研究与开发

基于GPT网络的日志文本故障关联规则挖掘研究

李冰，叶庆卫，王雨晞，何镇秦，王晓东
（宁波大学信息科学与工程学院，浙江 宁波 315211）

摘　要：日志文本异常检测及故障关联规则挖掘是现代系统运维和故障诊断中的关键环节，对于提升系统可靠性和降低运维成本具有重要意义。针对传统人工监控方式难以应对系统复杂性和规模增长的问题，提出了一种基于深度学习的通信网日志信息关联规则挖掘方法。该方法首先从通信网日志信息中提取主机IP地址、时间和故障类型等关键信息，并采用多种经典关联规则算法进行融合，生成关联规则标签，构建深度学习网络的数据集及标签集。然后，引入逐层学习机制，设计了一种高效的深度学习模型，在构建的数据集上进行训练，获得专门用于挖掘通信网故障事件之间关联关系的深度学习网络模型。在GAIA大型日志数据集上的实验结果表明，新算法展现出显著优势：在准确率方面达到64.2%，较FP-Growth算法（22.1%）提升190.5%，较Eclat算法（43.7%）提升46.9%，较Quant Matrix Miner算法（37.6%）提升70.7%；在运行时间方面仅需要11 s，较FP-Growth算法提速138.2%，较图神经网络（graph neural network，GNN）提速195.0%，同时保持了与Apriori算法相近的处理速度。此外，新算法在支持度（0.42）和置信度（0.75）等关键指标上也表现出良好的均衡性。该方法为故障事件关联规则的智能挖掘提供了新的工具，具有一定的实用价值和应用前景。

关键词：日志；计算机网络；异常检测；关联规则挖掘；深度学习；GPT网络
中图分类号：TN915
文献标志码：A
doi: 10.11959/j.issn.1000−0801.2025207

Research on log text fault association rule mining based on GPT network

LI Bing, YE Qingwei, WANG Yuxi, HE Zhenqin, WANG Xiaodong
Faculty of Electrical Engineering and Computer Science, Ningbo University, Ningbo 315211, China

Abstract: Log anomaly detection and fault association rule mining are critical components in modern system operation and maintenance, as well as fault diagnosis, playing a significant role in enhancing system reliability and reducing operational costs. To address the challenges posed by the increasing complexity and scale of systems, which traditional manual monitoring methods struggle to handle, a deep learning-based method for mining association rules from

收稿日期：2025−03−24；**修回日期：**2025−05−12
通信作者：叶庆卫，personalfree@163.com
基金项目：浙江省产学研合作项目（No.062400020）；大型横向项目网络运维平台研发项目（No.HK2022000189）
Foundation Items: The Industry-University-Research Cooperation Project of Zhejiang Province (No. 062400020), The Large-Scale Horizontal Project Network Operation and Maintenance Platform Research and Development Project (No.HK2022000189)

communication network logs was proposed. The method first extracted key information such as host IP addresses, timestamps, and fault types from communication network logs. Subsequently, it integrated multiple classic association rule algorithms to generate association rule labels, constructing a dataset and label set for the deep learning network. Subsequently, a hierarchical learning mechanism was introduced, and an efficient deep learning model was designed to train on the constructed dataset, resulting in a specialized deep learning network model for mining association relationships among communication network fault events. The experimental outcomes derived from the comprehensive GAIA log dataset underscore the pronounced superiority of the proposed method. It achieved an accuracy of 64.2%, representing a considerable enhancement of 190.5% over the FP-Growth algorithm's performance of 22.1%, 46.9% over the Eclat algorithm's 43.7%, and 70.7% over Quant Matrix Miner's 37.6%. With respect to operational time, the method exhibited remarkable speed, requiring only 11 seconds for execution, thereby outpacing the FP-Growth algorithm by 138.2% and the graph neural network (GNN) by 195.0%, while maintaining a comparable processing velocity to the Apriori algorithm. Furthermore, the method demonstrated a well-balanced performance in key metrics including support (0.42) and confidence (0.75). This method provides a new tool for the intelligent mining of fault event association rules, offering practical value and promising application prospects.

Key words: log, computer network, anomaly detection, association rule mining, deep learning, GPT network

0　引言

日志文本异常检测及故障关联规则挖掘是现代系统运维和故障诊断中的关键环节。随着信息化和智能化的发展，系统的复杂性和规模不断增加，传统的人工监控方式已难以满足需求。对系统日志进行自动化的异常检测，可以及时发现潜在的问题和故障，避免系统崩溃和数据丢失[1-4]。故障关联规则挖掘则通过分析日志数据中的关联模式，帮助运维人员快速定位故障根源，提高故障处理的效率和准确性。这不仅提升了系统的可靠性和稳定性，还显著降低了运维成本，为企业的数字化转型提供了有力支持。

在技术演进方面，当前关于日志文本异常检测的研究呈现出从传统方法向深度学习迁移的明显趋势。早期基于决策树（decision tree，DT）和支持向量机（support vector machine，SVM）等机器学习的方法虽然实现了基本的异常检测功能，但在处理非结构化日志数据时准确率普遍较低[5]。深度学习技术的快速发展为这一领域带来了新的突破。例如，文献[6]运用改进的卷积神经网络，通过设计特定的卷积核来提取日志数据中的局部特征，在多个公开日志数据集上展现出较高的准确率。文献[7]则采用生成对抗网络（generative adversarial network，GAN）的变体，让生成器和判别器在日志数据特征学习中相互博弈，从而精准识别异常情况。此外，文献[8]使用图神经网络（graph neural network，GNN）对日志数据构建的图结构进行分析，挖掘日志事件间复杂的关联关系，以实现异常检测。然而，这些方法对于长距离依赖的捕捉能力有限，近年来，基于Transformer架构的预训练模型在日志分析领域取得突破性进展，特别是基于Transformer的双向编码器表示（bidirectional encoder representation from Transformer，BERT）、生成式预训练变换器（generative pre-trained Transformer，GPT）等模型通过自注意力机制，能够有效捕捉日志文本中的深层语义特征，从而显著提升了检测的准确性。例如，文献[9]借助基于Transformer架构的模型，利用其强大的自注意力机制来捕捉日志序列中的长距离依赖关系，在日志异常检测任务中取得了良好的效果。

当前的主流关联规则挖掘算法在应用于现代日志分析时面临多重挑战。以Apriori算法[10]为例，其指数级别的时间复杂度使其难以应对海量

日志数据的处理需求；FP-Growth算法[11]虽然通过前缀树结构优化了存储效率，却无法有效捕捉日志事件间的时序关系；符号聚合近似（symbolic aggregate approximation，SAX）方法[12]在时间序列符号化的过程中则会不可避免地丢失关键语义信息。上述这些方法普遍依赖支持度和置信度等静态指标，不仅难以区分相关性与因果性，而且对时序特征的忽视导致时间敏感型规则准确率下降。更严重的是，面对日志数据中的噪声干扰，传统算法会产生无效规则，既增加计算负担又可能引发误诊风险[13-14]。相比之下，基于Transformer的深度学习方法通过自注意力机制动态学习事件关联，通过多头注意力结构全面捕获语义特征，利用位置编码技术保留时序信息，在保持可解释性的同时，还可以保证挖掘的准确率。

本文提出了一种基于深度学习的通信网日志信息关联规则挖掘方法。该方法从通信网日志信息中提取主机IP地址、时间戳和故障类型等关键信息，采用多种经典关联规则算法挖掘结果的融合来生成关联规则标签，进而构建深度学习网络所需的数据集及对应的标签集。本文构建了高效的深度学习模型，引入逐层学习机制，在上述数据集的基础上进行训练，以获得专门用于挖掘通信网故障事件之间关联关系的深度学习网络模型。经过在GAIA大型日志文本数据集上的训练和对比测试，证实了本文提出的用于挖掘关联规则的深度学习网络模型能够有效提取故障事件关联规则，并且与经典挖掘算法相比更加快速和高效，为故障事件关联规则的挖掘提供了一种智能工具。

1 日志文本故障关联规则挖掘模型

日志文件作为系统运行状态的记录，包含了大量关于系统行为和潜在故障的重要信息[15]。为了有效地从这些海量日志数据中提取有价值的信息，系统首先需要识别出日志中的异常事件，这可以通过现有的日志文本异常检测方法实现。这些方法利用统计模型、机器学习或深度学习技术[16]，对不同日志数据进行模式识别和异常检测，从而标记出可能指示系统故障的事件。然而，仅仅检测出异常事件并不足以全面理解系统的故障机制和潜在风险。为了深入分析故障之间的内在联系，本节进一步探讨故障关联规则挖掘模型。该模型在检测出日志中的异常事件后，挖掘出两个或多个故障事件之间的关联规则，从而揭示故障之间的因果关系或共现模式。

