备配置异常检测。目前,基于 AI 关联分析的方法是网络设备配置异常检测领域中相对前沿和热门的研究方向,AI 关联分析方法可以在海量配置文件中发现配置语句之间的依赖关系,如 FpGrowth 算法[23],能够从海量数据中自动挖掘出配置文本之间的关联关系。中国联通的 Liu 等提出了一种基于 AI 关联分析的网络设备配置异常检测方法[10],以 AI 关联分析中得到的弱关联规则构建检测模型,通过人工标注部分异常数据后训练自动标注模型,该方法能够从海量配置数据中快速发现配置异常。

上述工作都是近几年将机器学习用于异常检测方面比较突出的工作,这些工作大多是基于有监督或者半监督机器学习的,需要人工事先标注大量的数据和特征。对应到配置异常检测工作中,就意味着需要网络运维工程师事先标注大量的异常样本数据。然而,异常的配置本身就是"低频的、稀少的",无法找出大量可供标注的样本数据。另外,能否准确、全面地标注这些数据,取决于运维工程师们的技术和经验。基于有监督或半监督的配置异常检测方法需要大量的人工参与,这很可能导致检测不够全面、准确。

由于配置文件是一种重复性高、结构化程度高的文本,为了解决上述问题,本文提出了一种基于配置语句树(详见3.2节)的无监督异常检测算法,将配置文件构建成"配置语句树",根据语句树的结构提取配置文件的特征;然后通过无监督的聚类算法对配置文件进行分类,筛查出异常配置,并给出准确的定位以及推荐的修改方案。

3 基于配置语句树的异常检测

为了解决已有的问题,本文首先提出了一种将大数据分析、频率分析和共现语料分析相结合的方法,确定现网运营中的配置文件可能存在的异常类型;其次,提出了一种将每个配置文件构建成一棵结构化的配置语句树的的方法,通过该配置语句树提取配置文件的特征,然后使用无监督学习 *K*-Means 聚类算法筛查出异常的配置文件;最后,提出了一套高效的异常检测的方法,该方法能够准确高效地定位异常并且给出异常类型及推荐的修改方案。

本文采用了"传统的统计学方法"来确定配置文件中可能存在的异常类型,采用聚类的方法找出异常的配置文件结构,采用规则匹配的方法给出推荐的异常修改方案。本文没有将异常检测问题直接定义为有监督机器学习的问题,训练有监督模型进行配置异常检测,而是细化了异常检测的过程,为每个模块设计了最优的方案。由于配置文件是相对规则化的文本,在实际应用场景中,"异常"的配置数远少于正常工作的配置。因此本文采用统计学的方法提取出配置文件中可能存在的异常的类型,相比直接采用有监督的机器学习方法,统计学方法在一定程度上速度更快、准确率更高,比较适合大规模数据的处理,详见 3.1 节。虽然配置文件是高度规则化的文本,但与编程语言相比,它的书写规则和语法规则较为灵活,对前后顺序的要求较为宽松,并且配置文件有一定的结构特征,基于统计学的方法通常不能很好地兼顾到配置文件的结构,因此本文设计了一套基于配置语句树的异常定位方法,该方法能够通过聚类配置文件的语句树抛出异常的配置文件,配置语句记录了配置文件的结构信息,详见 3.2 节。最后,在异常修改方案的推荐方面,对于明确定义了种类的 7 种异常,本文采用了传统的规则匹配的方式,高效地给出了异常类型及推荐方案,而对于那些未定义类型的罕见异常,本文则直接给出异常配置的位置。相比采用神经网络等机器学习方案,规则匹配的速度更快,详见 3.3 节。

3.1 异常类型的确定

现网运行设备配置的厂家、版本众多,语法规则较为灵活,它们所产生的异常也同样是五花八门,现有的巡检工具几乎都是针对配置中词语的拼写、参数的固定格式等展开检测,而事实上,大量影响设备正常工作的几乎都是配置逻辑的

异常，并非简单的词语拼写或者参数格式错误。图 1 分别给出了华三和中兴两个厂家部分设备的部分配置语句，这些配置语句通常结构性较强，配置语句中有段落、句子、词语这些基本结构，以及 ip 地址、端口号等重要的参数，配置语句采用缩进表示层次结构。受运维工程师的技术、经验的限制，在对这些配置语句进行检测时，经常会出现漏检的情况。目前学者们通常采用神经网络、知识图谱等人工智能的方法对自然语言的推荐或异常检测进行建模，这一过程通常需要消耗大量的人力、时间，并且这些模型并不能完美适应复杂的自然语言。相比表达高度灵活、语法规范高度复杂的自然语言，网络设备的配置文件属于规则化文本的范畴，配置文件中也只包含了设备基本信息、ip 地址、端口号、密码等为数不多的几类信息。本文在综合分析上述因素后，决定采用传统的统计学方法确定配置中存在的异常类型。

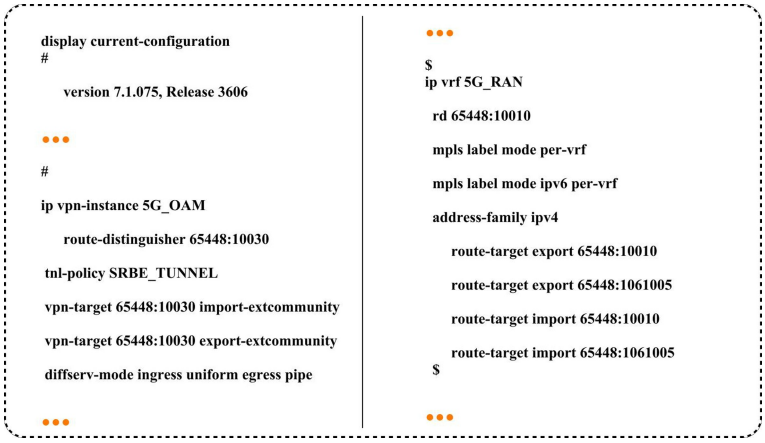


图 1 “华三”厂家某配置文件部分语句(左)和“中兴”厂家某配置文件部分语句(右)

Fig. 1 Statement of a configuration file of “Hua-San” manufacturer(left), and statement of a configuration file of “ZTE” manufacturer(right)

