HFLGuard：基于聚类和自编码器的异构联邦中毒防御方法

李勇飞[1]，方晨[1]，常朝稳[1]，郭渊博[2]，付春辉[1]
（1.信息工程大学密码工程学院，河南 郑州 450001；2.海南大学网络空间安全学院，海南 海口 570100）

摘　要：联邦学习的分布式特性使其很容易受到恶意客户端的攻击，造成隐私泄露和模型发散，而且现实场景下数据分布通常是多源异构的，进一步增加了抵御攻击的难度。为此，提出了一种异构联邦中毒防御方法HFLGuard。首先，计算客户端加权历史梯度，凸显恶意客户端“目标”。接着，利用K-means聚类算法对梯度进行初步分类，训练基于Transformer的自编码器，识别恶意梯度并计算偏离程度。最后，设计动态信任分数聚合机制，惩罚多次投毒的恶意客户端，基于信任分数为客户端赋予相应权重。在基准数据集MNIST、FMNIST、CIFAR-10和CIFAR-100上的实验结果表明，所提方法能够有效抵御目标和非目标中毒攻击。

关键词：联邦学习；中毒防御；自编码器；K-means聚类；动态权重
中图分类号：TP391
文献标志码：A
DOI：10.11959/j.issn.1000–436x.2025206

HFLGuard: clustering and autoencoder-based defense method against heterogeneous federated poisoning

LI Yongfei[1], FANG Chen[1], CHANG Chaowen[1], GUO Yuanbo[2], FU Chunhui[1]
1. Department of Cryptogram Engineering, Information Engineering University, Zhengzhou 450001, China
2. School of Cyberspace Security, Hainan University, Haikou 570100, China

Abstract: The distributed nature of federated learning made it vulnerable to attacks by malicious clients, leading to privacy leakage and model degradation. Additionally, the inherent heterogeneity and multi-source nature of real-world data distributions presented further challenges in mitigating these threats. To address these issues, a heterogeneous federated poisoning defense method HFLGuard was proposed. The method began by computing a weighted history gradient for each client to emphasize potential malicious “targets”. Subsequently, the K-means clustering algorithm provided an initial classification of gradients, followed by a Transformer-based autoencoder that detected malicious gradients and quantified their deviation. Finally, a dynamic trust score aggregation mechanism was then employed to penalize clients engaged in repeated poisoning attempts, assigning appropriate weights based on trust scores. Experiments conduct on benchmark datasets, including MNIST, FMNIST, CIFAR-10, and CIFAR-100, demonstrate that the proposed approach effectively mitigates both targeted and untargeted poisoning attacks.

Keywords: federated learning, poisoning defense, autoencoder, K-means cluster, dynamic weight

0　引言

随着信息技术的迭代演进，数字化建设进入了新的发展阶段，海量数据已成为国家基础性战略资源。各领域发展都离不开数据的支撑，如智能驾驶[1]、入侵检测[2]、推荐系统[3]、知识图谱构建[4]等领域，模型性能与数据质量密切相关。为了在保

收稿日期：2025–06–04；修回日期：2025–11–14
通信作者：方晨，17756230629@163.com
基金项目：河南省青年科学基金资助项目（No.242300420700）；先进计算与智能工程实验室基金资助项目（No.2023-LYJJ-01-023）
Foundation Items：The Natural Science Foundation of Henan Province (No.242300420700), The Foundation of Advanced Computing and Intelligent Engineering Laboratory (No.2023-LYJJ-01-023)

护数据隐私的前提下共享数据，提升模型训练效果，联邦学习[5]作为一种新的机器学习范式应运而生。

联邦学习采用分布式架构，系统内通常包含一个服务器和多个客户端，通过交互模型参数或更新梯度实现信息共享，使数据“可用不可见”，打破数据壁垒的同时避免隐私泄露。

然而，联邦学习的分布式特性使恶意客户端可以违背训练协议，破坏全局模型收敛和推断用户隐私信息。其中中毒攻击是联邦学习面临的最大威胁之一[6]。从目的上来看，中毒攻击可分为破坏特定标签映射的目标攻击和降低模型整体性能的非目标攻击。

目标攻击的核心意图是通过篡改模型行为，使模型对特定输入产生错误预测，同时保持其他输入的正常预测，具有很强的隐蔽性和针对性，攻击方式主要包括标签翻转（LF, label flipping）和后门注入。非目标攻击旨在破坏模型的整体性能，具有更强的破坏性，攻击方式主要包括模型中毒和本地数据污染。本文主要针对基于标签翻转的目标攻击和基于模型中毒的非目标攻击展开研究。

标签翻转攻击将数据集中的一类标签（称为源标签）替换为另一特定标签（称为目标标签），在不改变训练样本的前提下，使模型学习到错误的映射。由于仅改变源标签，标签翻转攻击易于实施，且难以被发现，潜在威胁很大。例如，在自动驾驶领域，受攻击模型可能将“停止”识别为“限速”，从而造成严重的后果[7]。模型中毒攻击是指恶意客户端篡改上传的模型参数或更新梯度，使全局模型偏离正确的方向，降低模型整体性能。例如，在物联网环境下，攻击者上传篡改后的参数，致使全局模型发散，破坏模型的可用性[8]。

针对中毒攻击，已有研究提出了缓解措施。例如，Sameera等[9]采用多类支持向量机（SVM, support vector machine）计算客户端模型到超平面的距离，识别恶意客户端。Lai等[10]引入局部异常因子（LOF, local outlier factor）比较各客户端的离群程度，之后利用辅助数据集对其进行准确性测试，该辅助数据集需要从与原始数据集相同分布的数据中采样。然而，这些方法通常具有不切实际的前提假设，在现实联邦学习场景中难以应用。同时，联邦学习面临严重的非独立同分布（Non-IID, non-independent and identically distributed）问题，通用领域的中毒防御方法在数据分布不平衡的条件下防御效果不佳。

在异构场景下，良性梯度呈现出多样性和偏移性，恶意梯度更易隐蔽其中，同时增加了良性客户端误判的可能性。具体来说，基于距离或统计度量的防御方法易受良性梯度之间极端参数的干扰，将良性梯度误判为恶意梯度。基于辅助数据集的防御方法要求服务器拥有与客户端本地分布相近的本地数据集，然而Non-IID问题使这一条件更加苛刻。基于相似度的防御方法通常假定“良性梯度之间的相似度应当远大于恶意梯度”，但在异构场景下，由于本地数据分布的差异，良性客户端相似度可能会大幅下降，仅依赖梯度之间的相似性将造成大量的误判漏判。另外，现有防御方法面对协同攻击时性能下降甚至完全失效。

针对上述挑战，本文提出了一种面向异构联邦场景的中毒防御方法，服务端不需要辅助数据集便可区分客户端上传的恶意梯度和良性梯度。本文的一个关键观察是，客户端历史梯度中通常包含其长期目标的信息，因此，本文设计加权历史梯度，结合历史信息识别恶意客户端。另外，为防止Non-IID场景下聚类算法误判良性梯度，提出利用自编码器（AE, autoencoder）与聚类结合的方法识别恶意梯度，同时解决了自编码器训练标签的问题。自编码器通过学习良性簇中的共性特征，采用标准差权重容忍良性梯度的多样性，并为重构损失大的恶意梯度赋予低权重。最后，本文构建了怀疑分数和信任分数2个列表，即使自编码器中仍然存在少量的误判漏判，动态分数记录的长期行为将会修正其错误，提高对频繁攻击的恶意客户端的惩罚，增强鲁棒性。本文的主要贡献如下。

1) 针对异构联邦场景下的中毒攻击，提出了一种“聚类初步筛选–自编码器精确校验–动态信任分数修正”的多阶段服务端侧防御方法HFL-Guard，仅依赖客户端上传的梯度信息，不需要额外辅助数据集恶意客户端数量等先验假设即可识别恶意梯度，利用动态权重对多次投毒的恶意客户端进行惩罚，在保持模型精度的同时提升防御适用性和鲁棒性。

2) 为进一步凸显客户端目标，提出了一种可衡量偏离程度的恶意梯度识别方法，设计时间衰减因子，计算各客户端的加权历史梯度，融合历史信

息的同时降低早期梯度对当前更新的影响，基于K-means聚类算法获得初步标签。利用自编码器学习客户端梯度的时空特征，重构客户端梯度，将重构损失过大的判定为恶意梯度，并量化偏离程度。

3) 为防止恶意客户端多次发起攻击对模型造成的影响，设计了一种动态信任分数聚合机制，服务端通过维护信任分数和怀疑分数2个列表，提高客户端多次投毒的惩罚，即使恶意客户端暂时伪装成良性，长期积累的怀疑分数将阻止其参与聚合，缓解了攻击者伪装其行为以绕开防御方法的影响。

4) 在4个基准数据集MNIST、FMNIST、CIFAR-10和CIFAR-100上展开广泛实验，实验结果表明，本文方法面对异构联邦中毒攻击时，能够在降低攻击成功率（ASR, attack success rate）的同时保持较小的精度损失，验证了HFLGuard的防御效果。