1.1 日志文本异常检测

日志文本异常检测的核心任务在于从海量日志数据中精准识别出显著偏离正常行为模式的异常事件[17]。随着数据规模的不断扩大和数据复杂性的日益增加，日志数据的来源日益多样化，导致不同来源的日志在文本格式和描述方式上存在显著差异。这些挑战使得传统的基于规则和统计模型的方法难以有效应对日志数据的复杂性和动态变化特性[18]。为了克服这些局限，学者们开始转向深度学习方法，特别是基于Transformer的预训练语言模型（如BERT）[19]，以利用其强大的上下文建模能力，捕捉日志序列中的异常模式。

日志中包含的异常类型主要分为5类：文件/文件夹操作异常、网络异常、数据库异常、系统异常（硬件异常与系统异常合并）和其他异常，每一类异常又包含多个细分类别。由于硬件异常数据量较少，可能导致实验结果较差，而硬件异常可以视为系统异常的一种，因此，将硬件异常与系统异常合并为同一类别。本文对每个日志序列l_i都标注异常类型标签$c \in \{1,2,3,4,5\}$，分别对应上述5类异常。

日志文本异常检测算法有很多，如文献[20]通过设计3种正则表达式的方法（regex1、regex2、regex3），从一般的日志信息文本中提取出3个关键特征：时间戳、节点IP地址和节点状态描述信息。3种正则表达式的定义如下。

Regex regex1 = new Regex (“\d{4} – \d{2} – \d {2} \ d{2}:\d{2}:\d{2}”)

Regex regex2 = new Regex (“\b(:[0 – 9]{1, 3} \.){3}[0 – 9]{1, 3}\b”)

Regex regex3 = new Regex (“\[\d{4} – \d{2} – \d {2} \ d{2}:\d{2}:\d{2}\][A – Z]+ : (.+)”)

文献[21]利用 BERT 强大的语义理解能力和自监督学习机制，捕捉正常序列的模式，实现对异常日志的检测。设日志序列为$L=\{l_1,l_2,\cdots,l_n\}$，经过日志解析后，L 被转换成日志序列模板 $T=\{t_{m1},t_{m2},\cdots,t_{mk}\}$，其中，$t_{mi}$ 表示第 i 个被掩码的日志模板。在向量化处理过程中，每一个日志模板 t_{mi} 被转换成向量 $\boldsymbol{x}_{mi}$，最终得到向量序列 $\boldsymbol{X}$：

$$\boldsymbol{X}=\left[x_1,x_2,\cdots,x_n\right] \tag{1}$$

其中，$\boldsymbol{x}_j=\boldsymbol{e}_{\text{token}}+\boldsymbol{e}_{\text{pos}}+\boldsymbol{e}_{\text{vol}}$，表示第 j 个日志模板的最终向量，$\boldsymbol{e}_{\text{token}}$、$\boldsymbol{e}_{\text{pos}}$ 和 $\boldsymbol{e}_{\text{vol}}$ 分别为词汇嵌入、位置嵌入和频次嵌入向量输出，$\boldsymbol{x}_j\in R^{d_{\text{model}}}$，$d_{\text{model}}$ 表示 BERT 模型中所有嵌入向量所处的向量空间维度。

向量序列 $\boldsymbol{X}$ 输入 LogBERT 模型中，得到模型输出：

$$\left[\boldsymbol{P},\boldsymbol{H}_L\right]=F\left(\boldsymbol{X},Q\right) \tag{2}$$

其中，F 为 LogBERT 模型，Q 为网络的权重参数，$\boldsymbol{P}=\left[p\left(t_{m1}\right),p\left(t_{m2}\right),\cdots,p\left(t_{mk}\right)\right]$ 表示掩码处的预测结果，$p\left(t_{mi}\right)$ 表示第 i 个被掩码的日志模板 t_{mi} 的预测概率，$\boldsymbol{H}_L=\left[h_1,h_2,\cdots,h_n\right]$ 表示日志序列的向量表示。

对于每一个被掩码的日志模板 t_{mi}，计算器预测概率分布 $p\left(t_{mi}\right)$，构建候选集 C_i，包含预测概率最高的 g 个日志模板，异常日志判定规则为：

$$\begin{cases} t_{mi}\in C_i, & \text{正常} \\ t_{mi}\notin C_i, & \text{异常} \end{cases} \tag{3}$$

一旦日志被判定为异常日志，LogBERT 将从异常日志中提取时间戳、IP 地址和节点状态描述信息。

1.2　关联规则挖掘模型

在日志文本异常检测之后，为进一步揭示故障之间的内在联系，本文提出了一种故障关联规则挖掘模型。首先，定义以下概念。

故障事件：指在日志文本中检测到的异常或错误信息，可以用 $e=\left[e_{\text{time}},\text{IP},e_{\text{type}}\right]$ 三元组来表示，其中 e_{time} 表示故障发生的时间，IP 为故障发生的 IP 地址，e_{type} 为发生的故障类型，每个三元组代表一个故障事件。

时间窗口：指在特定的时间范围内对数据进行处理，它以时间点来定义开始时间和结束时间，截取某一段时间内的数据，设 w 为时间窗口，t 为时间戳，l 为窗口长度，则在 t 时刻的时间窗口 $w(t)=\left[t-l,t\right]$。

故障事件集：由一个或者多个故障事件组成的集合。在日志文本数据集中，每个时间窗口内发生的所有故障事件可以组成一个商品集 $E=\{e_i,i=t_0,\cdots,t_k\}$，其中，$t_0$ 代表时间戳，记为 $I=\left\{e_{k1},e_{k2},\cdots,e_{kp}\right\}$。

支持度：表示一个故障事件或故障事件集在所有的事务（即时间窗口）中所占的比例，计算式如式（4）所示。其中，N 表示包含故障事件 X 的数量，M 表示所有事务的数量。

$$\text{Support}\left(X\right)=\frac{N}{M} \tag{4}$$

置信度：表示在包含故障事件 X 的事务中，同时包含故障事件 Y 的比例，计算式为：

$$\text{Confidence}\left(X\rightarrow Y\right)=P\left(Y|X\right)=\frac{P\left(XY\right)}{P\left(X\right)} \tag{5}$$

提升度：表示包含故障事件 X 的事务同时包含故障事件 Y 的比例，与 Y 的总体出现概率的比值，计算式如式（6）所示。提升度用来衡量关联规则的强度，提升度大于 1，表示 X 和 Y 之间存在正相关。

$$\text{Lift}\left(X\rightarrow Y\right)=\frac{\text{Confidence}\left(X\rightarrow Y\right)}{\text{Support}\left(Y\right)} \tag{6}$$

故障关联规则：指两个或多个故障事件之间存在的某种规律性联系，可以表达为“如果发生故障事件A，那么在一段时间内发生故障事件B的概率较高”。

基于上述这些定义，故障关联规则挖掘模型的流程如图1所示，主要步骤如下。

步骤1 数据预处理。将日志数据按照时间划分成多个等长的量化时间间隔$Q=\{q_1,q_2,\cdots,q_n\}$，每个量化时间间隔q_i包含固定长度的时间段。定义一个时间窗口$T1$，其长度为$T1$个连续的量化时间间隔，$T1=\{q_i,q_{i+1},\cdots,q_{i+T1-1}\}$，在时间窗口$T1$内发生的故障事件集合记为事务$T_i$，即$T_i=\{e_{i1},e_{i2},\cdots,e_{im}\}$，其中$e_{im}$表示第$i$个量化时间内的第$m$个故障事件，通过滑动窗口对量化时间内的故障事件进行采样，形成多个交易集$D=\{D_1,D_2,\cdots,D_k\}$，每个交易集D_i包含了L个连续的事务，即$D_i=\{T_{i1},T_{i2},\cdots,T_{iL}\}$

步骤2 生成标签。对上述预处理之后的数据集D_i，采用3种经典的关联规则挖掘算法（Apriori算法、FP-Growth算法和Eclat算法）进行关联规则挖掘，得到关联规则$R_A^{(i)}$、$R_P^{(i)}$和$R_E^{(i)}$。随后，统计每一条关联规则跨算法出现的次数，选择出现次数最多的关联规则作为交易集D_i的最终标签集合。FP-Growth算法和Eclat算法均是在Apriori算法上演变而来的，因此，本文以Apriori算法为例，详细介绍故障关联规则的提取过程，具体步骤如下。

（1）对于每一个故障事件e_k，按照式（4）计算其支持度$\text{Support}(e_k)$，筛选出满足最小支持度阈值min _sup的故障事件，记为$L_1=\{e_k\,\text{Support}|(e_k)\geqslant \text{min_sup}\}$。

（2）生成所有可能的故障事件组合$I_2=\{e_{k1},e_{k2}\}$，计算每一个组合的支持度$\text{Support}(I_2)$，筛选出满足最小阈值的组合，记为$L_2=\{I_2|\,\text{Support}(I_2)\geqslant \text{min_ sup}\}$。

