

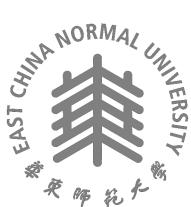
2022 届硕士专业学位研究生学位论文

分类号: _____

学校代码: 10269

密 级: _____

学 号: 51194501126



東華師範大學

East China Normal University

硕士专业学位论文

MASTER'S DISSERTATION (Professional)

论文题目：联邦学习中差分隐私和安全混淆的研究

院 系: 软件工程学院

专业学位类别: 工程硕士

专业学位领域: 软件工程

论文指导教师: 曹珍富 教授

论文作者: 何慧娴

2021 年 09 月 10 日

Thesis (Professional) for Master's Degree in 2022

School Code: 10269

Student Number:51194501126

EAST CHINA NORMAL UNIVERSITY

TITLE: A STUDY OF DIFFERENTIAL PRIVACY AND SECURE SHUFFLE IN FEDERATED LEARNING

Department: Software Engineering Institute

Major: Software Engineering

Research Direction: Cryptography and Network Security

Supervisor: Professor ZhenFu Cao

Candidate: HuiXian He

Nov 9, 2021

摘 要

随着数据孤岛的出现和隐私意识的普及，联邦学习作为一种新兴的数据共享和交换模型在金融、医疗、教育等诸多领域得到广泛应用。联邦学习的基本框架包含多个本地设备和一个中央服务器，所有训练数据保存在本地设备，所有设备共同协作训练一个全局模型。联邦学习的一个突出优点是它可以在服务器和客户端之间无需任何个人数据交换的情况下进行本地训练。但是，联邦学习也存在各种安全和隐私问题。近年来，大量的研究结果表明，联邦学习机制仍然存在安全问题，在训练过程中，本地设备与中央服务器之间的通信信道和传递的模型参数都有可能成为第三方窃取敏感信息的途径，联邦学习仍然面临各种安全和隐私威胁。针对联邦学习中用户本地训练数据、模型参数、模型架构等隐私的攻击方式包括投毒攻击、模型重建攻击、模型反演攻击、成员推理攻击等。

随着针对联邦学习框架的攻击模型增多，研究人员开始关注训练联邦学习模型时存在的隐私安全问题。针对联邦学习中的隐私保护的技术主要分为两类，一类是基于密码学技术，比如安全多方计算、同态加密等；另一类是基于系统安全技术，包括差分隐私、匿名技术、安全混淆等。基于安全多方计算的联邦学习隐私保护技术主要是在联邦学习中应用不经意传输、混淆电路、秘密共享等技术达到隐私保护的目的；基于同态加密的联邦学习隐私保护技术主要是通过加法同态、乘法同态或者全同态加密技术对用户上传的参数进行加密，以防止中央服务器或者恶意第三方服务器的攻击。虽然这两种密码学技术对数据的隐私保护效果很好，但应用于实际的联邦学习环境中，会出现模型难以收敛、计算成本过高、通信效率降低等问题，这阻碍了隐私保护的深度神经网络在当今工业中的全面应用，也是

研究领域中尚未解决的一个争论。而差分隐私等系统安全技术却在模型精度、通信效率上能达到更好的效果。因此，本文主要是基于系统安全技术对联邦学习中的隐私保护问题进行研究。

当前在联邦学习模型中应用差分隐私的主流方案是在本地训练随机梯度下降过程中，在梯度上添加噪声。Song 等人^[47]提出了一个 $(\epsilon_c + \epsilon_d)$ -差分隐私版本的随机梯度下降算，在本地模型的每一次迭代过程中，对梯度添加高斯噪声，并通过差分隐私的组合性和隐私放大效果，得到完全隐私损失的上界。差分隐私随机梯度下降 (DP-SGD) 严重降低了训练模型的效用，在数据集上训练和验证的损失率大大增加。因此本文针对模型效用和数据隐私保护的双重目标，设计、实现并评估了一个实用的联邦学习系统，该系统在保护数据隐私的前提下尽可能的维持了模型的精度和通信效率。本文主要的工作和贡献如下：

- (1) 在联邦学习差分隐私的场景下，本文设计了一种新型的、基于本地差分隐私的权重分配自适应干扰算法和梯度自适应裁剪算法。首先，我们考虑到不同的深度神经网络 (DNN) 层的模型权重可能会有很大的变化，提出了一种基于权重的贡献率添加自适应噪声的算法：在客户端本地训练的神经网络模型中，通过分析前向传播算法，计算每个属性类对于模型输出的贡献比，根据梯度的贡献率注入不同隐私预算的噪声。
- (2) 在传统的差分隐私随机梯度下降算法中，通常采用固定的裁剪阈值对梯度进行裁剪以限制函数的敏感度，然而固定的梯度裁剪可能添加额外的噪声。本文设计了一种自适应调整剪裁阈值的方案，通过计算梯度更新的方差和偏差，逐元素地对梯度进行裁剪，与之前的方案相比，通过使用梯度的坐标自适应剪裁实现了相同的隐私保证，而增加的噪声要少得多。之后我们利用“Moments Accountant”机制分析加噪累积产生的隐私预算，并证明了我们的方案满足 $(\epsilon_c + \epsilon_l)$ -差分隐私。与传统的注入噪声的方法相比，我们在相同的隐私保护程度下大大减少了噪声对模型输出结果的影响，提高了模型的准确性。

- (3) 由于本地差分隐私并不能有效防御针对联邦学习的生成对抗网络攻击，并且在通信轮数较大的联邦学习模型中，本地自适应差分隐私的强组合性质会导致总体隐私预算过高。本文提出了一种新的安全聚合机制，在本地客户端和中央服务器之间新增安全混洗器，在用户将参数上传到云服务器之前，先对参数进行拆分混洗，模型参数的更新被匿名的发送到混洗器，通过对模型参数的拆分和混洗实现客户端匿名，并减轻由联邦学习模型的高数据维度和大量查询迭代引起的隐私预算爆炸问题。
- (4) 为了验证本文的方案在实际生产环境中的可行性，本文模拟了联邦学习环境，分别在 MNIST、CIFAR-10、FMNIST 三种数据集上进行实验，首先通过控制变量法分析各个参数对于模型精度的影响和隐私保护的功用，并与前人的差分隐私方案和安全混洗方案进行对比，通过实验结果证明了自适应本地差分隐私方案和安全混洗框架的结合，在较低的隐私预算下还能使联邦学习模型维持较高的精度。

关键词： 联邦学习，隐私保护，差分隐私，安全混洗

ABSTRACT

With the emergence of data silos and the spread of privacy awareness, federated learning, an emerging model for data sharing and exchange, has a basic framework consisting of multiple local devices and a central server where all training data is stored locally and all devices work together to train a global model. One of the outstanding advantages of federation learning is that it allows local training without any individual data exchange between the server and the client, and is widely used in many fields such as finance, healthcare and education. However, there are various security and privacy issues associated with federation learning. In recent years, a large number of research results have shown that federal learning mechanisms still have security issues. During the training process, the communication channel between the local device and the central server and the model parameters passed may become a way for third parties to steal sensitive information, and federal learning still faces various security and privacy threats. Attacks against the privacy of the user's local training data, model parameters, and model architecture in federation learning include poisoning attacks, model reconstruction attacks, model inversion attacks, and membership inference attacks.

With the increase in attack models against the federation learning framework, researchers have begun to focus on the privacy security issues that exist when training federation learning models. The techniques for privacy protection in federation learning are mainly divided into two categories, one is based on cryptographic techniques, such as secure multi-party computation and homomorphic encryption, and the other is based on system security techniques, including differential privacy, ESA framework, and

mashup framework. Federated learning privacy protection techniques based on secure multi-party computation mainly apply techniques such as inadvertent transmission, obfuscated circuits and secret sharing in federation learning to achieve privacy protection; Federated learning privacy protection techniques based on homomorphic encryption mainly encrypt the parameters uploaded by users through additive homomorphic, multiplicative homomorphic or full homomorphic encryption techniques to prevent central servers or malicious third-party servers from attacks. Although these two cryptographic techniques are effective in protecting the privacy of data, when applied to a practical federal learning environment, problems such as difficult model convergence, high computational cost and reduced communication efficiency can occur; whereas system security techniques such as differential privacy can achieve better results in terms of model accuracy and communication efficiency. Therefore, this paper focuses on the study of privacy protection in federation learning based on system security techniques.

The current mainstream scheme for applying differential privacy in federation learning models is to add noise to the gradient during the locally trained stochastic gradient descent process. song et al. ^[47] proposed a $(\epsilon_c + \epsilon_d)$ -differential privacy version of the stochastic gradient descent arithmetic, in which the local During each iteration of the model, Gaussian noise is added to the gradient and an upper bound on the full privacy loss is obtained through the combinatorial nature of differential privacy and the privacy amplification effect. Differential privacy stochastic gradient descent (DP-SGD) severely reduces the utility of the training model, and the loss rate of training and validation on the dataset increases significantly. This paper therefore addresses the dual goals of model utility and data privacy preservation by designing, implementing and evaluating a practical federal learning system that maintains model accuracy and communication efficiency as much as possible while preserving data privacy. The main work and contributions of this paper are as follows.

The main work of this paper includes the following aspects:

1. In a federated learning differential privacy scenario, this paper designs a novel, lo-

cal differential privacy-based weight assignment adaptive interference algorithm and gradient adaptive cropping algorithm. First, we propose an algorithm for adding adaptive noise based on the contribution ratio of the weights, considering that the model weights may vary significantly from one deep neural network (DNN) layer to another: the contribution ratio of each attribute class to the model output is calculated in the client-side locally trained neural network model by analysing the forward propagation algorithm, and noise with different privacy budgets is injected according to the contribution ratio of the gradients. We further demonstrate how the proposed adaptive range setting can greatly improve the accuracy of aggregation models, especially in deeper model network structures.

2. In traditional differential privacy stochastic gradient descent algorithms, the gradient is usually clipped using a fixed clipping threshold to limit the sensitivity of the function, however a fixed gradient clipping may add extra noise. In this paper, we devise a scheme that adaptively adjusts the clipping threshold by calculating the variance and bias of the gradient update and cropping the gradient element by element, achieving the same privacy guarantee with much less added noise than previous schemes by using adaptive clipping of the gradient's coordinates. We then use the "Moments Account" mechanism to analyse the cumulative privacy budget generated by noise addition and show that our scheme satisfies $(\epsilon_c + \epsilon_l)$ -differential privacy. Compared with traditional methods of injecting noise, we greatly reduce the effect of noise on the model output results with the same degree of privacy protection and improve the accuracy of the model.
3. Since local differential privacy is not an effective defense against generative adversarial network attacks against federation learning, and in federation learning models with a large number of communication rounds, the strong combinatorial nature of local adaptive differential privacy and can lead to an excessive overall privacy budget. In this paper, we propose a new secure aggregation mechanism

by adding a secure mashup between the local client and the central server, where parameters are split and mashup before users upload them to the cloud server, and updates to model parameters are sent anonymously to the mashup, achieving client-side anonymity through splitting and mashup of model parameters, and mitigating the privacy budget explosion.

4. In order to verify the feasibility of this paper's scheme in a real production environment, this paper simulates a federal learning environment and conducts experiments on three datasets, namely MNIST, CIFAR-10 and FMNIST, respectively. Firstly, we analyse the effect of each parameter on the accuracy of the model and the utility of privacy protection through the control variables method, and compare it with the previous differential privacy scheme and secure mashup scheme. The results demonstrate that the combination of an adaptive local differential privacy scheme and a secure mashup framework can maintain high accuracy of the federated learning model even at a lower privacy budget.

Keywords: *Federated learning, Privacy preserving, Differential privacy, Security shuffle*

目录

第一章 绪 论	1
1.1 研究背景及意义	1
1.2 安全性和隐私威胁	4
1.3 国内外研究现状	6
1.4 本文工作与主要贡献	9
1.5 本文组织结构	10
1.6 本章小结	11
第二章 基础知识	12
2.1 神经网络	12
2.1.1 基本结构	12
2.1.2 随机梯度下降算法	13
2.1.3 经验风险最小化	14
2.2 联邦学习	14
2.3 差分隐私	15
2.3.1 基本定义	15
2.3.2 实现机制	17
2.3.3 相关定理	18
2.4 本章小结	19
第三章 本地自适应差分隐私 SGD 算法	20
3.1 引言	20

3.2	模型设计	22
3.2.1	模型概览	22
3.2.2	梯度的自适应加躁算法	24
3.2.3	梯度的自适应裁剪算法	28
3.2.4	基于自适应差分隐私的 SGD 算法	32
3.3	隐私参数分析	33
3.4	实验评估	36
3.4.1	实验准备	36
3.4.2	实验设计	37
3.4.3	结果分析	40
3.5	本章总结	47
第四章	基于 Top-K 安全混洗的联邦学习模型	48
4.1	引言	48
4.2	模型设计	49
4.2.1	模型概览	50
4.2.2	Top-K 梯度选择算法	53
4.2.3	拆分混洗算法	57
4.3	隐私性和收敛性证明	59
4.3.1	隐私性证明	59
4.3.2	模型收敛性分析	61
4.4	实验评估	63
4.4.1	实验准备	63
4.4.2	实验设计	64
4.4.3	结果分析	65
4.5	本章总结	69
第五章	总结与展望	70
5.1	论文总结	70
5.2	论文展望	71

参考文献	73
致谢	82
发表论文和科研情况	84

插图

1.1	联邦学习模型概况	3
1.2	联邦学习中的隐私威胁	5
1.3	联邦学习中的隐私保护技术	7
2.1	神经网络结构图	13
2.2	联邦学习模型工作流程	15
2.3	差分隐私的相邻数据集示意图	16
3.1	自适应差分隐私 SGD 算法流程图	23
3.2	逐层关联传播算法：根据前向传播计算总归因分数	26
3.3	逐层关联传播算法：根据反向传播计算各神经元的归因分数	27
3.4	MNIST 手写数字数据集	37
3.5	模型网络结构	37
3.6	仿真联邦系统模型概览	38
3.7	实现本地自适应差分隐私的伪代码片段	39
3.8	在 MNIST 数据集上噪声大小，裁剪阈值和隐藏层数量这三个参数对 于训练准确率的影响	40
3.9	在 MNIST 数据集上不同隐私预算下训练的准确率	42
3.10	不同隐私保护方案在 MNIST 数据集上训练的测试误差变化情况	44
3.11	在不同模型上进行成员推理攻击的准确率	46
4.1	基于安全混淆和 Top-K 梯度选择算法的联邦学习模型框架	51
4.2	top-K 梯度选择算法流程图	55

4.3	梯度元素的值及其效用评分	55
4.4	联邦学习安全混洗模型中执行参数拆分混洗的混洗器	58
4.5	安全混洗模型中本地客户端数量对联邦学习模型训练精度的影响 . .	66
4.6	安全混洗模型中客户端采样比对联邦学习模型训练精度的影响 . . .	66
4.7	安全混洗模型中梯度选择比率对联邦学习模型训练精度的影响 . . .	67
4.8	不同隐私保护方案在 MNIST 数据集上训练的模型分类准确率变化情况	68

表格

3.1	本地自适应差分隐私与其他四种基准方案	39
3.2	对比实验在数据集 MNIST 和 CIFAR-10 上的参数设置	43
3.3	本地自适应差分隐私与其他四种基准方案	43
4.1	安全混洗框架实验的模型网络结构	63
4.2	安全混洗框架的比较方案	65
4.3	不同梯度选择比率的通信开销	68

List of Algorithms

1	随机梯度下降算法	14
2	梯度的自适应加躁算法	28
3	梯度的自适应裁剪算法	31
4	基于自适应差分隐私的随机梯度下降算法	32
5	联邦学习中的安全混淆算法： $\mathcal{A}_{\text{csdp}}$	52
6	Top-K 梯度选择算法	56
7	混淆器中的拆分混淆算法	57

第一章 緒論

1.1 研究背景

在过去的近十年，人工智能（Artificial Intelligence, AI）取得了令人难以置信的进步，机器学习也越来越广泛地应用于各种领域，包括医疗健康、自动驾驶、金融贸易等。为了进一步提高模型的训练精度和学习能力，新兴的深度神经网络，也称为深度学习 (Deep Learning, DL) 随之提出。深度学习算法的目标是通过从数据中泛化来学习如何执行某些任务，比如说图像分类、语音识别、自然语言翻译等。深度学习作为最有前景的技术之一，已广泛应用于图像分类、自动驾驶、智慧医疗等各个方面。例如，智能图像识别系统已广泛部署在机场、火车站等公共场所。在识别可疑恐怖分子和检测违禁物品方面，它已被证明比人类更精确。基于患者的医疗数据，基于深度学习的回归技术可以帮助诊断和预防某些疾病（例如，遗传性和传染病）。很明显，基于深度学习的服务正在从旅行、社交、经济等诸多方面慢慢改变我们的生活。

深度学习算法的输入数据通常表示为一组样本。每个样本将包含一组特征值。例如，考虑一张 100x100 像素的照片，其中每个像素由一个数字（0-255 灰度）表示。我们可以用这些像素值组成一个长度为 10,000 的向量，通常称为特征向量。每张照片，表示为一个特征向量，可以与一个标签（例如，照片中人物的名字）相关联。深度学习算法将使用由多个特征向量及其相关标签组成的训练集来构建深度学习模型，这个过程称为模型的训练。当呈现一个新的测试样本时，深度学习模型应该给出预测的标签。模型准确预测标签的能力是衡量该模型对未知的数据的泛化程度的标准。它是通过测试误差（泛化误差）凭经验衡量的，它可以取决于用于

训练模型的数据的质量和数量、使用什么深度学习算法来构建模型、深度学习算法超参数的选择（例如使用交叉验证），甚至是特征的提取方法。

一般来说，训练数据量在某种程度上决定了模型的性能。为了支持基于机器学习/深度学习的开发不断增长的需求，许多互联网云提供商推动机器学习即服务（DLaaS），为机器学习/深度学习模型训练和服务提供计算平台和学习框架。典型的机器学习即服务平台包括 Amazon Sagemaker、Google Cloud ML Engine、Microsoft Azure ML Studio。要使用 MLaaS，客户需要向云提供商提供训练数据集和机器学习/深度学习算法。云提供商搭建深度学习环境，分配一定的计算资源，自动运行模型训练任务。云提供商还可以提供模型服务，将在云侧或客户侧训练的模型存储在云平台中。最后，云提供商向用户发布查询的 API 接口，使他们能够使用模型进行预测或分类。

训练数据集是训练和生成机器学习模型所必需的。数据集可能包含敏感样本，例如个人医疗记录、员工信息、财务数据等。我们的搜索查询、浏览历史、购买交易、我们观看的视频以及我们的电影的偏好都有可能被收集在使用的移动设备和计算机中。在街道上，以及即使在我们自己的办公室和家里。这种私人数据被用于各种深度学习应用。一些深度学习应用程序需要私人数据，此类私人数据以明文形式上传到集中的服务器，供深度学习算法学习数据中的规律，并从中构建模型。在 2018 年，中国互联网协会收到用户举报发现，腾讯音乐等多家应用软件以“通过深度学习向用户提供更好的服务”为由，长期收集并保存大量的用户个人数据，如照片、地址、电话等，甚至将这些包含了用户大量个人隐私的数据用作其他途径。问题不仅限于与将所有这些私人数据暴露在这些公司的内部威胁中，或外部威胁，如果持有这些数据集的公司遭到黑客攻击，那将会导致千万甚至亿万级的用户数据泄漏。