1　相关工作

1.1　联邦学习

联邦学习是由Google提出的机器学习范式[5]，服务端首先初始化全局模型，分发给各客户端，客户端本地训练后上传本地梯度更新，服务端对这些梯度采用某种聚合算法进行聚合，获得全局梯度对模型进行更新后再次下发，重复迭代此过程。

为适应联邦学习场景下面临的分布、异构等问题，聚合算法层出不穷，包括FedAvg[5]、FedProx[11]、FedAdam[12]、FedDyn[13]、FedCM[14]等，本文将FedAvg作为默认聚合算法，形式化表达为

$$g_{\text{global}}^{t}=\sum_{k=1}^{K}\frac{|D_k|}{|D|}g^{k,t} \tag{1}$$

$$\omega^{t+1}=\omega^{t}-\eta g_{\text{global}}^{t} \tag{2}$$

其中，$g^{k,t}$表示客户端k第t轮上传的梯度更新，η表示学习率。

1.2　联邦学习中毒攻击

Yang等[15]提出了一种模型洗牌的模型中毒攻击，设计了独特的方式打乱和缩放模型参数，在测试集上表现正常，但降低了全局模型的收敛速度，甚至造成模型发散。Yu等[16]针对联邦推荐系统提出了一种聚类攻击，通过上传中毒梯度，将条目嵌入收敛到几个密集簇中，使系统为这些条目生成相似的分数，并扰乱排名顺序。Kasyap等[17]提出了利用超维计算制作有毒样本，将有毒样本投影到超维空间，并在目标类样本中添加扰动，加入隐藏后门或噪声，并提出基于超维的置信度度量，用于检查模型一致性。女巫攻击的出现进一步增强了中毒攻击的效果，Xiao等[18]模拟多个节点加入系统，使用标签翻转攻击实现本地中毒，进行合谋攻击。

1.3　联邦学习中毒防御

为防御联邦学习场景下的中毒攻击，现有研究做出了许多努力。Blanchard等[19]提出了Krum算法，计算客户端梯度与剩余$n-m-2$个客户端梯度之间的距离，选择距离和最小的客户端作为本轮更新。Yin等[20]提出了Trimmed Mean算法，对客户端梯度的每一维排序，删除其中最大和最小的k个参数，并对剩下的参数求均值，作为本轮更新。然而，上述基于统计的方法在Non-IID场景下性能较差，同时，在客户端数量增加时，计算开销呈指数级增长。为此，Jebreel等[21]提出了FL-Defender方法，求出最后一层梯度之间的相似度，压缩相似度矩阵，求出压缩向量的质心，基于质心与其他压缩向量的偏离程度为客户端赋予相应权重。然而，上述方法在攻击者占比较高时逐渐失效。Gupta等[22]设置滑动窗口，对客户端历史梯度求和平均，利用K-means聚类算法将客户端梯度分为2个簇，其中较小的簇被识别为恶意梯度。然而，在客户端均为良性的场景下，仅基于聚类结果删除梯度将造成大量误报，严重损害模型精度。Nguyen等[23]使用HDBSCAN聚类识别并去除具有高角度偏差的中毒模型；自适应地调整裁剪边界，限制攻击者放大的后门模型的影响；根据裁剪边界估计适当的噪声水平，以消除后门。然而，在上述方法中，若20%的攻击者上传相同的更新，则HDBSCAN方法将完全失效。

2　威胁模型

2.1　攻击模型

根据先前研究，假定联邦学习系统中共有K个客户端，攻击者难以控制超过半数的客户端，因此最多有M个被攻击者控制的恶意客户端，满足$M\leqslant\frac{K-1}{2}$，这一设定符合常见的联邦学习场景[23-24]。攻击者了解模型架构、训练算法和全局模型参数。根据攻击者能力，主要考虑以下2种威胁场景：1) 攻击者能够污染受控客户端的数据集，篡改特定标签，实施标签翻转攻击；2) 攻击者能够直接

篡改上传梯度，实施模型中毒攻击，降低全局模型整体性能。在上述场景中，假定受控客户端之间均可以相互勾连，共谋相同目标，从而能够提升攻击效果。

2.2　攻击模型

为贴合实际联邦学习场景，本文舍弃现有防御方法中不切实际的假设。

1) 拥有辅助数据集。假定服务端包含与本地训练集相同分布的辅助数据集，对上传模型进行测试或训练有监督恶意分类器。然而，联邦学习的可用不可见属性使服务端无法获得客户端本地数据。

2) 知晓恶意客户端数量。假定服务端知晓系统中攻击者的数量，从而排除相应数量的客户端。然而，攻击者可能在任意轮次发起攻击，若依赖提前确定的固定攻击者数量执行防御，将会造成大量误报漏报。

与之相反，本文方法不需要采样辅助数据集和提前了解恶意客户端数量，服务端仅需获取客户端每轮上传的更新梯度。防御目标是在保持模型精度的同时抵御标签翻转攻击和模型中毒攻击。

3　方法

本文提出了一种异构联邦学习场景下的中毒防御方法 HFLGuard，框架如图 1 所示。HFLGuard 主要包含 3 个模块：加权历史梯度计算（WHGC, weighted history grads calculation）、恶意梯度识别和基于动态信任分数聚合。具体来说，服务端首先为各客户端不同轮次的梯度赋予相应权重，计算加权历史梯度，获取客户端的“目标”。接着，基于无监督聚类将历史梯度初步划分为两类，较大的簇视为良性梯度簇，用于训练自编码器。之后将本轮所有历史梯度输入自编码器，计算重构损失，大于阈值的视为恶意梯度。最后，设计怀疑分数和信任分数，当客户端被识别为恶意时，基于重构损失累加其怀疑分数，从而增加对多次投毒恶意客户端的惩罚力度，并降低信任分数，为各客户端动态赋予权重，实现加权聚合。完整过程如算法 1 所示。

算法 1　HFLGuard 中毒防御

输入　客户端数量 K，通信轮次 T，防御轮次 t_d，加权历史梯度 g_{his}，恶意客户端集 mc，时间衰减因子 α_t

输出　鲁棒全局模型 W^T

1)　初始化全局模型 ω_0 并分发

2)　for t = 1 to T do

3)　　for k = 1 to K do

4)　　//加权历史梯度计算

5)　　$g_{\text{his}}^{k,t}$ =WHGC ($g_{\text{his}}^{k,t-1}$,$g^{k,t}$,α_t)

6)　　end for

7)　//恶意梯度识别，计算损失阈值 TH^t，将识别到的恶意客户端加入恶意客户端集 mc 中

8)　mc,$\text{Loss}_{\text{re}}^t$,$\text{TH}^t$ =MGI(g_{his}^t)

9)　//自适应聚合，获得鲁棒全局模型 W^T

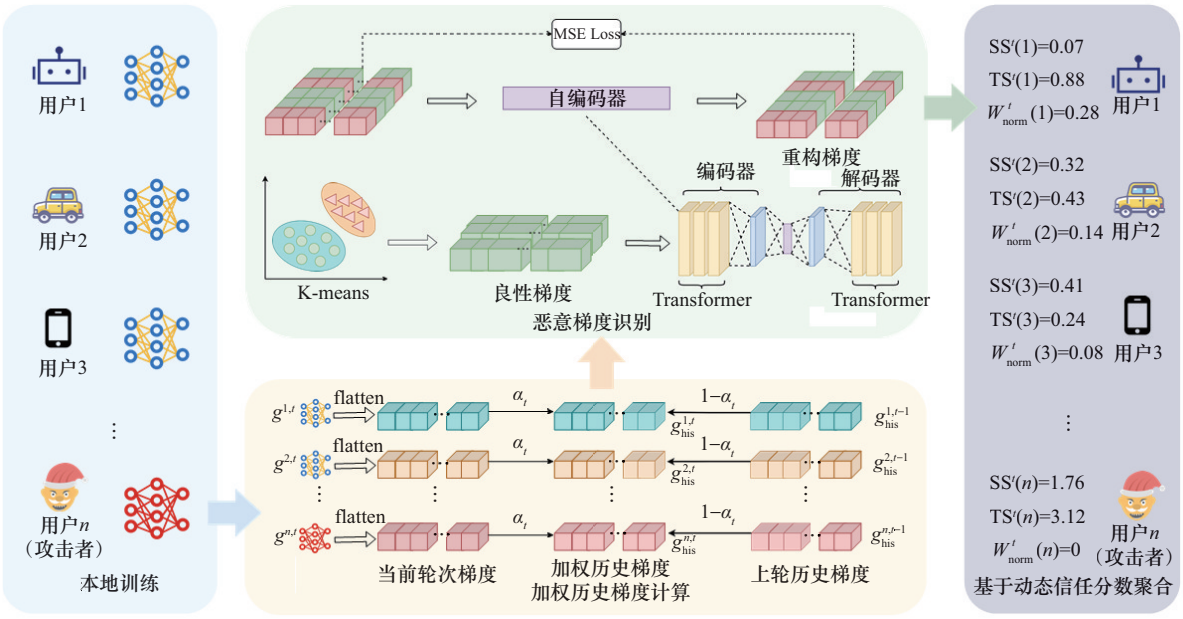


图 1　HFLGuard 框架

10)　W^t =AA(mc,$\text{Loss}_{\text{re}}^t$,$\text{TH}^t$)