（3）重复上述过程，生成3个、4个等更大规模的项集I_3、I_4等，计算每个项集的支持度，筛选出满足最小支持度阈值的项集，直到无法生

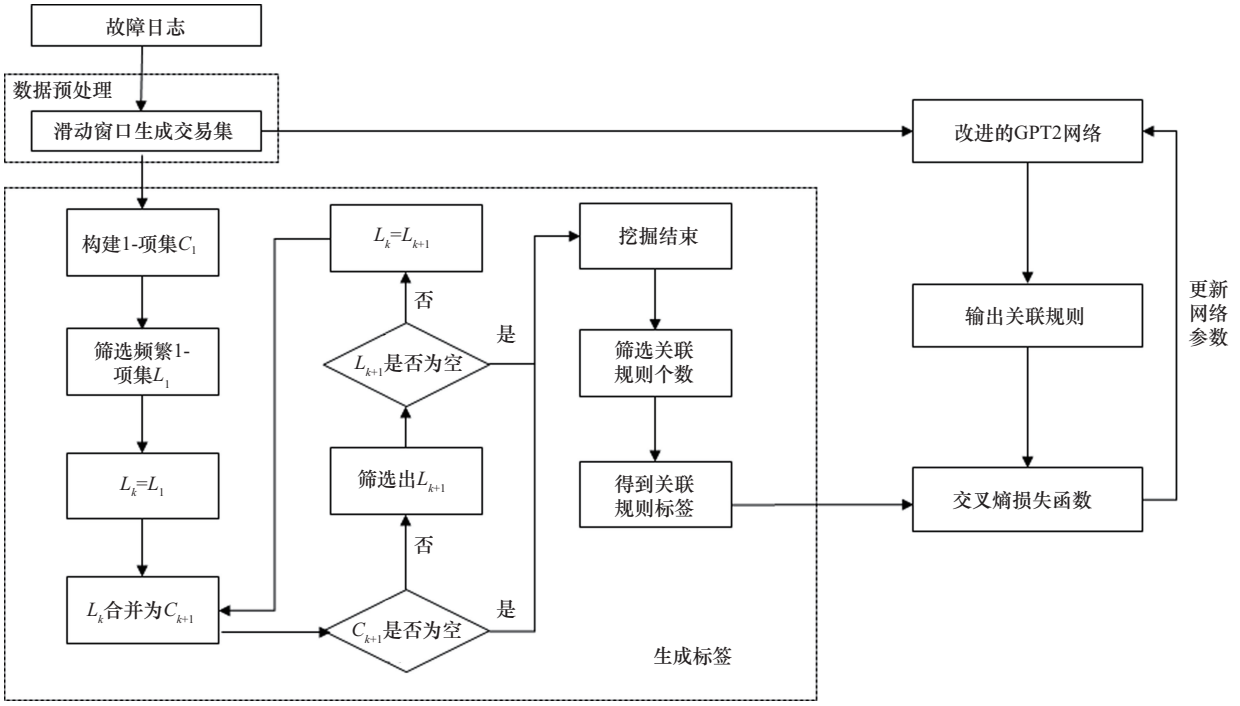


图1 故障关联规则挖掘模型的流程

成新的频繁项集为止。

（4）对于每一个频繁项集I，生成了所有可能的关联规则$X\to Y$，其中$X\in I, Y\in I, X\cap Y=0$，计算每个关联规则的置信度$\text{Confidence}(X\to Y)$和提升度$\text{Lift}(X\to Y)$，筛选出满足最小置信度阈值$\min_\text{conf}$和最小提升度阈值$\min_\text{lift}$的关联规则，将筛选后的关联规则按照置信度和提升度进行降序排序，并根据实际需求输出置信度与提升度最高的若干条关联规则，最后得到关联规则集$R=\{r_1, r_2, \cdots, r_m\}$。

步骤 3　网络搭建。经典的GPT-2模型由多层Transformer结构堆叠而成，其输出维度与输入维度保持一致。为了使其能够直接输出关联规则，本文对Transformer的输出部分进行了创新性调整。具体而言，对于序列中的特定节点，引入了一个全连接层，将该节点的输出映射到关联规则编码表上，从而得到每个关联规则编码ID的概率分布。通过这种方式，模型能够预测可能的关联规则，而其他节点则保持正常输出。此外，在GPT-2模型中引入了迭代结构，并设计了逐层学习损失函数，以实现逐层反向学习。这种设计使得模型能够逐步优化并输出多条关联规则，从而提升了模型在关联规则挖掘任务中的表现。

2　故障关联规则挖掘的GPT网络设计

在故障关联规则挖掘领域，传统方法主要依赖于统计分析和机器学习算法，在处理复杂的日志数据和捕捉深层次故障关联关系时存在一定局限性。随着自然语言处理技术的飞速发展，GPT凭借其强大的文本理解和生成能力，为故障关联规则挖掘提供了新的契机[17]。本文基于日志信息和深度学习技术，提出了一种利用GPT网络实现通信网故障关联规则挖掘的新方法：首先，收集通信网在运行过程中各个节点的日志信息，这些日志包含了大量关于系统行为和潜在故障的重要数据；其次，对日志信息进行预处理，提取出关键信息，如故障主机IP地址、故障时间和故障类型等，通过提取故障事件并构建故障事件字典，将这些事件转换为结构化的数据表示，并基于提取的故障事件和关联规则字典，生成相应的故障关联标签，同时，将数据集合理划分为训练集和测试集，以确保模型的泛化能力和评估准确性；最后，引入逐层学习机制，设计一种高效的深度学习模型，将经过处理的数据送入基于GPT的深度学习挖掘模型中进行训练，通过逐层学习，模型逐步学习和优化，从而实现对故障事件之间关联规则的深入挖掘。

2.1　故障事件字典

对原始日志信息中故障事件$e=\left[e_{\text{time}}, \text{IP}, e_{\text{type}}\right]$，按照时间$e_{\text{time}}$对故障进行处理，找到每一个时间节点$T_n$上发生的故障事件集合$E_{T_n}=\left\{e|e_{\text{time}}=T_n\right\}$。定义故障事件编码的格式为一个二元组，$I_m=\left[\text{host}, \text{type}\right]$，其中type为故障类型，host为主机IP地址。为了构建故障编码表，假设有n台主机和m种不同的故障类型，根据这些主机IP和故障类型的组合，一共生成了$n\times m$个唯一的编码。编码表的前几个位置预留给特殊符号{'<pad>': 0, '<unk>': 1, '<sep>': 2, '<no_error>': 3}，其中'<pad>'用于填充，'<unk>'表示未知或未定义的故障，'<sep>'用于分隔不同的故障事件，'<no_error>'表示无故障状态。从编码4开始，根据主机IP地址和故障类型的不同组合进行编码，如下所示：

$$\text{ID}=4+\left(\text{index of host}\right)\times m+\left(\text{index of type}\right)\quad(7)$$

其中，index of host 和index of type 分别表示主机IP地址和故障类型在各自集合中的索引。故障事件编码见表1，通过查询编码表，可以将特定的故障事件$\left(D_j, G_j\right)$映射到对应的编码ID，其中D_j为故障主机IP地址，G_j为故障类型，ID表示对D_j发生故障类型G_j的唯一编码。

表1 故障事件编码

主机IP，故障类型	编码ID
10.0.1.143,1	4
10.0.1.143,2	5
10.0.1.144,3	9
10.0.7.123,1	34
…	…

对于每一个故障编码I，它可能与除自己之外的其他$n\times m-1$个故障编码之间产生关联，此外，正常无故障事件也可以与其他故障编码产生关联，因此，关联关系编码表的总数量为$M=1+m\times n$，关联规则编码见表2。

表2 关联规则编码

关联规则	编码ID
4→5	1
5→4	2
4→6	3
6→4	4
…	…

2.2 日志文本预处理

本文在上述已经生成故障事件编码表的基础上构建故障事件集，具体步骤如下。

步骤1 时间窗口划分。日志数据在进行等长量化时间后，每个量化时间q_i内的日志数据为A_i，首先，将整个数据集按照固定步长s_1的滑动时间窗口进行采样，形成多个小的时间段T_1，记为$T_1=\{T_{1_1},T_{1_2},\cdots,T_{1_n}\}$，每一个$T_{1_j}$表示一个固定大小的滑动时间窗口，窗口大小为$w_1$，即每一个窗口包含了$w_1$个数据点。在每个小时间段内发生的故障事件被视为一个事务T_i。然后，以步长s_2和固定窗口大小w_2对T_1进行滑动，形成了多个大的时间段T_2，记为$T_2=\{T_{2_1},T_{2_2},\cdots,T_{2_m}\}$，在每一个大时间段$T_{2_i}$内，所有小时间段$T_1$的事务组合形成一条交易集$D_i=\{T_{i1},T_{i2},\cdots,T_{im}\}$，其中$T_{ij}$表示第$i$个大时间段内的第$j$个小时间段的事务。最后，所有大时间段内的交易集组合，形成了总的交易集$D=\{D_1,D_2,\cdots,D_n\}$。时间窗口划分过程如图2所示。