如何保护这些样本的隐私已成为机器学习中一个新的安全问题。这种隐私威胁在机器学习即服务中尤为严重。首先，在模型训练服务中，客户需要将训练数据上传到云提供商，提供商拥有数据的完全访问权限。过去的工作表明，恶意提供者

可以轻松窃取敏感数据，将它们嵌入模型中实现隐私的泄漏。其次，在模型训练服务中，客户需要将预训练好的模型上传到云提供商，并设置端点供远程用户使用模型。成功的深度学习模型包含有关训练集的基本信息。因此，即使恶意提供者无法直接访问数据集，他也可以从模型参数中提取有关训练数据的敏感信息。第三，即使云提供商是可信的，只有黑盒访问模型输出结果的恶意远程用户仍然能够通过使用精心设计的输入查询模型来检索有关训练数据的信息。

针对这些问题，Google 在 2016 年提出了联邦学习（Federated Learning, FL）的概念，它是一种典型的有助于解决多方计算下的隐私问题的学习方法。如图1.1所示，联邦学习的基本框架包含多个本地设备和一个中央服务器，所有训练数据保留在本地设备，所有设备共同协作训练一个全局模型。联邦学习实现了数据存储和模型训练的需求分离，使所有本地设备可以在不共享训练数据的情况下参与全局模型的训练。由于这种性质，联邦深度学习受到了工业界和学术界的广泛关注，并且已经大量研究者提出了各种分布式的学习架构来服务于特定场景。

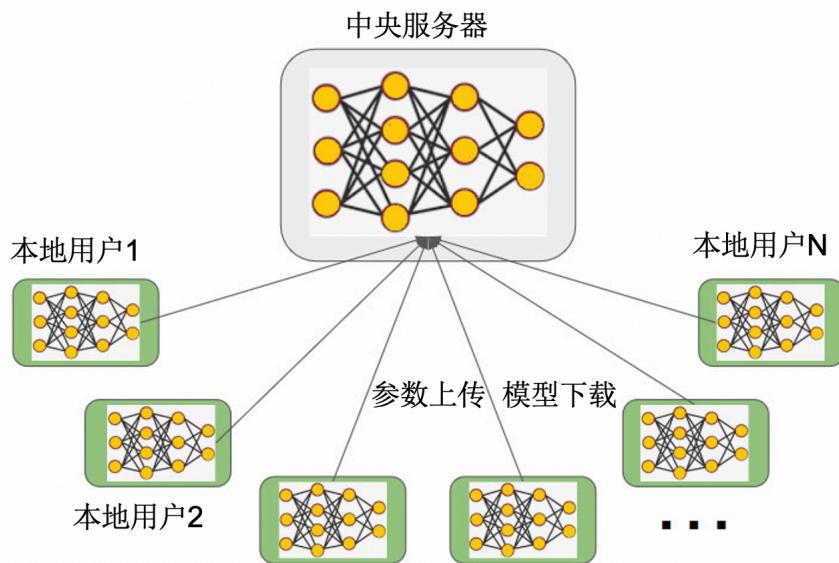


图 1.1: 联邦学习模型概况

联邦学习允许更智能的模型、更低的延迟和更低的功耗。在联邦学习框架中，由于数据通常分布和存储在不同的位置（例如，数据中心和医院），无需集中联合

学习使手机能够协作学习共享的预测模型，同时将所有训练数据保存在设备上，将深度学习的能力与数据存储在云中的需求分离开来，使用户可以在不共享本地数据的情况下在本地设备上进行预测。

1.2 安全性和隐私威胁

联邦学习的一个突出优点是它可以在服务器和客户端之间无需任何个人数据交换的情况下进行本地训练，从而防止客户端的数据被隐藏的对手窃听。尽管联邦学习的优势明显，而且与时俱进，但在实际应用之前，还需要对其安全性进行测试。近年来，大量的研究结果表明，联邦学习机制仍然存在安全问题，在训练过程中，本地设备与中央服务器之间的通信信道和传递的模型参数都有可能成为第三方窃取敏感信息的途径，联邦学习的框架仍然存在本地训练数据泄漏等隐私威胁。作为一种新的神经网络训练模型，攻击者可以从共享梯度中跟踪和获取参与者的隐私，联邦学习仍然面临各种安全和隐私威胁。

图1.2大致概括了针对联邦学习隐私攻击的攻击者、攻击内容、攻击类型、攻击方式和攻击时段。在联邦学习系统中，攻击方可能是内部攻击者，比如中央服务器、本地客户端；也有可能是外部攻击者。他们试图影响、破坏联邦学习模型的准确性，通过客户端上传的参数恶意的窃取用户的训练数据。

有一些恶意参与者会发送无效的模型参数更新到中央服务器，破坏全局模型的训练。比如，这些恶意参与方作为本地客户端参加训练，修改本地的训练数据，对本地数据注入一些有毒的数据，进行投毒攻击，从而损害全局模型的准确性，操纵模型的预测结果。

外部攻击主要通过本地客户端与中央服务器之间的通信信道发起。在训练过程中，局部模型更新和全局模型参数的结合过程，提供了关于训练数据的隐藏知识，用户的个人信息很有可能泄露给不受信任的服务器或其他恶意第三方。例如，白盒推理攻击和黑盒推理攻击^[21] 通过客户端上传的参数恶意的窃取用户的训练数据来生成样本原型。针对联邦学习中用户本地训练数据的攻击方式包括投毒攻击、

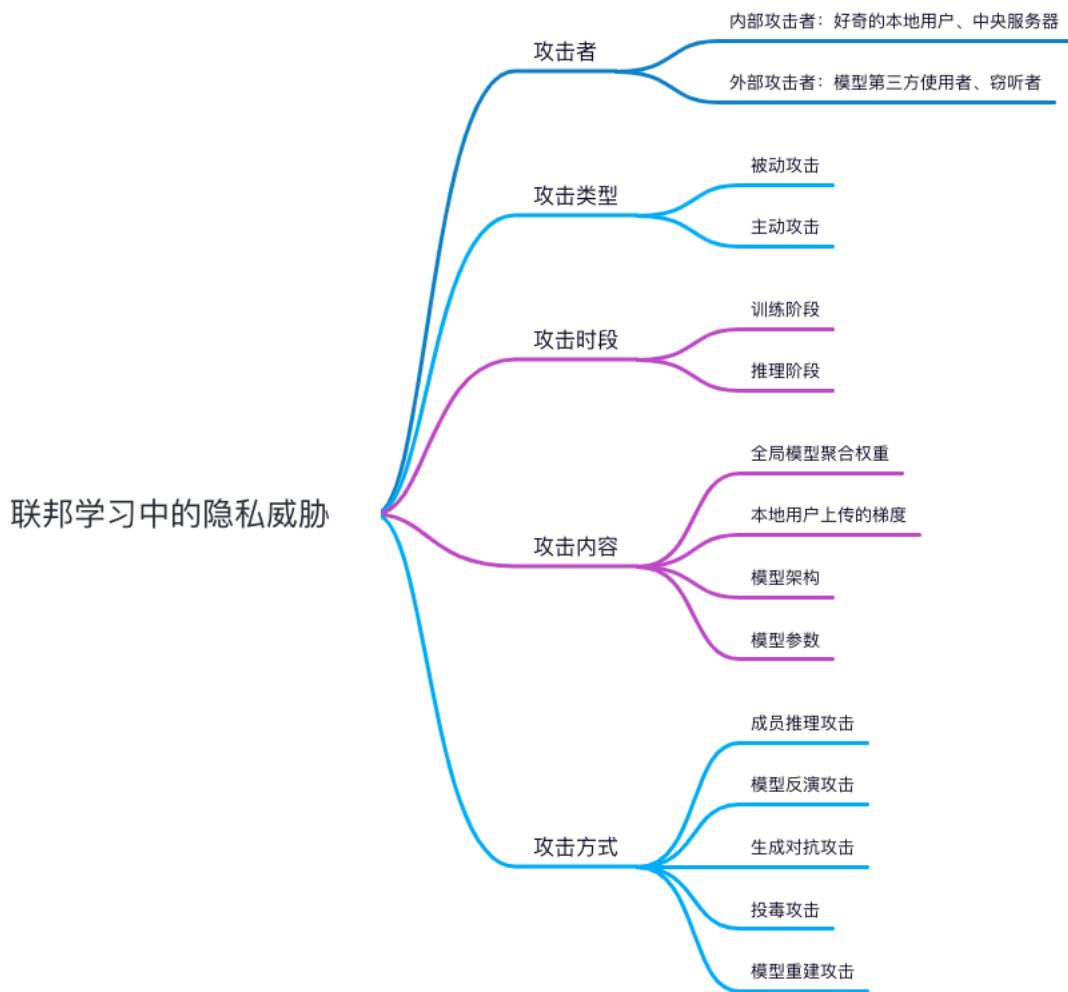


图 1.2: 联邦学习中的隐私威胁

模型重建攻击、模型反演攻击、成员推理攻击等。

投毒攻击：在联邦学习中，本地客户端在各自的设备上进行模型训练，将得到的训练参数上传给中央服务器。因为训练参数不需要通过可信机构的检查，所以有一些攻击者将恶意的训练样本注入自己的本地模型中，影响全局模型的更新结果，导致最终的模型预测结果错误甚至全局模型不可用。投毒攻击的影响对于许多企业和行业来说可能是致命的，在医疗部门、航空部门或道路安全方面甚至会危及生命。Marcus Comiter^[56] 曾使用投毒攻击进行实验，通过对熊猫的图像样本注入微小的恶意数据，导致算法预测结果发生重大变化，将熊猫识别为长臂猿。

模型重建攻击：在这种情况下，敌手的目标是窃取用户的原始训练数据。模型重建攻击需要白盒访问模型的权限，即模型中的特征向量对于敌手必须是已知的，敌手通过对特征向量的知识来重建用户的原始训练数据。对于一些机器学习算法，比如支持向量机（Support Vector Machine, SVM）或 K 最近邻算法（K-Nearest Neighbors, KNN），它们将特征向量存储在模型本身，容易受到重建攻击。攻击者通过解码用户上传的参数更新，反推出用户本地训练集中某条目标数据和其属性值。

模型反演攻击^[11]：利用用户上传的参数信息，以一种很简单的方式攻击用户数据，一旦用户的网络模型经过训练并达到收敛，攻击者就可以通过调整网络模型权重的梯度，获得网络模型中所有表示类的逆向工程试例。在模型反演攻击中，攻击者无需接触目标信息的标签类，攻击模型仍然能够恢复原始样本试例。这一攻击模型表明，任何经过精确训练的深度学习网络，无论是以何种方式进行训练收敛，都可以透露深度网络中不同标签类的信息。但是参数中包含的信息有限，模型反演攻击方式很难攻击卷积神经网络等复杂深度网络模型，在模型上添加了一定的隐私保护措施后，攻击也基本失效。

成员推理攻击^[13]：给定一个深度学习模型和一条数据样本（敌手的知识），成员推理攻击旨在确定样本是否为用于构建此深度学习模型的训练集成员（对手的目标）。这种攻击可能是被对手用来了解某个人的记录是否用于训练深度学习模型，此类攻击利用深度学习模型对训练集中使用的样本与未包含的样本的预测差异。Shokri 等人采用样本正确标签和目标模型预测的结果作为输入，训练影子模型作为攻击模型，达到推断任意样本是否在训练集中的目的。

1.3 国内外研究现状

随着针对联邦学习框架的攻击模型增多，研究人员开始关注训练联邦学习模型时存在的隐私安全问题。关于联邦学习的隐私定义主要分为全局隐私和局部隐私。在本地局部隐私中，每个客户端发送一个不同的隐私值，该值被安全的加密的

上传到中央服务器。在全局隐私中，服务器在最终输出中添加不同的噪音以实现隐私保护。安全多方计算、同态加密^[10] 和扰动技术是最常见的保证联邦学习中的安全和隐私的技术。

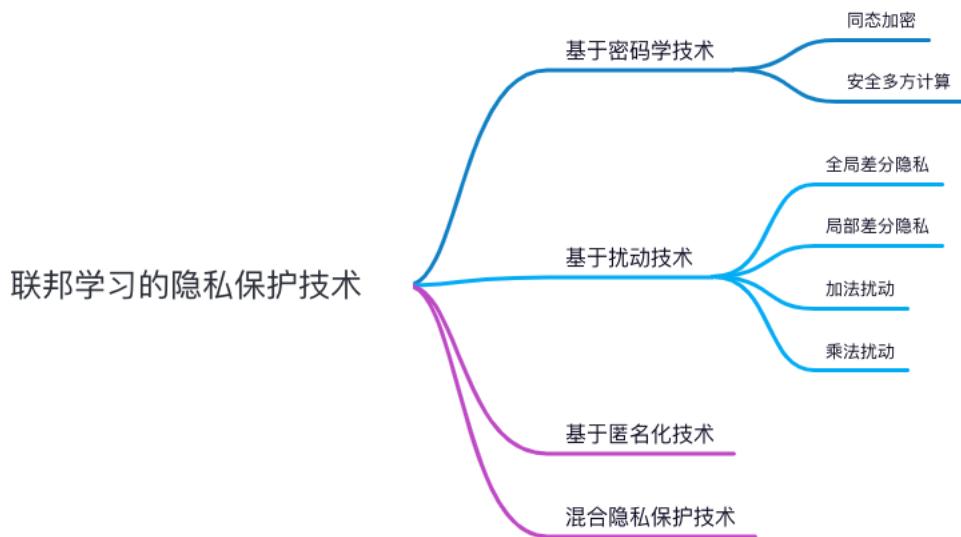


图 1.3: 联邦学习中的隐私保护技术

安全多方计算 (Secure Multi-Party Computation, SMC) 是由姚期智在 1982 年提出的^[16]，多个参与者在不泄露各自隐私数据情况下，利用隐私数据参与安全计算，共同完成某项计算任务。安全多方计算是解决协同计算问题的一种解决方案，它必须保证计算中各方信息的保密性、独立性和准确性。SMC 安全模型自然涉及多方，各方除了自己的输入和输出一无所知，以确保完整的零知识安全证明。当前，安全多方计算领域常见的技术主要包括混淆电路、零知识证明、不经意传输和秘密共享等。安全多方计算的重点是如何在没有可信第三方的情况下安全地计算一个指定的函数。同时，除了梯度的总和，每个参与者都不需要获得任何关于其他用户的信息。Bonawitz 等人^[3] 利用 SMC 设计了一个保护隐私的联邦学习框架，它可以安全地汇总用户的梯度，并且对用户的退出具有鲁棒性。然而，他们的模型由于涉及到学习过程中的多轮互动而导致通信开销巨大。

同态加密是一种加密形式，它允许用户对其加密数据执行计算，这些结果计

算以加密形式保留，解密后的输出与未加密时进行相同计算操作产生的结果相同。同态加密可以分为加法同态加密、乘法同态加密和全同态加密。同态加密在联邦学习中的应用主要通过防止服务器对本地客户端上传的权重进行逆向工程从而反推出训练数据，确保每个客户端对全局模型的更改都保持隐藏状态。因为服务端接收的是本地客户端通过同态加密算法处理后的数据，这种模型的安全性是以服务器上的计算成本为代价的^[23]，这种加密场景的高计算复杂度会严重危害分布式机器学习设置的性能。Phong 等人^[74] 提出了一个基于加法同态加密的联邦学习的隐私保护方案，保护梯度不被好奇的服务器发现。然而，一旦持有相同秘钥的用户相互串通，他们的方案将无法保护用户的隐私。

扰动技术的关键思想是在原始数据中加入噪声，使从扰动数据中计算出的统计信息在统计上与原始数据没有区别。三种广泛使用的扰动技术包括差分隐私、加法性扰动和乘法性扰动。差分隐私 (Differential Privacy, DP) 差分隐私技术是基于概率统计模型来量化数据集中实例的隐私信息的披露程度^[75]，主要原理是向数据添加噪音，或使用概括方法来掩盖某些敏感属性^[14]，使至多相差 1 条数据的 2 个数据集的查询结果概率不可区分，以保护用户的隐私。在联邦学习框架中，通过在本地模型和全局模型中对相关训练参数添加噪音，进行扰动，使敌手无法获得真实的模型参数，进而防御模型反演攻击、成员推理攻击等。在深度学习中，差分隐私可以作为一种局部隐私保护方案来保护用户梯度的隐私，Ding M 等人^[24] 提出了一种隐私保护的深度学习方法，主要通过添加噪音来扰乱本地模型的局部梯度，将差分隐私机制与模型训练中的随机梯度下降算法 (Stochastic Gradient Descent, SGD) 相结合。令人担忧的是，现有的差分隐私保护方案很难权衡隐私保护预算的成本和联邦学习模型的准确性，当使用较低的隐私预算达到较强的隐私保护的效果，可能使得模型难以收敛，可用性大幅下降；当隐私保护强度太低时，可能无法防御诸如 GAN 攻击等大规模的生成对抗网络攻击。

总的来说，安全多方计算基于复杂的计算协议，同态加密的运算成本非常高，而差分隐私破坏了数据的可用性，很难在模型性能和隐私成本上达到平衡，当前的

研究方向主要集中在对数据集和神经网络中的参数的加密和隐私保护机制上，较少关注到模型整体框架等过程。目前的联邦学习中的隐私保护方法还有许多不足，不能在隐私性和模型可用性上都达到一个相对满意的效果，此外，大部分方法是基于统一的、固定的参数设置，会导致模型迭代过程中累积大量隐私损失，使模型性能大幅下降。因此，在联邦学习场景下，保护用户隐私的同时维持模型准确性仍需大量的研究。

1.4 本文工作与主要贡献

针对联邦学习中隐私性和模型精度的双重指标，本文提出了本地自适应差分隐私算法和安全混淆框架，主要的工作和贡献包含以下三个方面：

- (1) 在联邦学习差分隐私的场景下，本文设计了一种新型的、基于本地差分隐私的权重分配自适应干扰算法和梯度自适应裁剪算法。首先，我们考虑到不同的深度神经网络（DNN）层的模型权重可能会有很大的变化，提出了一种基于权重的贡献率添加自适应噪声的算法：在客户端本地训练的神经网络模型中，通过分析前向传播算法，计算每个属性类对于模型输出的贡献比，根据梯度的贡献率注入不同隐私预算的噪声。
- (2) 本文设计了一种自适应调整剪裁阈值的方案，通过计算梯度更新的方差和偏差，逐元素地对梯度进行裁剪，与之前的方案相比，通过使用梯度的坐标自适应剪裁实现了相同的隐私保证，而增加的噪声要少得多。之后我们利用“*Moments Accountant*”机制分析加噪累积产生的隐私预算，并证明了我们的方案满足 $(\epsilon_c + \epsilon_l)$ -差分隐私。与传统的注入噪声的方法相比，我们在相同的隐私保护程度下大大减少了噪声对模型输出结果的影响，提高了模型的准确性。
- (3) 由于本地差分隐私并不能有效防御针对联邦学习的生成对抗网络攻击，并且在通信轮数较大的联邦学习模型中，本地自适应差分隐私的强组合性质和会