11) end for

12) return W^T

3.1　加权历史梯度计算

长期历史梯度能够更好地彰显客户端目标，通过分析研判历史梯度，收集中毒攻击的趋势和规律，能够为防御方法提供更好的特征表示。

Jebreel 等[21]和 Liu 等[25]发现，在训练的初始阶段，全局模型的目标是探索特定参数空间，而不是收敛到特定最优解，此时攻击者设计的模型更新更容易被后续正常更新所覆盖，发动多轮攻击不仅攻击效果不佳，且容易被恢复到正确的方向，而在模型接近收敛时投毒能够增强攻击效果，故攻击者更倾向于在此时进行投毒攻击。

因此，在计算历史梯度时，应当为近期梯度赋予更大的权重，逐渐削弱早期梯度的影响，避免其对当前更新方向产生过大干扰。本文提出了一种新的加权历史梯度计算方法，引入时间衰减因子 α_t，如算法 2 所示，具体计算式为

$$g_{\text{his}}^{k,t} = \alpha_t g_{\text{his}}^{k,t-1} + (1-\alpha_t) g^{k,t} \tag{3}$$

$$\alpha_t = \frac{1}{t - t_d + 1} \tag{4}$$

其中，$g_{\text{his}}^{k,t}$ 表示客户端 i 前 t 轮的加权历史梯度，α_t 表示第 t 轮时前 $t-1$ 轮历史梯度的权重系数，t_d 表示开始执行防御策略的轮次。即使早期恶意客户端设计伪装梯度，其影响在后续轮次中也在不断减弱，权重不断降低。

算法 2　加权历史梯度计算 WHGC

输入　加权历史梯度 $g_{\text{his}}^{k,t-1}$，当前轮次客户端梯度 $g^{k,t}$，时间衰减因子 α_t

输出　当前轮次加权历史梯度 $g_{\text{his}}^{k,t}$

1)　$\alpha_t = \dfrac{1}{t - t_d + 1}$

2)　$g_{\text{his}}^{k,t} = \alpha_t g_{\text{his}}^{k,t-1} + (1-\alpha_t) g^{k,t}$

3)　return $g_{\text{his}}^{k,t}$

3.2　恶意梯度识别

基于 K-means 聚类算法将上述加权历史梯度分为 2 个簇。根据 2.1 节中假设，恶意客户端数量必然少于良性客户端数量，同时，良性客户端基于本地数据训练模型，虽然在 Non-IID 场景下受分布差异的影响，但其整体优化目标与全局最优解的方向更加接近，这种共性使其梯度在特征空间中更加接近，呈现出可聚合的分布特征，即使存在一定的多样性，也能够聚合成更大的簇。反观恶意客户端，其目的是破坏全局模型在部分或所有数据上的性能，由于其攻击目标的局限性（如污染源数据实现标签翻转攻击、收集恶意梯度实现 LIE-B（little is enough-backdoor）攻击[26]），其梯度往往呈现出“偏离核心趋势”的特征，与良性梯度相比具有较为明显的差异，难以聚合为良性簇。因此，较小的簇代表恶意梯度。

为展示聚类效果，利用 t-SNE 技术[27]对 LIE-B 攻击下的客户端梯度进行降维表示，如图 2 所示，相同标签的节点（客户端）表示同一簇中的梯度，形状表示节点真实类型。根据图中聚类情况，K-means 聚类算法能够准确识别出恶意梯度和良性梯度。

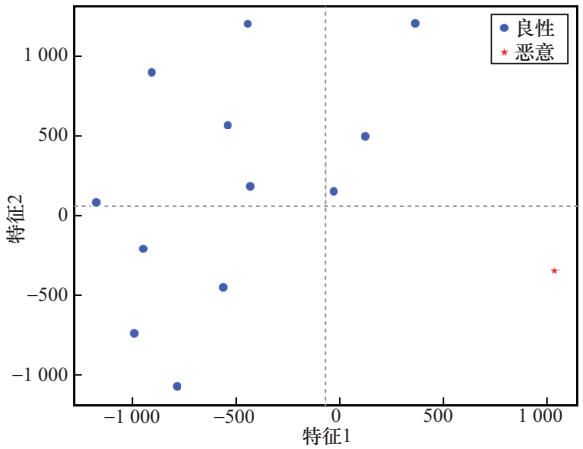


图 2　LIE-B 攻击下的客户端梯度降维表示

然而在攻击者未发动投毒攻击的轮次中，直接使用该结果会造成严重的误报，同时，聚类结果无法衡量恶意梯度与良性梯度的偏差程度。

为此，本文提出了结合 Transformer 架构的自编码器恶意梯度识别（MGI, malicious grads identification）方法，如算法 3 所示。自编码器是一个具有编码器–解码器结构的神经网络，利用 Transformer 架构提升网络表示能力。将聚类结果中较大簇中的梯度作为良性训练数据输入网络，首先由 Transformer 编码器将其嵌入低维向量空间，提取关键信息，减少噪声影响。接着利用 Transformer 解码器重构梯度，与原始梯度比较获得重构损失，表示为

$$\text{Loss}_{\text{re}}^t = \text{MSE}(\overline{g_{\text{his}}^t}, g_{\text{his}}^t) \tag{5}$$

通过缩小良性簇梯度的重构损失优化自编码器。

经过上述训练，自编码器学习到良性梯度的特征。将所有梯度输入网络，计算重构损失，基于本轮重构损失均值和标准差计算损失阈值，如式(6)所示。

$$\mathrm{TH}^t = \mathrm{Mean}(\mathrm{Loss}_{\mathrm{re}}^t) + \beta \mathrm{Std}(\mathrm{Loss}_{\mathrm{re}}^t) \tag{6}$$

其中，β为标准差权重，用于调节阈值对异常值的敏感度，β越大，阈值对异常值的容忍度越高，从而减少良性梯度被误判为恶意梯度的概率；β越小，阈值对异常值的容忍度越低，对恶意梯度的判断更加严格，从而减少恶意梯度被漏判的概率，超过损失阈值的为恶意梯度。另外，根据客户端重构损失与阈值的差值，可确定其梯度与正确方向的偏离程度。

为获得更好的防御效果，可设计自适应阈值。具体来说，当异常簇中节点数量较少（恶意客户端比例（PR, poisoning rate）较低）时，通过K-means聚类算法得到的良性簇“纯度”更高，其包含的良性梯度占比大，因此自编码器获得的训练数据（良性簇梯度）具有更强的一致性，其标准差$\mathrm{Std}(\mathrm{Loss}_{\mathrm{re}}^t)$较小，此时$\beta$应当较小，从而产生更严格的判断。当异常簇节点数量较多（PR较高）时，自编码器的训练数据产生的重构损失$\mathrm{Loss}_{\mathrm{re}}^t$可能会更加分散，此时需要适当增大$\beta$值，从而减少对良性梯度的误判。

算法3 恶意梯度识别MGI

输入 各客户端加权历史梯度g_{his}^t

输出 恶意客户端集mc，重构损失$\mathrm{Loss}_{\mathrm{re}}^t$，损失阈值$\mathrm{TH}^t$

```
b_grads, m_grads=K-means(g_his^t)
//基于良性梯度簇b_grads训练AE
AE(b_grads)
//重构所有梯度
ḡ_his^t =AE(g_his^t)
//计算重构损失和损失阈值
Loss_re^t = MSE(ḡ_his^t, g_his^t)
TH^t = Mean(Loss_re^t) + βStd(Loss_re^t)
for k = 1 to K do
    if Loss_re^{k,t} > TH^t then
        mc ← k
    end if
end for
return mc, Loss_re^t, TH^t
```

3.3 基于动态信任分数聚合

为增强攻击效果，提高中毒持久性，攻击者通常在模型接近收敛时发起多轮攻击。因此，应当根据客户端的投毒频次和力度进行惩罚，降低其权重系数。本文提出了一种基于动态信任分数的自适应聚合（AA, adaptive aggregation）方法，如算法4所示。具体来说，服务端维护信任分数和怀疑分数列表，当客户端在某轮中被判定为攻击者时，基于重构损失与损失阈值的差值增加其怀疑分数，降低信任分数，同时该客户端本轮权重置零。当客户端被判定为良性时，增加信任分数，并基于信任分数为客户端赋予相应权重。

$$\mathrm{SS}^t(k) = \mathrm{SS}^{t-1}(k) + (\mathrm{Loss}_{\mathrm{re}}^{k,t} - \mathrm{TH}^t) \tag{7}$$

$$\mathrm{TS}^t(k) = \begin{cases} \mathrm{TS}^{t-1}(k) - \mathrm{SS}^t(k), k \in \text{malicious} \\ \mathrm{TS}^{t-1}(k) + (\mathrm{TH}^t - \mathrm{Loss}_{\mathrm{re}}^{k,t}), k \in \text{benign} \end{cases} \tag{8}$$

$$W^t(k) = \max(\min(\mathrm{TS}^t(k),1),0) \tag{9}$$

最后，对各客户端权重归一化，聚合更新获得全局模型。归一化权重为

$$W_{\mathrm{norm}}^t(k) = \frac{W^t(k)}{\sum W^t(k)} \tag{10}$$

算法4 自适应聚合AA

输入 恶意客户端集mc，重构损失$\mathrm{Loss}_{\mathrm{re}}^t$，损失阈值$\mathrm{TH}^t$，怀疑分数$\mathrm{SS}^t$，信任分数$\mathrm{TS}^t$