本文基于事实合理推断设备配置的异常是小概率事件，是低频的、稀少的，如果设备存在大量异常，则运营商的网络服务根本不可能正常提供。基于“配置异常通常是低频的、稀少的”这一推断，本节提出了采用大数据分析、共现语料分析、频率分析的方法来确定现网配置中可能存在的异常类型。实验结果证明，本文通过统计学方法找出的异常类型可涵盖配置文件中 90%以上的异常，剩下不到 10%的异常通常是非常罕见的，是没有任何规律的异常。相比神经网络和知识图谱，在一定程度上采用传统的统计学的方法确定配置文件中的异常类型在准确率和速度上更有优势。

3.1.1 配置文件预处理

本节将配置文件中的 ip 地址、端口号、密码、用户信息等参数通过正则表达式匹配后分别采用统一的通配符进行替换。对于剩下的所有词语，通过频率分析的方式将其划分为“关键字”和“非关键字”，“关键字”指出现频率高、通用性强且有规律的词语，而“非关键字”指出现频率低、通用性差且规律性差的词语，这些“非关键字”通常是一些附加信息，如设备序号、地名、设备名等，它们不具备对其进行检测的必要。本节对“关键字”和“非关键字”进行划分的算法的过程如下。

1)获取高频词。为获取配置中的“关键字”并找到最值得挖掘和研究的配置词语，本节首先计算除上文中被替换的词语外剩下所有词语的词频。均匀抽取不同厂家的配置文件共 3000 份，总计 200 多万行配置语句。统计所有词语记为集合 S，$S=\{A_1, A_2, \cdots, A_n\}$，其中每个词 A_i 总共出现的次数为 $N(A_i)$，则词 A_i 的词频 $f(A_i)$ 为：

$$f(A_i)=\frac{N(A_i)}{\sum_{i\in S} N(A_i)} \tag{1}$$

计算集合 S 中每个词语 A_i 的词频 $f(A_i)$，接着将所有词语按照词频从高到低进行排序，从中取前 1%的词语，得到 $F=\{B_1, B_2, \cdots, B_p\}$ 为高频词集合。

2)获取“活跃”词。有些词语，如地名，在某个配置中大量重复出现，但是在别的配置文件中出现的频次很少，这些词在步骤 1)中也许会被归类到高频词的集合中，但是它们显然不是本节最终要寻找的“关键字”，因此在这一步中，我们需要计算所有词的“文档频率”，即找出那些“活跃”在各个配置文件中的词语。对于集合 S 中的词语，首先计算含有每个词的文档数 $D(A_i)$，则词 A_i 的文档频率 $w(A_i)$ 为：

$$w(A_i)=\frac{D(A_i)}{\sum_{i\in S} D(A_i)} \tag{2}$$

计算集合 S 中每个词语 A_i 的文档频率 $w(A_i)$，接着将所有词语按照文档频率从高到低进行排序，从中取前 1%的词语，得到 $G=\{C_1, C_2, \cdots, C_p\}$ 为“活跃”词集合。

3)划分“关键字”和“非关键字”。将上述两个步骤中计算得到的集合 F 和集合 G 做交集运算，即可得到“关键字集合”，记为集合 T。剩余不在集合 T 中的词为“非关键字”。

最后，根据上述步骤中得到的集合 T 保存配置文件中的关键字，将非关键字用统一的通配符进行替代，得到预处理后的配置文件，如图 2 所示。图中用色块圈注的部分则为根据上述规则被使用通配符替换的词语。

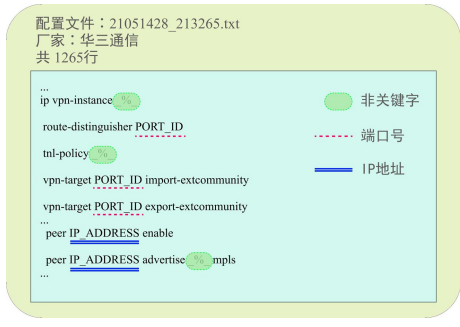


图 2 预处理后的配置文件示例

Fig. 2 Example of a pre-processed profile

3.1.2 共现语料分析

由于配置语句前后之间的关联性较大，本节采用“共现语料分析”的方法分析现网配置中可能存在的异常类型。首先依次统计配置文件中“两两共现”（相邻）的句子、词语作为一个整体出现的次数，构建共现语料库。图 3 为“共现语句”语料库的一部分。通过分析语料库，可以发现编号为 6 的共现语句‘pic import-route direct’出现频次为 5136，而 12431 号共现语句‘import-route direct pic’的顺序刚好和 6 号语句相反，其出现频次为 3。因此，本文认为 12431 号发生了句子顺序异常。在进行共现语料分析时，本节针对语料库中发现的频次异常的部分进行有针对性的分析，充分和运维工程师进行讨论，最终确定了在配置语句层面有 3 种类型的异常：配置语句顺序异常、配置语句冗余、配置语句缺失。类似地，本节还基于词语构建“共现词语”语料库，采用同样的方法进行分析，发现部分词语间可能漏打了空格。

	共现语句	厂家	出现频次
1	role name _%_ description Predefined _%_ role	华为	2356
2	vpn-target PORT_ID import-extcommunity vpn-target PORT_ID export-extcommunity	烽火通信	7429
3	aaa-authentication-template _%_ aaa-authentication-type _%_	中兴	1278
4	peer public key end public key code end	华为	7
5	ipv4 address IP_ADDRESS IP_ADDRESS carrier-delay up _%_ down _%	思科	976
6	pic import-route direct	华三通信	5136
7	interface _%_ shutdown	思科	7212
	.....	.....	.....
12431	import-route direct pic	华三通信	3
12432	ip mtu _%_ ipv6 enable	中兴	6975