导致总体隐私预算过高。本文提出了一种新的安全聚合机制，在本地客户端和中心服务器之间新增安全混洗器，在用户将参数上传到云服务器之前，先对参数进行拆分混洗，模型参数的更新被匿名的发送到混洗器，通过对模型参数的拆分和混洗实现客户端匿名，并减轻由联邦学习模型的高数据维度和大量查询迭代引起的隐私预算爆炸问题。

- (4) 为了验证本文的方案在实际生产环境中的可行性，本文模拟了联邦学习环境，分别在 MNIST、CIFAR-10、FMNIST 三种数据集上进行实验，首先通过控制变量法分析各个参数对于模型精度的影响和隐私保护的效用，并与前人的差分隐私方案和安全混洗方案进行对比，通过实验结果证明了自适应本地差分隐私方案和安全混洗框架的结合，在较低的隐私预算下还能使联邦学习模型维持较高的精度。

1.5 本文组织结构

本文一共六章，主要内容的组织安排如下：

第一章对本文研究内容：联邦学习的研究背景、国内外研究现状进行了阐述，介绍了目前联邦学习中的攻击模型和隐私保护的研究现状与发展方向。

第二章详细介绍本文研究内容所涉及的一些理论基础与背景知识，包含了联邦学习的相关概念，差分隐私的基础知识和神经网络的基本结构和训练算法。

第三章描述了本文所提出的本地自适应差分隐私算法的设计和实现，根据神经网络前向传播算法，分析属性值的贡献率，然后采用自适应梯度裁剪算法裁剪梯度以限制函数的敏感度，最后根据贡献比率添加对应隐私预算的高斯噪声，之后采用“*Moments Accountant*”机制分析添加的噪声大小，证明了在自适应差分隐私机制下的联邦学习算法满足 $(\epsilon_c + \epsilon_l)$ -差分隐私，并通过实验证明了此方案在模型精度和隐私效用方面的表现。

第四章针对上一章节的方法所存在的问题提出了一种联邦学习安全混洗框架，在联邦学习模型中新增安全混洗器，通过对客户端上传的梯度进行采样，然后按

照维度拆分混淆，再将混淆模型和自适应本地差分隐私保护方法结合在联邦学习系统中，降低系统的整体隐私预算，接着证明了安全混淆框架的隐私性和全局收敛性，最后通过实验证明了此方案在模型精度和隐私效用方面的表现。

第五章是对文本的工作内容的总结和未来研究方向的展望。首先对本文的研究内容进行了概括，并总结了现有方案的不足之处，之后对未来的研宄和改进方向进行了展望。

1.6 本章小结

这一章节为绪论，首先介绍了本文章的研究背景和意义，总结了当前联邦学习发展过程中的挑战和难点，并具体针对联邦学习中的隐私威胁和隐私保护的研宄现状做了具体的阐述，最后介绍了文章的主要研究、贡献和组织结构。

第二章 基础知识

在本章节中我们将介绍本文研究所需要的一些基本知识，有助于更好的理解之后章节的内容。

2.1 神经网络

2.1.1 基本结构

深度学习模型通常采用神经网络的形式。已经为不同的应用提出了各种神经网络架构，例如多层感知器、卷积神经网络和循环神经网络。神经网络^[34]最初的设计灵感来源于人脑的结构。我们知道，人类的大脑是处理信息的主要部分，也是人中枢神经系统中的重要部分。人脑中含有大量的神经元，它们像网状物一样复杂的相互连接。当人脑接收到外部环境或者感觉器官传入的刺激（兴奋），它随着神经元一层一层的将刺激（兴奋）传导到神经中枢（大脑或脊髓），神经中枢根据接收到的信号，作出不同的判断，最后传递到输出神经。不同的信号，大脑都可以进行学习和分辨，而这一通用的模型，就是神经网络。

神经网络的基本组成单元是神经元，一个神经网络可能包含数百亿个简单的神经元，它们按层排列，密集而复杂的相互连接着。神经网络中每一层有多个神经元，层与层之间是“前馈传播”的，也就是说，网络中的数据只在一个方向上移动。一个单独的神经元可能与它前面一层的几个神经元相连，它从这些神经元接收数据；与它后面一层的几个神经元相连，它向这些神经元发送数据。每一层的神经元只可能与其前一层和后一层的神经元相连接，不存在跨层连接。

如图2.1所示，一个神经网络由一个输入层、一个输出层以及输入和输出之间

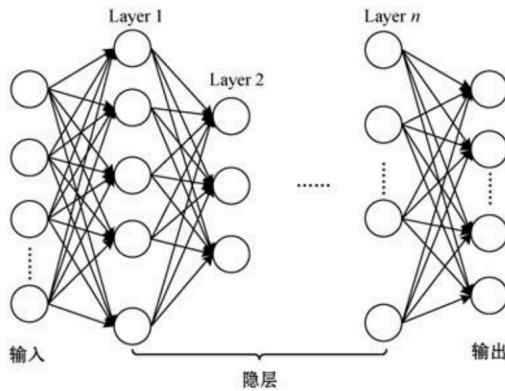


图 2.1: 神经网络结构图

的一系列隐藏层组成。每一层都是一组称为神经元的单元，它们连接到上一层和下一层的其他神经元。输入层用于接收信息，当一个神经网络被训练时，其所有的权重和阈值最初都被设置为随机值，然后训练数据被送入输入层；之后传入隐藏层进行特征的提取、网络权重的调整使得隐藏层的神经单元对某种模式形成反应；最后传导到输出层，输出模型判断的结果。神经元之间的每个连接都可以通过应用线性函数和元素级非线性激活函数（例如 sigmoid 或 ReLU）将信号传输到下一层的另一个神经元。通过这种方式，神经网络通过隐藏层转换输入，然后转换输出。

反向传播和随机梯度下降是训练深度学习模型和寻找最佳参数的常用方法。在深度神经网络中，对每个训练样本，通过前向传播算法从输入层、隐藏层到输出层依次训练，在输出层得到预测的结果，然后根据损失函数计算预测值与真实值之间的差异程度，之后根据反向传播算法调整权重系数，更新网络参数，使得损失函数的值最小，模型达到全局最优。

2.1.2 随机梯度下降算法

随机梯度下降算法（Stochastic Gradient Descent, SGD）是一种主流的用于机器学习和深度学习模型优化的迭代方法，从随机的权重系数开始，迭代运行梯度下降算法，使损失函数收敛到局部最小值，以找到使模型达到全局最优的权重系数，具体的算法如所示。

Algorithm 1 随机梯度下降算法

-
- 1: 输入: 学习率 α
 - 2: 输出: 初始参数 θ
 - 3: 初始化模型权重 θ , 作为梯度下降的起始点
 - 4: **while** 模型未达到全局最优点 **do**
 - 5: 从训练集中均匀抽出一小批量 (minibatch) 样本: $\mathbb{B} = \{x^{(1)}, \dots, x^{(m')}\}$
 - 6: 计算梯度估计:

$$g = \frac{1}{m'} \nabla_{\theta} \sum_{i=1}^{m'} L(x^{(i)}, y^{(i)}, \theta)$$

- 7: 梯度下降:

$$\theta = \theta - \epsilon g$$

- 8: **end while**
-

2.1.3 经验风险最小化

在神经网络中, 模型通过不断的学习数据集中的特征得到预测值, 通过损失函数计算预测值与真实值之间的误差, 之后再采用反向传播算法调整权重系数使得最终的损失函数的值最小。整个模型训练的过程可以理解为经验风险最小化 (Empirical risk minimization, ERM) 问题:

$$F(\boldsymbol{\theta}) := \frac{1}{n} \sum_{i=1}^n f_i(\boldsymbol{\theta}) \quad (2.1)$$

模型在训练集 $S = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$ 上进行训练, 其中, $F(\boldsymbol{\theta})$ 表示经验损失函数; $f_i(\boldsymbol{\theta}) = \ell(\boldsymbol{\theta}; \mathbf{x}_i, y_i)$ 表示在第 i 个训练样本 (\mathbf{x}_i, y_i) 上定义的损失函数; $\boldsymbol{\theta} \in \mathbb{R}^d$ 表示模型最终训练得到的权重参数。模型的训练目标是找到最终的权重参数 $\hat{\boldsymbol{\theta}} \in \mathbb{R}^d$, 使得公式3.1所计算得到的经验风险值最小。

2.2 联邦学习

传统的集中式深度学习需要将训练数据放在一起到数据中心。该模型以集中方式进行训练。而联邦学习允许数据所有者拥有一个私人学习网络, 该网络使用本地数据集进行训练。之后, 每个参与者将本地模型的梯度上传到云服务器。通过使用云服务器收集的全局梯度进行更新, 可以避免局部模型过度拟合。此外, 它还保

护本地数据不被其他参与者或云服务器直接知道。联邦学习的基本工作流程如下：

- **初始化:** 所有用户在各自的设备上都有一个预先分配的神经网络模型，并且可以自愿加入联邦学习协议，指定相同的深度学习和模型训练目标。
- **本地训练:** 在一个给定的通信回合中，联邦学习参与者首先从中央服务器下载全局模型参数，然后在各自的本地数据集 D_i 上进行模型训练，更新模型参数： $\omega_i^{r+1} \leftarrow \omega_i^r - \eta_i \nabla g(D_i^t, \omega_i^r)$
- **中央参数聚合:** 中央服务器等待所有本地客户端将更新后的模型参数 M_1, M_2, \dots, M_n 上传，聚合得到全局模型的参数，之后更新全局模型： $\omega^{r+1} \leftarrow \omega^r - \eta \frac{\sum_{U_i \in U^t} \Delta \omega_i}{\sum_{U_i \in U^t} |D_i^t|}$
- **迭代更新:** 迭代地执行上述步骤直至全局模型参数满足收敛条件，最终得到最优的全局模型。

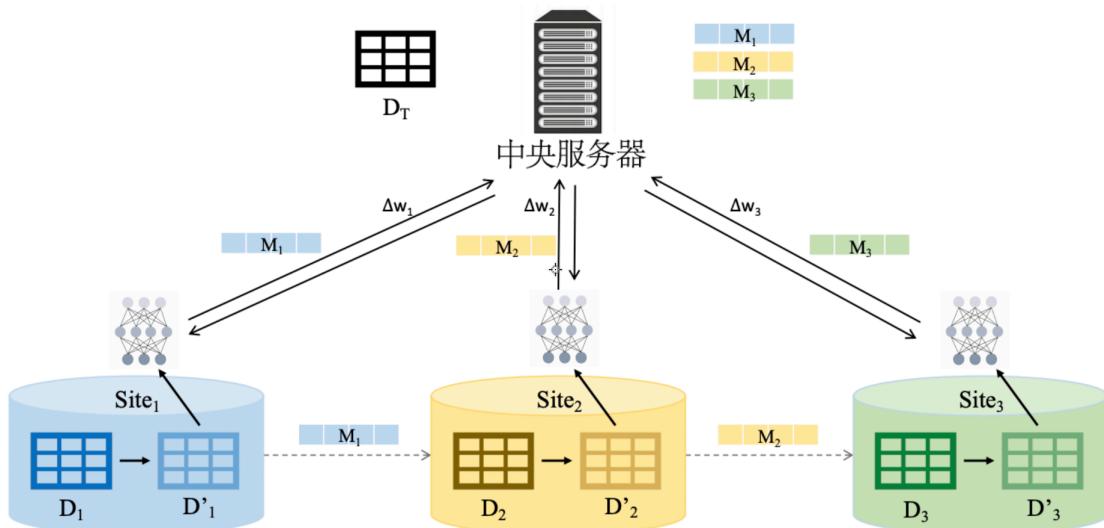


图 2.2: 联邦学习模型工作流程

2.3 差分隐私

2.3.1 基本定义

定义 2.3.1 (邻近数据集). 现有两个属性相近的数据集 D 和 D' ，他们的数据记录差为 $D \Delta D'$ ，如果 $|D \Delta D'| = 1$ ，则称数据集 D 和 D' 为邻近数据集 (*Adjacent Dataset*)。

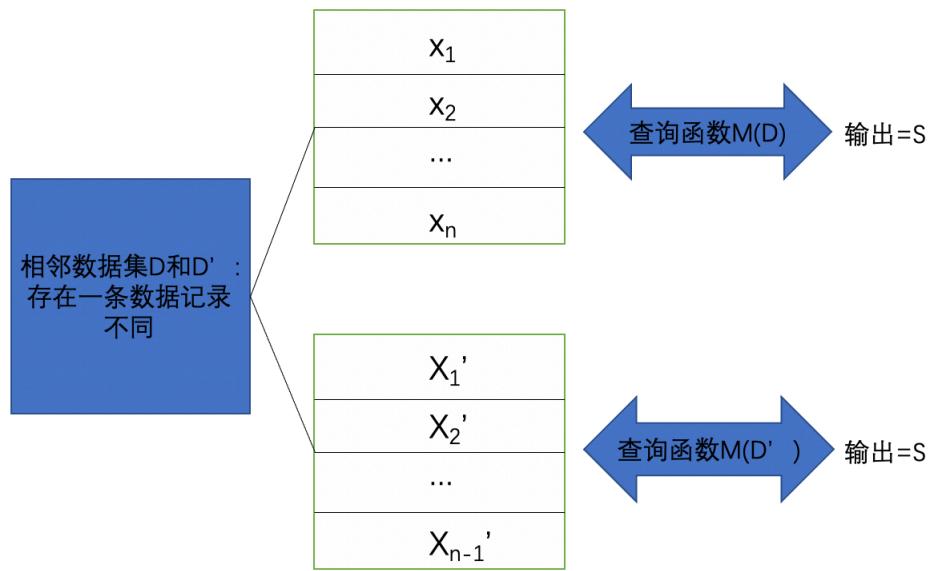


图 2.3: 差分隐私的相邻数据集示意图

2016 年，Dwork^[14] 等人首次提出了差分隐私的概念，它在针对数据隐私泄漏的新型隐私定义，目的是使数据库的查询函数对数据集中单条记录的变化不敏感。其思想是添加一定量的噪音来随机化给定算法的输出，从而使攻击者无法区分任何两个相邻的输入数据集的输出。具体的定义如下：

定义 2.3.2 ((ϵ, δ) -差分隐私). \mathcal{D} 表示数据集合， D 和 D' 为邻近数据集。现有随机算法 $M : D \rightarrow R$, D 表示定义域, R 表示值域。如果对于任意两个邻近数据集 $S, S' \in \mathcal{S}^n$ 和输出子集 $O \subseteq \mathcal{R}$ 时，总有

$$\Pr[M(D) \in S] \leq e^\epsilon \Pr[M(D') \in S] + \delta$$

， 则称该随机算法满足 (ϵ, δ) -差分隐私。

添加项 $\delta \in [0, 1]$ 表示以某种概率打破 $(\epsilon, 0)$ -差分隐私。当 $\delta = 0$ 时，则将 M 称为 ϵ -差分隐私。 ϵ 和 δ 表示隐私预算参数， ϵ 和 δ 越小，算法能提供的隐私保证程度越强。

差分隐私保护的实现是在查询函数的返回值中注入一定量的干扰噪声，但是注入的噪声量太大会影响最终结果的准确性，太少则无法保障数据的隐私性。那么

如何衡量添加的噪声量，既能保障数据的安全，又能维持数据的可用性呢？这里针对数据集提出敏感度的概念，加入的噪声量大小与数据集的敏感度息息相关。一个函数的灵敏度决定了需要在其输出中加入多少随机噪声来实现差异性隐私。对于相邻数据集 D 和 D' ，他们的敏感度代表某一个查询函数在这两个相邻数据集上输出的最大不同。

定义 2.3.3 (函数敏感度). 假设存在查询函数 $f : D \rightarrow R^d$ ，输入为一数据集，输出为 d 维的实数向量。对于任意的邻近数据集 D 和 D' ，函数 f 的 L_1 敏感度 (L_2 敏感度) 表示为 $\Delta_1(q)$ ($\Delta_2(q)$)，计算公式如下：

$$\Delta_1(q) = \max_{D \sim D'} \|f(D) - f(D')\|_1, \quad \Delta_2(q) = \max_{D \sim D'} \|f(D) - f(D')\|_2$$

称为函数 f 的全局敏感度。

2.3.2 实现机制

在差分隐私的实际应用中，如何针对不同的场景和问题设计添加噪声的机制使算法能满足差分隐私保护的要求呢？差分隐私的实现机制主要分为拉普拉斯机制 (Laplace Mechanism)^[9]、指数机制 (Exponential Mechanism)^[32] 与高斯机制 (Gaussian Mechanism)^[33]。其中，指数机制适用于非数值型结果的隐私保护，拉普拉斯机制和高斯机制适用于对数值型结果的隐私保护^[35]。

定理 2.3.4 (拉普拉斯机制). 给定一个基于数据集 D 的查询函数 $f(D)$ ，算法 $\ddot{f}(D)$ 满足 ϵ -差分隐私，当：

$$\ddot{f}(D) = f(D) + \text{Lap}\left(\frac{GS}{\epsilon}\right)$$

其中，噪声参数满足 $\text{Lap}\left(\frac{GS}{\epsilon}\right)$ 的 Laplace 分布， GS 表示数据集的敏感度。

与拉普拉斯机制类似，高斯机制对输入数据的所有维度添加满足高斯分布的噪声。

定理 2.3.5 (高斯机制). 一个查询函数 $f : D \rightarrow R$, 该算法的敏感度表示为 S_f , 算法 M 满足 ϵ -差分隐私, 当:

$$\mathcal{M}(d) \triangleq f(d) + \mathcal{N}(0, S_f^2 \cdot \sigma^2)$$

其中, $\mathcal{N}(0, S_f^2 \cdot \sigma^2)$ 是满足正态(高斯)分布的, 均值为 0, 标准差为 $S_f\sigma$ 。当 $\epsilon \in (0, 1]$, $\sigma \geq \sqrt{2 \ln(1.25/\delta)} \Delta_A / \epsilon$, 算法 M 满足 (ϵ, δ) -差分隐私。

但是对于离散型的查询结果或数据要如何处理呢? 这就产生了指数机制, 通常使用指数机制来随机选择离散的输出结果来满足差分隐私。指数机制整体的思想就是, 对于一个查询函数, 不是确定性的输出一个 R_i 结果, 而是以一定的概率值返回结果, 从而实现差分隐私。

定理 2.3.6 (指数机制). 指数机制满足差分隐私, 如果:

$$A(D, u) = \left\{ p : \Pr[p \in O] \propto \exp\left(\frac{\epsilon u(D, p)}{2\Delta u}\right) \right\}$$

其中 $u(D, p)$ 为评分函数, 评分越高, 则输出的概率越大^[53], Δu 表示 $u(D, p)$ 的全局敏感度。

2.3.3 相关定理

在解决一个复杂的差分隐私保护问题时, 可能在多个场景, 多个步骤多次应用差分隐私技术, 在这种情况下, 如何保证最终结果的差分隐私性, 以及隐私保护的程度该如何去度量呢? 这里引出差分隐私的三个最重要的性质: 可量化性、可组合性和后处理不变性^[35]。

可量化性指的是差分隐私算法在计算特定随机化过程时, 可以透明化、精准量化所施加的噪声大小, 即上文提及的隐私预算。这样使用者就可以清楚地知道算法的隐私保护力度; 差分隐私的后处理不变性, 确保了即使对算法的结果进行进一步处理, 只要不引入额外信息, 后续的处理就并不会削弱算法的隐私保护力度。组合性是指将独立的满足差分隐私的算法进行串行组合或者并行组合之后得到的算法依然满足差分隐私。

