面向服务质量感知云 API 推荐系统的数据投毒攻击检测方法

陈真[1,2]，乞文超[1]，鲍泰宇[1]，申利民[1,2]
（1. 燕山大学信息科学与工程学院，河北 秦皇岛 066004；
2. 燕山大学河北省计算机虚拟技术与系统集成重点实验室，河北 秦皇岛 066004）

摘 要：针对现有研究通常假设云 API 推荐系统的服务质量数据是可靠的，忽略了开放网络环境中恶意用户对云 API 推荐系统的数据投毒攻击的问题，提出了一种基于多特征融合的数据投毒攻击检测方法。首先，依据设计的相似性度量函数构建用户连通网络图，并利用 Node2vec 捕获用户邻域特征；其次，采用稀疏自编码器挖掘用户服务质量深度特征，并构建基于服务质量数据加权平均偏差的用户解释特征。进一步，融合用户邻域特征、服务质量深度特征和解释特征建立基于支持向量机的虚假用户检测模型，并使用网格搜索和交替迭代优化策略学习模型参数，继而实现虚假用户检测。最后，通过多组实验验证了所提方法的有效性和优越性，实现了服务质量感知云 API 推荐系统在数据端的投毒攻击防御。

关键词：推荐系统；云 API；服务质量；数据投毒；攻击检测
中图分类号：TP311
文献标志码：A
DOI: 10.11959/j.issn.1000−436x.2023161

Data poisoning attack detection approach for quality of service aware cloud API recommender system

CHEN Zhen[1,2], QI Wenchao[1], BAO Taiyu[1], SHEN Limin[1,2]
1. School of Information Science and Engineering, Yanshan University, Qinhuangdao 066004, China
2. Hebei Key Laboratory of Computer Virtual Technology and System Integration, Yanshan University, Qinhuangdao 066004, China

Abstract: To solve the problem that existing studies usually assumed that the QoS data of cloud API recommender system was reliable, ignoring the data poisoning attack on cloud API recommender system by malicious users in open network environment, a data poisoning attack detection approach based on multi-feature fusion was proposed. Firstly, a user connected network graph was constructed based on the designed similarity function, and users' neighborhood features were captured using Node2vec. Secondly, sparse auto-encoder was used to mine user QoS deep feature, and user interpretation feature based on QoS data weighted average deviation was designed. Furthermore, a fake user detection model based on support vector machine was established by integrating user neighborhood feature, QoS deep feature, and interpretation feature, the model parameters were learned using grid search and alternating iterative optimization strategy to complete fake user detection. Finally, the effectiveness and superiority of the proposed approach were verified through extensive experiments, realizing the poison attack defense against QoS aware cloud API recommender system at the data side.

Keywords: recommender system, cloud API, quality of service, data poisoning, attack detection

收稿日期：2023−05−05；**修回日期：**2023−07−27
基金项目：国家自然科学基金资助项目（No.62102348, No.62276226）；河北省自然科学基金资助项目（No.F2022203012）；中央引导地方科技发展资金资助项目（No.236Z0103G）；河北省教育厅高等学校科技计划基金资助项目（No.QN2020183）；河北省创新能力提升计划基金资助项目（No.22567626H）
Foundation Items: The National Natural Science Foundation of China (No.62102348, No.62276226), The Natural Science Foundation of Hebei Province (No.F2022203012), The Central Guiding Local Science and Technology Development Fund Project (No.236Z0103G), The Science and Technology Research Project of Hebei University (No.QN2020183), The Innovation Capability Improvement Plan Project of Hebei Province (No.22567626H)

0 引言

云时代，云应用程序接口（API, application programming interface）作为服务交付、能力复制和数据输出的最佳载体，已成为当今数字世界的基础设施[1]。云 API 正在成为人工智能技术“飞入寻常百姓家”的重要管道[2]，和实现面向服务软件开发与运行的核心要素。

然而，网络中云 API 的激增使用户越来越难从众多功能同质化的云 API 中选择高质量、个性化的云 API 进行面向服务的软件开发。为了应对上述问题，研究者引入服务质量（QoS, quality of service）来刻画云 API 的非功能侧特性[3]，以表征云 API 在某一方面的质量信息，如响应时间、吞吐量和可靠性等。但是由于高昂的时间成本、巨大的资源开销和服务质量情境依赖的特点，用户无法调用并测试全部候选云 API 的性能来获取相应的服务质量数据，云 API 提供商也很难提供符合用户情境特征的服务质量数据，故服务质量感知云 API 推荐系统成为解决这一矛盾的自然选择[4]。将具备良好性能的云 API 推荐给适用的人群，一方面可帮助用户从海量的云 API 中发现高质量、高价值的云 API；另一方面可辅助云 API 提供商了解所提供云 API 的服务质量，使其不断优化云 API 并将高质量云 API 提供给用户。

为了确保服务质量感知云 API 推荐系统的客观性和准确性，广泛采用的策略是使用调用过云 API 的其他用户的服务质量历史记录，将此历史交互记录作为云 API 推荐系统的数据基础[5]。值得注意的是，准确可信云 API 推荐系统的一个重要前提是确保用户提交的每个服务质量的评估信息真实可靠。然而，由于云 API 推荐系统自身的开放性，数据的真实性无法得到有效保证。例如，一些用户可能在服务质量评估时，为提高某些云 API 的利用率而故意降低其他云 API 的服务质量评分。服务提供商可能利用一些虚假用户故意提高自己的云 API 的服务质量，从而影响用户反馈信息的可信度，使推荐模型产生错误，使推荐方向遵循攻击者的意愿[6]。因此，如何有效利用服务质量的差异来检测虚假用户，以抵御数据投毒攻击成为服务质量感知云 API 推荐系统应用亟待解决的现实问题。

目前，鲜有服务质量感知云 API 推荐系统考虑虚假用户数据投毒攻击的影响。Zheng 等[7]提出服务质量感知云 API 推荐系统将受到用户贡献的服务质量数据可信度的高度影响。Ye 等[8]在进行云 API 推荐时，仅将离群点作为虚假用户的恶意行为，并引入改进的正则项来避免虚假用户对推荐性能的影响。Manikrao 等[9]和 Ran 等[10]提出通过第三方代理验证所有可用云 API 服务质量的注册表，以提高云 API 推荐系统的可信度。

可见，已有方法主要基于离群异常点检测[8,11]和代理验证[9-10,12]等策略进行服务质量数据安全防护，但没有考虑到在利用服务质量进行云 API 推荐之前，服务质量数据在收集、存储和预处理过程中可能受到的来自恶意用户的数据投毒攻击。基于此，本文提出从服务质量感知云 API 推荐系统数据端防御的视角出发，建立基于多特征融合的数据投毒攻击检测模型进行虚假用户检测，继而去除虚假用户以提高云 API 推荐系统的可靠性和可信性。

本文主要的贡献如下。

1) 提出了面向服务质量感知云 API 推荐系统的数据投毒攻击与检测问题，给出了服务质量感知云 API 推荐系统数据投毒攻击与检测的形式化定义。

2) 提出了融合用户邻域（neighborhood）特征、QoS 深度特征和解释（interpretation）特征的数据投毒攻击检测（NQI-Detector）方法，从多个维度捕获虚假用户和真实用户的特征差异，提升数据投毒攻击的检测效果。

3) 在真实云 API 服务质量数据集上进行大量实验，结果表明 NQI-Detector 在面向服务质量感知云 API 推荐系统的投毒攻击检测中展示出更优的检测效果。消融实验还解释了所提取特征的有效性。

1 基本概念

1.1 云 API 推荐系统数据投毒攻击定义

数据投毒攻击[13]是指攻击者通过向原始数据集中注入脏数据来达到污染数据分布的目的，使数据分布出现偏差或错误，造成云 API 推荐系统向用户推荐无质量保障的云 API，继而导致构建的服务化软件质量无法得到保障，给企业带来商业风险。

一般来说，攻击者为了避免投毒数据被检测出来，通常会在构建的虚假用户数据中加入真实数据或使虚假用户数据分布与真实用户相似。虚假用户构建完成后批量注入推荐模型中来影响模型的训练和推荐结果。

定义 1 云 API 推荐系统数据投毒攻击。服务质量感知云 API 推荐系统数据投毒攻击表示为

$$\boldsymbol{R}(A_u)\odot\boldsymbol{R}\left(\tilde{A}_{u'}\right)\Rightarrow\boldsymbol{R}\left(A_u,\tilde{A}_{u'}\right) \tag{1}$$

其中，$\odot$ 表示数据连接操作，$\boldsymbol{R}(A_u)$ 和 $\boldsymbol{R}\left(\tilde{A}_{u'}\right)$ 分别表示真实用户数据和虚假用户数据，A_u 和 $\tilde{A}_{u'}$ 分别表示真实用户 u 和虚假用户 u' 之前调用过的云 API 的集合，$\boldsymbol{R}\left(A_u,\tilde{A}_{u'}\right)$ 表示被虚假用户投毒后的用户−云 API 服务质量数据集合。