图 3 部分“共现语句”语料库

Fig. 3 Part of the "co-occurrence" corpus

另外，本节还将高频的词语和句子作为模板，对配置进行频率分析。将句子和词语分别按照频率高低进行排序，取前 1%的词句。然后计算剩余词句和这些高频词句之间的编辑距离[24]，通过分析与高频词句之间编辑距离较小的低频词句，发现了配置语句中存在拼写错误的情况，并且这些存在拼写错误的词语全部在关键字集合 T 中，因此将其确定为关键字拼写异常。经过上述的分析与探索，本节明确了 5 种类型的配置异常：关键字拼写错误、关键字之间漏打空格、配置语句顺序异常、配置缺失、配置冗余。再加上 ip 地址和端口号是配置异常的“高发区”，一共给出 7 种明确的、可检测的异常类型。

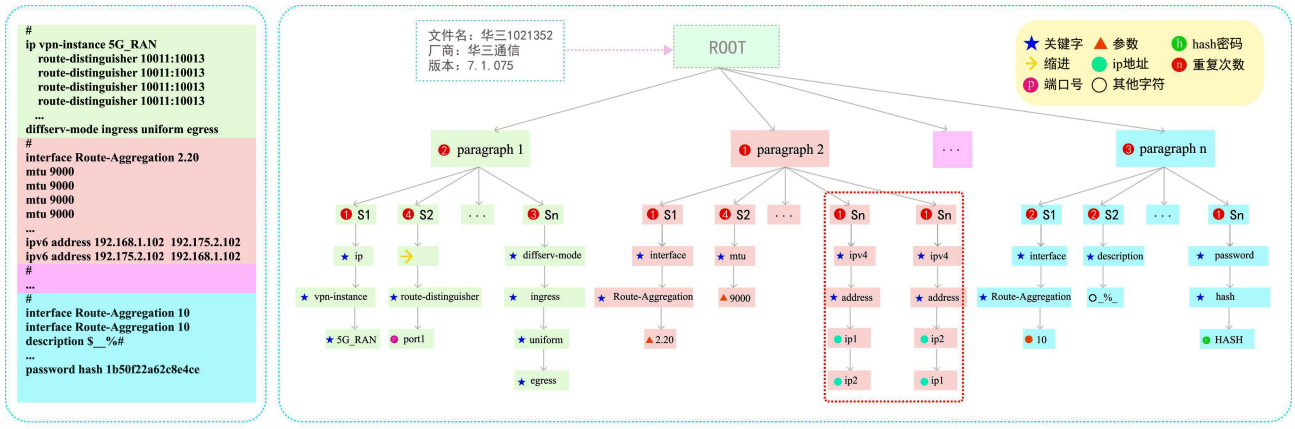
尽管到目前为止，本文可以依赖统计学的方法分析现网配置中存在的某些异常，但是由于本节在使用统计学的方法进行分析时忽略了 ip 地址、端口号等信息，导致目前无法具体地得出其中的异常。其次，在分析段落、句子、词语时，本节只是基于统计学指标考虑异常的类型，由于尚未考虑句子结构、段落结构等信息，因此无法完成对全部的异常进行定位和排查。在 3.2 节中，本文借鉴了抽象语法树的设计理念，创新性地提出了一种将配置文件构建成“配置语句树”以保留配置文件的结构信息，通过该配置语句树提取配置文件的特征，随后采用无监督的聚类算法找出异常的配置文件。

3.2 配置语句树的构建

由于传统的基于规则的配置异常检测方法通常检测规则较为固定死板，基于统计学的配置异常检测通常不能很好地兼顾到配置文件的结构。回顾编程语言的设计，编译器[25]能够通过快速扫描程序文件并将其解析成抽象语法树的方式生成可执行代码，因为抽象语法树刻画了编程文件所有可能存在的信息，包括内容和结构。本节借鉴了抽象语法树的设计理念，将配置文件构建成“配置语句树”，提取配置文件的特征，并采用无监督的聚类算法找出异常的配置文件。

本文构建的“配置语句树”是一棵 3 层以上的多叉树结构，语句树的第一层为配置文件信息节点，包含文件名、厂家、设备的版本等信息；第二层为段落节点，包含段落的起始和终止位置（行信息）以及该段落结构连续最大重复的次数；第三层为句子节点，包含该句子所在行的位置信息以及该句子结构连续最大重复次数；第四层起则为配置语句树的词节点，每个词节点均包含词的内容、该词语的类型、关键字还是非关键字或是 ip 地址、端口号等。如图 4 所示，这棵配置语句树表明，配置文件名为“华三 1021352”，厂家为华三通信，该配置的版本为 7.1.075。段落依次由‘paragraph 1’‘paragraph 2’…‘paragraph n’组成。第二层的 paragraph 节点中详细记录了段落的起止位置以及连续重复次数的信息，如图 4 所示，‘paragraph 1’重复了 2 次，‘paragraph 2’重复了 1 次，省略号中为省去未展现的段落，‘paragraph n’ 重复了 3 次。第三层的 S 节点代表句子，其中也记录了句子所在行的位置信息以及句子的连续重复次数。以图中的 paragraph1 为例，paragraph1 依次由‘句子 1($S1$)’‘句子 2($S2$)’，省略的部分为该段省去未显示的句子，‘句子 n(Sn)’组成。其中，‘$S1$’重复了 1 次，‘$S2$’重复了 4 次，…，‘Sn’重复了 3 次。S 节点以下则为句子节点，句子 1($S1$)由 3 个关键字组成，因此 $S1$ 为：‘ip vpn-instance 5G_RAN’，句子 2($S2$)依次由一个层次符、一个关键字和一个端口号组成，因此 $S2$ 为：‘quad route-distinguisher port’。