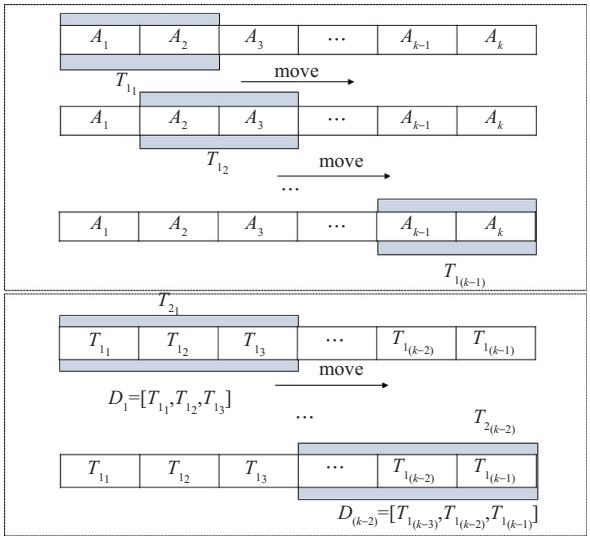


图2 时间窗口划分过程

步骤2 事务构建。对于每个大时间段T_2，首先提取该时间段内所有发生的故障事件，并依据2.1节中定义的故障事件编码，对于每个故障事件e_{ik}，将其逐一转换为对应的唯一ID编码，记为$\mathrm{ID}(e_{ik})$。在转换过程中，对于同一个小时间段T_1内的故障事件，使用逗号进行分隔。例如，某一个小时间段内包含故障事件e_{i1},e_{i2},e_{i3}，则转换后的事务表示为$T_i=\mathrm{ID}(e_{i1}),\mathrm{ID}(e_{i2}),\mathrm{ID}(e_{i3})$。对于不同大时间段$T_2$之间的事务，使用分隔符<sep>进行分隔，如某个大时间段T_{2i}内有T_i和T_{i+1}两个事务，则表示为$T_{2i}=T_i\|<\mathrm{sep}>\|T_{i+1}$。

为了确保提取到的关联规则具有足够的有效性和代表性，删除不满足故障事件编码数量大于30的事务，即对于每一个事务T_i，如果$|T_i|\leqslant 30$，则将该事务从交易集D中删除。这一筛选步骤旨在保证每个事务中包含足够多的故障事件，从而提高后续关联规则挖掘的准确性和可靠性。

2.3 标签设计

在完成上述日志文本预处理之后，每个数据

集D_i中都包含了w_2条事务T_i，每个事务T_i包含了w_1条日志数据A_i，鉴于选取时间跨度较短，假定这w_2条事务之间存在潜在的关联关系。在此基础上，运用经典的关联规则算法（Apriori 算法、FP-Growth 算法和 Eclat 算法）对每一个数据集D_i进行关联规则挖掘分析和融合，具体步骤如下。

步骤 1　频繁项集生成。在定义了最小支持度阈值 min_sup 和最小置信度阈值 min_conf 之后，对于每一个交易集D_i，用 3 种关联规则算法分别生成频繁项集集合，Apriori 算法、FP-Growth 算法和 Eclat 算法生成的频繁项集集合分别表示为：

$$F_{\mathrm{A}}^{(i)}=\left\{f_{\mathrm{A}_k}^{(i)}|\mathrm{Support}\left(f_{\mathrm{A}_k}^{(i)}\right)\geqslant\min\_\mathrm{sup}\right\} \tag{8}$$

$$F_{\mathrm{P}}^{(i)}=\left\{f_{\mathrm{P}_k}^{(i)}|\mathrm{Support}\left(f_{\mathrm{P}_k}^{(i)}\right)\geqslant\min\_\mathrm{sup}\right\} \tag{9}$$

$$F_{\mathrm{E}}^{(i)}=\left\{f_{\mathrm{E}_k}^{(i)}|\mathrm{Support}\left(f_{\mathrm{E}_k}^{(i)}\right)\geqslant\min\_\mathrm{sup}\right\} \tag{10}$$

步骤 2　关联规则提取。从每个算法生成的频繁项集中，对于每个频繁项集I，生成所有可能的关联规则$X\to Y$，其中$X\in I, Y\in I, X\cap Y=0$，计算每个关联规则的置信度 Confidence $(X\to Y)$和提升度 Lift $(X\to Y)$，筛选出满足最小置信度阈值 min_conf 和最小提升度阈值 min_lift 的关联规则，将筛选后的关联规则按照置信度和提升度大小进行降序排列，选择输出的关联规则数量，分别得到 3 种算法挖掘出的关联规则。Apriori 算法、FP-Growth 算法和 Eclat 算法输出的关联规则分别表示为：

$$R_{\mathrm{A}}^{(i)}=\left\{r_{\mathrm{A}_k}^{(i)}:\ X\to Y|\mathrm{Confidence}\left(r_{\mathrm{A}_k}^{(i)}\right)\geqslant\min\_\mathrm{conf},\mathrm{Lift}\left(r_{\mathrm{A}_k}^{(i)}\right)\geqslant\min\_\mathrm{lift}\right\} \tag{11}$$

$$R_{\mathrm{P}}^{(i)}=\left\{r_{\mathrm{P}_k}^{(i)}:\ X\to Y|\mathrm{Confidence}\left(r_{\mathrm{P}_k}^{(i)}\right)\geqslant\min\_\mathrm{conf},\mathrm{Lift}\left(r_{\mathrm{P}_k}^{(i)}\right)\geqslant\min\_\mathrm{lift}\right\} \tag{12}$$

$$R_{\mathrm{E}}^{(i)}=\left\{r_{\mathrm{E}_k}^{(i)}:\ X\to Y|\mathrm{Confidence}\left(r_{\mathrm{E}_k}^{(i)}\right)\geqslant\min\_\mathrm{conf},\mathrm{Lift}\left(r_{\mathrm{E}_k}^{(i)}\right)\geqslant\min\_\mathrm{lift}\right\} \tag{13}$$

步骤 3　标签融合。分别从 3 种算法的关联规则集合中提取出标签候选集，如下所示：

$$L_{\mathrm{A}}^{(i)}=\left\{l_{\mathrm{A}_k}^{(i)}|l_{\mathrm{A}_k}^{(i)}\in R_{\mathrm{A}}^{(i)}\right\} \tag{14}$$

$$L_{\mathrm{P}}^{(i)}=\left\{l_{\mathrm{P}_k}^{(i)}|l_{\mathrm{P}_k}^{(i)}\in R_{\mathrm{P}}^{(i)}\right\} \tag{15}$$

$$L_{\mathrm{E}}^{(i)}=\left\{l_{\mathrm{E}_k}^{(i)}|l_{\mathrm{E}_k}^{(i)}\in R_{\mathrm{E}}^{(i)}\right\} \tag{16}$$

统计 3 组标签中各规则跨算法出现的次数，计算式为：

$$\mathrm{Count}\left(l_k\right)=\sum_{M\in\{\mathrm{A,P,E}\}}I\left(l_k\in L_M^{(i)}\right) \tag{17}$$

其中，$I(\cdot)$为指示函数，当l_k存在于标签集合时，取值为 1，否则为 0。然后，选择出现次数最多的关联规则作为交易集D_i的最终标签集合，表示为式（18）。若存在多个规则出现次数相同，则优先选择支持度和置信度乘积最大的规则。

$$L_{\mathrm{final}}^{(i)}=\mathrm{argmax}\ \mathrm{Count}\left(l_k\right) \tag{18}$$

2.4　深度学习网络结构

经典的 GPT-2 网络是基于 Transformer 中的解码器部分构建的，主要是由嵌入层、位置编码层、3 层解码层和归一化层组成。经典 GPT-2 网络架构如图 3 所示。