1.2　云 API 推荐系统数据投毒攻击方式

为了避免被发现，攻击者通常会根据云 API 推荐系统的知识使用特定的攻击模型生成虚假用户配置文件[14]。不同攻击模型采用不同的方式来建立攻击文件，以模拟真实用户的服务质量数据。通常，一个虚假用户的攻击数据由四部分构成，即 $\tilde{A}_{u'}=\left(A_{\mathrm{S}},A_{\mathrm{F}},A_{\Phi},A_{\mathrm{T}}\right)$。其中，$A_{\mathrm{S}}$ 是选择填充项集合，通常与目标云 API 有一定的关联，以最大化攻击的有效性；A_{F} 是随机填充项集合，随机选取一组云 API 用来伪装攻击者的攻击行为；A_{Φ} 是空白项集合，即用户未调用过的云 API 集合；A_{T} 是目标项集合，一般包含一个或一组云 API，根据投毒攻击用户的攻击目的来给目标云 API 分配服务质量数据。

根据选择填充项的不同以及随机填充项的差异，数据投毒攻击可分为以下 3 种类型：随机（Rnd, random）攻击、潮流（Bdg, bandwagon）攻击和均值（Avg, average tack）攻击。上述 3 种数据投毒攻击需要消耗的攻击成本有所区别，所带来的攻击效果也将有所不同，具体对比如表 1 所示。

表 1　　**数据投毒攻击方式对比**

攻击模型	A_{S}	A_{F}	A_{Φ}	A_{T}
随机攻击	∅	r_{random}	∅	$r_{\mathrm{max/min}}$
潮流攻击	流行的云 API	r_{random}	∅	r_{max}
均值攻击	∅	r_{average}	∅	$r_{\mathrm{max/min}}$

为了表示攻击者想要同时提高或降低多少云 API 被推荐的概率，将 A_{T} 的数量定义为攻击强度 ε。因此通过数据投毒攻击模型 $f(\cdot)$ 生成虚假用户的过程可以表示为 $u'=f_x\left(A_{\mathrm{S}},A_{\mathrm{F}},A_{\Phi},A_{\mathrm{T}},\varepsilon\right)$，其中，$x=\{\mathrm{Avg},\mathrm{Rnd},\mathrm{Bdg}\}$。

此外，考虑到攻击成本的限制，攻击者只能将有限的虚假用户注入数据集中，引入攻击规模 b 来表示一次性注入虚假用户的数量。因此，虚假用户投毒数据可进一步表示为 $\boldsymbol{R}\left(\tilde{A}_{u'}\right)=\xi\left(u'_1,u'_2,\cdots,u'_n,b\right)$，其中，$\xi(\cdot)$ 表示虚假用户选取规则。

1.3　云 API 推荐系统数据投毒攻击检测

定义 2　云 API 推荐系统数据投毒攻击检测。给定一组用户 $U=\left\{u_1,u_2,\cdots,u_m,u'_1,u'_2,\cdots,u'_n\right\}$，其中包含真实用户 u 和虚假用户 u'。令 $\boldsymbol{R}\in\mathbb{R}^{|U|\times|A|}$ 表示用户−云 API 的服务质量矩阵，$r_{u,a}\in\boldsymbol{R}$ 表示用户 u 调用云 API a 的服务质量值。云 API 推荐系统数据投毒攻击检测过程定义为

$$f(\boldsymbol{R}(A_u,\tilde{A}_{u'})\,|\,\Omega(r_{u,a}),\Omega(r_{u',a'}))=U' \tag{2}$$

其中，$\Omega(r_{u,a})$ 和 $\Omega(r_{u',a'})$ 分别表示真实用户和虚假用户调用云 API 的服务质量分布，通过区分服务质量分布情况建立检测模型识别虚假用户集合 U'。

2　云 API 推荐系统的投毒攻击检测框架

面向服务质量感知云 API 推荐系统的数据投毒攻击检测框架如图 1 所示。该检测框架针对已被数据投毒攻击的云 API 推荐系统，从多个角度提取用户特征，包括邻域特征、服务质量深度特征和解释特征，并融合提取的多个用户特征建立模型完成数据投毒攻击检测。具体包括以下 4 个步骤。

Step1　邻域特征提取。通过用户相似性函数来计算边缘权重，据此构建基于用户相似性的连通网络图，并使用 Node2vec 算法捕获用户邻域特征。

Step2　服务质量深度特征提取。从服务质量数据出发，利用稀疏自编码器（SAE, sparse auto-encoder）模型挖掘用户潜在服务质量深度特征。

Step3　解释特征提取。引入用户数据加权平均偏差设计用户解释特征。

Step4　虚假用户检测。结合网格搜索算法优化虚假用户检测器得到最后的分类结果。

3　投毒攻击检测模型的建立

3.1　邻域特征提取

以往的研究中，皮尔逊相关系数通常用于衡量用户之间的相关性。然而，用户−云 API 的服务质量的交互历史数据十分稀疏，皮尔逊相关系数很容易受到数据稀疏性的影响，当共同交互项数量很少时，很难通过少量的共同交互数据来准确评估 2 个用户的相似性[15-16]。由表 1 可知，相同的攻击模型具有相似的选择填充方式，基于此，本节从用户与

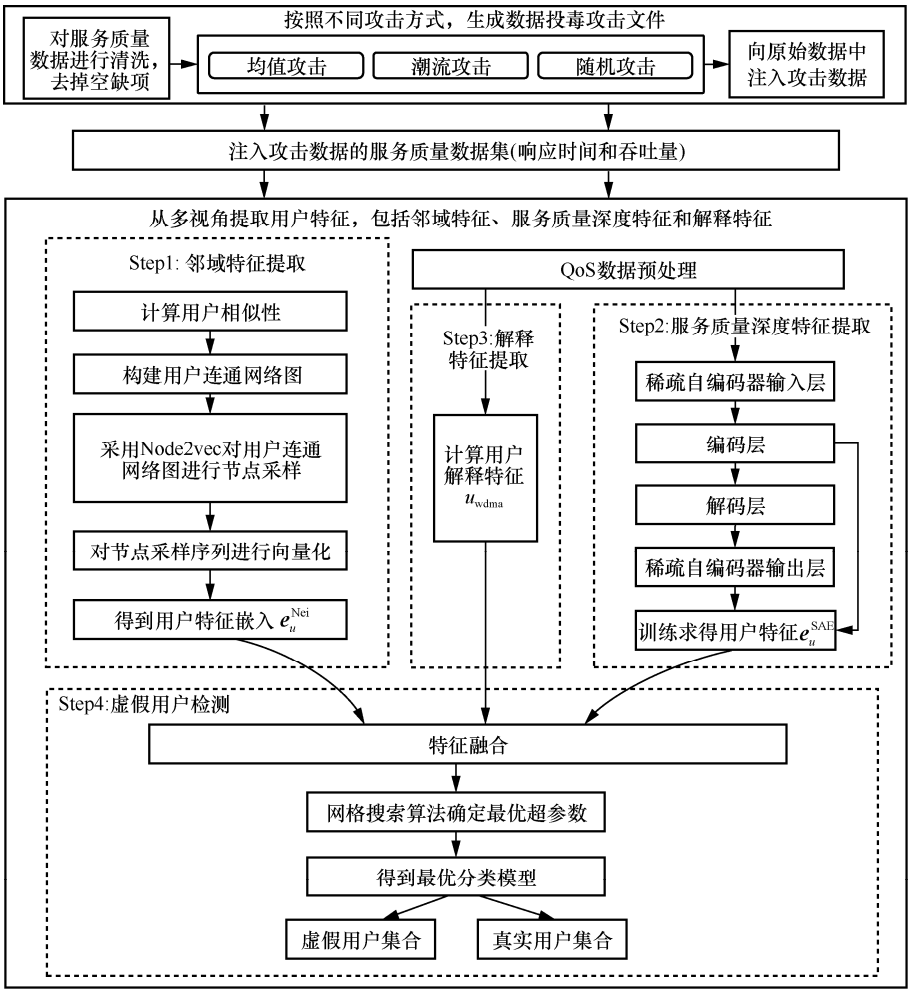


图 1　面向服务质量感知云 API 推荐系统的数据投毒攻击检测框架

云 API 交互行为的相似性中提出边缘权重度量，该度量根据服务质量的相关信息（如选择项、填充比率差异和评级偏差）进行计算，综合分析共同交互项的数量和交互数据偏差来衡量用户相似性。进一步，搭建用户连通网络进行邻域特征提取。