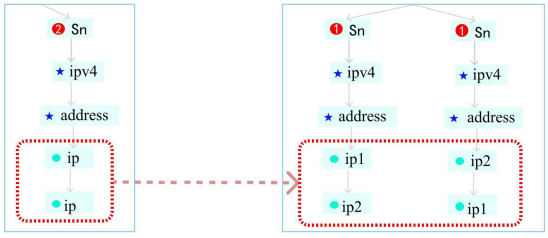
对于 ip 地址、端口号、hash 密码、参数等信息在构建配置语句树时会考虑它们本身的一些特征：同一段落内的 ip 地址和端口号等通常将具有较强的关联性，如图 4 中用红色虚线框圈注的两个配置语句，‘ipv4 address 192.168.1.102 192.75.2.10’与‘ipv4 address 192.75.2.102 192.168.1.102’，这两句话依次出现，按照上述的构建规则，它们会被抽象为两个相同的句式模板保存在同一个句子节点中。如图 5(a)所示，但由于这两句话的 ip 地址具有较强的相关性，因此在构建树时应该被当成两个独立的句子。配置语法树不仅需要以最少的代价、最简洁的方式尽可能多地记录配置文件的结构和内容信息，也同时需要尽可能更合理地记录配置语句中参数之间的逻辑关系。因此，在构建配置语句树时，对于同一个段落内的 ip 地址、端口号、hash 密码、注册号等参数，需要对同一个类型的不同内容加以区分。例如，上述依次出现的两个带有 ip 地址的语句，由于它们处于同一个段落内，因此将‘192.168.1.102’记录为 IP1，而将‘192.75.2.102’记录为 IP2。在生成配置语句树时，这两句话分别被抽象成‘ipv4 address IP1 IP2’与‘ipv4 address IP2 IP1’，如图 5(b)所示。由于考虑了 ip 地址的逻辑关系，这两句话也便不能被合并为一句话。在面对其他的参数构建配置语句树时，也是按照同样的方式进行处理。



(a)配置文件中某配置语句片段　　(b)该配置文件的“配置语句树”

图 4　某配置文件片段及其“配置语句树”

Fig. 4　A certain configuration file fragment and its “configuration statement tree”



(a)未考虑段落中参数内容之间关联构建的分支　　(b)考虑了参数内容之间的关联后构建的分支

图 5　两种在配置语句树中构建参数节点的方式

Fig. 5　Two ways to build parameter nodes in the configuration statement tree

根据上述步骤将每个配置文件都构建成一棵配置语句树，针对每棵配置语句树 i，分别统计其段落节点的数量，记为 α_i，句子节点的数量，记为 β_i，关键字数量，记为 γ_i，接着本文针对每棵配置语句树 i 生成一组特征向量 $\boldsymbol{T}_i=\{\alpha_i,\beta_i,\gamma_i\}$，将这些特征向量组 T 作为 K-Means 算法的特征输入，采用欧氏距离[26]作为距离测度。而 K 值的选择依照配置语句树节点中记录的配置文件的版本号，因为相同版本的配置文件的书写逻辑并无别差。本文选取了 20 000 个配置文件作为训练集，依照版本号共划分为 5 个类别，即 K 的值为 5。因此聚类算法的运行步骤为：1)随机选择初始化的 5 个样本作为初始聚类中心；2)针对数据集中的每个样本，计算它到 5 个聚类中心的欧氏距离并将其分到距离最小的聚类中心所对应的类中；3)针对每个类别重新计算它的聚类中心。将所有的配置文件进行聚类后，每个簇中的离群点则为异常的配置文件，而簇中心的文档可以视作正确的“模板”。

本节通过为每个配置文件创建配置语句树，最大程度地保留了配置文件的结构内容特征，在构建树的过程中，本节还提出了单独对 ip 地址、端口号、hash 密码等参数进行处理来保证配置语句树能最大程度地记录参数之间的逻辑关系。随后本节采用无监督的 K-Means 算法对这些语句树进行聚类，得到了异常配置文件检测的模型，提取那些离群点为异常文档，而那些簇中心点则被当作正确配置的“模板”。下一节将针对筛出的异常配置文件进行异常定位以及给出推荐的修改方案。

3.3　异常定位与修改方案推荐

至此，通过 3.2 节中的聚类模型，能正确地筛出存在异常的配置文件，而本节主要的问题是精准地定位配置文件中的异常所在，如果它们满足本文 3.1 节中确定的 7 种异常之一，则推荐出相应的修改方案，本节中的异常修改方案的推荐基于一种改进规则判断的方式。虽然，目前许多专家学者都在研究如何采用机器学习和神经网络技术来设计推荐算法以获得更好的通用性，但考虑到文本算法采用统计学方法确定了 7 种异常类型，涵盖了 90%以上的主要异常类型，这 7 种异常类型均具有非常明确的定义。并且，本文根据配置语句树的聚类结果中找出的异常，设计了一套高效的算法来快速定位到发生异常的句子或词语，因此传统的基于规则的修改方案推荐方法相比采用基于机器学习和神经网络技术的推荐算法具有更高的准确率和运行效率。

图 6 给出了异常定位与修改方案推荐的流程，首先将异常配置文件和上一节聚类后提取的“模板”配置文件按照段落节点、句子节点、词语节点的顺序逐一进行比对，便可以快速定位到异常所在段落、所在句子或是所在词语。如果定位到异常节点是单个词节点，则意味着是词语方面的异常。本文 3.1 节中提出的词语的异常包括：关键字拼写错误、关键字之间缺少空格，还有特别的 ip 地址和端口号异常。下面依次讨论如何分别确定这些异常及其推荐的修改方案。

1)关键字拼写异常，如果匹配到异常词语的类型为“其他词语”，则有可能为非关键字。这时，本文对比异常词语和“模板”中对应节点的关系，如果“模板”中的对应词语是关键字，并且计算两个词之间的“编辑距离”，如果编辑距离小于 2，则判定为关键字拼写错误，正确的则为“模板”中对应的词语。最后给出异常定位，异常类型和推荐的修改方案；如果上述过程中计算出的编辑距离大于 2，则去掉“关键字语料库”中匹配与之编辑距离小于 2 的词语。若能成功匹配到，则说明确为关键字拼写错误，正确的写法为语料库中的对应词，同样给出异常定位、异常类型和修改方案；否则，说明并不是关键字拼写错误。

2)关键字之间缺少空格，如果异常节点的类型为“其他词语”，并且可以确定不是关键字拼写错误，则以“关键字语料库”中的词语为参照进行语义分词，如果分词的结果都可以在

语料库中匹配到，则可以明确异常为“关键字之间缺少空格”。最后给出异常定位、异常类型和推荐的修改方案。

3）对于异常节点类型为“其他词语”的，若在匹配后确认不是上述两种异常，则可能是 ip 地址和端口号等参数的异常。如果“模板”中对应正常节点的类型为“ip 地址、端口号”等，则说明异常原因是这些参数的格式错误。由于语句树记录了段落中每个 ip、端口等参数间的逻辑规律，按照“模板”中的规律，即可找到异常的 ip 地址或端口号等参数的正确内容。