定理 2.3.7. 对于任意满足 (ε, δ) -差分隐私的算法 \mathcal{M}_1 和 \mathcal{M}_2 , 算法 \mathcal{M}_3 : $\mathcal{M}_3(\vec{x}) = (\mathcal{M}_1(\vec{x}), \mathcal{M}_2(\vec{x}))$ 也满足 (ε, δ) -差分隐私。

定理 2.3.8. 对于任意满足 (ε, δ) -差分隐私的算法 $\mathcal{M}_1, \dots, \mathcal{M}_d$, 算法 $\overline{\mathcal{M}}$: $\overline{\mathcal{M}}(\vec{x}) = (\mathcal{M}_1(\vec{x}), \dots, \mathcal{M}_k(\vec{x}))$ 也满足 $\left(\varepsilon \cdot \left(\sqrt{2d \ln(1/\delta)} + (e^\varepsilon - 1) \cdot d\right), \delta \cdot (d+1)\right)$ -差分隐私。

通过差分隐私的串并行组合定理, 人们可以利用基础的差分隐私算法设计出复杂的满足差分隐私的系统, 只要算法中的每一个步骤都满足差分隐私要求, 那么这个算法的最终结果将满足差分隐私特性, 这也是差分隐私的重要优势之一。在差分隐私的应用程序中, 通常结合串并行组合定理分析算法累积的总体隐私预算和隐私成本。

对于一个随机算法设计满足差分隐私的方案通常包括以下步骤:

- (1) 通过敏感度有界函数的组合来设计逼近的系统
- (2) 选择合适的噪声机制和参数实现差分隐私
- (3) 结合串并行组合定理分析算法累积的总体隐私预算和隐私成本

2.4 本章小结

本章节为基础知识, 对于论文的研究所涉及的基础知识定理进行了讲解。本章主要介绍了神经网络的结构和算法、联邦学习系统的学习协议以及差分隐私的基本概念、定义和定理。分布式联邦学习系统是本论文主要使用的系统架构, 本文所针对的攻击模型和隐私保护方案都是基于该分布式联邦学习系统。

第三章 本地自适应差分隐私 SGD 算法

3.1 引言

与传统的集中式深度学习相比，联邦学习通过分布式训练在一定程度上缓解了隐私泄漏的问题。然而，许多研究表明深度学习技术可以“记忆”模型中的训练数据信息，在训练过程中，本地设备与中央服务器之间的通信信道和传递的模型参数都有可能成为第三方窃取敏感信息的途径，联邦学习的框架仍然存在本地训练数据泄漏等隐私威胁^[48]。在这种情况下，敌方一旦通过白盒推理攻击或者黑盒推理攻击访问模型，就可以推演出客户端本地的训练数据。

在第二章的基础知识中我们介绍了联邦学习模型的整体流程，联邦学习模型的优化问题可以概括为 ERM（经验风险最小化）问题^[40]：

$$\arg \min_{\theta \in \mathcal{C}} \left(F(\theta) := \frac{1}{m} \sum_{i=1}^m F_i(\theta) \right) \quad (3.1)$$

通过优化 ERM 函数间接优化模型参数，使模型的实际输出与预测值更加接近，模型的准确率越高。从隐私保护的角度讲，我们只要截断了从原始输入到输出，在其中加入一道隐私保护屏障，根据在哪一步截断将差分隐私保护联邦学习的方法分为以下几种：

- **输入扰动：**输入扰动是在获取的训练数据上直接添加噪声，之后的模型训练和优化都是基于加噪后的训练数据^{[37][38][39]}。
- **输出扰动：**输出扰动沿袭了拉普拉斯机制最简单的思路，即考虑函数输出的敏感度来添加噪声，那么在 ERM 公式中我们只需要考虑 argmin 函数输出的

敏感度，基于这个敏感度来添加拉普拉斯噪声即可得到一个简单的满足差分隐私的 ERM 方法^[36]。

- **梯度扰动：**梯度扰动是在执行最小化损失函数的过程中，设计满足差分隐私的算法。
- **目标扰动：**目标扰动是在模型的目标函数中添加一个随机量，以使得最终模型的输出满足随机性。

基于输入的扰动和输出的扰动基本可以视为一个黑匣子模型，这种添加噪声的方式虽然简单直接，但无法对训练过程中数据的相互依赖性和输出有效性作出有用的、紧密的描述。在输入数据中加入过多的噪声，可能会影响模型训练的收敛性。在输出参数中加入过于保守的噪声，也就是根据最坏的攻击情况去添加噪声，可能会影响模型的实用性。

当前在深度学习模型中应用差分隐私的主流方案是在模型的梯度上添加噪声，方案的目标是在满足差分隐私的条件下，实现整体模型的最优可用性。Song 等人^[47]提出了一个 $(\epsilon_c + \epsilon_d)$ -差分隐私版本的随机梯度下降算法，在本地模型的每一次迭代过程中对梯度添加高斯噪声，并通过差分隐私的组合性和隐私放大效果，得到完全隐私损失的上界。Goodfellow^[64]提出了 ℓ_2 范式梯度裁剪的方式以限制函数敏感度，并设计了“Moments Accountant”(MA) 来计算更准确的隐私预算估计，在预训练过程中，该方法与 PCA 相结合，形成了一个满足 $(\epsilon_c + \epsilon_d)$ -差分隐私的 PCA。

由 Song 等人^[47] 中的实验数据可知，差分隐私随机梯度下降(DP-SGD)与 SGD 相比严重降低了训练模型的效用。当差分隐私提供的隐私强度增加时，在 MNIST 数据集上进行逻辑回归的训练和验证的损失率迅速增加。在 MNIST 上数据集上，采用 DP-SGD 训练的卷积神经网络(CNN) 的测试精度比 SGD 低得多。

在传统的基于差分隐私的联邦学习模型中，数据管理者倾向于给每个用户的数据相同的隐私预算，同样的隐私预算忽略了用户之间的差异。有些用户希望有更好的隐私保护，而有些用户对某些数据的隐私不敏感。由于联邦学习模型是分布

式结构，从一个大数据库到许多小数据库，所以对于每个用户来说，他们只需要关心他们自己的隐私。他们可以设置不同的隐私预算方案，而不是传统的统一分配，然后在最坏的情况下注入噪音，而且基于梯度扰动的方法的问题在于它们的迭代性质会导致隐私预算的飙升。因此，当前的主要挑战是设计一种新型的满足差分隐私的扰动算法，使其在联邦学习中能按照用户隐私标准，给不同的用户分配不同的隐私预算，既能保证模型的效用性，并且维持较高的计算效率，在模型准确率方面接近非差分隐私的深度神经网络模型。

本文的创新点在于采用一种更加复杂的方法来分析训练过程中训练数据对模型输出的贡献率。在预训练阶段，我们采用逐层关联传播算法在神经网络的前向传播和反向传播流程中分解模型输出，计算网络中每个神经元对模型输出的贡献率。基于每个神经元的贡献率，我们通过高斯机制生成一个噪音图元，然后根据每一层神经网络对模型输出的贡献率，在梯度上根据贡献率添加对应的噪声量，避免了无法量化的噪声，从而提升了隐私保护模型的准确性。在进行梯度下降之前，根据神经网络的层次对梯度进行自适应的裁剪，保证模型的快速瘦脸。我们的方法不仅产生了在模型精度方面最接近非差分隐私的模型，而且还降低了隐私预算的成本。

3.2 模型设计

3.2.1 模型概览

传统的联邦学习中使用差分隐私的主要流程如下所示：

- **本地计算：**客户端 i 根据本地数据库 D_i 和接受的服务器的全局模型 w_G^t 作为本地的参数，即 $w_i^t = w_G^t$ ，采用梯度下降策略进行本地模型训练得到 w_i^{t+1} (t 表示当前通信回合)。
- **模型扰动：**每个客户端产生一个随机噪音 n , n 是符合高斯分布的，使用 $\bar{w}_i^{t+1} = w_i^{t+1} + n$ 扰动本地模型 (这里注意 w 是一个矩阵， n 表示对矩阵的每一个元素添加噪音)。

- **模型聚合:** 中央服务器使用参数聚合算法聚合从客户端收到的 $\bar{w}_i t + 1$ ，得到新的全局模型参数 w_G^{t+1} ，也就是扰动过的模型参数。
- **模型广播:** 中央服务器将新的模型参数广播给每个客户端。
- **全局收敛:** 重复步骤 (1) - (4) 直至全局模型收敛。

在本文中，我们认为中央参数服务器是一个“诚实但好奇”(Honest but Curious, HbC) 的实体。也就是说，服务器将遵循与所有用户的协议。然而，通过利用通信信道访问用户梯度的便利，它也试图在训练过程中反推出关于客户端的额外的信息。出于这个原因，我们设计的本地自适应加噪算法目的是保护发送到服务器的本地梯度不会被中央服务器推断出任何关于用户的本地训练样本信息，并且尽量维持原有模型的精度。

本文的模型主要针对本地设备的模型训练添加自适应的噪声和梯度裁剪算法，保证中央服务器接收到满足 ϵ -差分隐私的权重，从而减轻成员推理攻击对联邦学习所构成的隐私威胁。本地客户端进行模型训练的基本流程如图3.1所示，通过从数据集 D 中随机采样构造各批次的样本，对于每一批样本通过梯度下降算法进行模型训练，在梯度上添加高斯噪声，使得最终的输出满足 ϵ -差分隐私，然后在梯度聚合之前通过梯度自适应裁剪算法加快模型的收敛速度。

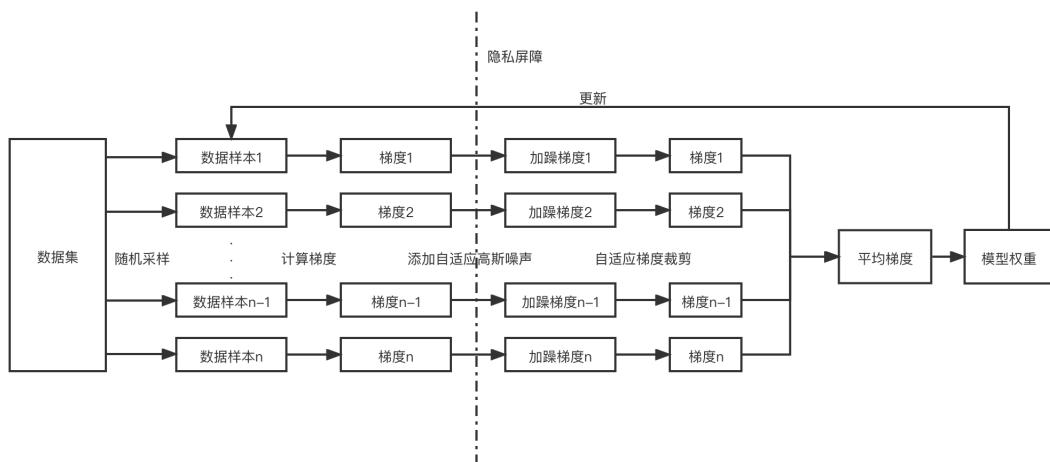


图 3.1: 自适应差分隐私 SGD 算法流程图

为实现相同的效用保证，我们算法的梯度复杂度（即计算的随机梯度总数）为 $O(n^{3/2})$ ，比之前的最佳结果高出 $\Theta(n^{1/2})$ 。之后我们在 MNIST 和 FICAR-10 数据集上进行实验，评估我们提出的方法，并与前人的四种差分隐私保护方案进行对比，发现我们的方法不仅产生了在模型精度方面最接近非差分隐私的模型，而且还降低了隐私预算。并且，我们针对添加本地自适应差分隐私方案的模型进行成员推理攻击，评估了该方案的隐私保护效用。

总的来说，本章提出的隐私保护方案是基于本地客户端的本地数据维度，从以下两个方面展开研究：

- (1) 通过在本地模型训练的梯度下降算法过程中针对不同神经元的贡献率添加对应比例的高斯噪声。
- (2) 根据训练轮数和梯度变化的偏差与方差动态的更新梯度裁剪阈值，对梯度进行自适应裁剪。

3.2.2 梯度的自适应加躁算法

在第二章我们详细介绍了联邦学习的流程和神经网络的结构，每个用户在本地设备的私有数据集上进行训练。本地训练的流程主要概括为前向传播和反向传播。神经网络中前向传播算法的第一步在输入层，我们使用前馈神经网络接收输入的 x 运行前向传播算法，得到预测值，然后通过反向传播算法不断调整参数使与预测值和真实值之间的误差降低。

Bach 等人提出了针对神经网络的逐层关联传播算法^[65]，这是一种用于计算特定图像区域对分类器输出影响的技术。该技术被认为是非常有效的，可以逐个像素地分解图像的预测值^[76]，可以应用在深度神经网络中对各神经层进行循环重正化^[77]。它允许分解深度神经网络的预测值。由于经典的反向传播算法是通过链式法则对模型的输出进行反向求导，根据损失函数计算每个神经元对模型总误差的贡献值，然后来调整模型的权重，以降低总误差。根据这一思路，在我们的工作中我们利用逐层关联传播算法将神经网络的输出值按层进行分解，得到每层的神经元

对于模型输出的贡献比，然后根据神经元的贡献率，在梯度下降的过程中添加对应比例隐私预算的高斯噪声。

在逐层关联传播算法中，根据神经网络的结构自前向后计算归因值，也称为相关性分数，它假设一个分类器可以被分解成多个编译层。在卷积神经网络结构中，每个隐藏神经元的转化过程表示为 $y = \sigma(\mathbf{x} * \omega + b)$ ，其中 \mathbf{x} 代表输入向量， y 是输出， b 和 ω 分别代表偏置项和权重矩阵。 $\sigma()$ 是一个激活函数，用于结合线性变换和非线性变换。在前向传播过程中，图像输入以像素的形式进入网络，然后与网络权重 w 相乘，再加上一个偏置 b ，并通过激活函数使其成为非线性计算，这个过程一直持续到输出层得到模型输出。

在逐层关联传播算法中，网络输出表示为 $F(\mathbf{x})$ ，其中 F 表示一个分类器， \mathbf{x} 是以像素形式存在的图像 $(x_1, x_2, x_3, \dots, x_n)$ 。网络的输出值 $F(\mathbf{x})$ 以反向传播的方式从输出层反馈到隐藏层，逐层分解每个像素的归因分数，直到到达输入层，反向传播的过程就结束了。对于卷积神经网络等网络结构的前向传播算法，模型输出表示为 F_θ ，输出层的神经元表示为 z ， Cr_z 表示网络总归因分数，逐层关联传播算法利用 F_θ 和 Cr_z 在反向传播算法中逐层分解计算各层神经元对应的归因分数。输出层的归因分数通常作为输出层的预激活值，在反向传播过程中，每一层的所有神经元的归因分数总和是恒定。

接着，我们具体阐述如何计算网络中某一个神经元 a_j 的归因分数。箭头 (\rightarrow) 和 (\leftarrow) 分别代表第 l_m 层和第 l_{m+1} 层的前向连接和反向连接关系， $r_{n \rightarrow z}$ 表示由神经元 n 到神经元 z 的前向传播关系， $R_{n \leftarrow z}^{(l_m, l_{m+1})}$ 表示第 $m+1$ 层的神经元 z 到第 m 层的神经元 n 的反向传播关系。如图3.2所示，在前向传播算法中，神经元 a_{i1} 接受上一层输入的像素值 x_1 并计算 $r_{a_{i1} \rightarrow a_j}^{(l_m, l_{m+1})} = \sigma(x_1 * \omega + b)$ ，然后将计算结果传输给第 l_{m+1} 层的神经元 a_j ；同理，输出层 l_{m+1} 的神经元 a_j 接受来自上一层神经元 a_{i2} 的计算结果 $r_{a_{i2} \rightarrow a_j}^{(l_m, l_{m+1})} = \sigma(x_2 * \omega + b)$ ，那么模型输出的结果为 $r_{a_j} = r_{a_{i1} \rightarrow a_j} + r_{a_{i2} \rightarrow a_j} + b_{a_j}$ 。具体的来说，神经元 a_4 接收来自输入层的神经元 a_1, a_2, a_3 的计算结果，分别为 $\sigma(x_1 * \omega + b), \sigma(x_2 * \omega + b), \sigma(x_3 * \omega + b)$ ，神经元 a_4 接收的总输

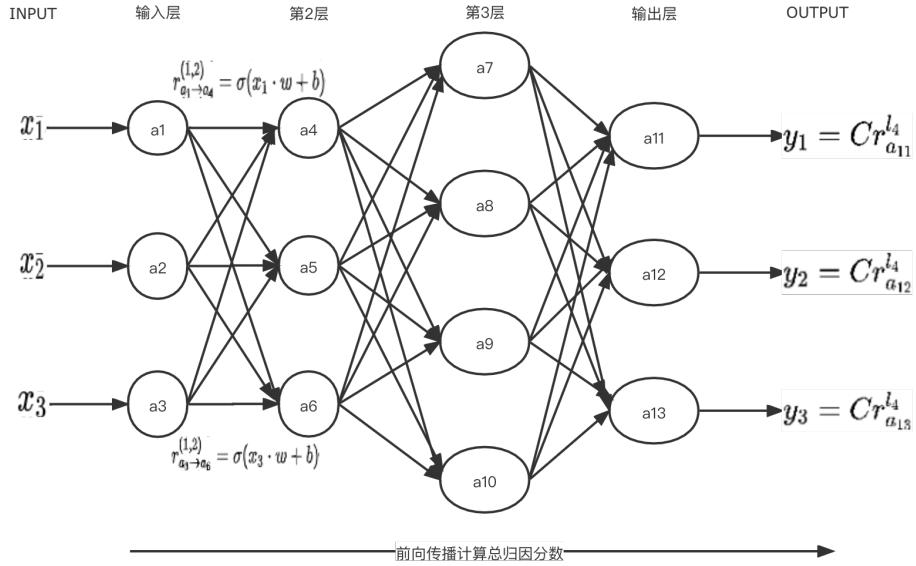


图 3.2: 逐层关联传播算法: 根据前向传播计算总归因分数

入值为 $r_{a_1 \rightarrow a_4}^{(l_1, l_2)} + r_{a_2 \rightarrow a_4}^{(l_1, l_2)} + r_{a_3 \rightarrow a_4}^{(l_1, l_2)} + b_{a_4}$ 。

$Cr_{a_i}^{l_m}(x_i)$ 表示第 m 层的神经元 a_i 对于模型输出的归因分数。因为输出层的神经元 a_4 的归因分数等于模型的输出, 所以有 $Cr_{a_4}^{l_4}(x_i) = y_1$ 。

根据神经网络相邻层间的线形关系和反向传播算法, 神经元 a_i 的归因分数 $Cr_{a_i}^{l_m}(x_i)$ 即为与之相邻的第 $m+1$ 层的所有神经元 $a_j \in l_{m+1}$ 的归因分数总和:

$$Cr_{a_i}^{l_m}(x_i) = \sum_{a_j \in l_{m+1}} Cr_{a_i \leftarrow a_j}^{l_m \leftarrow l_{m+1}}(x_i)$$

根据层间关联传播算法, 第 m 层和第 $m+1$ 层的反向关联性表示为:

$$R_{a_i \leftarrow a_j}^{(l_m, l_{m+1})} = \begin{cases} \frac{r_{a_i \rightarrow a_j}}{r_{a_j} + b_{a_i}} \cdot Cr_{a_j}^{l_m}, & r_{a_j} \geq 0 \\ \frac{r_{a_i \rightarrow a_j}}{r_{a_j} - b_{a_i}} \cdot Cr_{a_j}^{l_m}, & r_{a_j} < 0 \end{cases}$$