定义 3　用户相似性。2 个用户之间的相似性是指用户的服务质量数据的相似程度。具体地，用户 u 和 v 之间的相似性为

$$S(u,v)=\sum_{a=1}^{|A_u\cap A_v|}\frac{1}{|U_a|}\mathrm{e}^{-\lambda\delta(r_{u,a},r_{v,a})} \tag{3}$$

其中，U_a 表示与云 API a 产生过交互的用户 u 集合，λ 表示用户 u 和 v 所调用云 API 的影响参数。用户 u 和 v 的服务质量偏差判定函数 $\delta(r_{u,a},r_{v,a})$ 为

$$\delta(r_{u,a},r_{v,a})=\begin{cases}1, & |r_{u,a}-r_{v,a}|>\tau\\ 0, & \text{其他}\end{cases} \tag{4}$$

其中，τ 表示 2 个用户与同一个云 API 交互产生的服务质量偏差阈值，当偏差超过 τ 时，有理由认为 2 个用户的服务质量体验不同，用户之间的相似性不强。当 2 个用户之间的相似性够强时，即 $S(u,v)$ 的值足够大时，会在用户连通网络图中的 2 个顶点（或用户）之间创建一条边。由于虚假用户和真实用户之间具有不同的相似性模式，因此可以利用距离阈值拉近相似性更强的用户。但是这一阶段计算的用户距离不足以充分表示虚假用户和真实用户之间的差异。为了发现更深层和更稳定的关系，利用 Node2vec 算法基于用户连通网络图对用户的邻域特征进行进一步提取。

Node2vec 算法同时考虑了网络结构的同质性和结构相似性[17]，在随机游走模型的基础上，通过参数 p 和参数 q 控制随机游走过程中的邻域采样策略。当 $p>1$ 时，偏向广度优先搜索（BFS, breadth first search）策略；当 $q<1$ 时，偏向深度优先搜索

（DFS, depth first search）策略。Node2vec 随机游走采样策略如图 2 所示，在广度优先采样策略下，节点 u_1 的向量与其邻居节点 u_2～u_5 相近，表现出较大的同质性；在深度优先采样策略下，首先通过深度延伸，节点 u_1 的向量和 u_7 的向量相近，表现出结构的相似性。

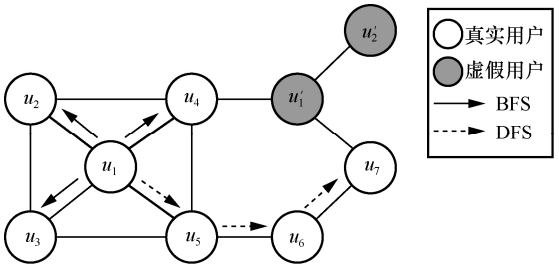


图 2　Node2vec 随机游走采样策略

用户邻居信息嵌入是将用户实体和邻居关系参数化为向量表示。在搭建好的用户连通网络图中，优化的目标是在给定目标节点的条件下，使目标节点和其邻居节点同时出现的概率最大，确保当前节点下一步能转移到邻居节点。因此，目标函数为

$$\max \sum_{u\in U} \log P_r(N_S(u)\,|\,u) \tag{5}$$

其中，u 表示任意节点（用户），$N_S(u)$ 表示采样序列中节点 u 的邻居节点。在一个节点游走序列中，任意节点出现的概率等于该节点之前的所有节点出现概率的乘积，具体形式为

$$P_r(N_S(u)\,|\,u) = \prod_{n_i\in N_S(u)} P_r(n_i\,|\,u) \tag{6}$$

假设一个节点作为源节点和作为近邻节点时共享相同的特征向量空间。那么，式(6)中的条件概率式可进一步表示为

$$P_r(n_i\,|\,u) = \frac{\exp(E(u)E(n_i))}{\sum_{v\in V}\exp(E(u)E(v))} \tag{7}$$

其中，$E(\cdot)$ 是节点特征向量映射函数，V 是连通网络图中节点集合。综上，式(5)中的目标函数可更新为

$$\arg\max \sum_{u\in U}\left[-\log Z_u + \sum_{n_i\in N_S(u)} E(u)E(n_i)\right] \tag{8}$$

其中，$Z_u = \sum_{v\in V}\exp(E(u)E(v))$。在给定节点 u 和采样序列中邻居节点 $N_S(u)$的情况下，使目标函数最大化可得到嵌入邻域信息的节点 u 的表示 $E(u)=\boldsymbol{e}_u^{\mathrm{Nei}}$，继而可得到每个节点的特征嵌入。

3.2　服务质量深度特征提取

自编码器（AE, autoencoder）能够从原始输入中提取特征，其结构通常包含输入层、编码层和解码层。将服务质量数据 x 作为 AE 的输入，并通过式(9)得到编码 $\hat{x}$ 作为隐式特征。

$$\hat{x} = f_p^{\mathrm{AE}}(\boldsymbol{W}^{(1)}x + \boldsymbol{s}) \tag{9}$$

其中，f_p 表示编码层的激活函数，$\boldsymbol{W}^{(1)}$ 表示编码层的权重矩阵，$\boldsymbol{s}$ 表示编码层偏置向量。

类似地，解码层的向量可以通过式(10)来重构输入向量。

$$\boldsymbol{e}_u^{\mathrm{AE}} = f_q^{\mathrm{AE}}(\boldsymbol{W}^{(2)}\hat{x} + \boldsymbol{t}) \tag{10}$$

其中，f_q 表示解码层的激活函数，$\boldsymbol{W}^{(2)}$ 表示解码层的权重矩阵，$\boldsymbol{t}$ 表示解码层的偏置向量。

AE 中的特征提取可以被描述为最小化输入和输出之间的误差的过程，将 AE 中的参数表示为 $\Theta=\{\boldsymbol{W}^{(1)},\boldsymbol{W}^{(2)},\boldsymbol{s},\boldsymbol{t}\}$，AE 的目标函数为

$$J_{(\mathrm{AE})}(\Theta) = \sum L\left(x, f_q^{\mathrm{AE}}\left(f_p^{\mathrm{AE}}(x)\right)\right) \tag{11}$$

其中，$L(\cdot)$ 是输入空间的损失函数。在目标函数的推进下更新所有参数，获得最优解。

在进行分类任务提取特征时，原始的自编码器只是简单地充当恒等函数，不能准确反映训练服务质量数据集的独特统计特征。一般在重建训练数据的过程中，需要使隐藏层节点数小于输入节点数，但是也可以在增加隐藏层节点数的同时，对其加入一定的稀疏限制来达到同样的效果，确保每次得到的特征编码尽量稀疏。

为了限制隐藏层节点数和活跃度，使大部分隐藏层的节点被抑制，小部分被激活，达到稀疏化数据特征的目的。本文采用稀疏自编码器提取用户服务质量深度特征，故原始基于 AE 的编码和解码过程式(9)和式(10)更新为

$$\hat{x} = f_p^{\mathrm{SAE}}(\boldsymbol{W}^{(1)}x + \boldsymbol{s}) \tag{12}$$

$$\boldsymbol{e}_u^{\mathrm{SAE}} = f_q^{\mathrm{SAE}}(\boldsymbol{W}^{(2)}\hat{x} + \boldsymbol{t}) \tag{13}$$

本文引入 KL（Kullback-Leibler）散度为模型添加稀疏性限制，使隐藏层大部分节点的活跃度很小。因此，式(11)的重构误差可以进一步表示为式(14)，当目标函数 $J_{(\mathrm{SAE})}(\Theta)$ 取最优值时，挖掘得到隐式特征。

$$J_{(\mathrm{SAE})}(\Theta) = \sum L\left(x, f_q^{\mathrm{SAE}}\left(f_p^{\mathrm{SAE}}(x)\right)\right) + \beta\sum_j \mathrm{KL}\left(\rho\,\|\,\hat{\rho}_j\right) \tag{14}$$

其中，β 表示惩罚因子，选取 Sigmoid 函数作为激活函数，神经元的输出接近激活函数上限 1 时，该神经元状态为激活；当神经元的输出接近激活函数的下限 0 时，该神经元状态为抑制。那么当某个约束或规则使神经网络中大部分神经元的状态为抑制时，该约束为稀疏性限制，KL 散度作为交叉熵与信息熵的差值，其计算式为

$$\mathrm{KL}(\rho \| \hat{\rho}_j) = \rho \log \frac{\rho}{\hat{\rho}_j} + (1-\rho) \log \frac{1-\rho}{1-\hat{\rho}_j} \tag{15}$$

其中，ρ 表示所期望的平均激活值，$\hat{\rho}_j$ 表示第 j 个神经元节点的平均激活程度，每个 $\hat{\rho}_j$ 都会向与 ρ 更相近的方向进行更新，即让每个神经元节点的平均激活值更接近期望值。对于第 j 个神经元，如果后续的 batch-size 设置为 N，则第 j 个神经元将会经过 N 次激活，输出 N 个激活值，那么 $\hat{\rho}_j$ 将取到这些激活值的平均值。