输出 归一化权重W_{norm}^t

```
for k = 1 to K do
    if k ∈ mc then
    //计算损失偏差，增加怀疑分数，减少信任分数
        SS^t(k) = SS^{t-1}(k) + (Loss_re^{k,t} - TH^t)
        TS^t(k) = TS^{t-1}(k) - SS^t(k)
    else
        //增加信任分数
        TS^t(k) = TS^{t-1}(k) + (TH^t - Loss_re^{k,t})
    end if
    W^t(k) = max(min(TS^t(k),1),0)
end for
//权重归一化
W_norm^t(k) = W^t(k) / ΣW^t(k)
return W_norm^t
```

4　实验

4.1　实验设置

4.1.1　数据集

本文实验采用联邦学习常用基准数据集，分别为 MNIST[28]、FMNIST[29]、CIFAR-10[30] 和 CIFAR-100[30]。

MNIST 是拥有 70 000 张手写数字图像的灰度数据集，包含 0~9 共 10 种类别。每种类别有 6 000 张训练数据和 1 000 张测试数据，图片大小为 28×28。

FMNIST 是拥有 70 000 张衣物图像的灰度数据集，包含“Shirt”等共 10 种类别，与 MNIST 类似，每种类别有 6 000 张训练数据和 1 000 张测试数据，图像大小为 28×28。

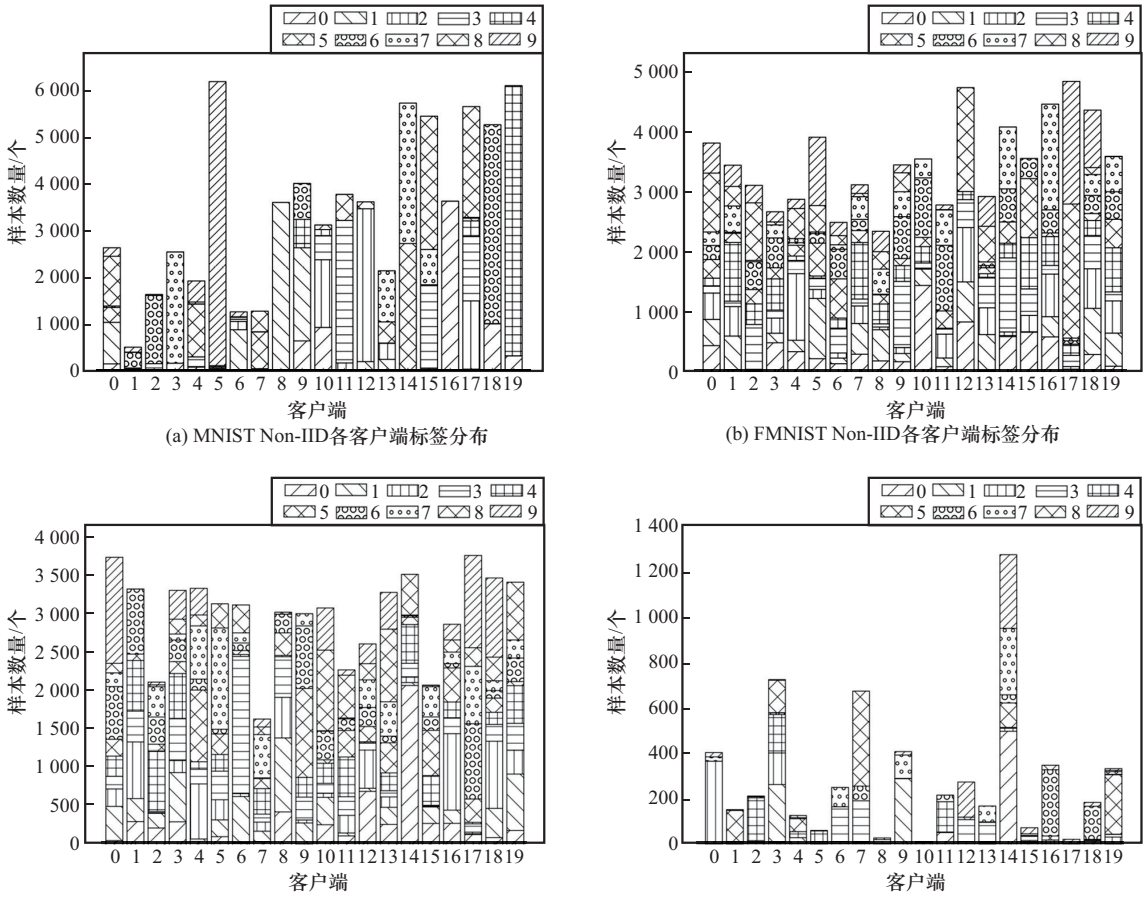
CIFAR-10 是拥有 60 000 张普适物体图像的彩色数据集，包含“deer”等共 10 种类别。每种类别有 5 000 张训练数据和 1 000 张测试数据，图像大小为 3×28×28。

CIFAR-100 是拥有 60 000 张普适物体图像的彩色数据集，包含“公共汽车”等 100 种类别，每种类别有 500 张训练数据和 100 张测试数据，图像大小为 3×28×28。

4.1.2　模型和超参数设置

HFLGuard 基于 Pytorch 框架实现，在上述基准数据集的不同分布上展开广泛实验，所有实验在 NVIDIA RTX 3090 GPU 上进行。根据文献[21]，为每个数据集分别设定 IID 场景和 Non-IID 场景，以模拟真实环境下防御方法的性能。Non-IID 场景基于狄利克雷分布进行模拟，各客户端数据分布如图 3 所示，为简化表示，CIFAR-100 取前 10 类样本。

联邦学习架构中包含 20 个客户端，MNIST、FMNIST、CIFAR-10 和 CIFAR-100 均采用两层卷积神经网络（CNN, convolutional neural network）架构，全局训练轮次分别为 250、300、400 和 400 轮，本地迭代次数分别为 1、2、2 和 2 次，学习率均为 0.005。对于自编码器，编码器和解码器均采用 8 头 3 层的 Transformer 结构，隐藏层维度分别为 128、



(a) MNIST Non-IID各客户端标签分布　(b) FMNIST Non-IID各客户端标签分布

(c) CIFAR-10 Non-IID各客户端标签分布　(d) CIFAR-100 Non-IID各客户端标签分布

图3　Non-IID场景下各客户端数据分布情况

128、256和256，训练轮次为200轮，学习率设置为0.001。

4.1.3 攻击方法和基线设置

本文假设攻击者能够执行2种攻击方法，分别为标签翻转目标攻击LF和非目标攻击LIE-B[26]。攻击者数量占比0~0.4，即联邦学习系统中存在0、4和8个攻击者。

在LF中，为最大化攻击效果，本文选择相似标签进行翻转。为降低被发现的概率，攻击者在训练初始阶段不进行投毒，而是表现得和正常客户端一致，直到80轮后实施攻击。

在LIE-B中，攻击者相互勾连，通过在恶意客户端上传梯度中添加轻微扰动，使现有防御方法将良性客户端识别为异常值。

本文采用FedAvg[5]、Krum[19]、Trimmed Mean（TMean）[20]、FL-Defender[21]和MUD-HoG[22]防御方法作为基线设置，对比HFLGuard防御效果。

4.1.4 评价指标

对于目标攻击，本文主要设置了如下3个评价指标。

1) 主任务精度（ACC, accuracy）：预测正确的样本数除以样本总数，表示模型的整体性能。

2) 源类精度（S_ACC, source accuracy）：预测正确的源类样本数除以源类样本总数，表示模型在源类样本上的预测效果。

3) 攻击成功率（ASR）：被预测为目标类的源类样本数除以源类样本总数，表示目标攻击模型的攻击效果。

对于非目标攻击，由于其目的是破坏模型整体性能，因此采用“主任务精度”作为评价指标。有效的防御方法应在保持模型性能的前提下降低攻击成功率。

4.2 性能评价

4.2.1 针对LF攻击的鲁棒性

如表1~表8所示，FedAvg在面对标签翻转攻击时，源类准确率均有不同程度的下降，攻击成功率随着PR的增加而提升。由于攻击仅针对特定标签，因此模型的整体性能变化相对较小。同时，攻击改变了模型决策边界，当原始模型在源类和目标类样本上“波动”时，中毒模型更倾向于将其预测为目标类。表中加粗和加下划线数据分别表示最优结果和次优结果。