因为位于输出层的神经元 a_j 的贡献等于模型的输出, 第 $m-1$ 层的神经元 a_j 对于第 m 层的神经元的归因分数 $Cr_{a_i \leftarrow a_j}^{l_{m-1} \leftarrow l_m}(x_i)$ 等于:

$$Cr_{a_i \leftarrow a_j}^{l_{m-1} \leftarrow l_m}(x_i) = \begin{cases} \frac{a_i w_{i,j}}{\sum_{a_i \in l_{m-1}} a_i w_{i,j}} Cr_{a_j}^{l_m}(x_i) & \sum_{a_i \in l_{m-1}} a_i w_{i,j} \neq 0 \\ \mu & \sum_{a_i \in l_{m-1}} a_i w_{i,j} = 0 \end{cases} \quad (3.2)$$

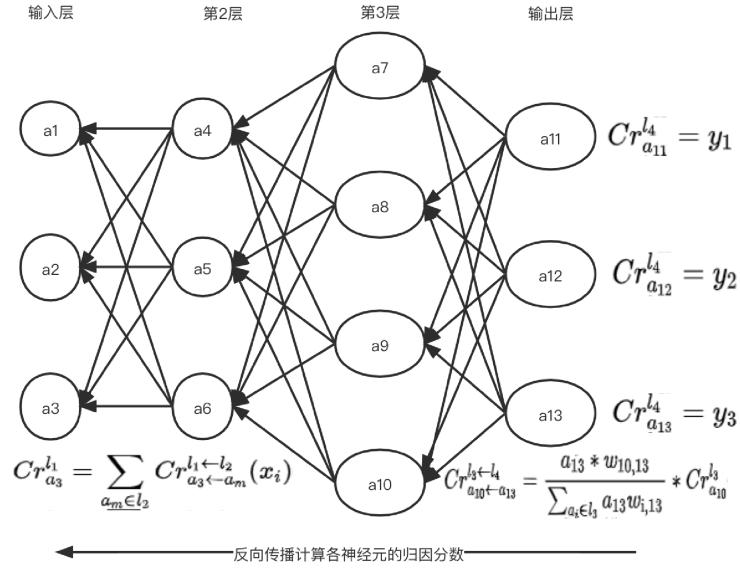


图 3.3: 逐层关联传播算法: 根据反向传播计算各神经元的归因分数

其中 μ 是一个无限接近于零, 但大于零的数字。从上述公式中, 我们可以认为每一层的贡献是相等的, 而且贡献是逐层传递的。根据以上公式的推导, 我们能得到神经网络模型中每一层以及每个神经元的贡献值。那么模型的输出可以表示为各层神经元归因分数的累加和:

$$\begin{aligned} \sum f(x_i, \omega_i^r) &= Cr_{a_{11}}^{l_4}(x_i) + Cr_{a_{12}}^{l_4}(x_i) + Cr_{a_{13}}^{l_4}(x_i) \\ &\quad + Cr_{a_7}^{l_3}(x_i) + Cr_{a_8}^{l_3}(x_i) + Cr_{a_9}^{l_3}(x_i) + Cr_{a_{10}}^{l_3}(x_i) \\ &\quad + Cr_{a_4}^{l_2}(x_i) + Cr_{a_5}^{l_2}(x_i) + Cr_{a_6}^{l_2}(x_i) \\ &\quad + Cr_{a_1}^{l_1}(x_i) + Cr_{a_2}^{l_1}(x_i) + Cr_{a_3}^{l_1}(x_i) \end{aligned}$$

通过从上述公式中提取同一属性的贡献, 我们可以计算出每个属性类对模型输出的平均贡献:

$$Cr_j(x_i) = \frac{1}{n} \sum_{i=1}^n Cr_{x_{i,j}}(x_i), j \in [1, u] \quad (3.3)$$

本文提出一种自适应噪声添加算法, 针对每个梯度计算其贡献值, 根据属性对于模型输出的贡献率添加对应比例的高斯噪声。在此方案中, 隐私预算 σ_l 是根据各自的归因分数按比例分配给每个梯度: $\sigma = \sigma_j = \frac{u * |\bar{Cr}_j|}{\sum_{j=1}^u |\bar{Cr}_j|} * \sigma_l$, 自适应高斯噪声

按以下方式注入梯度中:

$$\tilde{\mathbf{g}}_t \leftarrow \frac{1}{L} \sum_i (\bar{\mathbf{g}}_t(x_i) + \mathcal{N}(0, S_f^2 \cdot \sigma^2)) \quad (3.4)$$

算法2详细描述了梯度的自适应加躁算法，我们将自适应噪声添加算法与随机梯度下降算法相结合，关键的步骤包括：计算梯度及其归因分数，根据归因分数分配相应的隐私预算，在梯度上添加高斯噪声，最后进行梯度下降，在梯度下降的过程中不断消耗隐私预算，当隐私预算消耗殆尽时，模型停止训练，输出权重。

Algorithm 2 梯度的自适应加躁算法

- 1: 输入: 数据集 $\{x_1, \dots, x_N\}$, 损失函数 $\mathcal{L}(\theta) = \frac{1}{N} \sum_i \mathcal{L}(\theta, x_i)$, 学习率 η_t , 初始隐私预算 σ_l , 批大小 L
 - 2: 初始化: 模型权重 θ_0
 - 3: **for** $t \in [T]$ **do**
 - 4: 以概率 L/N 随机采样一批数据集 L_t
 - 5: **for** $x_i \in L_t$ **do**
 - 6: 根据逐层传播算法计算神经元对于模型输出的归因分数: $Cr_{a_i}^{l_m}(x_i) = \sum_{a_j \in l_{m+1}} Cr_{a_i \leftarrow a_j}^{l_m \leftarrow l_{m+1}}(x_i)$
 - 7: 计算神经元对模型输出的平均贡献: $Cr_j(x_i) = \frac{1}{n} \sum_{i=1}^n Cr_{x_i,j}(x_i), j \in [1, u]$
 - 8: 计算神经元对模型输出的贡献率: $\alpha = \frac{|Cr_j|}{\sum_{j=1}^u |Cr_j|}$
 - 9: 计算梯度: $\mathbf{g}_t(x_i) \leftarrow \nabla_{\theta_t} \mathcal{L}(\theta_t, x_i)$
 - 10: 根据贡献率分配相应的隐私预算: $\sigma = \frac{u * |Cr_j|}{\sum_{j=1}^u |Cr_j|} * \sigma_l$
 - 11: 在梯度上添加高斯噪声: $\tilde{\mathbf{g}}_t \leftarrow (\bar{\mathbf{g}}_t(x_i) + \mathcal{N}(0, S_f^2 \cdot \sigma^2))$
 - 12: **end for**
 - 13: 计算平均加躁梯度: $\tilde{w}^t = \frac{1}{L} \sum_{x_i \in L_t} \tilde{w}^t(x_i)$
 - 14: 梯度下降: $\theta^{t+1} = \theta^t - \eta^t \tilde{w}^t$
 - 15: **end for**
 - 16: 输出: θ_t
-

3.2.3 梯度的自适应裁剪算法

在传统的差分隐私随机梯度下降算法中，提供隐私保护的常用技术是限制函数的敏感度并添加与敏感度界限成比例的高斯噪声。为此，我们需要在每一轮 SGD 上限制梯度的敏感性。而且，相关研究表明合适的梯度裁剪能加快模型的收敛速度。

度。Abadi^[65] 等人提出通过梯度裁剪使得梯度保持在 [-C,C] 的范围内，以保证函数的敏感度有界。如果损失函数是可微的（如果不可微，则使用子梯度）和 Lipschitz 有界的，用 Lipschitz 界限制梯度范数，并用它来推导出梯度的敏感度。如果损失函数导数作为输入的函数有界（例如，在逻辑回归的情况下，可以通过最大可能的输入正则来限制梯度），从而推导出梯度敏感度。如果损失函数不像深度学习应用中那样具有已知的 Lipschitz 界，则很难推导出梯度范数的先验界。

在固定的梯度裁剪算法中，由于梯度的大小没有一个先验的界限，我们采用 ℓ_2 -范数的固定值对每个梯度进行裁剪。假设用户上传的梯度向量为 $\tilde{\mathbf{g}}_t$ ，根据固定梯度范数进行裁剪后，梯度缩放为 $\mathbf{g} / \max\left(1, \frac{\|\mathbf{g}\|_2}{C}\right)$ ，其中 C 是梯度阈值。对于梯度的裁剪能保证梯度值小于梯度阈值时，也就是当 $\|\mathbf{g}\|_2 \leq C$ ， \mathbf{g} 保持不变；当 $\|\mathbf{g}\|_2 > C$ 时，它会按照裁剪比例缩小为 C 。在每次训练迭代中，可以使用经验值来获得梯度正则的近似界限，并在损失函数近似界限处裁剪梯度。然而，经验值的可用性是一个强有力的假设，在没有经验值的情况下如何针对自适应添加的噪声裁剪梯度是一个难题。如果梯度阈值 C 的值太小，那么裁剪后的梯度会较小，算法添加的噪声较小时可能会破坏梯度估计的无偏性；另一方面，如果不对梯度进行裁剪，大量的噪声添加到每个梯度会导致模型的可用性大大降低。神经网络的架构、损失函数本身、数据的缩放都会影响裁剪范数的选择。本章节所设计的方案根据训练轮数和梯度变化的偏差与方差动态的更新梯度裁剪阈值，对梯度进行自适应裁剪。

在深度学习的模型训练中，模型的泛化能力取决于预测值的方差、偏差和数据的噪声。偏差度量的是模型预测值与真实值之间的偏离程度；方差度量的是训练数据的变动给模型预测结果带来的影响，也就是噪声的添加会影响梯度的方差，而随机梯度的方差决定了 SGD 算法的收敛速度，梯度的裁剪会影响偏差。因此我们更关注梯度更新的方差和偏差来决定如何对梯度进行裁剪。之前的梯度裁剪算法给梯度本身添加了额外的噪声，因此我们考虑根据训练时观察到的历史梯度的统计数据来设置梯度阈值，通过计算梯度更新的偏差和方差在每轮随机梯度下降中更新梯度梯度阈值。

首先，我们对梯度进行裁剪，裁剪后的梯度为 \hat{w}^t :

$$\hat{w}^t = \text{clip}(w^t, C^t) \triangleq w^t \cdot \min\left(1, \frac{C^t}{\|w^t\|_2}\right) \quad (3.5)$$

然后对保留的梯度 \hat{w}^t 根据神经元的归因分数添加高斯噪声 $\mathcal{N}(0, S_f^2 \cdot \sigma^2)$:

$$\tilde{w}^t = \hat{w}^t + N^t \quad N^t \sim \mathcal{N}(0, \sigma^2 I) \quad (3.6)$$

理想情况下，裁剪后的梯度对于模型的收敛的影响应该很小，因此我们希望在每一轮梯度下降算法中找到最佳的裁剪阈值 C^t 使 $\mathbb{E} \|\tilde{w}^t - w^t\|^2$ 最小。

根据三角不等式和 Jensen 不等式，更新前后的梯度方差与偏差可以表示为以下公式：

$$\text{bias}(\tilde{w}^t) \leq \text{bias}(w^t) + 2\mathbb{E} \|\tilde{w}^t - w^t\| \text{ 和 } \text{Var}(\tilde{w}^t) \leq 3\text{Var}(w^t) + 6\mathbb{E} \|\tilde{w}^t - w^t\|^2 \quad (3.7)$$

我们通过约束 $\mathbb{E} \|\tilde{w}^t - w^t\|$ 找到最佳的裁剪阈值 C^t ，将上述公式转换为：

$$\mathbb{E} \|\tilde{w}^t - w^t\|^2 = \|w^t\|^2 \left(1 - \frac{1}{\max(1, \|w^t\|)}\right)^2 + C^{t2} \sigma^2 \quad (3.8)$$

公式3.8中的第一项对应于变换后的梯度 w_t 可能被裁剪的情况，第二项对应于注入到裁剪梯度中的高斯噪声。理想情况下，我们希望能找到使上述表达式3.8最小化的裁剪阈值 C^t 。

为了使预测的梯度值的偏差最小，根据上一轮迭代得到的加躁梯度，通过指数渐进平均估计可得：

$$m^t = \beta_1 m^{t-1} + (1 - \beta_1) \tilde{w}^t \quad (3.9)$$

其中 β_1 是指数移动平均线的衰减参数。

为了使预测的梯度值的方差最小，假使梯度没有被裁剪时，根据 $\tilde{w}^t = w^t + C^t N^t$ ，从 $\mathbb{E} (\tilde{w}_i^t - m_i^t)^2$ 推导出 $\mathbb{E} (w_i^t - m_i^t)^2$:

$$\begin{aligned} \mathbb{E} (w_i^t - m_i^t)^2 &= \mathbb{E} (\tilde{w}_i^t - m_i^t)^2 + \mathbb{E} (C^t N_i^t)^2 + 2\mathbb{E} (-C^t N_i^t) (w_i^t + C^t N_i^t - m_i^t) \\ &= \mathbb{E} (\tilde{w}_i^t - m_i^t)^2 - \mathbb{E} (C^t N_i^t)^2 - 2\mathbb{E} (C^t N_i^t) (w_i^t - m_i^t) \\ &= \mathbb{E} (\tilde{w}_i^t - m_i^t)^2 - C^{t2} \sigma^2 \end{aligned}$$

我们需要确保 $(w_i^t - m_i^t)^2$ 满足上限和下限：

$$(w_i^t - m_i^t)^2 \approx \min \left(\max \left((\tilde{w}_i^t - m_i^t)^2 - C^{t2} \sigma^2, h_1 \right), h_2 \right)$$

其中， h_1 和 h_2 为常数，我们使用上式的指数移动平均值来估计方差：

$$\begin{aligned} v_t &= \min \left(\max \left((\tilde{g}_i^t - m_i^t)^2 - C^{t2} \sigma^2, h_1 \right), h_2 \right) \\ (s_i^t)^2 &= \beta_2 (s_i^{t-1})^2 + (1 - \beta_2) v_t \end{aligned} \quad (3.10)$$

梯度自适应裁剪算法在每个训练迭代时刻 t 设置梯度裁剪阈值 C^t ，其中每个迭代对应于一个 minibatch 的处理，接着跟踪训练过程中看到的每个批次的梯度规范。在每一轮的梯度聚合之后，根据公式3.9和3.10计算梯度变化的方差和偏差，更新梯度裁剪阈值 C^t 使 $\mathbb{E} \|\tilde{w}^t - w^t\|^2$ 最小。将自适应梯度裁剪应用到随机梯度下降算法中，具体算法如3所示。梯度自适应裁剪算法的动态性导致了 C^t 的自适应设置，该设置由数据、网络和损失动态决定，而不是由用户在训练初始阶段设置固定的裁剪值，根据神经网络各层的均值和统计特征进行梯度裁剪既能限制敏感度有界，也能保留有效的梯度信息。

Algorithm 3 梯度的自适应裁剪算法

- 1: 输入：数据集 $\{x_1, \dots, x_N\}$ ，损失函数 $\mathcal{L}(\theta) = \frac{1}{N} \sum_i \mathcal{L}(\theta, x_i)$ ，学习率 η_t ，隐私预算 σ ，批大小 L ，裁剪阈值 C^0
 - 2: 初始化：模型权重 θ_0
 - 3: **for** $t \in [T]$ **do**
 - 4: 以概率 L/N 随机采样一批数据集 L_t
 - 5: **for** $x_i \in L_t$ **do**
 - 6: 计算梯度： $\mathbf{w}_t(x_i) \leftarrow \nabla_{\theta_t} \mathcal{L}(\theta_t, x_i)$
 - 7: 梯度裁剪： $\hat{w}^t = \text{clip}(w^t, C^t) \triangleq w^t \cdot \min \left(1, \frac{c}{\|w^t\|_2} \right)$
 - 8: 在梯度上添加高斯噪声： $\tilde{w}^t \leftarrow (\hat{w}^t(x_i) + \mathcal{N}(0, S_f^2 \cdot \sigma^2))$
 - 9: **end for**
 - 10: 计算平均加躁梯度： $\tilde{w}^t = \frac{1}{L} \sum_{x_i \in L_t} \tilde{w}^t(x_i)$
 - 11: 梯度下降： $\theta_{t+1} \leftarrow \theta_t - \eta_t \tilde{\mathbf{w}}_t$
 - 12: 根据公式3.10和3.9计算梯度变化的方差和偏差，更新 C^t
 - 13: **end for**
 - 14: 输出： θ_t
-

3.2.4 基于自适应差分隐私的 SGD 算法

结合前两节所提出的自适应梯度加躁和裁剪算法，我们设计了算法4。在本地客户端训练过程中，在随机梯度下降算法中添加自适应噪声使算法整体满足 (ϵ, δ) -差分隐私，最小化目标函数 $f(\theta) = \frac{1}{N} \sum_{k=1}^N f_k(\theta)$ ，并使用自适应梯度裁剪算法更新梯度裁剪阈值。在随机梯度下降算法中，每一轮用户随机采样小批次的 B 个样本，对于样本集中的每条数据记录，根据逐层关联传播算法计算神经元的归因分数和梯度，根据梯度更新的方差和偏差选择合适的梯度裁剪阈值 C^t 裁剪梯度。完成梯度裁剪后，对梯度添加同归因分数等比例的高斯噪声。之后计算批次大小为 L 的样本集的平均加躁梯度，然后进行梯度下降，计算得到梯度更新的均值和标准差。之后不断重复采样迭代的进行梯度下降的训练，使目标函数最小，输出模型权重 θ_t 。

Algorithm 4 基于自适应差分隐私的随机梯度下降算法

- 1: 输入：目标函数 $f(\theta) = \frac{1}{N} \sum_{k=1}^N f_k(\theta)$ ，学习率 η^t ，训练批次大小 L ，高斯噪声参数 σ_l ，裁剪阈值 C^0
 - 2: 初始化： $m^0 = 0 \cdot 1, s^0 = \sqrt{h_1 h_2} \cdot 1$
 - 3: 随机初始化模型参数： θ^0
 - 4: **for** $t \in [T]$ **do**
 - 5: 以概率 L/N 随机采样一批数据集 L_t
 - 6: **for** $x_i \in L_t$ **do**
 - 7: 根据逐层传播算法计算神经元对于模型输出的归因分数： $Cr_{a_i}^{l_m \leftarrow l_{m+1}}(x_i) = \sum_{a_j \in l_{m+1}} Cr_{a_i \leftarrow a_j}^{l_m \leftarrow l_{m+1}}(x_i)$
 - 8: 计算神经元对模型输出的平均贡献： $Cr_j(x_i) = \frac{1}{n} \sum_{i=1}^n Cr_{x_{i,j}}(x_i), j \in [1, u]$
 - 9: 计算神经元对模型输出的贡献率： $\alpha = \frac{|Cr_j|}{\sum_{j=1}^u |Cr_j|}$
 - 10: 计算梯度： $\mathbf{g}_t(x_i) \leftarrow \nabla_{\theta_t} \mathcal{L}(\theta_t, x_i)$
 - 11: 梯度裁剪： $\hat{g}^t = \text{clip}(g^t, C^t) \triangleq g^t \cdot \min\left(1, \frac{c}{\|g^t\|_2}\right)$
 - 12: 根据贡献率分配相应的隐私预算： $\sigma = \frac{u * |Cr_j|}{\sum_{j=1}^u |Cr_j|} * \sigma_l$
 - 13: 在梯度上添加高斯噪声： $\tilde{g}^t \leftarrow (\hat{g}^t(x_i) + \mathcal{N}(0, S_f^2 \cdot \sigma^2))$
 - 14: **end for**
 - 15: 计算平均加躁梯度： $\bar{g}^t = \frac{1}{L} \sum_{x_i \in L_t} \tilde{g}^t(x_i)$
 - 16: 梯度下降： $\theta^{t+1} = \theta^t - \eta^t \bar{g}^t$
 - 17: 根据公式3.10和3.9计算梯度变化的方差和偏差，更新 C^t
 - 18: **end for**
 - 19: 输出： θ_t
-