3.3 解释特征提取

服务质量感知云 API 推荐系统的数据投毒攻击的特点在于，攻击者在预测平台上注入大量虚假用户的服务质量数据，以获取符合自身意图的推荐结果。服务质量感知云 API 推荐系统的脆弱性与其过于依赖底层交互数据（如用户提交的服务质量）来训练模型以及无法有效区分真实用户和虚假用户有关。由于解释特征能很好地标记带给推荐系统性能变化的攻击者，本文引入用户数据加权平均偏差（WDMA, weighted deviation from mean agreement）u_{wdma} 作为用户解释特征，其具体形式为

$$u_{\mathrm{wdma}} = \frac{\sum_{a=0}^{N_u} \frac{|r_{u,a} - \bar{r}_a|}{N_a^2}}{N_u} \tag{16}$$

其中，N_u 表示所有与用户 u 交互的云 API 的数量，N_a 表示云 API a 被调用的数量。用户解释特征 u_{wdma} 能很好地衡量用户对一组云 API 的服务质量相对于其他用户的偏差，同时针对稀疏云 API 的服务质量偏差给予较高权重，平衡服务质量稀疏性所带来的问题，带来很高的信息增益。

3.4 虚假用户检测

邻域特征提取模块、服务质量深度特征提取模块和解释特征提取模块从多角度得到了用户的多元嵌入表示，将多特征进行融合作为最终虚假用户检测模块的输入，即

$$\boldsymbol{e}_u = \boldsymbol{e}_u^{\mathrm{Nei}} \oplus \boldsymbol{e}_u^{\mathrm{SAE}} \oplus u_{\mathrm{wdma}} \tag{17}$$

其中，$\oplus$ 表示特征的拼接操作，$\boldsymbol{e}_u^{\mathrm{Nei}}$ 表示用户邻域特征向量，$\boldsymbol{e}_u^{\mathrm{SAE}}$ 表示从稀疏自编码器中获得的用户服务质量深度特征向量，u_{wdma} 表示用户解释特征。

在虚假用户检测模块中使用改进的网格搜索算法[18]对 SVM 模型中的超参数进行优化，通过穷举搜索找到分类模型中的最优参数。改进的网格搜索算法流程如图 3 所示。

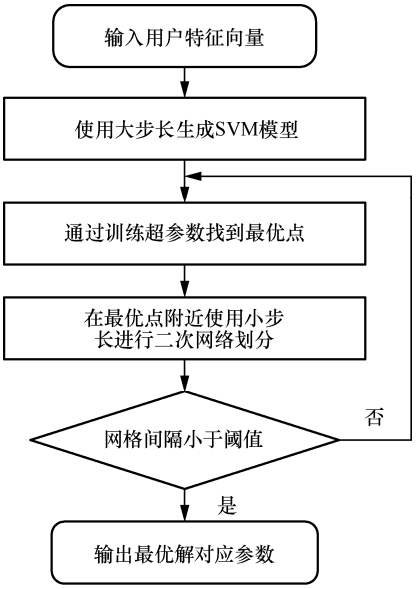


图 3 改进的网格搜索算法流程

本文中使用的改进的网格搜索算法将惩罚参数 c 和不同核函数中的核参数 a 分别取 M 和 N 个预设超参数值域，以笛卡儿积的形式生成 $M \times N$ 个 (c,a) 的参数组合，利用不同的参数组合训练不同的 SVM 分类器，从而在 $M \times N$ 个参数组合中得到准确率最高的分类器对应的参数作为最优参数。

4 算法设计

检测方法 NQI-Detector 将数据投毒攻击检测分为用户特征提取和虚假用户检测 2 个步骤。

4.1 用户特征提取

用户特征提取步骤如算法 1 所示。

算法 1 用户特征提取

输入 用户−云 API 服务质量矩阵，偏差阈值 τ，相似关系阈值 ζ，采样策略控制参数 p 和 q，节点游走路径长度 l，窗宽 k

输出 用户特征 $\boldsymbol{e}_u^{\mathrm{Nei}}$、$\boldsymbol{e}_u^{\mathrm{SAE}}$、$u_{\mathrm{wdma}}$

1) //生成用户连通网络图 G

2) for each user $u \in U$ do
3) $\text{Sim}_{u,v} \leftarrow S(u,v)$
4) if $\text{Sim}_{u,v} > \zeta$ then
5) G 中节点 u 和 v 建立一条边
6) end if
7) end for
8) //提取用户邻域特征
9) walks=∅
10) for each node $u \in U$ do
11) $\text{walk} \leftarrow \text{node2vecWalk}(G,u,l)$
12) $\text{walks} \leftarrow \text{walks} \bigcap \text{walk}$
13) $\boldsymbol{e}_u^{\text{Nei}} \leftarrow \text{skipgram}(k,d,\text{walks})$
14) end for
15) //提取用户服务质量特征
16) $\text{DN} = \text{SAESetup}(u_{\text{QoS}})$
17) $\text{SAE} = \text{SAETrain}(\text{DN}, u_{\text{QoS}})$
18) $\boldsymbol{e}_u^{\text{SAE}} = \text{getSAEVector}(\text{SAE})$
19) //提取用户解释特征
20) $u_{\text{wdma}} \leftarrow \text{calculate}(N_u, N_a)$
21) return $\boldsymbol{e}_u^{\text{Nei}}$、$\boldsymbol{e}_u^{\text{SAE}}$、$u_{\text{wdma}}$

算法 1 中第 1)～7)行是计算用户相似度，并构建基于相似性的用户连通网络图 G。第 8)～14)行是用户邻域特征的表示学习，通过优化学习参数计算节点采样序列，得到节点采样序列的向量化表示。第 15)～18)行得到了用户−云 API 服务质量深度特征表示，通过反向传播训练稀疏自编码器更新参数。第 19)～20)行按照式(16)得到用户解释特征。

在时间复杂度方面，算法 1 在生成用户连通网络图 G 时，第 2)行遍历$|U|$次，在第 3)行计算目标用户 u 遍历所有邻居节点 $v \in U$ 计算相似性，故第 1)～7)行的时间复杂度为 $O(|U|^2)$。在用户邻域特征提取中，问题规模为$|U|$，故第 8)～14)行的时间复杂度为 $O(|U|)$。第 15)～20)行计算时间复杂度为 $O(1)$。综上，算法 1 的时间复杂度为 $O(|U|^2+|U|)$。在空间复杂度方面，第 1)～7)行需借助一个大小为$|U|\times|U|$的辅助矩阵来存储用户连通网络图 G，故其空间复杂度为 $O(|U|^2)$。在第 8)～14)行提取用户邻域特征时，需要借助 2 个变量 walk、walks 以及用户邻域特征向量 $\boldsymbol{e}_u^{\text{Nei}}$，与问题规模即用户集合 U 的大小无关，故其空间复杂度为 $O(|\boldsymbol{e}_u^{\text{Nei}}|)$。在第 15)～20)行服务质量深度特征和解释特征提取中，仅需要借助 3 个变量 DN、SAE、u_{wdma} 以及用户邻域特征向量 $\boldsymbol{e}_u^{\text{SAE}}$，故其空间复杂度为 $O(|\boldsymbol{e}_u^{\text{SAE}}|)$。综上，算法 1 的空间复杂度为 $O(|U|^2+|\boldsymbol{e}_u^{\text{Nei}}|+|\boldsymbol{e}_u^{\text{SAE}}|)$。

4.2　虚假用户检测

虚假用户检测步骤如算法 2 所示。

算法 2　虚假用户检测

输入　惩罚参数集合 C，核参数集合 Kernel，用户特征 $\boldsymbol{e}_u^{\text{Nei}}$、$\boldsymbol{e}_u^{\text{SAE}}$、$u_{\text{wdma}}$

输出　用户标签 label

1) $\boldsymbol{e}_u = \boldsymbol{e}_u^{\text{Nei}} \oplus \boldsymbol{e}_u^{\text{SAE}} \oplus u_{\text{wdma}}$
2) //遍历惩罚参数
3) for $c \in C$ do
4) //遍历核参数
5) for $a \in \text{Kernel}$ do
6) $\text{svm} = \text{SVMTrain}(c,a,\boldsymbol{e}_u)$
7) if $\text{svm}_{\text{score}} > \text{best}_{\text{score}}$ then
8) $\text{best}_{\text{score}} \leftarrow \text{svm}_{\text{score}}$
9) $\text{best\_parameters} = \{c,a\}$
10) end if
11) end for
12) end for
13) //对测试用户 u 进行检测
14) $\text{label} = \text{SVM}_{\text{best\_parameters}}(\boldsymbol{e}_u)$
15) return label