4）ip 地址、端口号的逻辑顺序异常，如果异常节点类型成功匹配为“ip 地址”或“端口号”，则说明 ip 与端口号的格式未发生错误，可以推断是这些参数的逻辑配置出现了异常。因此，本文参照“模板”中的格式，学习正确的参数的书写逻辑，调整 ip 地址和端口号的逻辑顺序异常。

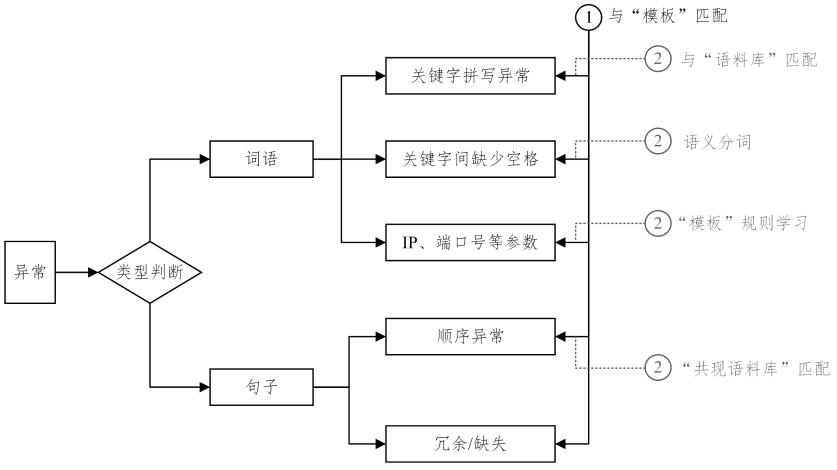


图 6　异常定位与修改方法推荐算法的工作流程

Fig. 6　Workflow of anomaly locating and modification plan recommendation algorithm

如果定位到的异常是句子层面的异常，根据 3.1 中确定的异常类型，句子可能出现顺序异常、配置语句缺失或冗余。语句的异常判断相比词语更为简单，下面依次讨论如何分别确定这些异常及其推荐的修改方案。

1）确定句子前后顺序是否发生异常有两个途径。首先尝试与“模板”树的对应分支进行匹配，判断与之匹配的句子是否与发生异常的句子存在对应的顺序关系，如果可以，则给出异常定位、异常类型和推荐的修改方案。但由于语句的合并，“模板”未必能直接匹配到。对于发生异常的句子，如果无法直接通过匹配“模板树”得出异常类型，则本节将发生异常的语句及其下一个配置语句捆绑成一组语料 R，在 3.1 节构建的“共现语句语料库”中查询 R 是否为“低频次语料”，如果确为“低频次语料”，则给出异常定位、异常类型以及推荐的修改方案。

2）对于语句缺失或冗余，本文通过与“模板”树进行匹配，来寻找是否有漏配、多配的情况，如果是，则给出异常定位、异常类型和推荐修改方案。

4　算法评估

本节主要从两方面对本文的异常配置检测算法进行评估，首先将本文算法与传统的基于规则的 ptn 网络巡检工具——ZXSEM/TIM400、广东联通提出的基于 AI 关联分析的方法[10]、基于贝叶斯网络（Bayesian Networks）的异常检测方法[22]以及基于 FpGrowth 的异常检测算法进行对比分析[23]，突出本文算法对于异常捕捉的全面性与丰富性。其次，从准确率、召回率以及检测效率 3 个方面对该算法进行量化评价，进一步证明本文算法在异常检测中相比其他两种算法的优势。本文实验环境为 Intel（R）Core i7-9700 CPU、16GB 内存的 Windows 10 系统台式机，算法在 python3.6 环境下运行。程序采用单线程实现，运行过程中只用到了 CPU 的一个核心。

4.1　实验数据来源与安全性问题

在算法实验和评估的过程中，本文采用的数据集均来自国内某运营商提供的其现网运营的部分网络设备配置文件，这些配置文件来自 5 个不同的厂家，包括中兴、华三通信、华为、烽火通信、思科。由于现网运营的配置对隐私保护等安全问题要求很高，因此本文在使用这些数据进行实验前对其进行了加密。对于 ip 地址和端口号信息，本文参考了 Zhao、Yi 等的工作[28-29]，在加密过程中分别对每个 ip 地址以及端口号的每一位都随机分配一个数值，随后将其每一位的 ASCII 码都加上这个数值作为加密后的结果，而将这些随机生成的数值作为密钥。本文在大量的调研和实验中发现，ip 地址和端口号的异常主要是相互顺序关系，前后配置顺序错乱等顺序异常。本文采用的增加 ASCII 编码的加密方式保留了 ip 地址和端口号之间的顺序信息和相互关系信息，在异常检测任务需要第三方介入或有其他保密需求保密的情况下，这种加密方式可以在不解密的情况下支持异常检测工作的顺利开展；而对于路由器的用户名、密码以及用户名等敏感信息文本，本文采用了基于 Hash 函数和三元组的密码表加密算法[27]对其进行加密，确保了数据集安全以及对用户隐私的保护。

4.2　可检测异常类型对比分析

本节采用了来自华三、华为、思科、中兴以及烽火通信 5 个厂家的配置，每个厂家分别随机取 200 个城市，每个城市取 100 份配置文件，总计 100 000 份配置文件作为数据集，将本文算法与传统的基于规则以及基于 AI 关联分析的方法分别在此数据集上进行多次实验。

实验对本文提出的 7 种类型的配置异常进行检测。如

表 1 所列，传统的基于规则的 ptn 网络巡检工具只能识别 3 种异常，为关键字拼写与空格缺失，以及 ip 地址和端口号的格式异常，这些异常都是简单的、不涉及到配置逻辑的异常，虽然目前基于 AI 上下文关联分析算法可以解决逻辑关联上的问题，但是在传统的错误检测方面表现不佳，而 Bayesian Networks 和 FpGrowth 支持的异常检测类型也不如本文方法丰富。显然，与其他几种方法进行对比，本文算法对于异常的检测能力更为全面，本文算法的泛化性较强。