在下一节，我们将给出自适应差分隐私 SGD 算法的隐私性证明和整体隐私预

算 (ϵ, δ) 分析。

3.3 隐私参数分析

自适应差分隐私 SGD 算法在梯度下降过程中根据神经元的归因分数在梯度上添加高斯噪声，算法整体满足 (ϵ, δ) -差分隐私。证明如下：

证明. 假设现有相邻数据集 D 和 D' ，两个数据集上的第 n 条数据记录 x_n 和 x'_n 不同，在模型上的输出分别为 F_θ 和 F'_{θ} ， $C(D)$ 和 $C(D')$ 分别代表这两个数据集上所有属性值的归因分数之和：

$$Cr(D) = \{Cr_j(x_i)\}, j \in [1, u], \text{ 其中 } Cr_j(x_i) = \frac{1}{n} \sum_{i=1}^n Cr_{x_{i,j}}(x_i), j \in [1, u], x_i \in D \quad (3.11)$$

$$Cr(D') = \{Cr_j(x'_i)\}, j \in [1, u], \text{ 其中 } Cr_j(x'_i) = \frac{1}{n} \sum_{i=1}^n Cr_{x'_{i,j}}(x'_i), j \in [1, u], x'_i \in D' \quad (3.12)$$

根据神经元的归因分数在其梯度 $\mathbf{g}_t(x_i) \leftarrow \nabla_{\theta_t} \mathcal{L}(\theta_t, x_i)$ 上添加自适应的噪声，其中隐私预算与归因分数成正比： $\sigma = \sigma_j = \frac{u * |\bar{Cr}_j|}{\sum_{j=1}^u |\bar{Cr}_j|} * \sigma_l$ ：

$$\tilde{\mathbf{g}}_t \leftarrow \frac{1}{L} \sum_i (\bar{\mathbf{g}}_t(x_i) + \mathcal{N}(0, S_f^2 \cdot \sigma^2)) \quad (3.13)$$

通过敏感度公式计算梯度归因分数的 L_2 敏感度 S_f ：

$$\begin{aligned} S_f &= \frac{1}{|D|} \sum_{j=1}^u \left\| \sum_{x_i \in D} Cr_{x_{i,j}}(x_i) - \sum_{x'_i \in D'} Cr_{x'_{i,j}}(x'_i) \right\|_2 \\ &= \frac{1}{|D|} \sum_{j=1}^u \left\| Cr_{x_{n,j}}(x_n) - Cr_{x'_{n,j}}(x'_n) \right\|_2 \\ &\leq \frac{2}{|D|} \max \sum_{j=1}^u \|Cr_{x_{i,j}}(x_i)\|_2 \\ &\leq \frac{2u}{|D|} \end{aligned} \quad (3.14)$$

其中, u 和 $|D|$ 分别表示神经网络属性和元组的数量, 然后可以得到:

$$\begin{aligned}
 \frac{\Pr(\ddot{C}r(D))}{\Pr(\ddot{C}r(D'))} &= \frac{\prod_{j=1}^u \exp\left(\frac{\epsilon_c \left\|\frac{1}{|D|} \sum_{x_i \in D} Cr_j(x_i) - \ddot{C}r_j(x_i)\right\|_2}{S_f}\right)}{\prod_{j=1}^u \exp\left(\frac{\epsilon_c \left\|\frac{1}{|D'|} \sum_{x'_i \in D'} Cr_j(x'_i) - \ddot{C}r_j(x'_i)\right\|_2}{S_f}\right)} \\
 &= \prod_{j=1}^u \exp\left(\frac{\epsilon_c}{|D|S_f} \|Cr_j(x_n) - Cr_j(x'_n)\|_2\right) \\
 &\leq \prod_{j=1}^u \exp\left(\frac{\epsilon_c}{|D|S_f} \max \|Cr_j(x_n)\|_2\right) \\
 &= \exp\left(\epsilon_c \frac{\max_{x_i \in D} \sum_{j=1}^u \|Cr_j(x_n)\|_2}{|D|S_f}\right) \\
 &\leq \exp(\epsilon_c)
 \end{aligned} \tag{3.15}$$

根据上述推倒证明可知, 在联邦学习的神经网络中添加自适应噪声后, 所上传的梯度是满足 ϵ_c -差分隐私的。在满足差分隐私的基础上, 在下一节我们结合差分隐私的组合定理, 计算累积的隐私预算。 \square

本章所提出的自适应差分隐私保护方案是通过在随机梯度下降算法上添加自适应的高斯噪声, 保护数据的隐私性。在上一部分我们已经证明了此算法满足 ϵ_c 差分隐私, 那另外一个非常重要的问题就是评估在训练过程中添加噪声所累积的隐私预算成本。

我们向梯度中添加高斯噪声以得到加噪后的数据, 根据第二章所给出的高斯机制的定理可知, 当隐私参数 $\sigma = \sqrt{2 \log \frac{1.25}{\delta}} / \epsilon$ 时, 每一批次的输出都满足 (ϵ, δ) -差分隐私。考虑到训练批次是以概率 $q = L/N$ 从数据集中随机采样的, 根据隐私放大定理和差分隐私的强组合定理, 隐私性由 (ϵ, δ) -差分隐私扩大到 $(q\epsilon, q\delta)$ -差分隐私。

然而, 组合定理并没有考虑到特定的噪声分布, 给出的隐私边界较为松散。在本节中, 我们采用“Moments Accountant”(MA) 机制, 去计算算法迭代过程中添加噪声所累积的隐私预算成本。MA 机制通过跟踪隐私损失随机变量随时间的变化, 并结合组合定理, 对于采样的高斯机制给出更严格的隐私损失估计。

隐私损失是一个随机变量，取决于添加到算法中的随机噪声。假使算法 \mathcal{M} 是满足 (ε, δ) -差分隐私的，那么 \mathcal{M} 中的隐私损失随机变量是存在严格的尾部边界。尾部边界在概率分布中是一个非常重要的信息。我们通过计算隐私损失随机变量的对数矩，结合标准的马尔科夫不等式，得到尾部边界，也差分隐私算法的隐私损失。

对于邻近数据集 D, D' ，算法 \mathcal{M} ，额外输入 aux ，算法的输出 $o \in \mathcal{R}$ 的隐私损失表示为：

$$c(o; \mathcal{M}, \text{aux}, D, D') \triangleq \log \frac{\Pr[\mathcal{M}(\text{aux}, D) = o]}{\Pr[\mathcal{M}(\text{aux}, D') = o]}$$

由于在随机梯度下降算法 \mathcal{M} 中，通过迭代的梯度下降更新权重，每一轮输出的梯度满足差分隐私，根据差分隐私的串行组合定理，最终的模型输出也满足差分隐私。我们定义第 λ^{th} 个时刻的时刻生成函数：

$$\alpha_{\mathcal{M}}(\lambda; \text{aux}, D, D') \triangleq \log \mathbb{E}_{o \sim \mathcal{M}(\text{aux}, D)} [\exp(\lambda c(o; \mathcal{M}, \text{aux}, D, D'))]$$

为了证明给定算法 \mathcal{M} 的隐私保障，我们对于每一轮的 $\alpha_{\mathcal{M}}(\lambda; \text{aux}, D, D')$ 给出严格的尾部边界，考虑到所有可能的额外输入和邻近数据集 (D, D') ，整体的时刻函数为：

$$\alpha_{\mathcal{M}}(\lambda) \triangleq \max_{\text{aux}, D, D'} \alpha_{\mathcal{M}}(\lambda; \text{aux}, D, D')$$

$\alpha_{\mathcal{M}}(\lambda)$ 满足串行组合定理和尾部边界定理：

定理 3.3.1 (时刻函数的串行组合). 假设算法 \mathcal{M} 是由一系列自适应算法 $\mathcal{M}_1, \dots, \mathcal{M}_k$ 组合而成的，满足 $\mathcal{M}_i : \prod_{j=1}^{i-1} \mathcal{R}_j \times \mathcal{D} \rightarrow \mathcal{R}_i$ ，那么对于任意的 λ 都满足：

$$\alpha_{\mathcal{M}}(\lambda) \leq \sum_{i=1}^k \alpha_{\mathcal{M}_i}(\lambda)$$

定理 3.3.2 (时刻函数的尾部边界). 对于任意的 $\varepsilon > 0$ ，算法 \mathcal{M} 是满足 (ε, δ) -差分隐私的，当

$$\delta = \min_{\lambda} \exp(\alpha_{\mathcal{M}}(\lambda) - \lambda \varepsilon)$$

根据定理3.3.2，在算法的每一次迭代中，通过计算 $\alpha_{\mathcal{M}_i}(\lambda)$ 然后限定其尾部边界，结合3.3.1，就可以得到整体时刻函数的尾部边界，通过隐私损失变量的尾部边界计算得到整体的隐私损失。

3.4 实验评估

3.4.1 实验准备

在这一节中，我们进行实验来评估本地自适应差分隐私在联邦学习系统中的性能。在模拟的联邦学习系统中，我们假设有四台本地设备，每个本地用户由配备 6GB 内存、四核 2.36GHz Cortex A73 处理器和四核 Cortex A53 1.8GHz 处理器的华为 nova3 安卓手机模拟。中央服务器由一台联想服务器模拟的，服务器有 2 个英特尔 (R) 至强 (R) E5-2620 2.10GHZ CPU，32GB 内存，512SSD，2TB 机械硬盘，运行于 Ubuntu 18.04 操作系统。

在实验过程中，我们选择了深度学习中常用的两个经典数据集-MNIST 手写体数字识别数据集^[46] 和 CIFAR-10 数据集^[68] 进行实验。

- MNIST 数据集包含 60000 个训练样本和 10000 个测试样本。MNIST 是用于分类任务的经典数据集，来源于美国国家标准与技术研究所。总共包含 60000 个训练样本和 10000 个测试样本。每个样本为 28x28 像素的手写数字图像，每个像素点用灰度值表示，灰度值范围为 0 到 255，图像包含十个类别，如下图3.4所示。
- CIFAR-10 数据集包含了十个类别的 RGB 彩色图像，总共包含 50000 个训练样本和 10,000 个测试样本。每张图像的大小为 32*32。这些图像有以下十个类别：飞机、汽车、鸟、猫、鹿、狗、青蛙、马、船和卡车。

此外，我们让所有用户离线训练一个统一的卷积神经网络，以获得本地用户的梯度。在我们的实验中采用的模型网络结构为 CNN，包括 2 个卷积层（分别包含 20 个特征图和 50 个特征图），两个池化层和两个全连接层（分别为 256 和 10 个

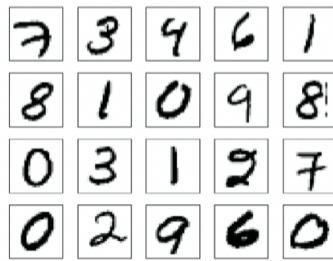


图 3.4: MNIST 手写数字数据集

神经元)。模型的激活函数为 ReLU，并引入了 Dropout 正则以提高模型的泛化能力。图3.5展示了 CNN 的网络结构。

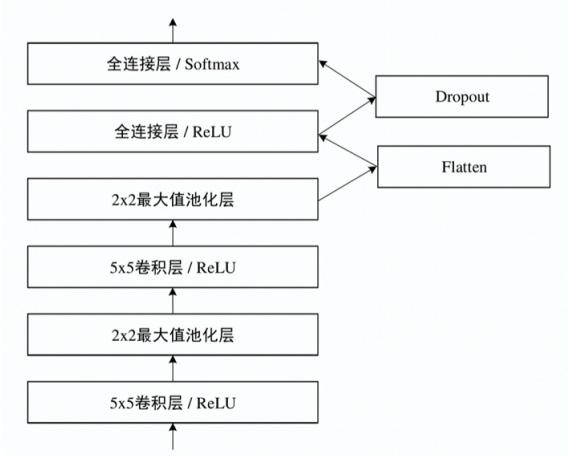


图 3.5: 模型网络结构

所有的实验都是用 PYTHON 语言编译的，我们使用 Tensorflow 去实现本地差分隐私算法，这是一个流行的深度学习库。本文使用了 TensorFlow-Federated，这是 TensorFlow 中的一个联邦学习库。我们在 Python 的基础上二次开发了该算法，并通过将该算法部署到多个边缘设备上构建了一个真实的联邦学习环境。以本地训练集 MNIST 为例，图3.6展示了联邦学习的仿真模型概览。

3.4.2 实验设计

为了实现隐私保护，我们需要在本地训练的随机梯度下降算法中实现梯度的自适应扰动和裁剪，并采用 MA 机制跟踪每一次梯度扰动所增加的隐私预算。因此，

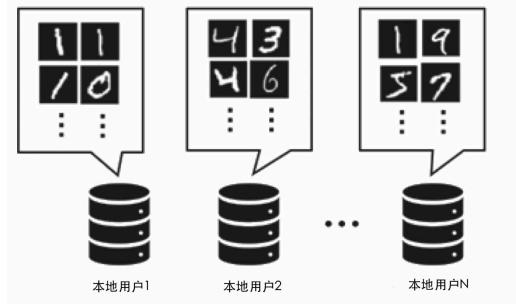


图 3.6: 仿真联邦系统模型概览

代码的实现主要分为两个部分：梯度扰动 (Sanitizer)，隐私预算跟踪 (Accountant)。

图3.7展示了实现梯度扰动和裁剪、隐私预算跟踪的代码片段，为了实现整体算法满足 ϵ -差分隐私，Sanitizer 需要完成以下三个步骤：首先，通过前向传播算法计算每批样本的梯度及其归因分数；其次，根据每个样本的梯度范数进行梯度裁剪以限制函数敏感度；最后，根据归因分数在梯度上添加自适应的噪声然后更新权重。

在 TensorFlow 中，由于性能原因，梯度计算是分批进行的，所以在训练过程，随机采样一批训练子样本 B : $\mathbf{g}_B = 1/|B| \sum_{x \in B} \nabla_{\theta} \mathcal{L}(\theta, x)$ 。为了限制梯度更新的敏感度，我们需要计算每个批次的梯度 $\nabla_{\theta} \mathcal{L}(\theta, x)$ ，具体由 `per_example_gradients` 函数实现。这样即使是大批量的训练，训练速度也不会大幅下降。在每个批次的训练中，我们会单独计算损失函数 \mathcal{L} ，也就是每个数据样本 x_i 都有单独的损失函数结果 \mathcal{L} 。一旦我们获得了每批数据样本的梯度，我们可以很容易地使用 TensorFlow 操作符来对梯度进行裁剪，添加高斯噪声。

我们的实验主要分为三个部分：

- (1) 针对本地自适应差分隐私 SGD 方案，分析噪声水平、裁剪阈值、隐藏层数量这些超参数对模型分类准确率影响。
- (2) 将本地自适应差分隐私 SGD 方案与非隐私的 SGD、前人提出的差分隐私 SGD 方案（如表所示）进行对比，比较各个方案在相同隐私预算的情况下模型分类所能达到的准确率和模型收敛速度。

```

class DPSGD_Optimizer():
    def __init__(self, accountant, sanitizer):
        self._accountant = accountant
        self._sanitizer = sanitizer
    def Minimize(self, loss, params, batch_size, noise_options):
        # Accumulate privacy spending before computing
        priv_accum_op = self._accountant.AccumulatePrivacySpending(batch_size, noise_options)
        with tf.control_dependencies(priv_accum_op):
            #计算每批样本的梯度和其归因分数
            px_grads, px_grads_socre = per_example_gradients(loss, params)
            #裁剪梯度
            px_grads = clip_gradients(px_grads, threshold)
            #梯度加躁
            sanitized_grads = self._sanitizer.Sanitize(px_grads, noise_options)
            #梯度下降
            return apply_gradients(params, sanitized_grads)
    def DPTrain(self, loss, params, batch_size, noise_options):
        accountant = PrivacyAccountant()
        sanitizer = Sanitizer()
        dp_opt = DPSGD_Optimizer(accountant, sanitizer)
        sgd_op = dp_opt.Minimize(loss, params, batch_size, noise_options)
        eps, delta = (0, 0)
        #只要隐私预算在预先设定的限度内，就继续训练。
        while within_limit(eps, delta):
            sgd_op.run()
            eps, delta = accountant.GetSpentPrivacy()

```

图 3.7: 实现本地自适应差分隐私的伪代码片段

- (3) 在本地自适应差分隐私的联邦学习模型上应用成员推理攻击进行实验，评估模型的隐私保护效用。

基准方案名称	具体算法
SGD	没有实现差分隐私的随机梯度下降算法
DP-SGD ^[57]	在梯度上添加固定噪声大小的差分隐私随机梯度下降算法
DS-SGD ^[67]	在梯度下降过程中，选择性的进行参数共享实现隐私保护
LDP-SGD	本地差分隐私方案
ADP-SGD	我们的改进方案，使用梯度自适应加躁与裁剪

表 3.1: 本地自适应差分隐私与其他四种基准方案

3.4.3 结果分析

实验一（分析各个参数对模型准确率的影响）

分类模型的精度由多个因素决定，这些因素包括网络的拓扑结构、隐藏单元的数量以及模型训练的参数，如批量大小和学习率，必须仔细调整以获得最佳性能。有些参数是针对隐私的，如梯度范数裁剪阈值和噪声水平。本节实验重点研究噪声大小，裁剪阈值和隐藏层数量这三种参数对于模型分类准确率的影响。为了准确的反映每种参数对于准确率的影响，我们控制变量的进行实验。参考值如下：1,000个隐性单元，600个批量，初始梯度范数裁剪阈值为4，初始学习率为0.1，在10个训练轮次中递减到最终学习率为0.052，噪声参数 σ 分别为2和5，用于训练CNN网络模型。对于每一种参数组合进行模型训练，直至隐私预算累积至 $(2, 10^{-5})$ -差分隐私。具体的实验结果如图3.8所示。