算法 2 的第 1)行将算法 1 中提取的特征通过拼接操作进行特征融合，第 2)～12)行从网格优化搜索中找到最合适的参数，第 13)～14)行利用训练得到的参数完成虚假用户检测。

在时间复杂度方面，算法 2 采用改进的网格优化算法训练 SVM，在第 2)～5)行以嵌套方式通过 2 个 for 循环获得 SVM 最优的惩罚参数 c 和核参数 a。在第 6)行训练 SVM 执行时，需要遍历第 1)行获取的用户特征向量 $\boldsymbol{e}_u$，故算法 2 的时间复杂度为 $O(M\times N\times|\boldsymbol{e}_u|)$，其中，$M$ 为惩罚参数 c 的预设值集合的大小，N 为核参数 a 的预设值集合的大小。在空间复杂度方面，算法 2 第 1)行需借助一个大小为 $|\boldsymbol{e}_u|$的辅助数组存储用户特征向量，故其空间复杂度为 $O(|\boldsymbol{e}_u|)$。第 2)～12)行进行 SVM 模型训练时，仅需借助 $\text{svm}_{\text{score}}$、$\text{best}_{\text{score}}$ 和 best_parameters 等变量，与问题规模 $M\times N$ 无关，故其空间复杂度为 $O(1)$。综上，算法 2 的空间复杂度为 $O(|\boldsymbol{e}_u|)$。

由于算法 1 和算法 2 以线性方式顺序运行，故 NQI-Detector 整体的时间复杂度为 $O(|U|^2+|U|+M\times N\times$

$|\boldsymbol{e}_u|$)，空间复杂度为 $O(|U|^2+|\boldsymbol{e}_u^{\mathrm{Nei}}|+|\boldsymbol{e}_u^{\mathrm{SAE}}|+|\boldsymbol{e}_u|)$，其中$|\boldsymbol{e}_u^{\mathrm{Nei}}|<|\boldsymbol{e}_u|$，$|\boldsymbol{e}_u^{\mathrm{SAE}}|<|\boldsymbol{e}_u|$。由于 M、N 和$|\boldsymbol{e}_u|$在算法实际执行时均为有限值，故 NQI-Detector 的时空复杂度主要依赖于问题规模$|U|$，NQI-Detector 可应用于大规模云 API 推荐系统的数据投毒攻击检测。另外在数据给定后，以预处理方式执行算法 1 进行用户特征提取，可进一步提升 NQI-Detector 对投毒攻击检测的整体时空效率。

5 实验与分析

5.1 实验准备

1) 实验数据。本文实验数据来自真实世界的云 API 服务质量数据集 WS-DREAM[19]。WS-DREAM 记录了真实世界的多个云 API 被分布在不同地理位置的 339 个用户调用所产生的服务质量数据。实验数据统计信息如表 2 所示。

表 2 实验数据统计信息

数据	值
响应时间范围/s	(0,20]
平均响应时间/s	0.908
响应时间数据集用户数量/个	339
响应时间数据集云 API 数量/个	5 818
吞吐量范围/(kbit·s^{-1})	(0,1 000]
平均吞吐量/(kbit·s^{-1})	47.561
吞吐量数据集用户数量/个	339
吞吐量数据集云 API 数量/个	5 801

在开放网络环境下，网络中可用云 API 数量很多，但是单个用户通常只调用过少量云 API，因此用户–云 API 服务质量矩阵是十分稀疏的。为了使实验与真实的云 API 应用场景保持一致，在用户–云 API 服务质量矩阵上采用 A/B 模式来稀疏化服务质量数据集。

2) 参数设置。由于不同类别的服务质量数据变化范围不同，服务质量偏差阈值 τ 的取值依赖于具体服务质量类型。因此，在响应时间数据集中设置偏差阈值 τ =5，吞吐量数据集中设置偏差阈值 τ 固定为 250。此外，SVM 模型惩罚参数 c 和核参数 a 在响应时间数据集中设置为(5, 0.5)，在吞吐量数据集中设置为(20, 0.5)。

3) 评价指标。为了评估虚假用户检测效果，采用广泛使用的准确率（Precision）、召回率（Recall）和 F1 值作为评价指标，分别定义如下

$$\mathrm{Precision}=\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}} \tag{18}$$

$$\mathrm{Recall}=\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}} \tag{19}$$

$$\mathrm{F1}=\frac{2\times\mathrm{Precision}\times\mathrm{Recall}}{\mathrm{Precision}+\mathrm{Recall}} \tag{20}$$

其中，TP 表示正确判断出的正例数量，FP 表示被误认为正例的数量，FN 表示被误认为负例的数量。

4) 实验环境。硬件环境：英特尔处理器（四核），内存 32 GB。软件环境：编程语言 Python3.0，编程环境 VS Code。

5.2 对比实验

考虑到目前还没有专门的针对服务质量感知云 API 推荐系统的数据投毒攻击检测算法，为了证明所提方法检测性能的准确性和优越性，选择 4 种先进的数据投毒攻击检测算法，将其迁移到云 API 服务质量感知推荐领域，进行对比分析。4 种对比数据投毒攻击检测算法如下。

1) PCA-VarSelect[20]。典型的无监督投毒攻击检测算法，可自动构建用户特征，通过概率潜在语义分析（PLSA）找到具有相似偏好的用户群体，结合 PCA-VarSelect 从多元统计学角度重新描述评级矩阵。

2) Bayes-Detector[21]。一种有监督虚假用户检测方法，利用矩阵分解为每个用户构建隐式特征，随后使用贝叶斯模型生成潜在标签信息来更新隐式特征以完成虚假用户检测。

3) DL-DRA[22]。一种有监督方法，使用双三次插值算法减少评级矩阵的稀疏性，并使用结构化深度学习网络进行检测。

4) DSAE-EDM[23]。一种深度学习投毒攻击集成检测方法，直接采用深度自动编码器自动提取用户的潜在特征以达到攻击检测的效果。

为了验证所提方法能否很好地解决面向服务质量感知云 API 推荐系统的数据投毒攻击检测问题，实验分别在响应时间数据集和吞吐量数据集上进行数据投毒攻击检测方法的比较分析。此外，考虑到当数据投毒攻击规模较小时，通常很难区分虚假用户和真实用户。在实验过程中，将攻击强度设置为 4%，攻击规模分别设置为 3%、5%、10%和 25%，以 F1 值为评价指标，实验结果如图 4 所示，具体分析如下。

1) 在攻击强度确定的前提下，当随机攻击的攻击规模设置为 3%时，PCA-VarSelect 的 F1 值低于 0.8，无法有效区分真实用户和虚假用户。这是由于传统攻击检测算法 PCA-VarSelect 的数据降维过程是线性的，在恢复数据时会有一定程度的失真。