表 1 不同算法可检测的种类

Table 1 Types that can be detected by different algorithms

异常类型	本文算法	传统算法	AI 关联分析	Bayesian Networks	FpGrowth
关键字拼写	√	√	×	√	√
空格缺失	√	√	×	√	√
ip 地址异常	√	√	×	×	×
端口号异常	√	√	×	×	×
命令缺失	√	×	√	√	√
命令冗余	√	×	√	√	√
句子顺序异常	√	×	√	√	√

不仅如此，本文算法采用的配置语句树是无监督的，即便在异常类型未知的情况下，即除了本文采用统计学方法定义的这 7 种类型的异常外，通过配置语句树的聚类算法仍然可以定位出其他异常的位置，并且将这些结构异常实时返回给运维人员进行分析。通常，传统的基于规则或者基于关联分析的方法是有监督或是半监督的。若想实现异常检测，需要人为归纳出异常类型，并对数据集进行标注，这样的方法显然会受到人们思维局限性的影响，既费时费力，又不全面。比如，在某个配置文件中，有配置语句‘aaa session-limit telnet’和‘aaa session-limit ssh’，这两句话看上去不属于任何一类异常，也都是高频语句，两句话共现频次也很高，工程师们很难发现这两句话在一起有何异常。但是在对语句树进行聚类分析时，这两句话被当成异常抛出。经过运维工程师的确认，对配置段落进行分析，才发现是由于该设备的权限不够导致无法用这两条配置，而并非这两条配置语句本身有问题。本文算法是一种无监督的方法，无需人工标注异常数据，可以自动检测出各种异常。经过对比分析，本文算法实现的异常检测更具全面性与智能性。

4.3 量化评估

4.3.1 准确率

准确率是衡量一种算法有效性最核心的指标。由于本文采用的是无监督的聚类算法，因此无法直接评估算法检测所有异常的准确率。故本节的准确率评估仅针对本文基于共现频率分析的统计学方法找出的 7 种类型的概率最高的异常展开。本节仍然采用来自 5 家厂商的 10000 份数据进行测试，首先使用本文算法扫描出数据集异常并纠正，得到在本文算法下无异常的数据集。其次，针对 7 种异常类型，每份配置的每种类型人为添加 20 处错误，再次对修改后的数据集使用此算法，统计准确率。

由于传统的 ptn 网络巡检工具是基于传统的匹配规则进行异常检测的，匹配规则非常明确，并非采用了统计学方法或机器学习的方法，因此统计传统 ptn 巡检工具的准确率没有太大的意义，故本文算法主要和基于 AI 关联分析的算法、基于贝叶斯网络(Bayesian Networks)的异常检测算法以及基于 FpGrowth 的异常检测算法进行对比。

如表 2 所列，本节就关键字拼写及缺失空格、ip 地址、端口号、命令缺失、命令冗余以及句子顺序等异常检测的准确率进行统计分析，本文算法的平均准确率达到 86%，在关键字拼写和关键字缺失空格这两个独立场景中甚至可以达到 90%以上，基于关联分析算法的平均准确率为 85%，且没有能达到 90%以上的。通过比较分析可以得出，本文算法的准确率提升显著。

表 2 准确率对比分析

Table 2 Comparative analysis of accuracy

(单位：%)

异常类型	本文算法	AI 关联分析	Bayesian Networks	FpGrowth
关键字拼写	91	×	87	85
空格缺失	94	×	91	87
ip 地址异常	89	×	×	×
端口号异常	82	×	×	×
命令缺失	88	86	71	44
命令冗余	84	83	68	79
句子顺序异常	82	82	75	77

4.3.2 召回率

召回率是衡量一种算法可靠程度的指标，指算法实际测出的异常占理论应测异常的比例。由于本文采用的是无监督的聚类算法，因此无法直接评估算法检测所有异常的召回率。故本节的准确率评估仅针对本文基于共现频率分析的统计学方法找出的 7 种类型的概率最高的异常展开。本节继续采用 4.3.1 节中提到的 10000 份数据，在经过纠正数据集与人为添加异常后，使用本文算法进行分析。本节将本文算法和基于 AI 关联分析算法、基于贝叶斯网络(Bayesian Networks)的配置异常检测算法以及基于 FpGrowth 的配置异常检测算法进行对比。由于传统的基于规则的 ptn 巡检工具没有采用统计学方法或机器学习的方法，不涉及召回率的问题，因此在统计召回率时本节未考虑与传统 ptn 巡检算法进行对比。

表 3 召回率对比分析

Table 3 Comparative analysis of recall rates

(单位：%)

异常类型	本文算法	AI 关联分析	Bayesian Networks	FpGrowth
关键字拼写	93	×	90	88
空格缺失	97	×	93	89
ip 地址异常	86	×	×	×
端口号异常	42	×	×	×
命令缺失	83	82	78	74
命令冗余	85	86	51	62
句子顺序异常	87	24	75	38

实验结果如表 3 所列，关键字拼写、关键字缺失空格、ip 地址异常以及命令缺失的召回率都能达到 90%以上，平均召回率达到了 85.1%。在异常检测中，本文算法与基于 AI 的关联分析算法以及基于 Bayesian Networks 或 FpGrowth 的方法在拼写错误与空格缺失方面差距并不大，例如正确示例‘port link-mode route’中‘route’错写为‘rote’，‘address-family ipv4 unicast’漏掉空格写成‘ipv4unicast’，这种异常的召回率通常都很高，但若出现形如 A+B 与 B+A 这种语句顺序问题，本文算法的表现就极大地领先了其他几种算法，例如语句 A 为‘address-family ipv4’而语句 B 为‘route-distinguisher

65448:20001'，A 与 B 的前后顺序调换并不会造成语法错误，A 在前 B 在后或 A 在后 B 在前的写法在语法上都是正确的。然而，对于传统的 ptn 巡检工具、关联分析算法或基于 Bayesian Networks 或 FpGrowth 的配置异常检测算法，由于它们采用的是人为制定的规则或者它们在训练时需要人工标注一部分数据，AB 语句的顺序一经调换，算法中没有相应的规则或者算法本身没有学习到相应的规则与之匹配，因此找不到相应的异常，导致训练的异常检测模型并没有学习到这类异常的特征，这就导致了其较低的召回率。本文使用的配置语法树采用无监督的聚类算法，通过将异常配置的语句树与正常的“模板”语句树进行对照来匹配出相应的异常配置文件，采用统计学方法确定了可能存在异常的种类，抛弃了人工输入规则或人工标注部分异常的方式，鲁棒性更强，可靠性更高，因此本文算法的召回率更高。