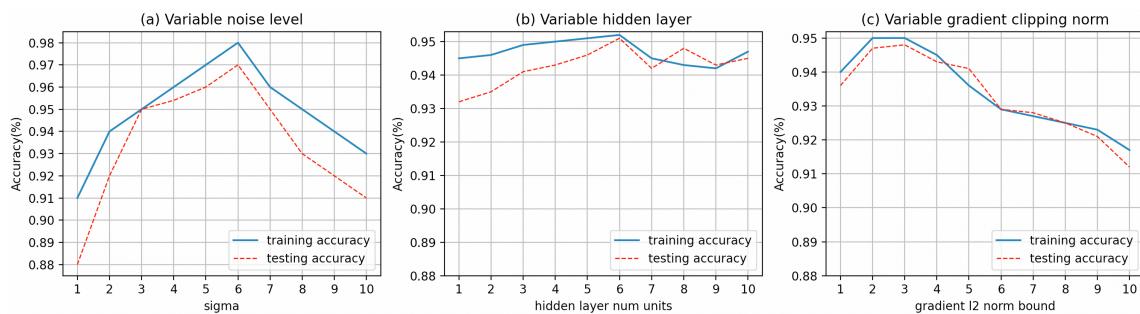


图 3.8: 在 MNIST 数据集上噪声大小，裁剪阈值和隐藏层数量这三个参数对于训练准确率的影响

如图3.8 (a) 展示的是噪声参数 σ 对模型准确率的影响，X轴是噪声水平，这个值的选择对准确性有很大影响。由于噪声参数 σ 与噪声采样的分布的方差是呈反比的，这意味着 σ 越大，添加的噪声量越小。通过添加更多的噪音，每轮训练步骤的隐私损失成比例缩小，所以我们可以在给定的累积隐私预算内运行更多的训练轮次。该模型在训练集和验证集上模型的准确率维持在0.88%至0.98%之间，我们的框架允许对训练参数进行自适应控制减少了过拟合情况，自适应的噪声添加根据训练轮次合理分配隐私预算，同时提高了模型的准确性和训练性能。

图3.8 (b) 展示了隐藏层数量对模型准确率的影响, X 轴表示隐藏层单元数量。对于非差分隐私模型, 更多的隐藏单元能有效避免过度拟合, 因为更多的隐藏单元会让我们的训练更有针对性。然而, 添加了差分隐私的模型训练, 隐藏层数量的增加可能会影响梯度的敏感度, 使每次梯度更新时添加更多的噪声。针对这个问题, 我们的根据梯度的归因分数自适应添加噪声的方案能有效的控制敏感度有界, 随着隐藏单位的数量增加, 模型的准确率依然维持在 93% 以上。

图3.8 (c) 展示了梯度范数裁剪阈值对模型准确率的影响, X 轴表示梯度 ℓ_2 -范数裁剪阈值 C^0 。当 C^0 为 2-3 时, 模型准确率最高, 接着随着 C^0 的增加, 模型准确率逐渐降低至 91% 左右, 限制梯度范数会产生两个相反的效果: 剪裁破坏了梯度估计的无偏性, 如果剪裁参数太小, 被剪切的平均梯度可能与真实梯度的方向大不相同。另一方面, 增加裁剪阈值迫使我们在梯度中加入更多的噪声, 也就是以 σC^0 的比例添加噪声。而我们的自适应梯度裁剪方案能有效的考虑上一轮训练得到的梯度偏差和方差, 取训练过程中未被剪辑的梯度范数的中值, 模型准确率最高依然能达到 95% 左右。

对于不同隐私预算的训练版本, 我们用同样的架构进行了实验: 一个包括 1000 个神经元的 ReLU 隐藏层, 以及 600 个批量大小。为了限制敏感度, 我们设置初始梯度范数裁剪阈值 C^0 为 3。我们报告了四种隐私参数的训练结果, 分别为小 ($\epsilon=0.5$)、中 ($\epsilon=2$)、大 ($\epsilon=4$) 和更大 ($\epsilon=8$), 固定 $\delta = 10^{-5}$, $\sigma=6$, 这里 ϵ 代表训练神经网络的隐私保护水平。学习率最初设置为 0.1, 在 10 个训练回合后线性下降到 0.052, 然后固定为 0.052。

图3.9显示了不同隐私预算下的训练的结果, 在每张图中, 我们都显示了训练集和测试集上准确率的变化情况。对于隐私预算为 $(0.5, 10^{-5})$, $(2, 10^{-5})$, $(4, 10^{-5})$ 和 $(8, 10^{-5})$ 的差分隐私, 分别达到 81%、94%、95% 和 97% 的测试集准确性。由图3.9 (a) 所示, 当 $\epsilon=0.5$ 时, 由于给定的隐私预算较小, 在训练的第 30 个轮次隐私预算消耗殆尽, 模型基本趋近收敛, 模型的准确率最高达到 81%。虽然在传统的差分隐私保护中, $0 < \epsilon < 1$ 时被认为能提供较强的隐私保护。而在联邦学习的深度神经网络

中应用差分隐私时,由于网络的复杂性和训练多次大量迭代,导致隐私预算消耗的很快, $0 < \epsilon < 1$ 的取值会导致模型训练的精度大大下降。在深度学习方面应用差分隐私的大量研究表明,当 $0 < \epsilon \leq 10$ 时,能提供较强的隐私保护效果。如图3.9(b)(c)(d)所示,当 $\epsilon=2, 4, 8$ 时,随着训练轮数的增加,模型均能达到收敛。我们的自适应差分隐私 SGD 方案,使模型在训练集和测试集上的准确度差异很小,这与理论上的观点一致,即添加噪声后的梯度训练依然有很好的泛化作用。相比之下,非差分隐私 SGD 的训练和测试准确率之间的差距随着训练轮次的增加而增加,容易造成过拟合。在噪声参数为 $(8, 10^{-5})$ 时,模型在 600 个训练轮次后能达到接近非差分隐私模型的准确率。

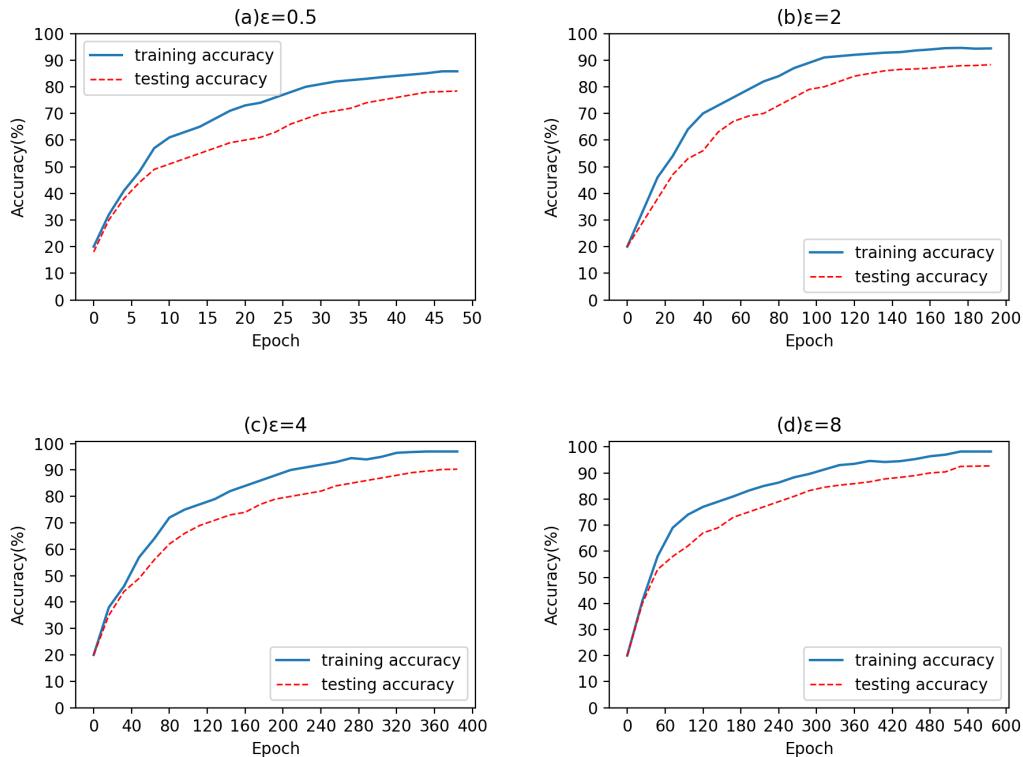


图 3.9: 在 MNIST 数据集上不同隐私预算下训练的准确率

实验二（与前人的隐私保护方案进行对比实验）

我们将本文提出的本地自适应差分隐私方案（ADP-SGD）与 SGD、DP-SGD、DS-SGD、LDP-SGD 方案进行对比实验，选取的数据集为 MNIST 和 CIFAR-10，网络模型为 CNN5，具体的参数设置如下表所示。我们比较了不同方案在给定相同的隐私预算情况下，在测试集上所计算的目标函数的平均损失误差变化情况。

参数	MNIST	CIFAR-10
隐私预算 ϵ	0.5/2/4/8	0.5/2/4/8
批大小	600	600
初始梯度裁剪阈值	3	3
学习率	0.05	0.05
本地设备数量	1000	100
训练轮数	100	100

表 3.2: 对比实验在数据集 MNIST 和 CIFAR-10 上的参数设置

图3.10显示了当隐私预算为 $(0.5, 10^{-5})$, $(2, 10^{-5})$, $(4, 10^{-5})$ 和 $(8, 10^{-5})$ 时，不同方案在 MNIST 数据集上平均测试误差随训练轮次的变化情况。

算法	隐私预算	
	$2, 10^{-5}$	$4, 10^{-5}$
SGD	98.6%	98.6%
DP-SGD	88.9%	94.0%
DS-SGD	88.9%	94.0%
LDP-SGD	88.9%	94.0%
ADP-SGD	94.6%	95.7%

表 3.3: 本地自适应差分隐私与其他四种基准方案

首先，对于 MNIST 数据集，在没有添加差分隐私保护的原始 CNN 模型上进行梯度下降训练，经过 20 个训练轮次后模型在训练集上得到的基准准确率为 98.6%，我们的方案（Adaptive Differential Privacy-SGD,ADP-SGD）在训练刚开始的一个轮次，训练集的训练误差下降较慢，这是由于刚开始训练时模型中所有梯度的归因

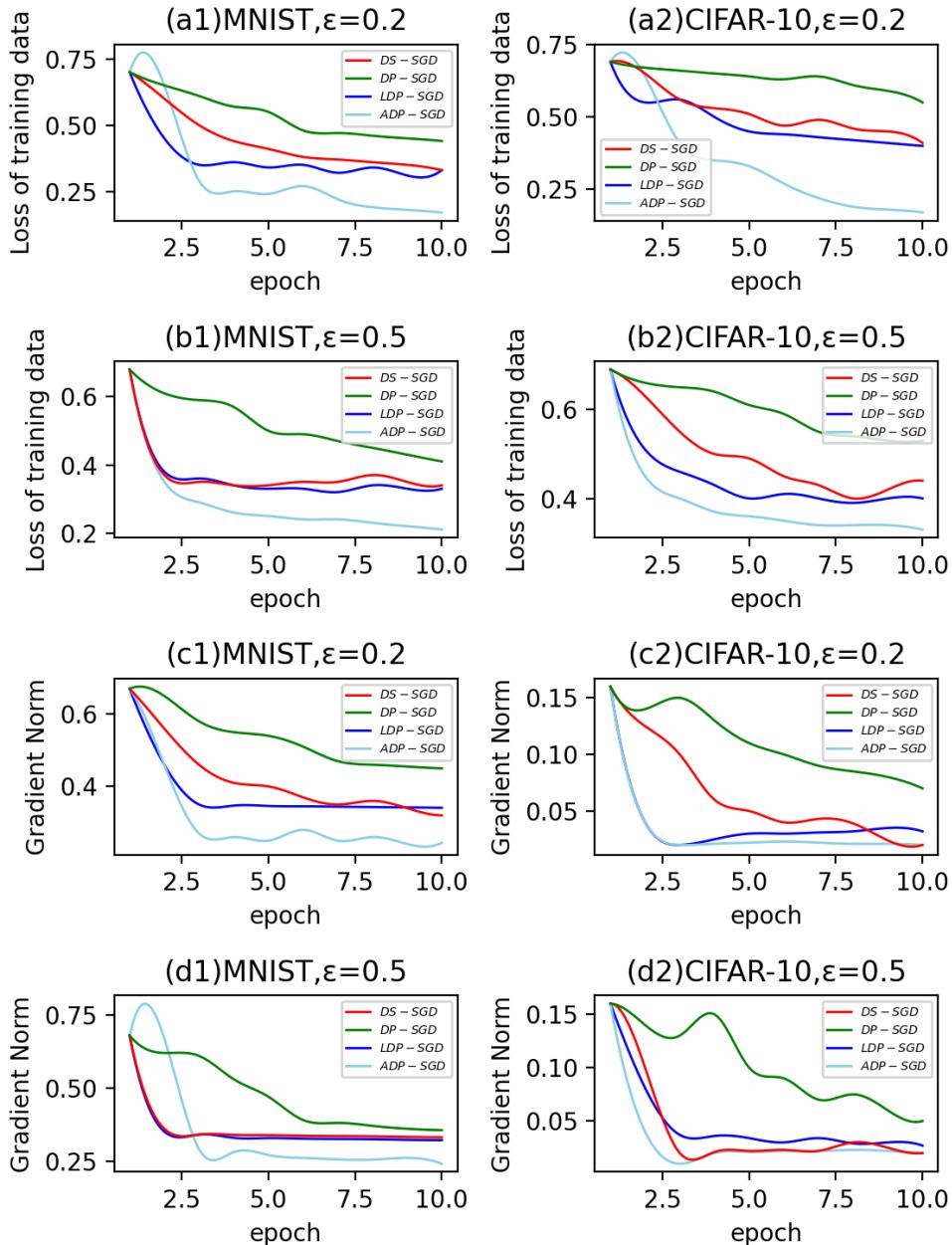


图 3.10: 不同隐私保护方案在 MNIST 数据集上训练的测试误差变化情况

分数较高，导致对于每个梯度分配的隐私预算较高，加躁后与原始值差异较大。然而，在 20 个轮次过后，模型收敛的速度远远超过其他三个基准差分隐私方案。在 50 个训练轮次之后，训练集的损失率降低至 25% 以下，而 DS-SGD 和 DP-SGD 的

训练损失值在 10 个轮次之后仅降低至 35% 左右，DP-SGD 更次之，在 40% 左右。图3.10中的（c1）和（c2）展示了模型训练中梯度的下降情况，在相同的隐私预算下，ADP-SGD 能在 2 个训练轮次，梯度范数接近 0.01 且趋于平稳，意味着模型趋近收敛的速度最快。

其次，对于 CIFAR-10 数据集，由于图像本身复杂度的增加，在没有添加差分隐私保护的原始模型上进行梯度下降训练，经过 20 个训练轮次后模型在训练集上得到的基准准确率为 97%。LDP 在第一个训练轮次的模型收敛速率最高，然而在第二个训练轮次过后，ADP-SGD 达到 35% 的模型准确率和 0.05 左右的梯度范数，训练数据损失值和梯度范数均比另外三个基准方案的效果优。

综上，我们的方案在调整梯度自适应加躁和自适应裁剪后使得模型收敛率和准确率大大提升，无论是在模型的准确率还是收敛速度方面都更加接近原始无隐私保护的模型。

实验三（针对攻击模型，分析该方案的隐私保护效用）

我们曾在第一章介绍了针对联邦学习模型的隐私攻击，其中成员推理攻击是最流行的一类攻击，旨在确定一个输入样本 x 是否存在于模型训练集 D 中。该攻击只攻击者假设知道模型的输出预测向量（黑盒），并且是针对有监督的机器学习模型进行的。为了推断成员属性，对抗者使用与目标模型的相同算法训练了很多影子模型。对于每个影子模型，对手随机选择一些数据样本来形成一个训练集和一个验证集。影子模型是在训练集上训练的。然后，他将训练特征数据和验证特征数据都送入影子模型，并获得相应结果，即每个类别的概率。所以对手可以建立一个分类器，以特征数据的预测结果为输入，以数据是来自训练集（表示为 1）还是验证集（表示为 0）为标签。有了这个分类器，敌手将考虑为敏感样本的特征值发送到目标模型并检索预测结果。然后，他将预测结果发送给分类器，并得到敏感样本是否是目标模型训练集的成员的结论。

我们重现了文献^[70] 中的成员推理攻击算法，配置和参数与之相同：目标模型是卷积神经网络，数据集为 CIFAR-10。我们为目标模型和影子模型选择了两种规

模的训练集：2500 和 10000，验证集与训练集的大小相同。敌手用不同的训练和验证样本子集运行 100 个影子模型，使用逻辑回归算法根据预测结果对样本是否包含在训练集中进行分类。

该实验的评估指标为模型的隐私保护效用，因此我们对比了在自适应差分隐私和无隐私保护的模型上进行成员推理攻击的攻击准确率，结果如图3.11所示。

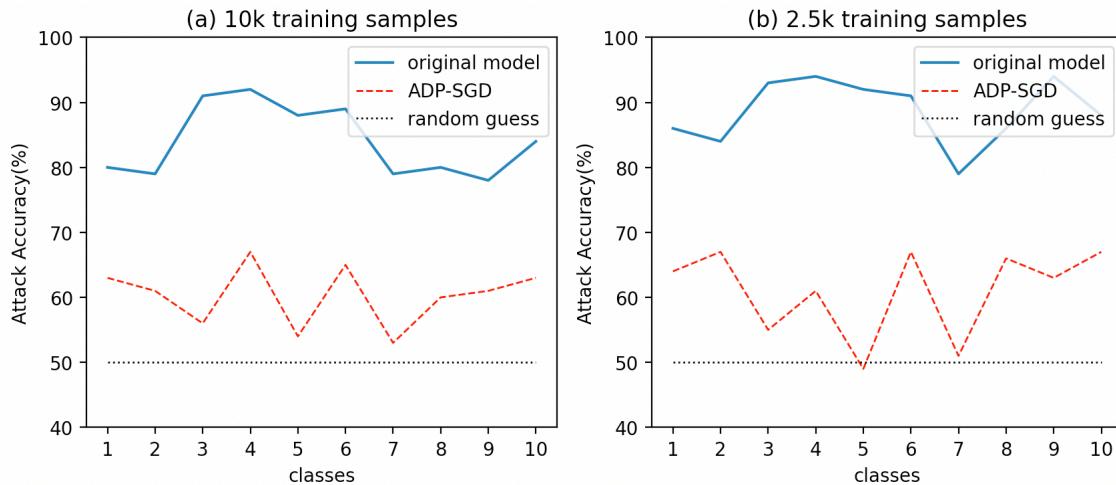


图 3.11: 在不同模型上进行成员推理攻击的准确率

我们分别在 10k 和 2.5k 个数据样本上进行攻击实验，图3.11中的 (a) 表示在 10k 数据样本上进行攻击实验，x 轴表示对抗攻击的类别，蓝线（original model）表示在原始无隐私保护的模型上进行成员推理攻击的各个类别的攻击准确率，在不同类别上基本都可达到 80%~90% 的准确率。我们可以看到，敌手对识别包含在训练集中的样本有很高的置信值。当样本不在训练集中时，错误率相对较高。图3.11中的红线（ADP-SGD）显示了在添加了自适应差分隐私的模型上进行成员推理攻击的准确性。基准线是 50%（黑色虚线），这是敌手通过随机猜测达到的准确率。在应用差分隐私的情况下，敌手的推断准确率下降到 50%~65%，接近于随机猜测。这比原始模型的攻击准确率要低得多。而在 2.5k 的数据样本上进行攻击，我们的方案针对攻击实验能在两个类别上将准确率降低至 50% 左右，证明了本地自适应差分隐私针对成员推理攻击的隐私保护效用。