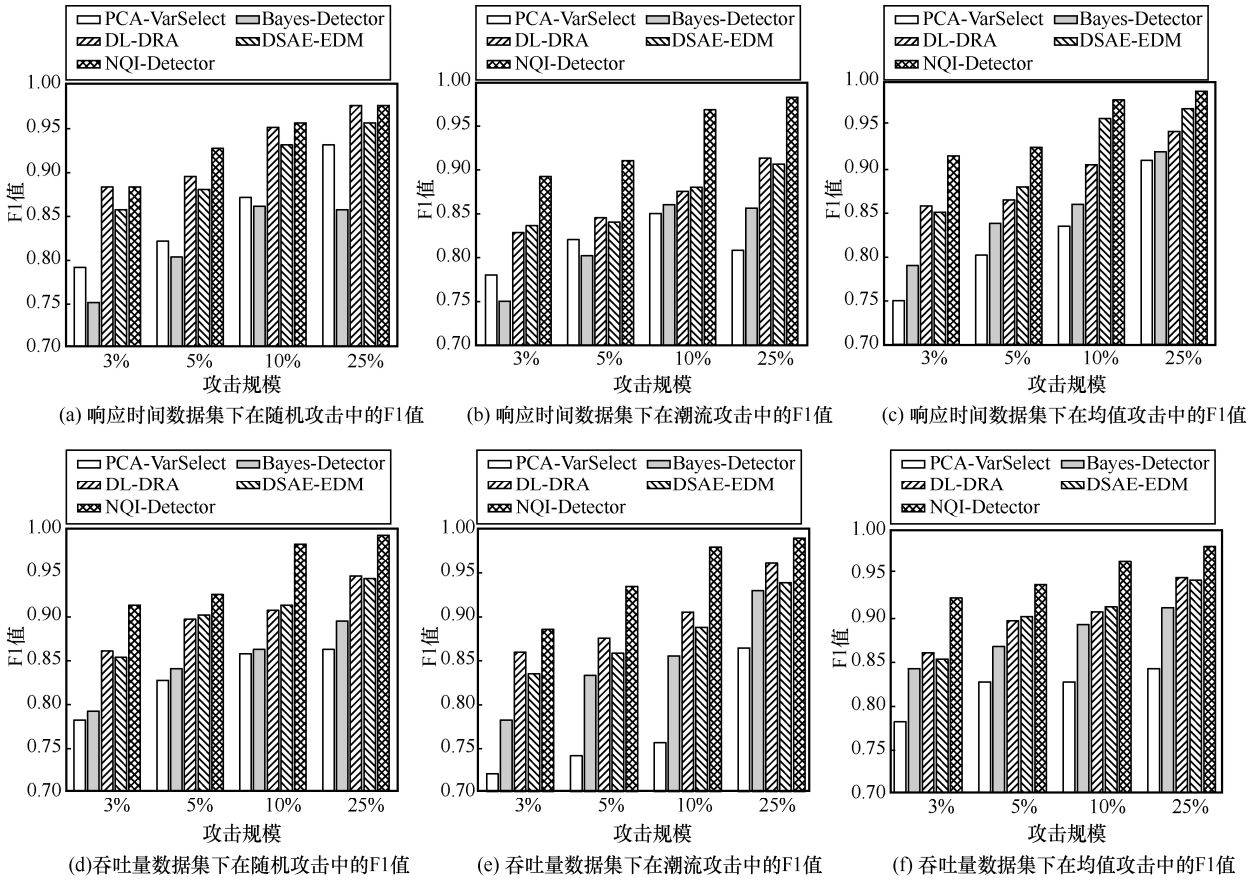


图 4　不同数据投毒攻击检测方法的对比结果

NQI-Detector 能够非线性提取用户特征，在低攻击强度下达到很好的效果。当攻击强度增大到 25%时，DSAE-EDM 和 DL-DRA 都可以有效检测。

2) 潮流攻击中，在低攻击规模下的响应时间数据集中，DSAE-EDM 和 DL-DRA 的 F1 值都不到 0.85，没有很好地完成虚假用户的潮流攻击检测。考虑到具有相似情境的用户所具备的服务质量也相似，只挖掘服务质量数据特征是不够的，NQI-Detector 与 DSAE-EDM 相比，构建了基于相似性的连通网络，增加了对用户邻域特征的挖掘，达到了强化用户特征的目的，不但克服了传统的人工检测特征在面对不同类别的攻击时所展现出来的普适性不强的问题，而且增加了虚假用户识别的准确性。

3) 均值攻击下吞吐量数据集的检测任务的完成情况要明显好于响应时间数据集，其原因可能在于均值攻击自身的攻击特点，由于现实因素，吞吐量数据集的数据范围远大于响应时间数据集，导致依靠求取均值完成虚假用户填充项的均值攻击会更大地偏离真实用户的服务质量，这有利于几种虚假用户检测算法捕获到更明显的用户特征，从而取得更优的效果。

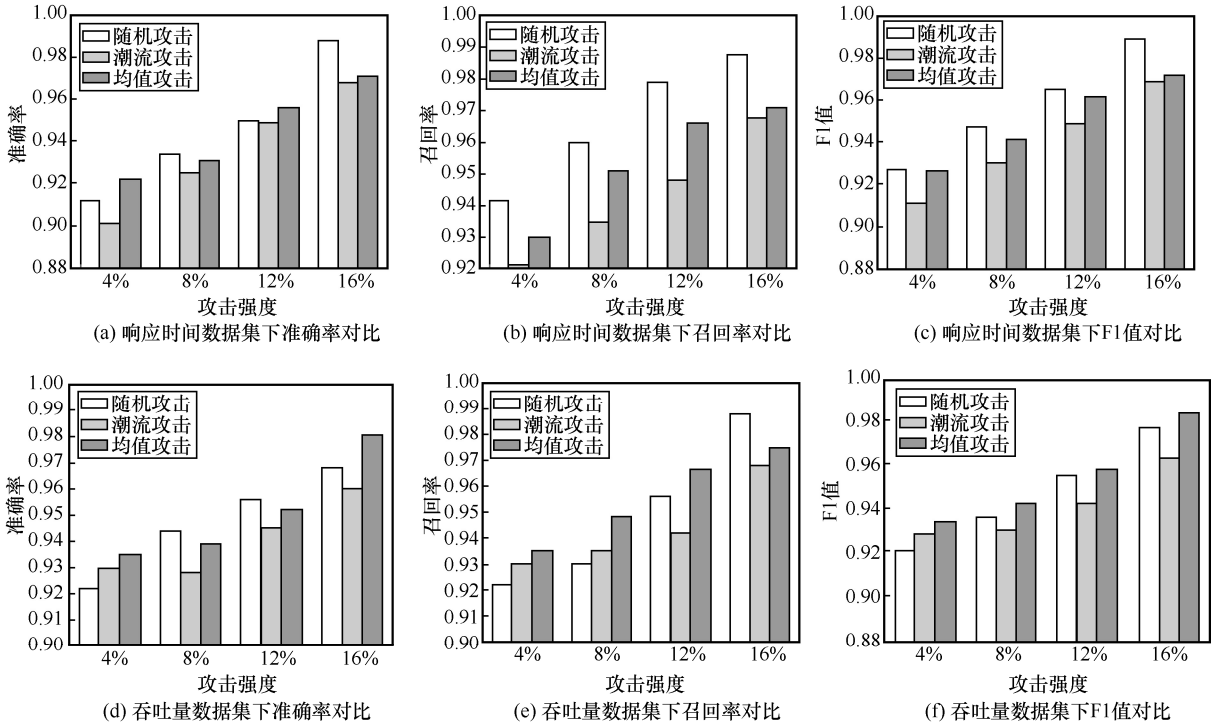
5.3　不同攻击强度下的检测效果分析

本节实验通过对云 API 服务质量数据集注入不同攻击强度的数据投毒攻击来评价 NQI-Detector 的性能，设置攻击规模为 5%。所提方法固定攻击规模为 5%的虚假用户时，分别注入不同攻击强度（4%、8%、12%和 16%）的随机攻击、潮流攻击以及均值攻击的数据投毒攻击检测结果如图 5 所示。从图 5 可以看出，随着攻击强度的增大，NQI-Detector 的召回率和准确率会逐渐提高，检测效果逐步增强。这也与具有较高攻击强度的数据投毒攻击更容易检测出攻击行为的预期一致。

5.4　不同攻击规模下的检测效果分析

本节实验通过对云 API 服务质量数据集注入不同攻击规模的数据投毒攻击来评价 NQI-Detector 的性能，设置数据投毒攻击强度为 4%，NQI-Detector 对不同攻击规模的 3 种攻击在不同评价指标下检测效果对比如图 6 所示。