另外，本节实验过程中除了在 10 000 份样本文件中对采用统计学方法确定的 7 种定义的异常进行召回率验证，还另外抛出了 27 处不在这 7 种异常范围内的异常，这 27 处异常分布在同一个骨干网同一省份的 16 份配置文件中。工程师们参考这一信息快速对该省份的骨干网节点配置展开了有针对性的筛查，最终根据本文算法提供的信息成功解决了该省份的配置异常。本文算法充分利用了配置语句的“强规则性”和书写时的“灵活性”，采用无监督机器学习模型和统计学理论相结合的方法，融合了无监督方法不需要人工事先标注数据的优势以及统计学方法在处理大规模数据中少量异常时的高效和准确的优势。相比纯统计学方法和纯机器学习的方法，本文算法无论是在全面性、准确性、效率方面都有优势。

4.3.3 效率

本节分别采用本文算法与基于规则的 ptn 巡检工具进行测试，执行两种算法，统计其完成相同任务所耗费的时间。本节分别在 5 个不同厂家的配置中各随机抽取 2 000 份，共计 10 000 份配置文件和 17 521 900 行配置语句。

在采用 ptn 自动巡检工具为以上配置进行异常检测时，由于工具本身鲁棒性不高，第一次实验运行了 86 min 后，进程被强制中断。因此从头进行第二次实验，花费 8 h 32 min，顺利完成异常检测任务，采用该方法检测平均每份文件需要花费 3.07 s。而采用本文算法对同样的配置语句进行异常检测时，一次性完成任务，共耗时 1 h 18 min，平均每份配置文件的检测时间约为 0.47 s，而采用基于 AI 关联分析的配置异常检测以及采用基于 Bayesian Networks 的配置异常检测算法和基于 FpGrowth 的配置异常检测算法完成上述同样的任务，耗时均在 1 h 35 min 左右，略慢于本文算法。

综上可以清晰地看出，基于规则的 ptn 巡检工具在数据集上的检测时间大约为本文算法的 6 倍，本文算法在效率上远远领先于传统算法。对于基于 Bayesian Networks 的配置异常检测算法和基于 FpGrowth 的配置异常检测算法，本文算法在效率上稍占优势，但本文算法的准确率、召回率和泛化能力远高于这两种算法。

4.3.4 算法稳定性

为了评估本文算法的稳定性，本文主要就聚类模型的稳定性展开评估。本节分别在 5 个不同厂家的配置文件中各随机抽取 2 000 份，共抽取 100 组，用于训练 100 个不同的聚类模型。随后随机抽取 10 000 份配置文件，采用自动化脚本批量为每个文件随机制作 20 组异常配置，共计 200 000 处异常。随后，将这 10 000 份配置文件分别输入进这 100 个模型中，统计其检出率。本节实验选取了 3 种不同的特征，第一种为本文最终采用的段落节点数、句子节点数和关键字数，将其作为特征输入（以下称 PSK 特征）；第二种为将最底层节点数、段落节点数作为特征输入（以下称 PD 特征）；第三种将句子节点数和关键字作为特征输入（以下称 SK 特征）。本节分别对这 3 种不同的特征输入进行了上述实验。

如图 7 所示，x 轴为 100 个模型的序号，y 轴为该模型检测出的异常占预设的异常的比例，其中蓝线对应的模型是本文最终采用的，通过该图可以看到，折线相对平稳，表示本文模型的稳定性较高。而红色和绿色的折线分别对应的是采用其他两种特征选取时的结果，可以看到，此时算法的稳定性较差，经过分析，出现该现象的原因是特征选取时未覆盖所有厂家不同版本的文件的段落句子等特征，导致不同版本的配置文件使用该算法检测时的结果差异较大。而本文选取的段落节点数、句子节点数和关键字数在一定程度上能代表一个配置文件的全部信息，而无关乎厂家与版本。实验结果证明，本文算法以及采用的模型和特征选取具有较强的稳定性和泛化性，鲁棒性较高。

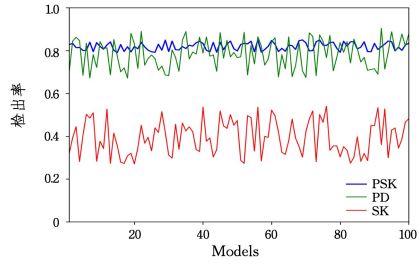


图 7 100 组模型的检出率（电子版为彩图）

Fig. 7 Detection rate of 100 group models

结束语 本文提出了一种基于配置语句树的无监督网络设备配置异常检测方法，该方法借鉴了抽象语法树的设计理念，首创性地提出了将网络设备的配置文件构建成“配置语句树”以保留配置文件的信息，通过该配置语句树提取配置文件的特征，随后采用无监督的 K-Means 聚类算法找出异常的配置文件。本文对配置文件进行频率分析、共现语料分析，确定了 7 种可检测的异常类型，并给出了异常定位及推荐的修改意见。从量化评估的结果看，算法的平均准确率和平均召回率均大于 85%，部分场景达到了 90%以上。对千万行配置的规模，相比传统的 ptn 巡检工具，本文算法的速度快了约 6 倍。此外，除了确定的 7 种类型异常外，对于其他可能存在的异常，算法可以给出它们的定位，然后交由运维工程师人工判断。但是，本文算法仍有以下不足：配置语句树未与跨段落的 ip 地址、端口号等参数之间建立联系，无法检测段与段之间相关的异常；本文算法有一定的通用性，但是将过多不同厂家的配置文件放在一起进行异常检测，准确率和召回率会降低；在异常定位、异常筛查方面，本文表现较为出色，但在给出异常推荐修改方案时仍然是基于传统的规则匹配的方法进行的。

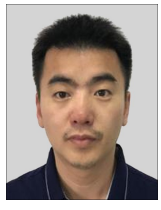
参 考 文 献

[1] WILLIS P J. The challenges in building a carrier-scale IP network[J]. BT Technology, 2000, 18(3): 11-14.