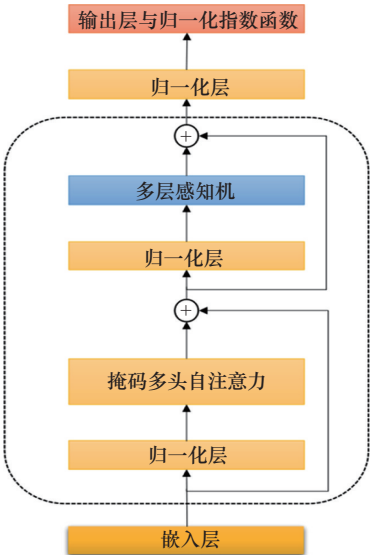


图 3　经典 GPT-2 网络架构

改进后的GPT子网架构在上述模型的基础上，引入迭代机制，增强了模型的表达能力。对于序列中的特定节点（如最后1个或最后2个节点），新增全连接映射层，将其映射到关联规则编码表的字典空间，生成关联规则编码ID的概率分布，用于输出关联规则编码ID，其他节点仍通过线性层映射到词汇表，生成常规单词的概率分布。最终，模型将常规词汇预测结果与关联规则编码ID预测结果拼接融合，既保留了GPT-2的文本生成能力，又提升了对复杂关联关系的建模能力。本文通过串联两个子网架构，构建了完整的深度学习网络，用于关联规则挖掘的GPT改进网络结构如图4所示，其中，FC表示全连接层。

由图3、图4和前文介绍可知，在第1个子网Net_1中，输入数据是经过预处理得到的整数编码$\boldsymbol{X}=\left[\mathrm{ID}_{ij}\right]_{n\times m}$，其中$n$表示数据集$D$中交易集$D_i$的数量，$m$表示交易集$D_i$中包含的故障事件的数量，$\mathrm{ID}_{ij}$表示矩阵中第$i$个数据集$D_i$中第$j$个故障事件的编码ID。$\boldsymbol{X}$包含了不同时间内出现的故障事件编码ID，首先经过嵌入层和位置编码层转化为向量表示$\boldsymbol{X}_{1\mathrm{emb}}$，$\boldsymbol{X}_{1\mathrm{emb}}$随后被输入包含12层的解码层模块中进行处理，每一层依次执行以下操作：归一化处理、多头自注意力机制计算、残差连接与二次归一化、前馈神经网络运算以及最终的残差连接与归一化操作。处理后的输出$\boldsymbol{X}_{1\mathrm{trans}}$通过线性层生成输出预测文本结果$\boldsymbol{Y}_{1\mathrm{text}}$。接着将$\boldsymbol{Y}_{1\mathrm{text}}$分别通过$M$个全连接层，每一层对应计算机网络中的一个节点，每层输出由ReLU函数激活，得到M个输出；对于前$M-1$个节点的输出结果直接经过一个softmax分类器，得到包含各类的概率向量，经过argmax操作找到最大概率对应的类别索引，即预测输出结果$\boldsymbol{X}_1'=\left[\mathrm{ID}_{ij}'\right]_{n\times(m-1)}$。对于最后一个节点，通过一个全连接映射层，将其映射到关联规则编码表上，然后得到关于关联规则编码表上各个关联规则的概率向量，经过argmax操作找到最大概率对应的关联规则类别索引，即预测输出的第1条故障关联规则$\hat{\boldsymbol{y}}_1'=\left[\mathrm{ID}_{im}'\right]_{n\times1}$。最后，将预测输出的$M$个节点的内容进行拼接，得到第1个子网的预测输出$\boldsymbol{Y}_1=\left[\boldsymbol{X}_1'\hat{\boldsymbol{y}}_1'\right]$。

将$\boldsymbol{Y}_1$作为第2个子网Net_2的输入，经过和第1个子网Net_1相同的处理，得到预测文本结果$\boldsymbol{Y}_{2\mathrm{text}}$。将$\boldsymbol{Y}_{1\mathrm{text}}$通过$M$个全连接层，对于前$M-2$个节点，输出结果经过softmax分类器得到包含各类的概率向量，经过argmax操作找到最大概率对应的类别索引，即预测输出结果$\boldsymbol{X}_2'=\left[\mathrm{ID}_{ij}''\right]_{n\times(m-2)}$。对于最后2个节点，通过一个全连接映射层，将其映射到关联规则编码表上，然后经过argmax操作找到最大概率对应的关联规则类别索引，即预测输出的2条故障关联规则$\hat{\boldsymbol{y}}_2'=\left[\mathrm{ID}_{i(m-1)}'',\mathrm{ID}_{im}''\right]_{n\times2}$。最

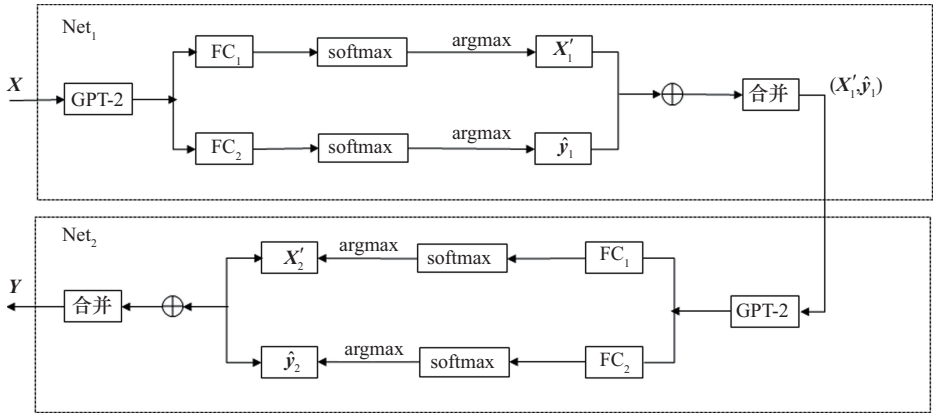


图4　用于关联规则挖掘的GPT改进网络结构

后，将预测输出的M个节点的内容进行拼接，得到第2个子网的预测输出$\boldsymbol{Y}_2=\left[\boldsymbol{X}_2'\hat{\boldsymbol{y}}_2'\right]$。

2.5　损失函数设计

2个子网的损失函数都采用交叉熵损失函数，并且每个子网的损失函数都由两部分组成：一部分为正常输出的前$M-i(i=1,2)$个节点和标签X_i之间的损失函数，如式（19）、式（22）所示，另一部分是经过映射之后i个节点和标签$\hat{\boldsymbol{y}}_i$之间的损失函数，如式（20）、式（23）所示，两部分相加得到子网的损失函数，如式（21）、式（24）所示，定义分别如下。

$$\mathrm{Loss}_{1,1}=\mathrm{CrossEntropy}\left(\boldsymbol{X}_1',\boldsymbol{X}_1\right) \tag{19}$$

$$\mathrm{Loss}_{1,2}=\mathrm{CrossEntropy}\left(\hat{\boldsymbol{y}}_1',\hat{\boldsymbol{y}}_1\right) \tag{20}$$

$$\mathrm{Loss}_1=\mathrm{Loss}_{1,1}+\mathrm{Loss}_{1,2} \tag{21}$$

$$\mathrm{Loss}_{2,1}=\mathrm{CrossEntropy}\left(\boldsymbol{X}_2',\boldsymbol{X}_2\right) \tag{22}$$

$$\mathrm{Loss}_{2,2}=\mathrm{CrossEntropy}\left(\hat{\boldsymbol{y}}_1',\hat{\boldsymbol{y}}_2\right) \tag{23}$$

$$\mathrm{Loss}_2=\mathrm{Loss}_{2,1}+\mathrm{Loss}_{2,2} \tag{24}$$

其中，$\boldsymbol{X}_1'$为前$n-1$个节点的映射输出，$\hat{\boldsymbol{y}}_1'$为最后一个节点映射到关联规则编码表字典上的预测输出的关联规则编码ID，$\boldsymbol{X}_1=\boldsymbol{X}[:,2{:}\mathrm{m}]$，$\hat{\boldsymbol{y}}_1$为真实的标签中的一个编码ID。$\boldsymbol{X}_2'$为前$n-2$个节点的映射输出，$\hat{\boldsymbol{y}}_2'$为最后2个节点映射到关联规则编码表字典上的预测输出的关联规则编码ID，$\boldsymbol{X}_2=\boldsymbol{X}[:,3{:}\mathrm{m}]$，$\hat{\boldsymbol{y}}_2$为真实的标签中的两个编码ID。

整个网络的损失函数Loss为第1个子网的损失函数Loss_1和第2个子网的损失函数Loss_2的加权和，如式（25）所示。两个子网的结构相同，但是第2个子网的输入数据是第1个子网的预测输出，因此，第1个子网对于整个网络的影响是最大的，获得较大的权重值。

$$\mathrm{Loss}=0.6\mathrm{Loss}_1+0.4\mathrm{Loss}_2 \tag{25}$$

2.6　深度学习网络训练流程

深度学习网络训练流程如图5所示，主要步骤如下。

步骤1　将样本数据$\boldsymbol{X}$输入GPT-2模型，通过模型得到输出结果$\boldsymbol{Y}_1=\left[\boldsymbol{X}_1'\hat{\boldsymbol{y}}_1'\right]$。