表1和表2展示了MNIST中，各防御方法均能够在保持模型整体性能的同时抵抗标签翻转攻击，原因在于该数据集中图像较为简单，即使部分图像标签被翻转，模型仍能够从大量正常标注的图像中学习到正确的特征。

表3和表4展示了FMNIST中，各防御方法在主任务性能上均能够取得与FedAvg相似的效果。然而，随着攻击者数量的增加，只有Krum和HFLGuard能够在IID和Non-IID场景下保持较好的性能。TMean源类精度在高攻击占比（PR=0.4）时分别降至39.50%和28.94%，仅略高于无防御方法FedAvg的性能。原因在于攻击比例较高时，恶意梯度的部分维度将占据主体地位，TMean剪枝仅能消除部分恶意参数，剪枝后的结果仍由攻击者主导，攻击成功率分别达到51.89%和60.27%，进一步验证了这一分析。FL-Defender在Non-IID场景下性能大幅下降，在PR=0.2和PR=0.4时源类精度只有64.55%和46.86%，原因在于该方法依赖余弦相似

表1 MNIST IID场景下不同防御方法针对LF的性能

PR	指标	FedAvg	Krum	TMean	FL-Defender	MUD-HoG	HFLGuard
0	ACC	**98.45%**	97.16%	98.38%	98.41%	98.42%	98.31%
	S_ACC	99.34%	98.71%	99.30%	99.30%	99.35%	**99.40%**
	ASR	**0.10%**	**0.10%**	0.15%	0.15%	0.15%	0.15%
0.2	ACC	98.37%	96.40%	**98.43%**	98.38%	98.42%	98.42%
	S_ACC	98.31%	98.01%	98.96%	**99.35%**	99.30%	99.30%
	ASR	0.60%	0.15%	0.35%	**0.10%**	0.15%	0.15%
0.4	ACC	97.56%	96.93%	98.05%	98.33%	98.31%	**98.34%**
	S_ACC	91.31%	98.21%	95.08%	**99.35%**	99.26%	99.26%
	ASR	6.95%	0.20%	3.18%	**0.10%**	**0.10%**	**0.10%**

表 2　MNIST Non-IID 场景下不同防御方法针对 LF 的性能

PR	指标	FedAvg	Krum	TMean	FL-Defender	MUD-HoG	HFLGuard
0	ACC	98.13%	95.46%	97.96%	98.13%	98.17%	**98.19%**
	S_ACC	99.54%	98.05%	**99.64%**	99.33%	99.59%	99.54%
	ASR	**0.10%**	0.21%	**0.10%**	**0.10%**	**0.10%**	**0.10%**
0.2	ACC	**98.14%**	95.13%	98.02%	97.95%	98.13%	98.13%
	S_ACC	98.97%	98.10%	99.28%	99.54%	**99.64%**	99.38%
	ASR	0.31%	0.26%	0.10%	**0.05%**	0.10%	0.10%
0.4	ACC	97.61%	94.88%	97.60%	**98.14%**	98.06%	98.09%
	S_ACC	94.76%	98.15%	95.22%	**99.59%**	**99.59%**	98.87%
	ASR	3.90%	0.15%	3.44%	**0.10%**	**0.10%**	0.36%

表 3　FMNIST IID 场景下不同防御方法针对 LF 的性能

PR	指标	FedAvg	Krum	TMean	FL-Defender	MUD-HoG	HFLGuard
0	ACC	88.79%	87.45%	**88.80%**	88.58%	88.63%	88.39%
	S_ACC	87.34%	85.88%	87.11%	86.44%	87.64%	**88.46%**
	ASR	6.58%	8.78%	6.98%	8.16%	6.08%	**5.85%**
0.2	ACC	87.78%	87.47%	88.37%	88.55%	**88.64%**	88.62%
	S_ACC	66.52%	85.37%	76.42%	86.21%	**86.94%**	86.89%
	ASR	25.10%	9.62%	16.83%	7.77%	**7.37%**	7.43%
0.4	ACC	84.29%	86.37%	85.67%	88.41%	88.36%	**88.49%**
	S_ACC	29.68%	83.06%	39.50%	85.88%	86.72%	**87.45%**
	ASR	61.23%	10.41%	51.89%	7.93%	7.09%	**6.58%**

表 4　FMNIST Non-IID 场景下不同防御方法针对 LF 的性能

PR	指标	FedAvg	Krum	TMean	FL-Defender	MUD-HoG	HFLGuard
0	ACC	**87.93%**	85.60%	87.12%	87.58%	87.40%	87.33%
	S_ACC	**86.09%**	81.58%	82.03%	82.97%	77.85%	84.86%
	ASR	**7.51%**	13.08%	9.74%	10.07%	12.63%	8.83%
0.2	ACC	86.64%	85.09%	86.71%	86.48%	84.23%	**87.02%**
	S_ACC	67.56%	75.13%	73.57%	64.55%	42.96%	**83.42%**
	ASR	23.26%	18.36%	17.64%	27.10%	44.85%	**8.68%**
0.4	ACC	82.82%	81.17%	83.43%	84.85%	82.97%	**86.22%**
	S_ACC	24.60%	58.88%	28.94%	46.86%	27.05%	**64.83%**
	ASR	64.83%	31.72%	60.27%	46.24%	67.27%	**23.98%**

度识别恶意梯度。然而在 Non-IID 场景下，良性梯度之间也具有较大偏差。MUD-HoG 同样在该场景下失效，同时本文在实验中观察到，在 PR=0.4 时，源类精度出现剧烈波动，降至 27.05%。原因在于，MUD-HoG 基于 K-means 聚类算法将梯度分为两类，然而该方法对聚类中心的选择很敏感，当攻击者数量较多时，随机选择的质心很有可能落在攻击者上，从而造成错误的结果。而本文方法对每个客户端维护怀疑分数，即使在某轮中攻击者躲过了聚类检测，由于长期恶意行为，其更新权重依然为 0。

因此，即使攻击强度较高，HFLGuard仍能保持最高的源类精度，证明其对LF的鲁棒性。

表5和表6展示了CIFAR-10中各防御方法的防御效果，由于该数据集包含各类物体的三通道彩色图像，主任务难度提升。观察表中数据可知，Krum方法使整体性能出现大幅下降，原因在于其仅选择单个客户端，难以适应困难任务，这一现象在Non-IID场景下表现得更加明显，在无攻击者参与时，Kurm方法的主任务精度仅为45.26%，远低于FedAvg方法。其他方法在主任务精度上能够保持良好的性能，然而，与FMNIST相同，TMean随着攻击比例提高逐渐失效，在PR=0.4时源类精度降至11.29%和9.28%。FL-Defender在IID场景下性能表现良好，而在Non-IID场景下源类精度分别下降至21.15%和35.61%，攻击成功率分别升至54.64%和34.62%。MUD-HoG同样面临质心敏感和防御效果剧烈波动的问题，在PR=0.4的Non-IID场景下源类精度仅有29.18%，攻击成功率达到40.58%。相比之下，HFLGuard表现稳定，即使在PR=0.4时源类精度也能达到60.25%和44.30%。

表7和表8展示了CIFAR-100中各防御方法的防御效果。该数据集共包含100种类别，且各类别仅有500个训练样本，任务难度较之CIFAR-10进一步提高。根据表中数据，只有HFLGuard能够抵御不同程度的攻击，即使在PR=0.4时也能保持35.10%和29.75%的源类精度，远高于其他防御方法。

从实验结果来看，在表2中，当PR=0时，HFLGuard与FedAvg源类精度均为99.54%，攻击成功率均为0.10%，说明几乎无良性梯度被误判为恶意梯度；当攻击比例增加时，HFLGuard均能大幅降低攻击成功率，说明对恶意梯度漏判较少。在表4中，当PR=0时，HFLGuard相较于其余防御方法，其源类精度与FedAvg的86.09%最为接近，达到了84.86%，说明其误判率较低；随着攻击比例的增加，HFLGuard源类精度显著高于其余防御方

表5　CIFAR-10 IID场景下不同防御方法针对LF的性能

PR	指标	FedAvg	Krum	TMean	FL-Defender	MUD-HoG	HFLGuard
0	ACC	**71.58%**	55.65%	71.37%	68.33%	70.73%	69.43%
	S_ACC	63.45%	46.15%	62.53%	55.68%	**63.64%**	60.84%
	ASR	16.71%	19.19%	16.32%	18.02%	**15.40%**	17.95%
0.2	ACC	**70.51%**	54.07%	70.26%	68.35%	70.20%	70.47%
	S_ACC	38.12%	46.28%	37.34%	58.55%	60.31%	**61.88%**
	ASR	37.08%	20.10%	35.97%	18.60%	16.78%	**15.21%**
0.4	ACC	68.89%	55.22%	68.15%	68.21%	68.55%	**68.91%**
	S_ACC	16.84%	0%	11.29%	57.70%	57.83%	**60.25%**
	ASR	55.61%	67.75%	60.97%	17.10%	17.62%	**14.95%**