[2] GOZDE B, ALIDSMAN A. AHP integrated TOPSIS and VIKOR

methods with Pythagorean fuzzy sets to prioritize risks in self-driving vehicles[J]. Applied Soft Computing, 2021, 99(3): 1568-4946.
[3] SIRIWARDHANA Y, PORAMBAGE P, LIYANAGE M, et al. A survey on mobile augmented reality with 5G mobile edge computing: architectures, applications, and technical aspects [J]. IEEE Communications Surveys & Tutorials, 2021, 23(2): 1160-1192.
[4] LIU G H, MENG X C, ZHOU X R, et al. Exploring the optimization of China Unicom packet domain IP bearer network architecture for 5G[J]. Telecommunications Technology, 2019(12): 95-98.
[5] WANG W Q. PTN network inspection solution for LTE[J]. Science and Technology Innovation, 2020(27): 62-63.
[6] LIU H M, CHEN G. Innovative research and practice of network operation and maintenance system based on centralization and intelligence[J]. China New Communication, 2015, 17(2): 68-71.
[7] CUI J. Introduction to the construction of intelligent operation and maintenance mode of 5G network[J]. Technology and Market, 2021, 28(5): 126-127.
[8] THEO A, NATALI H, SANNE K, et al. In AI we trust? Perceptions about automated decision-making by artificial intelligence[J]. AI & SOCIETY, 2020, 35(3): 611-623.
[9] GUPTA S, SACHIN M, SAMADRITA, et al. Artificial intelligence for decision support systems in the field of operations research: review and future scope of research[J]. Annals of Operations Research, 2022, 308(1): 215-274.
[10] LIU X W, MA D D, YE X B, et al. Application of AI based Configuration Audit System in 5G Backhaul Network[J]. Designing Techniques of Posts and Telecommunications, 2021(8): 15-19.
[11] LIN T L, CHEN J G, GUO W J, et al. Application of big data analysis methods in 5G precision construction[J]. Changjiang Information and Communication, 2022, 35(6): 230-232.
[12] HOFMANN M J, BIEMANN C, WESTBURY C, et al. Simple Co-Occurrence Statistics Reproducibly Predict Association Ratings[J]. Cogn Sci, 2018, 42(7): 2287-2312.
[13] ZHANG J, WANG X, ZHANG H, et al. A novel neural source code representation based on abstract syntax tree[C]// 2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE). IEEE, 2019: 783-794.
[14] SINAGA K P, YANG M S. Unsupervised K-Means clustering algorithm[J]. IEEE access, 2020, 8: 80716-80727.
[15] LIU X, ZHU P D, MI Q, et al. Rule-based anomaly detection for inter-domain routing systems[J]. Journal of the National University of Defense Technology, 2006(3): 71-76.
[16] SMRITHY G S, RAMADOSS B. A Statistical-Based Light-Weight Anomaly Detection Framework for Wireless Body Area Networks[J]. The Computer Journal, 2022, 65(7): 1752-1759.
[17] YU Y J, YIN Y F, LIU Q. Analysis of the distribution pattern of high-frequency Chinese character string mutual information based on large-scale corpus [J]. Computer Science, 2014, 41(10): 276-282.
[18] PINCOMBE B. Anomaly Detection in Time Series of Graphs Using ARMA Processes[J]. Asor Bulletin, 2005, 24(1): 67-75.
[19] ROODBANDI J, SADAT A, CHOOBINEH A, et al. Research outputs in ergonomics and human factors engineering: a bibliometric and co-word analysis of content and contributions[J]. International Journal of Occupational Safety and Ergonomics, 2022, 28(4): 2010-2021.
[20] LIU D P, ZHAO Y J, XU H W, et al. Opprentice: Towards Practical and Automatic Anomaly Detection through Machine Learning[C]// 15th Internet Measurement Conference. Tokyo, Japan. New York: ACM, 2015: 211-224.
[21] YANG X W, LATECKI L J, POKRAJAC D. Outlier Detection with Globally Optimal Exemplar-based GMM[C]// International Conference on Data Mining. SDM, Sparks, Nevada, USA. New York: SDM, 2009: 145-154.
[22] RASHIDI L, HASHEMI S, HAMZEH A. Anomaly detection in categorical datasets using bayesian networks[C]// International Conference on Artificial Intelligence and Computational Intelligence. 2011: 610-619.
[23] SHABTAY, LIOR, et al. A guided FP-Growth algorithm for mining multitude-targeted item-sets and class association rules in imbalanced data[J]. Information Sciences, 2021, 553(1): 353-375.
[24] MAHDI B, SOHEIL E, MOHAMMAD G, et al. Approximating edit distance in truly subquadratic time: Quantum and mapreduce[J]. Journal of the ACM, 2021, 68(3): 1-41.
[25] MERIGOUX D, MONAT R, PROTZENKO J. A modern compiler for the french tax code[C]// Proceedings of the 30th ACM SIGPLAN International Conference on Compiler Construction. 2021.
[26] DONG Z B. Analytical and Research on 3D Point Cloud Segmentation Algorithm Based on Improved Euclidean Distance [D]. Beijing: North China Electric Power University, 2022: 4-38.
[27] CAO J D. Research on cryptographic table encryption algorithm based on Hash function and triplet [J]. Software Guide, 2012, 11(11): 54-56.
[28] ZHAO X H. Research on encryption method based on DNA computing[D]. Zhengzhou: Zhengzhou Institute of Light Industry, 2013.
[29] YI J, QIU M X. Design of user password authentication scheme based on ACSII code and random numbers[J]. Computer and Digital Engineering, 2011, 39(3): 102-104.

SHEN Yuancheng, born in 1998, postgraduate. His main research interests include data visualization and interactive data exploration and analysis system.



WANG Yunhai, born in 1984, Ph.D, professor, Ph.D supervisor, is a member of China Computer Federation. His main research interests include visual analysis of big data, human-computer interaction and computer graphics.