步骤2　计算$\boldsymbol{X}_1'$和标签$\boldsymbol{X}_1$之间的损失函数、$\hat{\boldsymbol{y}}_1'$和标签$\hat{\boldsymbol{y}}_1$之间的损失函数，二者相加得到第1个子网Net_1的损失函数Loss_1。

步骤3　将步骤1输出的$\boldsymbol{Y}_1=\left(\boldsymbol{X}_1',\boldsymbol{y}_1'\right)$作为输入数据，输入第2个子网$\mathrm{Net}_2$中，得到网络输出为$\boldsymbol{Y}_2=\left[\boldsymbol{X}_2'\hat{\boldsymbol{y}}_2'\right]$。

步骤4　计算$\boldsymbol{X}_2'$和标签$\boldsymbol{X}_2$之间的损失函数、$\hat{\boldsymbol{y}}_2'$和标签$\hat{\boldsymbol{y}}_2$之间的损失函数，二者相加得到第2个子网Net_2的损失函数Loss_2。

步骤5　输出完毕，得到2条频繁关联规则。将Loss_1和Loss_2代入式（25）得到该网络的整体

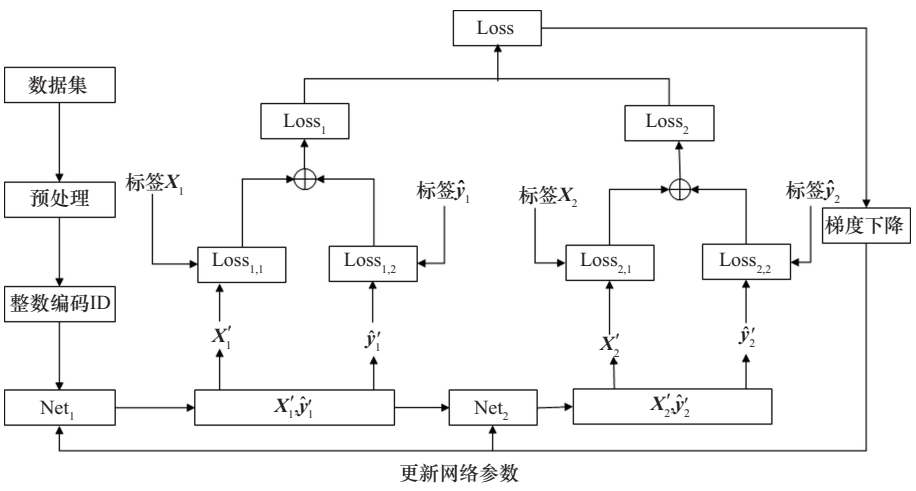


图5　深度学习网络训练流程

的损失函数Loss，随后进行梯度下降，更新Net_1和Net_2的网络参数。

对于需要$m(m>2)$条关联规则的情况，在标签生成时需要选择输出m条关联规则，得到m个标签$L=\{L_1,L_2,\cdots,L_m\}$，其中每个标签$L_i=(\boldsymbol{X}_i,\hat{\boldsymbol{y}}_i)$。接下来，构建$m$个子网$\mathrm{Net}_1,\mathrm{Net}_2,\cdots,\mathrm{Net}_m$，每个子网负责处理不同长度的输出，对于第$i$个子网$\mathrm{Net}_i$，输出为$\boldsymbol{Y}_i=(\boldsymbol{X}_i',\boldsymbol{y}_i')$，$\boldsymbol{X}_i'$为前$n-i$个节点映射输出，$\boldsymbol{y}_i'$为最后$i$个节点映射到关联规则编码表上预测输出$i$个关联规则编码ID。对于每个子网$\mathrm{Net}_i$，分别计算$\boldsymbol{X}_i'$和标签$\boldsymbol{X}_i$之间的损失函数以及$\hat{\boldsymbol{y}}_i'$和标签$\hat{\boldsymbol{y}}_i$之间的损失函数，二者相加得到子网$\mathrm{Net}_i$的损失函数$\mathrm{Loss}_i$，然后将所有子网的损失函数$\mathrm{Loss}_i$代入整体损失函数Loss中（如式（25）所示），随后进行梯度下降，更新所有子网Net_i的网络参数。

3 故障关联规则挖掘模型实现与测试

3.1 数据集介绍及预处理

GAIA数据集源自云智慧AIOps社区，是针对运维领域异常检测、日志解析及故障根因定位等任务构建的综合性基准数据集。该数据集基于云智慧自主研发的业务模拟系统MicroSS生成，聚焦于小型通信网络场景的典型运维需求。MicroSS系统通过全方位监控业务实体运行状态，完整采集了包括业务执行日志、实体关联指标及对象特征描述信息在内的全维度运维数据。GAIA数据集涵盖日志结构化解析、语义级异常检测和命名实体识别3类任务数据，总计包含218 736条日志记录。其中，语义异常检测数据部分整合了多IP主机产生的信息（info）与错误（error）两级日志元数据及状态特征。info级日志均表征正常业务行为，error级日志包含3类数字标记（1～3）的故障状态，分别对应网络连接异常、数据库访问异常及其他类型异常事件，为算法模型训练提供了细粒度标注支持。

GAIA数据集中收集了来自13个不同节点的日志信息及其对应的状态，为了便于处理，首先对所有节点的日志记录进行时间离散化，即将连续的时间轴划分为一系列离散的时点。随后，对这些离散时间点进行去重处理，最终得到1 414个独立的离散时间节点，以及每个时间节点对应的节点状态。

接下来，将上述1 414个独立离散时间节点划分为多个大时间段T_2，每个大时间段由w_2个小时间段T_1组成。在每个大时间段T_{2_i}内，提取所有发生故障事件，并根据2.1节定义的故障事件编码，将这些故障事件逐一转换为对应的唯一ID编码。最后，将处理后的数据集D按照8∶2的比例划分为训练集和测试集，用于后续模型的训练和评估。

3.2 训练结果

在参数设置中，嵌入层维度为768，前馈层维度为2 048，查询（Q）、键（K）和值（V）的维度均为64，多头自注意力机制的头数为16，解码层数为6。基于上述配置，本文利用生成的训练集数据对模型进行训练。训练过程中，训练集损失函数变化曲线、训练集准确率变化曲线和测试集准确率变化曲线分别如图6～图8所示。从图6可以看出，随训练轮次的变化，第1个子网和第2个子网的损失均呈现明显的下降趋势，表明模型在逐渐学习并优化。图7和图8显示了模型的平均准确率随训练轮次的提升，准确率从初始的较低值逐渐上升并趋于稳定，最后为95%，这表明模型的性能在训练过程中得到了显著提高。总体来看，模型训练过程稳定，损失减少且准确率提高，显示出良好的学习能力和泛化能力。

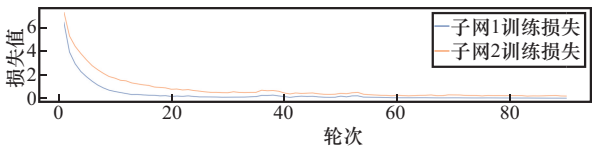


图6　训练集损失函数变化曲线

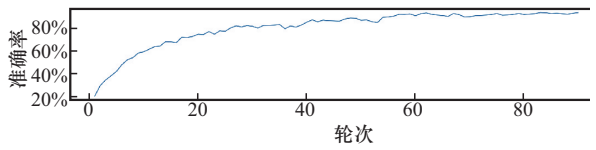


图7　训练集准确率变化曲线

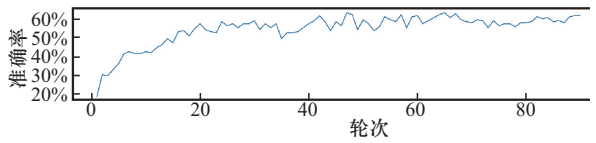


图8　测试集准确率变化曲线

3.3　评价指标

在评估关联规则挖掘算法的性能时，4 个关键指标被广泛采纳应用：运行时间、准确率、支持度和置信度[22]。由于本文所提算法输出的是多条关联规则，因此采用的是平均支持度和平均置信度。准确率衡量的是正确挖掘出来的关联规则占总数的比例，是衡量算法的直接指标，计算式如式（26）所示；运行时间用来衡量算法处理数据集所需要的时间，反映了算法的效率，计算式如式（27）所示；平均支持度表示在所有的关联规则中，前项和后项同时出现的平均概率，计算式如式（28）所示；平均置信度表示在所有的关联规则中，在前项出现的情况下，后项出现的平均概率，计算式如式（29）所示：

$$\text{Accuracy}=\frac{\text{TP}}{P}\times 100\% \tag{26}$$

$$T=t_{\text{end}}-t_{\text{start}} \tag{27}$$

$$\overline{\text{Support}}=\frac{1}{n}\sum_{i}^{n}\text{Support}_i \tag{28}$$

$$\overline{\text{Confidence}}=\frac{1}{n}\sum_{i}^{n}\text{Confidence}_i \tag{29}$$