表6　CIFAR-10 Non-IID场景下不同防御方法针对LF的性能

PR	指标	FedAvg	Krum	TMean	FL-Defender	MUD-HoG	HFLGuard
0	ACC	**69.13%**	45.26%	68.63%	66.75%	68.46%	68.66%
	S_ACC	63.46%	38.40%	62.86%	53.32%	65.12%	**66.98%**
	ASR	17.44%	**11.27%**	17.51%	22.48%	12.60%	11.47%
0.2	ACC	**68.52%**	45.88%	66.43%	63.86%	68.04%	68.19%
	S_ACC	40.19%	2.59%	39.32%	21.15%	60.81%	**63.06%**
	ASR	35.94%	46.62%	36.47%	54.64%	16.11%	**15.58%**
0.4	ACC	66.40%	44.81%	64.30%	65.47%	**66.45%**	66.10%
	S_ACC	13.53%	1.86%	9.28%	35.61%	29.18%	**44.30%**
	ASR	63.06%	49.87%	69.50%	34.62%	40.58%	**28.58%**

表 7　CIFAR-100 IID 场景下不同防御方法针对 LF 的性能

PR	指标	FedAvg	Krum	TMean	FL-Defender	MUD-HoG	HFLGuard
0	ACC	**33.44%**	14.95%	32.87%	27.59%	33.16%	31.46%
	S_ACC	45.03%	23.18%	**47.68%**	37.75%	43.05%	41.72%
	ASR	**6.62%**	13.25%	7.28%	7.78%	7.56%	7.95%
0.2	ACC	**33.34%**	15.95%	32.89%	27.04%	33.09%	31.72%
	S_ACC	31.79%	17.22%	29.80%	20.53%	26.49%	**43.05%**
	ASR	11.92%	12.58%	13.91%	11.26%	13.91%	**4.64%**
0.4	ACC	**33.27%**	15.25%	32.90%	27.23%	**33.12%**	30.37%
	S_ACC	21.19%	0%	17.88%	24.50%	19.87%	**35.10%**
	ASR	21.85%	25.83%	26.49%	14.57%	22.52%	**6.62%**

表 8　CIFAR-100 Non-IID 场景下不同防御方法针对 LF 的性能

PR	指标	FedAvg	Krum	TMean	FL-Defender	MUD-HoG	HFLGuard
0	ACC	**30.12%**	11.27%	27.65%	23.82%	29.73%	29.48%
	S_ACC	**41.77%**	12.32%	25.32%	39.24%	34.81%	41.14%
	ASR	5.70%	18.84%	7.59%	5.06%	6.96%	**4.43%**
0.2	ACC	**29.96%**	11.86%	27.63%	24.13%	29.56%	28.76%
	S_ACC	35.44%	2.17%	21.52%	10.76%	29.11%	**39.87%**
	ASR	8.23%	7.97%	9.49%	13.29%	7.59%	**4.16%**
0.4	ACC	**29.80%**	8.67%	27.43%	23.55%	29.38%	28.58%
	S_ACC	20.25%	0%	6.96%	10.13%	24.68%	**29.75%**
	ASR	23.42%	20.29%	28.48%	32.91%	12.66%	**8.86%**

法，使攻击成功率分别降低 14.58% 和 40.85%。在表 6 中，当 PR=0 时，HFLGuard 将源类精度提升至 66.98%，说明其在一定程度上缓解了异构对源类精度带来的负面影响；当中毒客户端数量增加时，其余方法虽精度下降相对较小，但源类精度大幅下降，说明对恶意梯度存在大量漏判，而 HFLGuard 在保持良好性能的同时使攻击成功率分别降低 20.36% 和 34.48%。在表 8 中，当 PR=0 时，HFLGuard 源类精度获得了次优的性能 41.14%，仅低于 FedAvg 源类精度的 41.77%，说明即使在困难数据集 CIFAR-100 上，HFLGuard 也不会过度误判良性梯度；当攻击比例提高时，HFLGuard 均获得最优的防御方法，分别将攻击成功率限制在 5% 和 10% 以内，且源类精度仅下降约 1.5%。

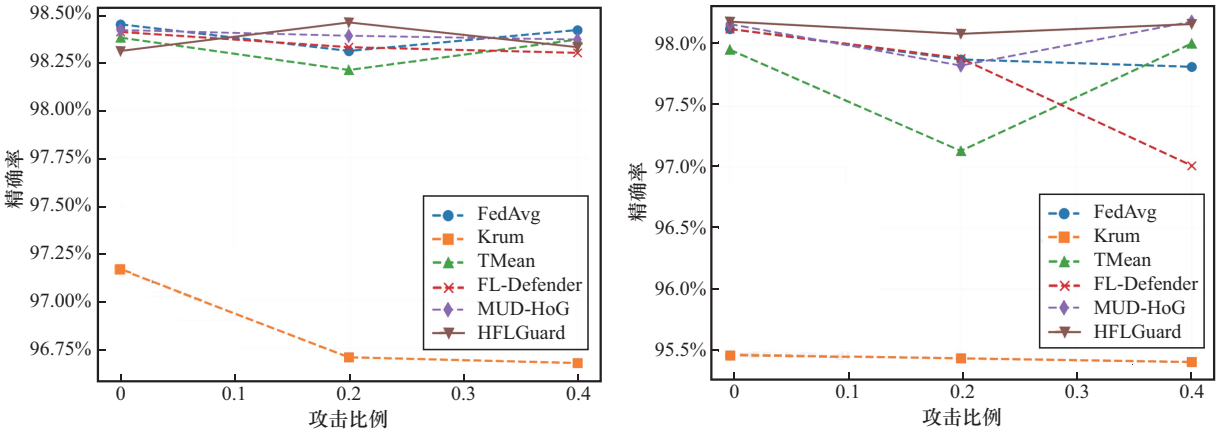
4.2.2　针对 LIE-B 攻击的鲁棒性

由于 LIE-B 为非目标攻击，仅考察模型整体性能指标。

图 4 展示了 MNIST 中各防御方法针对 LIE-B 攻击时的鲁棒性，观察图中数据可得，LIE-B 攻击对 MNIST 影响较小，除 Krum 外，其余防御方法均能保持良好的性能，而 Krum 则由于选择到中毒客户端性能明显下降。

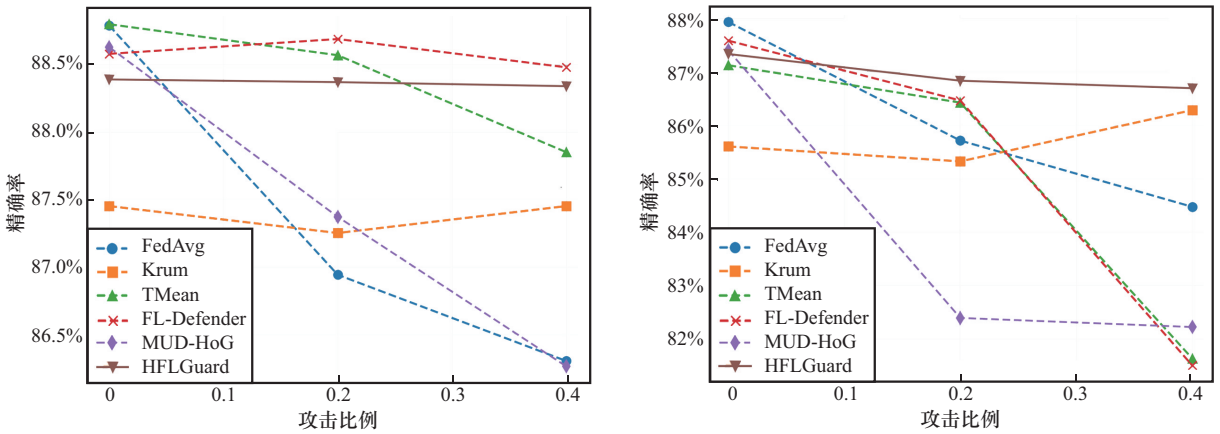
图 5 展示了 FMNIST 中各防御方法针对 LIE-B 攻击时的鲁棒性，观察图中数据可得，由于 FMNIST 数据相对简单，在 IID 场景下攻击影响较小，各方法均能保持良好的性能。在 Non-IID 场景下，当 PR=0.4 时，TMean、FL-Defender 和 MUD-HoG 方法精确率分别降至 81.63%、81.50% 和 82.22%，而 Krum 和 HFLGuard 能成功抵御攻击。

图 6 展示了 CIFAR-10 中各防御方法的防御效果。在 IID 和 Non-IID 场景下各防御方法整体性能均有不同程度的下降，Krum 和 TMean 方法完全失效，性能均低于 FedAvg。MUD-HoG 在高攻击比例时性能骤降至 56.63% 和 56.50%，略高于 FedAvg。