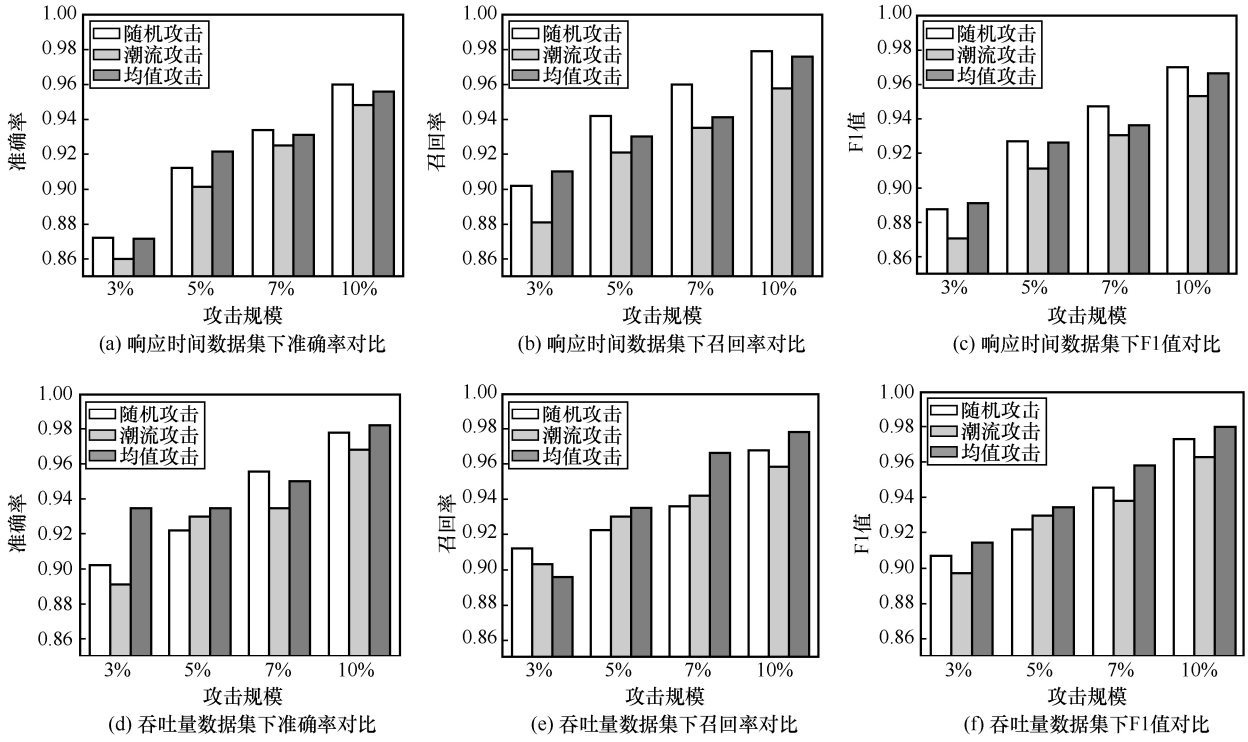
一般情况下，注入 5%规模的数据投毒攻击已



(a) 响应时间数据集下准确率对比　(b) 响应时间数据集下召回率对比　(c) 响应时间数据集下F1值对比

(d) 吞吐量数据集下准确率对比　(e) 吞吐量数据集下召回率对比　(f) 吞吐量数据集下F1值对比

图 5　不同攻击强度下的检测结果



(a) 响应时间数据集下准确率对比　(b) 响应时间数据集下召回率对比　(c) 响应时间数据集下F1值对比

(d) 吞吐量数据集下准确率对比　(e) 吞吐量数据集下召回率对比　(f) 吞吐量数据集下F1值对比

图 6　不同攻击规模下的检测结果

经消耗了恶意用户很高的知识成本，选择攻击规模分别为 3%、5%、7%、10%。从图 6 可以看出，在较小和较大的攻击规模下，NQI-Detector 依然保持良好的检测能力，这也验证了其在云 API 服务质量数据集上的检测有效性。在图 6(c)和图 6(f)的 F1 值检测结果中，NQI-Detector 对潮流攻击的检测性能低于均值攻击的性能指标。这是因为采用潮流攻击生成虚假用户时，攻击者利用赋予流行云 API 更优的服

务质量来包装自己，将自己伪装成经常使用流行云 API 的真实用户，使检测方法不易检测出来。此外，由于随机攻击在生成虚假用户时，虚假用户与云 API 交互产生的服务质量值具有不确定性，继而导致了随机攻击和潮流攻击、均值攻击之间的检测结果关系是不固定的。

5.5　敏感特征消融实验分析

为了进一步研究所提取的 3 个检测特征的有效性，本节将设计 6 个消融变体实验，以分析这些特征对提高数据投毒攻击检测的准确率是否有正向影响。给出组件 $x \in \{N, S, W, \mathrm{NS}, \mathrm{NW}, \mathrm{SW}\}$，其中，$N$、$S$ 和 W 分别表示 $\boldsymbol{e}_u^{\mathrm{Nei}}$、$\boldsymbol{e}_u^{\mathrm{SAE}}$ 和 u_{wdma}，NS、NW 和 SW 分别表示相应特征组合，使用 NQI-x 表示实验对照组没有嵌入对应组件。表 3 和表 4 展示了在响应时间数据集和吞吐量数据集上的实验结果，攻击强度设置为 4%，攻击规模设置为 10%。

如表 3 和表 4 所示，NQI-N 相对于 NQI-S 和 NQI-W 表现较差，这表明用户邻域特征在检测任务中起到了主要作用。此外 NQI-NS 在面对各类投毒攻击时性能最差，这表明解释特征在单独应用于检测时表现出了较差的性能。然而，将解释特征和服务质量深度特征相结合可以有效地检测攻击者，这种改进在响应时间数据集上表现并不明显，但是在吞吐量数据集上，这 2 种特征的结合更加有效。

5.6　提取特征可视化分析

Bayes-Detector 作为先进的使用隐式特征进行数据投毒攻击检测的方法。5.2 节实验结果表明所提 NQI-Detector 比 Bayes-Detector 方法更加有效。为了探究这一原因，采用可视化方法来分析所提取的用户特征表示。本节实验使用 Maaten 等[24]提出的 T-SNE（T-distributed stochastic neighbor embedding）对提取的特征可视化处理，T-SNE 将高维数据内部的特征放大，使相似的数据在低维中距离更近，不相似的数据在低维中距离更远。使用 T-SNE 分析 2 种方法所提取的服务质量数据集上的用户特征，响应时间数据集和吞吐量服务质量数据集上 2 种数据投毒攻击检测方法所提取的用户特征的视化结果如图 7 所示。

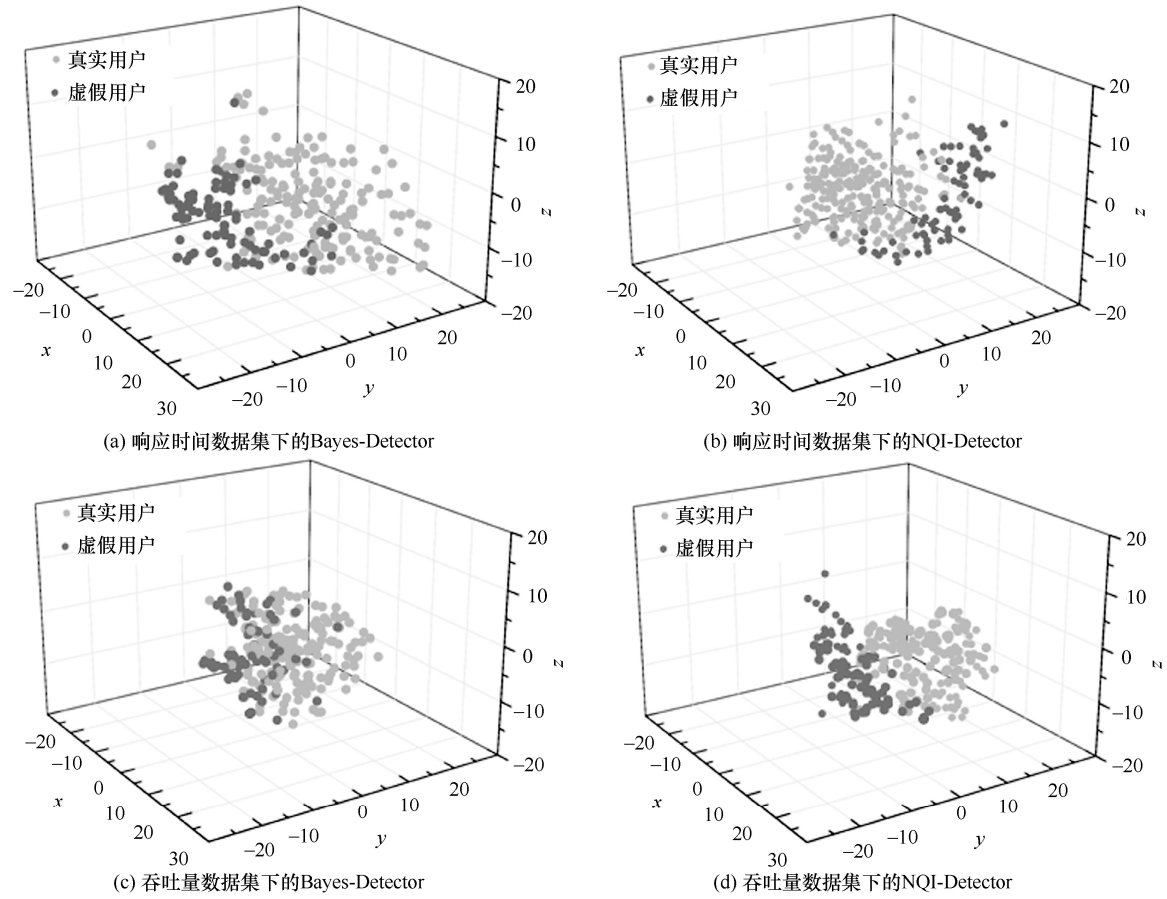
从图 7 可知，与 Bayes-Detector 相比，NQI-Detector 所提取的用户特征中重叠的用户数量更少，相似用户的特征分布更紧密，虚假用户与真实用户之间的边界更清晰。因此，NQI-Detector 在检测云 API 服务质量数据集中的虚假用户时更有效。