其中，TP 表示真正例（即正确预测的关联规则），P 表示预测出的总的关联规则数量，t_{end} 表示程序结束的时间，t_{start} 表示程序开始的时间，n 为关联规则的数量，Support_i 表示第 i 个关联规则的支持度，Confidence_i 表示第 i 个关联规则的置信度。

3.4　测试结果与对比

3.4.1　采样步长 s_1 对关联规则挖掘的影响

本文在上述预处理后的数据集上进行实验，探讨采样步长 s_1 对关联规则挖掘的影响。本文考虑了采样步长 12、15、18，数据集在不同的采样步长 s_1 下测试集的各项指标对比见表 3。从表 3 可以看出，随着采样步长 s_1 的增加，模型的准确率提高明显，但是对平均支持度和平均置信度的影响较小，且在 s_1 从 15 增加到 18 时，关联规则的平均支持度出现了下降。综合来看，采样步长 s_1 设置为 18 时，整个网络的关联规则的挖掘效果更好，其中准确率达到了 64.2%，平均支持度为 0.42，平均置信度为 0.75。

表3　不同的采样步长 s_1 下测试集的各项指标对比

步长 s_1	准确率	平均支持度	平均置信度
12	58.1%	0.42	0.74
15	63.3%	0.43	0.75
18	64.2%	0.42	0.75

为了验证上述采样步长 s_1 对关联规则挖掘结果影响的猜想，本文引入了子网 Net_1、Apriori 算法、GNN 和 Quant Matrix Miner 进行论证实验。

不同算法在不同的采样步长 s_1 下的各项指标对比如图 9 所示。由图 9 可知，随着采样步长 s_1 的增加，关联规则挖掘的准确率大部分呈增加趋势，平均支持度和置信度变化不是特别明显。采样步长 s_1 从 12 到 15，测试集的挖掘效果提升明显，采样步长 s_1 从 15 到 18，测试集的挖掘效果提升较小。Quant Matrix Mine 在采样步长 s_1 为 15 时取得了较好的准确率，在采样步长 s_1 为 18 时取得了较大的置信度，而 Net_1、Apriori 算法和

GNN 都在采样步长 s_1 为 18 时取得了较好的准确率、支持度和置信度。综合 3 种评价指标，图 9 中所有的网络算法都在采样步长 s_1 时取得了更好的挖掘效果，这也验证了上述猜想，因此，选择在采样步长 s_1 为 18 作为本文的采样步长，并在此基础上进行后续实验。

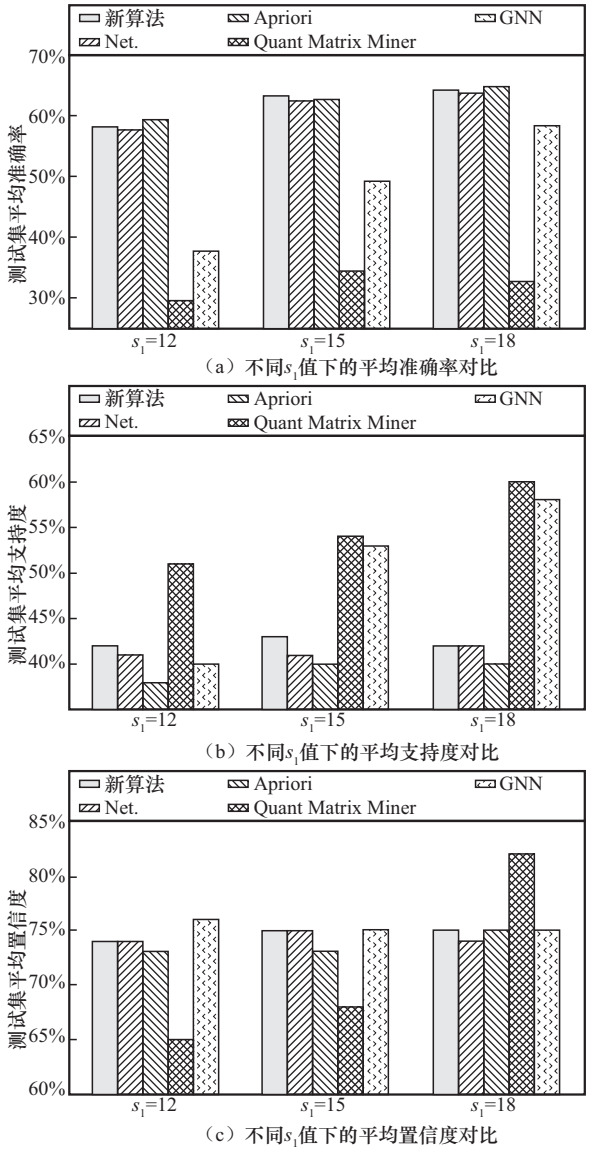


图 9　不同算法在不同的采样步长 s_1 下的各项指标对比

3.4.2　本文算法的有效性评估

本文在对预处理后的数据集进行深入分析的基础上，利用所构建的网络模型进行了训练，并成功实现了模型的收敛。训练完成后，调用了训练好的模型对测试集数据进行关联规则的挖掘。测试集共包含 193 条数据样本，每个样本的长度均为 4 614 个字符，为了验证本文提出的算法中迭代结构的有效性和可行性，将其和 Net_1，以及几个关联规则挖掘算法（包括 Apriori 算法、FP-Growth 算法、Eclat 算法、GNN 和 Quant Matrix Miner 算法）进行了对比实验。不同算法在测试集上的各项指标对比见表 4。

表 4　不同算法在测试集上的各项指标对比

算法	准确率	时延/s	支持度	置信度
新算法	64.2%	11	0.42	0.75
Net_1	63.7%	8	0.42	0.73
Apriori	64.8%	12.99	0.40	0.75
FP-Growth	22.1%	26.20	0.37	0.72
Eclat	43.7%	6.29	0.56	0.82
GNN	58.3%	32.44	0.58	0.75
Quant Matrix Miner	37.6%	8.12	0.60	0.82

此外，为了全面评估不同算法在数据规模和事务复杂度变化时的性能，本文还进行了两项对比实验。首先，对本文提出的算法与 Apriori、Eclat 算法、GNN 和 Quant Matrix Miner 算法在每一条样本长度固定、样本数量变化的情况下的运行时间进行了对比分析，该分析旨在评估算法在处理不同规模数据集时的效率差异。数据条数增加时算法运行时间对比如图 10 所示。其次，修改滑动时间窗口的采样步长 s_1，增加事务 T_i 的复杂度，在样本数量相同的情况下进行对比分析，该分析旨在揭示事务内部结构变化对算法运行效率的影响。事务项集增加时算法运行时间对比如图 11 所示。

由表 4 可知，新算法在多个关键性能指标上展现出了优异的均衡性。具体来说，新算法的准确率达到了 64.2%，与表现最佳的 Apriori 算法的 64.8% 相比，差距仅为 0.6%，几乎可以忽略不计。同时，在运行时间方面，新算法取得了 11 s

的成绩，相较于 Apriori 算法的 12.9 s，提升了 14.7%，显示出更高的效率。此外，新算法在支持度和置信度方面也维持了良好的水平。相比之下，Quant Matrix Miner 算法虽然在规则质量指标上表现突出，支持度达到 0.60，置信度为 0.82，但其准确率仅为 37.6%，显著偏低。这是由于 Quant Matrix Miner 算法过度依赖频繁项集挖掘，而忽略了日志数据中的全局特征关联，导致在整体性能上表现不佳。FP-Growth 算法表现则更为不佳，其准确率仅为 22.1%，时延却高达 26.20 s，这是因为需要多次扫描数据集，导致计算效率偏低。GNN 算法的准确率为 58.3%，时延为 32.44 s，虽然在准确率上表现尚可，但是时间成本太高，这是因为 GNN 需要学习节点的嵌入表示，涉及多层的图卷积操作，计算量比较大。总体而言，新算法在准确率、运行效率和规则质量 3 个关键维度上实现了最佳平衡。