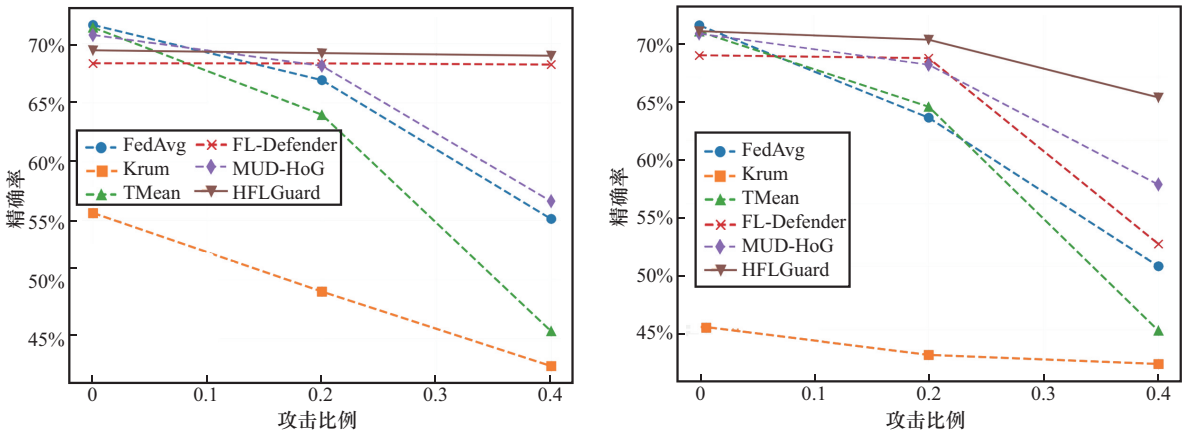
(a) 各防御方法在MNIST IID上的性能　　(b) 各防御方法在MNIST Non-IID上的性能

图4　各防御方法在MNIST上的性能



(a) 各防御方法在FMNIST IID上的性能　　(b) 各防御方法在FMNIST Non-IID上的性能

图5　各防御方法在FMNIST上的性能

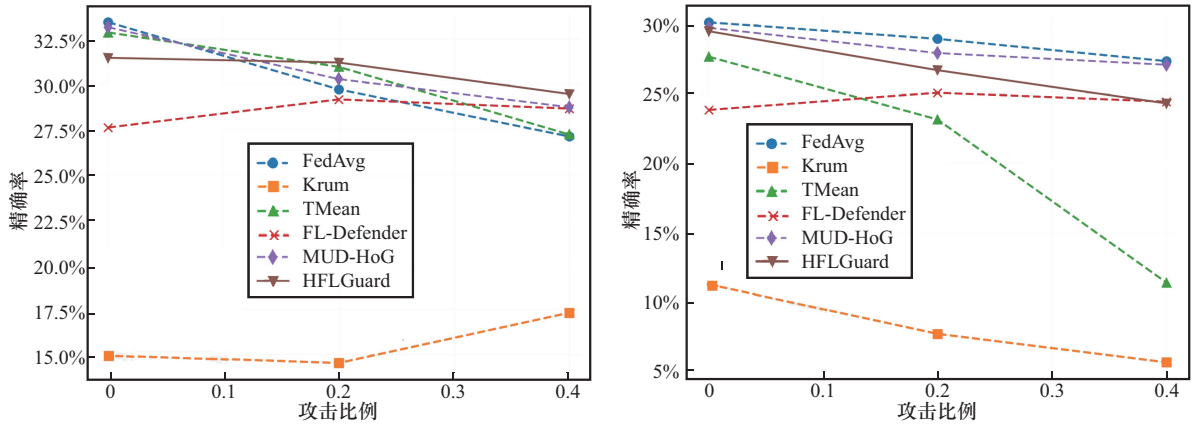


(a) 各防御方法在CIFAR-10 IID上的性能　　(b) 各防御方法在CIFAR-10 Non-IID上的性能

图6　各防御方法在CIFAR-10上的性能

FL-Defender虽然在IID场景下性能较为平稳，但在Non-IID场景下难以抵御高强度攻击。相反，HFL-Guard能够有效抵御LIE-B攻击，即使在PR=0.4时仍能保持68.97%和63.41%的主任务精度。

图7展示了CIFAR-100中不同防御方法的性能。CIFAR-100任务的复杂性增加了防御LIE-B攻击的难度，Krum和TMean方法完全失效。在Non-IID场景下，数据分布不平衡进一步给各防御方法



(a) 各防御方法在CIFAR-100 IID上的性能　　(b) 各防御方法在CIFAR-100 Non-IID上的性能

图7　各防御方法在CIFAR-100上的性能

造成了困难，FL-Defender在无攻击者时使精确率下降约6%。实验证明，HFLGuard能够在各种条件下准确识别出投毒客户端，然而，在有攻击者存在的条件下，多数方法精确率略低于FedAvg，原因在于CIFAR-100本身复杂度高，各客户端数据量小，相比之下，排除恶意客户端带来的影响比攻击本身的影响更大。

4.2.3　HFLGuard稳定性分析

在目标攻击中，主要通过主任务精度ACC、源类精度S_ACC和攻击成功率ASR衡量HFLGuard在不同攻击强度下的表现。由表1~表8可以看出，随着攻击比例的增加，HFLGuard的主任务精度均能够维持在较高水平，而基线方法Krum和TMean对主任务精度影响较大。同时，S_ACC下降的幅度较小，攻击成功率增长缓慢，说明HFLGuard能够准确识别出恶意梯度，阻止其参与聚合，降低恶意影响，而其余方法的防御效果随着攻击比例的提升均有不同程度的下降。

在非目标攻击中，主要通过主任务精度分析防御方法在不同攻击强度下的表现。由图4~图7可以看出，随着攻击比例的增加，在所有数据集上，HFLGuard的主任务精度均呈现出缓慢下降的趋势，如在CIFAR-10中，ACC从68.66%下降至63.41%（下降5.23%），而其余方法降至56.50%以下。

综上，HFLGuard在面对不同强度的攻击时具有鲁棒性和稳定性，主要原因在于加权历史梯度能够准确反映恶意客户端的长期目标；自编码器能够通过重构损失捕获良性梯度与恶意梯度之间的偏差；动态信任分数增大了恶意客户端的惩罚力度，即使攻击者通过暂时伪装成良性客户端规避当前轮次的检测，长期累积的信任分数也会阻止其参与模型聚合。

4.3　时间衰减因子分析

为验证时间衰减因子的有效性与鲁棒性，本文设计了不同衰减策略，并将攻击强度设置为最高的PR=0.4，恶意客户端早期输出伪装的良性梯度，结果如表9所示，其中Average表示为所有轮次的梯度赋予相同权重。根据表9中结果，本文提出的时序衰减策略在各数据集上都取得了较好的防御效果，在抵御攻击的同时能维持模型精度。当$\alpha_t = 0$，即不使用历史梯度时，防御效果较差，说明历史梯度中包含的信息能够提高防御效果。当$\alpha_t = 0.7$和等权重时，历史梯度衰减过慢，恶意客户端刻意伪造的信息对后续防御产生的影响过大，干扰了自编码器的学习过程，导致其在复杂任务上的攻击识别准确率下降。

4.4　大规模应用分析

为量化训练轮数增加对模型性能的影响，在CIFAR-10 Non-IID场景下，设定PR=0.4，改变训练轮次观察HFLGuard性能变化。如表10所示，随着训练轮次增加，主任务精度仅出现较小波动，且1 000轮时已完全收敛，与2 000轮主任务时精度相同。同时也表明HFLGuard通过动态信任分数长期跟踪客户端行为，避免恶意客户端规避防御机制。

为量化客户端数量增加对模型性能和计算开销的影响，在CIFAR-10 Non-IID场景下，设定PR=0.4，改变客户端数量观察HFLGuard的性能变化。如表11所示，随着客户端数量的增加，主任务精度呈现出轻微的下降趋势，说明HFLGuard能够在客户端数量增加时保持稳定的防御效果。同时，HFLGuard中的聚类时间、自编码器训练时间和推理时间缓慢增长，证明了其在大规模场景下的应用潜力。