表 3　在响应时间数据集上消融变体实验对比结果

方法	随机攻击			潮流攻击			均值攻击		
	准确率	召回率	F1 值	准确率	召回率	F1 值	准确率	召回率	F1 值
NQI-N	0.565	0.500	0.529	0.617	0.667	0.635	0.577	0.594	0.580
NQI-S	0.841	0.817	0.829	0.927	0.951	0.941	0.902	0.947	0.927
NQI-W	0.972	0.951	0.972	0.941	0.918	0.928	0.958	0.922	0.948
NQI-NS	0.554	0.482	0.512	0.524	0.584	0.552	0.590	0.549	0.552
NQI-NW	0.594	0.569	0.573	0.561	0.538	0.545	0.561	0.527	0.548
NQI-SW	0.958	0.945	0.950	0.932	0.910	0.924	0.945	0.962	0.955
NQI	**0.989**	**0.964**	**0.955**	**0.975**	**0.959**	**0.968**	**0.970**	**0.982**	**0.979**

表 4　在吞吐量数据集上消融变体实验对比结果

方法	随机攻击			潮流攻击			均值攻击		
	准确率	召回率	F1 值	准确率	召回率	F1 值	准确率	召回率	F1 值
NQI-N	0.732	0.754	0.742	0.695	0.752	0.617	0.715	0.744	0.729
NQI-S	0.941	0.964	0.955	0.849	0.869	0.854	0.874	0.832	0.857
NQI-W	0.964	0.970	0.966	0.957	0.922	0.941	0.958	0.967	0.962
NQI-NS	0.533	0.630	0.574	0.585	0.562	0.542	0.523	0.583	0.554
NQI-NW	0.583	0.659	0.611	0.570	0.551	0.561	0.614	0.655	0.639
NQI-SW	0.961	0.957	0.958	0.953	0.948	0.955	0.938	0.920	0.928
NQI	**0.978**	**0.985**	**0.980**	**0.964**	**0.982**	**0.975**	**0.972**	**0.955**	**0.961**



(a) 响应时间数据集下的Bayes-Detector　(b) 响应时间数据集下的NQI-Detector

(c) 吞吐量数据集下的Bayes-Detector　(d) 吞吐量数据集下的NQI-Detector

图 7　不同服务质量数据集下的用户特征可视化分析

6　结束语

为了解决服务质量感知云 API 推荐系统中的数据投毒攻击问题，本文提出了一种面向服务质量感知云 API 推荐系统的基于多特征融合的数据投毒攻击检测方法。首先，根据设计的相似性函数构建了用户连通网络图，利用 Node2vec 算法通过图上的随机游走捕获用户邻域特征；其次，从服务质量数据视角，利用稀疏自编码器挖掘用户服务质量深度特征表示，并构建了用户解释特征。然后，采用特征嵌入技术连接用户的邻域特征、服务质量深度特征与解释特征，实现了用户的多元嵌入表示。在此基础上，结合网格搜索优化虚假用户检测器，提升高维稀疏环境下虚假用户的检测效果。最后，从多个方面进行实验，系统地分析了所提出的数据投毒攻击检测方法在服务质量感知云 API 推荐系统中的有效性。

参考文献：

[1] GONG W W, ZHANG X Y, CHEN Y F, et al. DAWAR: diversity-aware Web APIs recommendation for mashup creation based on correlation graph[C]//Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval. New York: ACM Press, 2022: 395-404.

[2] QI L Y, LIN W M, ZHANG X Y, et al. A correlation graph based approach for personalized and compatible Web APIs recommendation in mobile APP development[J]. IEEE Transactions on Knowledge and Data Engineering, 2023, 35(6): 5444-5457.

[3] CHEN Z, PAN M S, HE P F, et al. Context and auto-interaction are all you need: towards context embedding based QoS prediction via automatic feature interaction for high quality cloud API delivery[J]. Future Generation Computer Systems, 2022, 128: 265-281.

[4] SHU Y J, ZHANG J H, ZHANG W E, et al. IQSrec: an efficient and diversified skyline services recommendation on incomplete QoS[J]. IEEE Transactions on Services Computing, 2023, 16(3): 1934-1948.

[5] CAO B, LIU J, WEN Y, et al. QoS-aware service recommendation based on relational topic model and factorization machines for IoT mashup applications[J]. Journal of Parallel and Distributed Computing, 2019, 132: 177-189.

[6] 陈真, 乞文超, 贺鹏飞, 等. 云应用程序编程接口安全研究综述：威胁与防护[J]. 电子与信息学报, 2023, 45(1): 371-382.
CHEN Z, QI W C, HE P F, et al. A survey for cloud application programming interface security: threats and protection [J]. Journal of Electronics & Information Technology, 2023, 45(1): 371-382.

[7] ZHENG Z B, LI X L, TANG M D, et al. Web service QoS prediction via collaborative filtering: a survey[J]. IEEE Transactions on Services

Computing, 2022, 15(4): 2455-2472.

[8] YE F H, LIN Z W, CHEN C, et al. Outlier-resilient Web service QoS prediction[C]//Proceedings of the Web Conference 2021. New York: ACM Press, 2021: 3099-3110.

[9] MANIKRAO U S, PRABHAKAR T V. Dynamic selection of Web services with recommendation system[C]//Proceedings of International Conference on Next Generation Web Services Practices. Piscataway: IEEE Press, 2005: 117-121.

[10] RAN S P. A model for Web services discovery with QoS[J]. ACM SIGecom Exchanges, 2003, 4(1): 1-10.

[11] YANG J, CHEN Y, RAHARDJA S. Neighborhood representative for improving outlier detectors[J]. Information Sciences, 2023, 625: 192-205.

[12] BURRA V B, PACHALA S. An improved proxy re-encryption scheme using resource optimization and authentication protocol[J]. International Journal of System Assurance Engineering and Management, 2023: doi.org/10.1007/s13198-022-01809-9.

[13] GUPTA P, YADAV K, GUPTA B B, et al. A novel data poisoning attack in federated learning based on inverted loss function[J]. Computers & Security, 2023, 130: 1-8.

[14] ZHANG H X, WANG D Y, ZHANG W, et al. QoS prediction in intelligent edge computing based on feature learning[J]. Journal of Cloud Computing, 2023, 12(1): 1-16.

[15] LI J H, WU H, CHEN J P, et al. Topology-aware neural model for highly accurate QoS prediction[J]. IEEE Transactions on Parallel and Distributed Systems, 2022, 33(7): 1538-1552.

[16] ZHU J, LI B, WANG J, et al. BGCL: bi-subgraph network based on graph contrastive learning for cold-start QoS prediction[J]. Knowledge-Based Systems, 2023, 263: 1-11.

[17] HA J, PARK S. NCMD: Node2vec-based neural collaborative filtering for predicting MiRNA-disease association[J]. IEEE/ACM Transactions on Computational Biology and Bioinformatics, 2022, 20(2): 1257-1268.

[18] XIA D W, ZHENG Y L, BAI Y, et al. A parallel grid-search-based SVM optimization algorithm on Spark for passenger hotspot prediction[J]. Multimedia Tools and Applications, 2022, 81(19): 27523-27549.

[19] YANG Y T, ZHENG Z B, NIU X D, et al. A location-based factorization machine model for Web service QoS prediction[J]. IEEE Transactions on Services Computing, 2021, 14(5): 1264-1277.

[20] MEHTA B, NEJDL W. Unsupervised strategies for shilling detection and robust collaborative filtering[J]. User Modeling and User-Adapted Interaction, 2009, 19(1): 65-97.

[21] YANG F, GAO M, YU J L, et al. Detection of shilling attack based on Bayesian model and user embedding[C]//Proceedings of 2018 IEEE 30th International Conference on Tools with Artificial Intelligence. Piscataway: IEEE Press, 2018: 639-646.

[22] ZHOU Q, WU J, DUAN L. Recommendation attack detection based on deep learning[J]. Journal of Information Security and Applications, 2020, 52: 1-13.

[23] 郝耀军, 张付志. 基于深度自动编码器的托攻击集成检测方法[J]. 计算机工程与应用, 2019, 55(1): 9-22.
HAO Y, ZHANG F. Ensemble detection method for shilling attacks based on deep sparse autoencoder [J]. Computer Engineering and Applications, 2019, 55(1): 9-22.

[24] MAATEN L, HINTON G. Visualizing data using t-SNE [J]. Journal of Machine Learning Research, 2008, 9: 2579-2605.

[作者简介]

陈真（1987− ），男，陕西宝鸡人，博士，燕山大学副教授、博士生导师，主要研究方向为服务计算、推荐系统和服务化软件开发等。

乞文超（1998− ），女，河北邢台人，燕山大学硕士生，主要研究方向为服务计算、云 API 安全和推荐算法。

鲍泰宇（1999− ），男，河北石家庄人，燕山大学硕士生，主要研究方向为服务计算、协同推荐算法和数据投毒攻击与检测等。

申利民（1962− ），男，黑龙江佳木斯人，博士，燕山大学教授、博士生导师，主要研究方向为协同计算、服务计算和信息安全等。