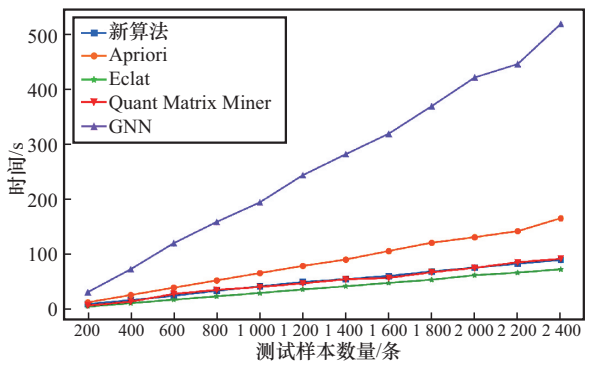


图 10　数据条数增加时算法运行时间对比

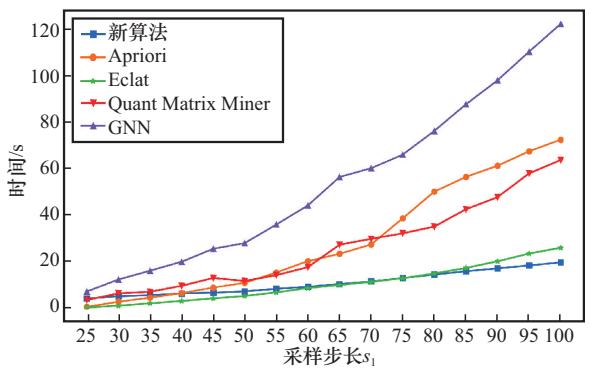


图 11　事务项集增加时算法运行时间对比

进一步地，本文通过对图 10 和图 11 的性能对比分析，可以清晰地观察到不同算法在数据规模扩展时的运行效率差异。GNN 因其复杂的图结构构建和节点迭代更新机制，运行时间显著高于其他算法，且在大规模数据集上呈现明显的非线性增长趋势。相比之下，Apriori 算法虽然在大数据处理上相对高效，但其运行时间随事务项集增加而快速上升，这主要源于频繁项集生成和剪枝过程的计算复杂度提升。Eclat 算法和 Quant Matrix Miner 算法在中小规模数据集上展现出优异的效率优势，这得益于其优化的频繁项集挖掘策略有效减少了冗余。值得关注的是，新算法在不同数据规模下均表现出卓越的性能稳定性，其运行时间随数据规模的增长保持线性上升趋势，既避免了 GNN 的指数级复杂度，又显著优于 Apriori 算法在项集增加时的性能衰减。这一结果表明，新算法通过创新的计算架构设计，成功实现了处理效率与规模扩展性的最佳平衡，为大规模日志数据分析提供了可靠的技术方案。

综上所述，通过引入迭代结构，本文所提出的算法在保证较高的准确率、支持度和置信度的同时，显著缩短了在大数据上的运行时间，相较于经典的挖掘算法，新算法有效提升了关联规则的挖掘性能。

4　结束语

本文提出的改进型 GPT 子网架构在保持经典 GPT-2 模型多层 Transformer 解码器的基础上，通过对输出部分进行创新性调整，成功提高了模型的文本生成能力和对复杂关联关系的表达能力。实验结果表明，该方法能够有效捕捉序列中的关键节点，并通过引入额外的全连接映射层来增强对这些节点的理解和处理能力，此外，通过与另一个子网架构串联形成深度学习网络，进一步提升了整体性能。因此，这种改

进型的GPT子网架构在实际应用中具有较高的实用价值和潜力。

然而，尽管本文所提出的算法在多项指标上表现良好，但仍有进一步优化的空间，以应对更加复杂和多样化的数据挖掘需求。未来的研究将聚焦于提升算法的支持度和置信度，以期实现算法性能的全面优化，从而更好地满足实际应用的需求。

参考文献：

[1] 樊志强, 刘日昇. 网络设备智能化管理的研究与应用[J]. 物联网技术, 2023, 13(8): 51-55.
FAN Z Q, LIU R S. Research and application of intelligent management of network equipment[J]. Internet of Things Technologies, 2023, 13(8): 51-55.

[2] GIRACCA D M, PIRES F L, BARÁN B, et al. A deep learning approach for anomaly detection for industrial control systems[C]// Proceedings of the 2024 L Latin American Computer Conference (CLEI). Piscataway: IEEE Press, 2024: 1-10.

[3] YANG J, SHENG Y Q, WANG J L, et al. CAGCN: centrality-aware graph convolution network for anomaly detection in industrial control systems[J]. Journal of Computer Science and Technology, 2024, 39(4): 967-983.

[4] CHOE H O, LEE M H. Artificial intelligence-based fault diagnosis and prediction for smart farm information and communication technology equipment[J]. Agriculture, 2023, 13(11): 2124.

[5] SHETH V, TRIPATHI U, SHARMA A. A comparative analysis of machine learning algorithms for classification purpose[J]. Procedia Computer Science, 2022, 215: 422-431.

[6] WANG K L. Attention-based BiGRU-CNN for system log anomaly detection[C]//Proceedings of the 2024 4th International Symposium on Computer Technology and Information Science (ISCTIS). Piscataway: IEEE Press, 2024: 151-156.

[7] DONG Y, ZHANG Z W, PENG H Y, et al. New perspective on progressive GANs distillation for one-class anomaly detection[J]. Journal of Imaging Science and Technology, 2023, 67(6): 1-11.

[8] LI Z, SHI J Y, VAN LEEUWEN M. Graph neural networks based log anomaly detection and explanation[C]//Proceedings of the 2024 IEEE/ACM 46th International Conference on Software Engineering: Companion Proceedings. New York: ACM, 2024: 306-307.

[9] LOU Z Y, CHAI X L, SHANG C, et al. Generalizable and robust log anomaly detection based on Transformer[C]//Wireless Sensor Networks. Singapore: Springer, 2025: 168-177.

[10] WANG C M, FU A Q, LI W D, et al. Intelligent identification of hidden dangers in hydrogen pipeline transmission station using GWO-optimized Apriori algorithm[J]. Energies, 2024, 17(18): 4539.

[11] XIAO Y, MENG L H, ZHANG Y R, et al. An improved FP-Growth algorithm with time decay factor and element attention weight[C]//Proceedings of the 2024 IEEE 4th International Conference on Power, Electronics and Computer Applications (ICPECA). Piscataway: IEEE Press, 2024: 860-865.

[12] PAPPA L, KARVELIS P, STYLIOS C. Exploring the diverse world of SAX-based methodologies[J]. Data Mining and Knowledge Discovery, 2024, 39(1): 4.

[13] WANG S Z, JIANG R X, WANG Z Q, et al. Deep learning-based anomaly detection and log analysis for computer networks[J]. arXiv preprint, 2024: 2407.05639.

[14] JIN G, JING C, ZONG T Q, et al. Data redundancy elimination and noise processing via large language model prompt engineering[C]//Proceedings of the 2024 10th International Conference on Big Data and Information Analytics (BigDIA). Piscataway: IEEE Press, 2024: 818-825.

[15] MARJAI P, KISS A. The usage of template mining in log file classification[J]. IEEE Access, 2024, 12: 96378-96386.

[16] DU X Y, XU C, LI L, et al. Multigranularity feature automatic marking-based deep learning for anomaly detection of industrial control systems[J]. IEEE Open Journal of Instrumentation and Measurement, 2024, 3: 2500110.

[17] CHAI X L, ZHANG H, ZHANG J, et al. Log sequence anomaly detection based on template and parameter parsing via BERT[J]. IEEE Transactions on Dependable and Secure Computing, 2025, 22(2): 1150-1167.

[18] LIU M, HU S, ZHANG J, et al. Methods for identifying complex lithologies from log data based on machine learning[J]. Unconventional Resources, 2023, 3: 20-29.

[19] LI Z L, TU X Z, GAO H, et al. LogCSS: log anomaly detection based on BERT-CNN with context-semantics-statistics features[J]. Journal of Intelligent & Fuzzy Systems, 2024, 46(4): 7659-7676.

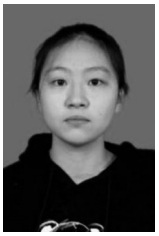
[20] 王雨晞, 叶庆卫, 周鹏, 等. 日志信息驱动的计算机网络节点故障预测研究[J]. 电信科学, 2024, 40(8): 11-22.
WANG Y X, YE Q W, ZHOU P, et al. Research on fault prediction of computer network nodes driven by log information[J]. Telecommunications Science, 2024, 40(8): 11-22.

[21] GUO H X, YUAN S H, WU X T. LogBERT: log anomaly de-

tection via BERT[C]//Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN). Piscataway: IEEE Press, 2021: 1-8.

[22] BAO F G, MAO L H, ZHU Y L, et al. An improved evaluation methodology for mining association rules[J]. Axioms, 2022, 11(1): 17.

[作者简介]



李冰（2001– ），女，宁波大学信息科学与工程学院硕士生，主要研究方向为通信网故障诊断。

叶庆卫（1970– ），男，宁波大学信息科学与工程学院教授、硕士生导师，主要研究方向为通信信号处理、信号传输与检测、分类器与机器学习等。

王雨晞（1999– ），男，宁波大学信息科学与工程学院硕士生，主要研究方向为通信网智能运维。

何镇秦（2001– ），男，宁波大学信息科学与工程学院硕士生，主要研究方向为通信网智能运维。

王晓东（1970– ），男，宁波大学信息科学与工程学院教授、硕士生导师，主要研究方向为多媒体信号处理、图像处理等。