表 9 不同衰减因子对防御性能的影响

数据集	指标	$\alpha_t = 0$	$\alpha_t = 0.3$	$\alpha_t = 0.5$	$\alpha_t = 0.7$	Average	$\frac{1}{t - t_d + 1}$
MNIST	ACC	98.47%	98.29%	98.32%	98.32%	98.42%	**98.48%**
	S_ACC	98.44%	98.34%	98.59%	**98.69%**	98.24%	98.59%
	ASR	0.45%	0.45%	**0.35%**	**0.35%**	0.60%	0.45%
FMNIST	ACC	90.99%	90.95%	91.00%	**91.04%**	90.97%	**91.04%**
	S_ACC	84.14%	84.97%	**85.64%**	84.31%	84.64%	85.48%
	ASR	11.19%	10.85%	10.41%	11.41%	11.07%	**10.35%**
CIFAR-10	ACC	69.11%	69.14%	**69.18%**	69.05%	68.96%	69.16%
	S_ACC	60.68%	60.74%	60.74%	**61.14%**	52.98%	60.68%
	ASR	17.51%	17.37%	16.98%	**16.38%**	22.75%	17.77%
CIFAR-100	ACC	31.34%	31.42%	31.32%	31.33%	**31.45%**	31.25%
	S_ACC	37.34%	38.61%	37.34%	22.15%	36.71%	**41.77%**
	ASR	6.83%	6.22%	6.76%	10.45%	7.15%	**5.06%**

表 10 训练轮次对模型性能的影响

训练轮次/轮	ACC
400	63.41%
1 000	65.23%
2 000	65.23%

表 11 客户端数量对模型性能和计算开销的影响

客户端数量/个	ACC	聚类时间/s	自编码器训练时间/s	推理时间/s
20	63.41%	0.32	2.42	0.003 5
40	62.84%	0.35	2.42	0.003 7
60	62.10%	0.41	2.43	0.004 1
80	62.66%	0.52	2.44	0.004 4
100	62.01%	0.53	2.67	0.005 0

5 结束语

本文提出了一种异构联邦中毒防御方法，旨在防御多源异构的联邦学习场景下的中毒攻击。为识别长期“目标”，计算客户端的加权历史梯度。基于 K-means 聚类算法进行初步分类，利用良性梯度训练 Transformer 自编码器，重构客户端梯度，去除冗余特征，从而识别恶意客户端。基于动态信任分数惩罚长期投毒的恶意客户端，为其赋予较低的权重。实验结果表明，本文方法能够抵御标签翻转攻击和 LIE-B 攻击，保护了模型的准确性和完整性。未来将进一步研究梯度加密情况下的联邦中毒防御方法，以提高系统的隐私性，避免隐私泄露。

参考文献：

[1] XIAO Q F, PAN X D, LU Y F, et al. Exorcising “Wraith”: protecting LiDAR-based object detector in automated driving system from appearing attacks[J]. arXiv Preprint, arXiv: 2303.09731, 2023.

[2] HUANG K, XIAN R D, XIAN M, et al. A comprehensive intrusion detection method for the Internet of vehicles based on federated learning architecture[J]. Computers & Security, 2024, 147: 104067.

[3] WANG L Y, ZHOU H L, BAO Y W, et al. Horizontal federated recommender system: a survey[J]. ACM Computing Surveys, 2024, 56(9): 1-42.

[4] WANG Y Z, WANG H Z, LIU X L, et al. GFedKG: GNN-based federated embedding model for knowledge graph completion[J]. Knowledge-Based Systems, 2024, 301: 112290.

[5] MCMAHAN H B, MOORE E, RAMAGE D, et al. Communication-efficient learning of deep networks from decentralized data[J]. arXiv Preprint, arXiv: 1602.05629, 2016.

[6] NGUYEN T D, RIEGER P, MIETTINEN M, et al. Poisoning attacks on federated learning-based IoT intrusion detection system[C]//Proceedings 2020 Workshop on Decentralized IoT Systems and Security. Virginia: The Internet Society, 2020: 23-26.

[7] WANG S, LI Q M, CUI Z Y, et al. Bandit-based data poisoning attack against federated learning for autonomous driving models[J]. Expert Systems with Applications, 2023, 227: 120295.

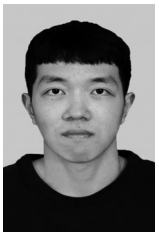
[8] KASYAP H, TRIPATHY S. Sine: similarity is not enough for mitigating local model poisoning attacks in federated learning[J]. IEEE Transactions on Dependable and Secure Computing, 2024, 21(5): 4481-4494.

[9] SAMEERA K M, VINOD P, RAFIDHA R K A, et al. LFGurad: a defense against label flipping attack in federated learning for vehicular network[J]. Computer Networks, 2024, 254: 110768.

[10] LAI Y C, LIN J Y, LIN Y D, et al. Two-phase defense against poisoning attacks on federated learning-based intrusion detection[J]. Computers & Security, 2023, 129: 103205.

[11] SAHU A K, LI T, SANJABI M, et al. Federated optimization in heterogeneous networks[J]. arXiv Preprint, arXiv: 1812.06127, 2018.
[12] REDDI S, CHARLES Z, ZAHEER M, et al. Adaptive federated optimization[J]. arXiv Preprint, arXiv: 2003.00295, 2020.
[13] ACAR D A E, ZHAO Y, NAVARRO R M, et al. Federated learning based on dynamic regularization[J]. arXiv Preprint, arXiv: 2111.04263, 2021.
[14] YAN B J, LIU B Y, WANG L J, et al. FedCM: a real-time contribution measurement method for participants in federated learning[C]//Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN). Piscataway: IEEE Press, 2021: 1-8.
[15] YANG M, CHENG H, CHEN F, et al. Model poisoning attack in differential privacy-based federated learning[J]. Information Sciences, 2023, 630: 158-172.
[16] YU Y, LIU Q, WU L K, et al. Untargeted attack against federated recommendation systems via poisonous item embeddings and the defense[C]//Proceedings of the AAAI Conference on Artificial Intelligence. Palo Alto: AAAI Press, 2023: 4854-4863.
[17] KASYAP H, TRIPATHY S. Beyond data poisoning in federated learning[J]. Expert Systems with Applications, 2024, 235: 121192.
[18] XIAO X, TANG Z, LI C Y, et al. SCA: sybil-based collusion attacks of IIoT data poisoning in federated learning[J]. IEEE Transactions on Industrial Informatics, 2022, 19(3): 2608-2618.
[19] BLANCHARD P, MHAMDI E M E, GUERRAOUI R, et al. Machine learning with adversaries: Byzantine tolerant gradient descent[C]//Proceedings of the Neural Information Processing Systems. Massachusetts: MIT Press, 2017: 118-128.
[20] YIN D, CHEN Y D, RAMCHANDRAN K, et al. Byzantine-robust distributed learning: towards optimal statistical rates[C]//International Conference on Machine Learning. New York: ACM Press, 2018: 5650-5659.
[21] JEBREEL N M, DOMINGO-FERRER J. FL-Defender: combating targeted attacks in federated learning[J]. Knowledge-Based Systems, 2023, 260: 110178.
[22] GUPTA A, LUO T, NGO M V, et al. Long-short history of gradients is all you need: detecting malicious and unreliable clients in federated learning[C]//Computer Security-ESORICS 2022. Berlin: Springer, 2022: 445-465.
[23] NGUYEN T D, RIEGER P, CHEN H, et al. {FLAME}: taming backdoors in federated learning[C]//31st USENIX Security Symposium. Berkeley: USENIX Association, 2022: 1415-1432.
[24] ZHOU X J, CHEN X Z, LIU S K, et al. FLGuardian: defending against model poisoning attacks via fine-grained detection in federated learning[J]. IEEE Transactions on Information Forensics and Security, 2025, 20: 5396-5410.
[25] LIU T, ZHANG Y H, FENG Z, et al. Beyond traditional threats: a persistent backdoor attack on federated learning[C]//Proceedings of the AAAI Conference on Artificial Intelligence. Palo Alto: AAAI Press, 2024: 21359-21367.
[26] BARUCH M, BARUCH G, GOLDBERG Y. A little is enough: circumventing defenses for distributed learning[C]//Advances in Neural Information Processing Systems. Massachusetts: MIT Press, 2019: 8632-8642.
[27] MAATEN L V D, HINTON G. Visualizing data using t-SNE[J]. Journal of Machine Learning Research, 2008, 9(2605): 2579-2605.
[28] LECUN Y, BOTTOU L, BENGIO Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324.
[29] XIAO H, RASUL K, VOLLGRAF R. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms[J]. arXiv Preprint, arXiv: 1708.07747, 2017.
[30] KRIZHEVSKY A. Learning multiple layers of features from tiny images[J]. Handbook of Systemic Autoimmune Diseases, 2009, 1(4): 1-60.

[作者简介]

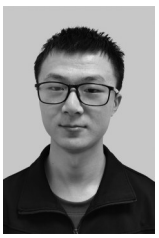


李勇飞（1998– ），男，河南开封人，信息工程大学博士生，主要研究方向为联邦学习、网络安全、人工智能安全等。

方晨（1993– ），男，安徽宿松人，信息工程大学讲师，主要研究方向为机器学习、隐私安全。

常朝稳（1966– ），男，河南滑县人，信息工程大学教授，主要研究方向为大数据与信息安全。

郭渊博（1975– ），男，陕西周至人，博士，海南大学教授、博士生导师，主要研究方向为网络防御、数据挖掘、机器学习、人工智能安全等。



付春辉（1999– ），男，山东潍坊人，信息工程大学博士生，主要研究方向为智能化流量分类。