3.5 本章总结

本章详细介绍了如何在深度学习模型的随机梯度下降算法中添加自适应的高斯噪声以及对梯度进行自适应裁剪。我们设计了一个自适应噪声添加的算法，在神经网络前向传播算法中，根据梯度对于模型输出的贡献率注入不同程度的隐私预算的噪声。与传统的注入噪声的方法相比，本文的方案在联邦学习的架构下避免了由统一本地用户的敏感度而噪声的隐私预算的浪费，在相同的隐私保护程度下最大限度地提高了模型的准确性，并且证明了算法能满足 ϵ_c -差分隐私。接着，本文设计了一种自适应调整剪裁阈值的方案，通过计算梯度更新的方差和偏差，逐元素地对梯度进行裁剪，与之前的方案相比，通过使用梯度的自适应剪裁实现了相同的隐私保证，而模型的收敛速度大大提升。之后我们利用“*Moments Accountant*”机制分析加噪累积产生的隐私预算，与传统的注入噪声的方法相比，我们在相同的隐私保护程度下大大减少了噪声对模型输出结果的影响，提高了模型的准确性。然后，我们在 MNIST 和 CIFAR-10 数据集上分别进行实验，评估了方案的隐私保护效用以及对模型性能的影响。

然而，在客户端的本地数据集添加的噪声只能保证本地数据的匿名性，不能够防止外部攻击者针对通信信道的攻击。如果客户端在每次迭代中同时上传了大量的权重更新，中央云服务器仍然可以将它们链接在一起，推导出参数信息。而且，当参与一次迭代的客户端数量达到上千人时，会导致聚合任务升级成一个高维任务，隐私预算暴增。因此，下一章我们对联邦学习模型框架进行了改进，在现有的联邦学习模型上新增混洗器，实现联邦学习框架的隐私安全，提高整体联邦学习模型的精度。

第四章 基于 Top-K 安全混洗的联邦学习模型

4.1 引言

上一章节中所提出的本地自适应差分隐私方案是通过在客户端将梯度上传至参数服务器前，对梯度添加自适应噪声，尽管方案采用了本地差分技术减少一定程度的隐私预算，但 Truex 等人^[49]指出的，一个复杂的隐私保护系统将多个本地差分隐私的算法进行组合，会导致这些算法的隐私成本增长。也就是说，隐私预算为 ϵ_1 和 ϵ_2 的本地差分算法的组合会消耗的隐私预算总和为 $\epsilon_1+\epsilon_2$ 。使用联邦学习训练的联合模型需要客户在多次迭代中向中央服务器上传梯度更新。如果在迭代训练过程中的每一次迭代都应用本地差分隐私，隐私预算就会累积起来，从而导致总隐私预算的爆炸。在实际的联邦学习应用场景中，本地客户端的数量可能超过千万量级，中央服务器对所有本地上传的加躁梯度进行聚合时，可能因为噪声量的聚合而导致原有的梯度信息被累积的噪声淹没。现有的本地差分隐私协议对于多维聚集的联邦学习框架可能是不可行的，局部噪声带来的误差会随着维度系数的增加而加剧，从而大大降低模型的精度。而且，当参与一次迭代的客户端数量达到千万量级时，会导致聚合任务升级成一个高维任务，隐私预算暴增^[43]。在本地设备的模型训练上采用差分隐私技术，对于聚合后的梯度平均估计误差能达到 $O\left(\frac{\sqrt{d \log d}}{\epsilon \sqrt{n}}\right)$ 。

在联邦学习的背景下，传统的经验风险最小化(ERM)存在以下挑战：(i) 需要为客户的 data 提供隐私保证，(ii) 压缩客户和服务器之间的通信，因为客户提供的连接可能为低带宽。提高联邦学习的通信效率分为两种策略，其一，在联邦学习训练期间减少服务器和客户端之间的通信回合数；其二，在每一次服务器和客户端

的通信过程中传输更少的参数。现有的研究包括使用静态抽样来选择一部分客户模型参与全局更新，或者对客户端上传的参数使用压缩算法来提高通信效率。

在最近的研究工作中，人们提出了一个新的隐私框架，使用匿名化的方式上传模型参数到中央服务器，即所谓的安全混洗模型 (Secure Shuffle Model, SS)。安全混洗是一个针对 n 个客户和一个中央服务器的隐私保护框架。它使每个客户可以提交一个或多个消息，服务器从所有客户那里了解到一个无序的消息集合，除了本地客户上传的信息，服务器没有能力将任何信息与信息所属的客户联系起来。

安全混洗模型的准确性增益来自于隐私放大效应，对本地设备的输出进行混洗后在差分隐私的中心视图中比没有混洗的输出提供更强的隐私效用。因此，在安全混洗模型中，对于不受信任的中央服务器要达到相同的隐私保护水平，所需要添加的本地噪音更少。然而，目前还不清楚如何在联邦学习中使用 SS 模型。虽然有一些作品对基本任务进行了研究，如位/实数求和与直方图，但现有的协议对于多维聚合的联邦学习可能是不可行的。

为了应对以上两种挑战，本文根据前人的研究思路，设计了客户端梯度的 Top-K 采样算法和参数的拆分混洗算法，有效地减少了服务器和客户端的通信过程中传输数据的带宽，提供通信性能的同时也保证了算法整体满足差分隐私。

4.2 模型设计

在安全混洗模型中，我们假设敌手为恶意的第三方服务器和中央服务器，因为它们持有用户本地梯度的所有加密版本。在我们的威胁模型中，我们假设这两种服务器是诚实而好奇的，这意味着每个服务器都诚实地遵守预先商定的程序来完成其任务。然而，它也可能试图通过利用掌握的先验知识来损害用户的数据隐私。此外，我们假设第三方服务器和中央服务器之间不存在串通，混洗器和中央服务器之间不存在串通。

在上述威胁模型下，我们将隐私要求表述如下：

- 用户的本地梯度的保密性：敌手如云服务器，可以通过利用共享梯度和全局

参数来恢复用户的敏感信息，如数据标签和成员信息。为了保护用户的隐私，每个用户的本地梯度在被发送到服务器之前应该通过安全加密。

- 用户所选择的 Top-K 梯度值对应的索引信息：虽然用户上传的梯度值是添加噪声之后的，但是由于梯度的绝对值和其索引信息是一并发送给混洗器的，本方案需要约束中央服务器成功预测一个索引是否在用户本地向量上传的 Top-k 元素中。
- 对用户的可靠性和聚合结果进行隐私保护：为了使学习过程公平和非歧视性，每个用户的可靠性，即用户的“数据质量”信息，应该被保密，在训练过程中不能被服务器和任何用户获取。另一方面，模型聚合的结果可以被视为有价值的知识产权，它是用大量的资源产生的，甚至包含一些用户的专有信息。因此，除了参与训练的用户之外，聚合的结果对敌手来说应该是保密的。

本文基于梯度稀疏化的思想，创造性地开发了 Top-k 梯度选择算法，应用在安全混洗框架中，与随机扰动相结合，设计了满足 $(\epsilon_1 + \epsilon_2)$ -本地差分隐私的算法。在本文，我们将客户端采样和梯度混洗这两种隐私放大效应相结合，实现的方案能提高全局模型的精度，也保证在更低的隐私成本下达到相同的隐私预算，且降低了通信成本，缓解了由维度系数增加而带来的隐私预算暴增和模型精度下降的问题。

我们将在本节详细的描述该框架中各个模块的设计和实现过程。

4.2.1 模型概览

如图4.1所示，该框架主要由本地客户端、混洗器和中央服务器 3 部分组成：

- 本地客户端：本地设备通过模型训练得到梯度向量 \mathbf{g} ，通过 Top-K 梯度选择算法计算得到满足 $(\epsilon_1 + \epsilon_2) - LDP$ 的索引列表和梯度元素列表。
- 混洗器：对梯度进行拆分混洗，通过隐私放大效应使得算法满足 ϵ_0 -差分隐私，达到梯度匿名机制，最后将混洗后的结果发送至中央服务器。

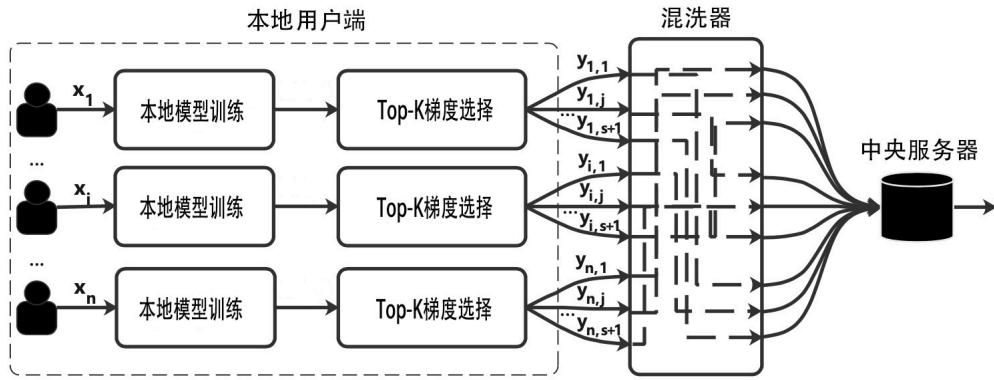


图 4.1: 基于安全混洗和 Top-K 梯度选择算法的联邦学习模型框架

- 中央服务器：一个诚实但好奇的第三方。服务器接受混洗器上传的梯度并进行聚合，然后更新全局模型。

本地客户端在本地进行模型训练后得到梯度向量，根据向量中每一元素的绝对值进行降序排序，以较大的概率选择为 top-K 元素的梯度值，在梯度上添加拉普拉斯噪声，再以较小的概率选择非 top-K 元素的梯度值，将得到的索引列表和梯度元素列表上传至安全混洗器。然后安全混洗器将收集到的梯度以维度进行拆分，打乱次序，达到隐私放大效果，再发送给中央服务器进行聚合。安全混洗器独立于服务器并专门用于本地客户端梯度的子采样、拆分混洗、上传。这个模型通过子采样和拆分混洗两者的结合达到隐私放大效应，降低了隐私预算，从而提高了整体联邦学习模型的精度。当本地差分隐私添加更少的噪音时，对于同样的中央服务器能达到相同水平的隐私预算，但通信成本要低得多。

假设现在有 m 个本地客户端，每个客户端表示为 $i \in [m]$ ，有本地数据集 $\mathcal{D}_i = \{d_{i1}, \dots, d_{ir}\} \in \mathbb{S}^r$ ，由 r 个数据集合构成。 $F_i(\theta)$ 表示在客户端 i 的本地数据集 \mathcal{D}_i 上进行训练，对于模型梯度 $\theta \in \mathbb{R}^d$ 进行衡量的损失函数，其中 $F_i(\theta) = \frac{1}{r} \sum_{j=1}^r f(\theta; d_{ij})$ ， $f(\theta; \cdot) : \mathcal{C} \rightarrow \mathbb{R}$ 是凸函数。中央服务器的目标是找到一个最佳的模型参数向量 $\theta^* \in \mathcal{C}$ 使得损失函数 $\min_{\theta \in \mathcal{C}} (F(\theta) = \frac{1}{m} \sum_{i=1}^m F_i(\theta))$ 最小，其中隐私性满足单个客户端的隐私预算，也就是满足 ϵ -差分隐私。

在算法5中，首先我们从 m 个客户端中随机挑选 k 个客户端，表示为集合 \mathcal{U}_t ，

其中 $k \leq m$ 。每个客户端 $i \in \mathcal{U}_t$ 从本地数据集中抽样 \mathcal{S}_{it} 个样本训练模型，计算梯度 $\nabla_{\theta_t} f(\theta_t; d_{ij})$ 。第 i 个客户端根据 Top-K 梯度选择算法对梯度进行采样扰动，得到满足 $(\epsilon_1 + \epsilon_2) - LDP$ 的梯度元素列表和索引列表 $\langle index_{i,j}, y_{i,j} \rangle$ 。混洗器对收到的梯度 $\langle y_{i,j} \rangle$ 进行拆分混洗（梯度维度的拆分、输出随机排序的结果），然后发送给中央服务器。最后，中央服务器对混洗后的梯度进行聚合求均值，更新全局模型。

Algorithm 5 联邦学习中的安全混洗算法: \mathcal{A}_{csdp}

```

1: 输入: 数据集  $\mathcal{D} = \bigcup_{i \in [m]} \mathcal{D}_i, \mathcal{D}_i = \{d_{i1}, \dots, d_{ir}\}$ , 损失函数  $F(\theta) = \frac{1}{mr} \sum_{i=1}^m \sum_{j=1}^r f(\theta; d_{ij})$ ,  

   本地差分隐私预算  $\epsilon$ , 梯度范数阈值  $C$ , 模型学习率  $\eta_t$ , K  

2: 初始化:  $\theta_0 \in \mathcal{C}$   

3: for  $t \in [T]$  do  

4:   本地更新:  

5:   for 客户端  $i \in \mathcal{U}_t$  do  

6:     for 样本  $j \in \mathcal{S}_{it}$  do  

7:        $\mathbf{g}_t(d_{ij}) \leftarrow \nabla_{\theta_t} f(\theta_t; d_{ij})$   

8:       梯度裁剪:  $\mathbf{g}_t(d_{ij}) \leftarrow \mathbf{g}_t(d_{ij}) / \max \left\{ 1, \frac{\|\mathbf{g}_t(d_{ij})\|_p}{C} \right\}^3$   

9:       Top-K 梯度选择:  $\langle index_{i,j}, y_{i,j} \rangle = \text{Top-k}(\mathbf{g}_t(d_{ij}), K, \epsilon)$   

10:    end for  

11:    客户端 i 将  $\langle index_{i,j}, y_{i,j} \rangle$  发送给混洗器  

12:  end for  

13:  全局更新:  

14:  混洗器对于  $\langle index_{i,j}, y_{i,j} \rangle$  中的权重进行拆分混洗，然后上传给中央服务器  

15:  中央服务器聚合梯度:  $\bar{\mathbf{g}}_t \leftarrow \frac{1}{ks} \sum_{i \in \mathcal{U}_t, j \in \mathcal{S}_{it}} \langle y_{i,j} \rangle$   

16:  梯度下降:  $\theta_{t+1} \leftarrow \prod_{\mathcal{C}} (\theta_t - \eta_t \bar{\mathbf{g}}_t)$   

17: end for  

18: 输出: 最终全局模型参数  $\theta_T$ 

```

4.2.2 Top-K 梯度选择算法

在神经网络的训练过程中，每次迭代的梯度都是从训练样本的小批量子样本中计算得到。直观地说，应用于子样本的算法比应用于全样本的算法具有更强的隐私保证，这种隐私的放大是由于一条数据如果没有在子样本中被选中，就会享有完美的隐私。因此我们在本地用户上传的梯度向量中进行子采样，上传到混洗器。

然而随机子抽样对所有梯度向量的维度一视同仁，因此可能会丢弃“重要”的维度。每个本地客户端从 d 维的梯度向量中随机采样并扰动 k 个维度，扰动的值被放大了 d/k 倍以获得无偏的平均估计，因此注入的噪声也被放大了。对于梯度向量高维的情况（比如，梯度是 n 维向量，算法从中随机抽取 k 个维度的梯度值，当 $n \gg k$ 时），从一个向量中随机抽出一小部分 β 值就会减慢训练的收敛率。根据文献^[78] 提出的梯度稀疏化技术，作者通过移除梯度向量中绝对值最小的 $R\%$ 的梯度，使梯度更新算法稀疏化。由于绝对值较小的梯度会随着时间的推移而累积，会破坏模型的收敛性。梯度稀疏化技术通过评判梯度的重要性，仅上传更重要的梯度值而降低通信成本，避免维度爆炸。本文根据梯度稀疏的想法设计了 Top-K 梯度选择算法，它的主要思想是基于这样一个事实，即具有较大绝对值的梯度可以为模型收敛做出更多贡献，并基于差分隐私选择的技术。

具体来说，算法发生在本地客户端进行随机梯度下降算法得到梯度向量后、将梯度向量上传至混洗器前。首先，本地用户通过 SGD 算法得到梯度向量 \mathbf{g} ，求得向量中的每一个元素 $\mathbf{g}[i]$ 的绝对值 $abs(\mathbf{g}[i])$ ，然后根据每一维度的绝对值进行降序排序，得到绝对值最大的 K 个梯度值。

因为算法的思想是具有较大绝对值的梯度可以为模型收敛做出更多贡献，可以理解为具有最大绝对值的维度应该以最高的概率输出。根据文献^[35] 提出的指数机制，指数机制是对于任意非数值型的查询，返回最佳的响应，并且满足差分隐私的一种隐私保护机制。指数机制是为以下情况设计的：例如在拍卖中设定价格，目标是使收益最大化，而向最优的价格添加上少量的正噪音（为了保护投标的隐私）会大大减少所产生的收入。对于用户的每个投标价格是需要满足隐私的，但我们

又不希望在价格上添加过多的噪音。在本文的场景下，我们希望选择最佳的梯度值，但直接向梯度中添加噪音会完全破坏模型收敛，两个场景的核心问题是相似的。所以首先考虑通过指数机制实现满足差分隐私的 top-k 梯度选择算法。

在第二章的基础知识中，我们已经介绍了差分隐私中的指数机制。对于一个任意的范围 \mathcal{R} ，通过定义一个实用性函数 $u : \mathbb{N}^{|\mathcal{X}|} \times \mathcal{R} \rightarrow \mathbb{R}$ ，为用户的数据打分。比如本文中的梯度绝对值越高，那么实用性函数所给的评分也越高。直观地说，用户通过指数机制中的实用性函数能得到给定数据库 X 中的效用评分最高的数据记录，即 top-1 数据。实用性函数的敏感度表示为：

$$\Delta u \equiv \max_{r \in \mathcal{R}} \max_{x, y: \|x-y\|_1 \leq 1} |u(x, r) - u(y, r)|$$

对于指数机制 $\mathcal{M}_E(x, u, \mathcal{R})$ ，按照正比于 $\exp\left(\frac{\varepsilon u(x, r)}{2\Delta u}\right)$ 的概率从数据集 X 中选择，并输出最优元素 $r \in \mathcal{R}$ ，是满足 $(\varepsilon, 0)$ - 差分隐私。

在差分隐私的指数机制中，K 值为 1，为了适应各种学习任务，K 值应该是可调整的，因此本文设计了一种类似于指数机制的扰动采样算法，具体算法流程如图4.2所示。

我们设计了一个二维数组，分别存储梯度值的索引 i ，梯度绝对值 $\mathbf{g}[i]$ 。梯度的置信值 u_i 为 1 或者 0，1 表示该梯度值属于 top-K，0 反之。对于第 i 个索引的梯度值 $\mathbf{g}[i]$ ，我们将根据其索引对应的梯度置信值进行划分，得到 top-K 集合 $S_{\text{top}} \leftarrow \{i \mid i \in \text{Top}(|\tilde{x}_i|)\}$ 和非 top-K 集合 $S_{\text{non-top}} \leftarrow \{i \mid i \in [n] \setminus S_{\text{top}}\}$ 。

如图4.3所示，当 $k=2$ 时，根据梯度元素的绝对值进行降序排序，然后根据其索引对应的梯度值是否为 top-k 梯度值进行划分，得到，梯度置信向量 $u = \{u_1, \dots, u_n\} = \{1, 0, 0, 0, 1, 0\}$ ，其中 $S_{\text{top}} = \{1, 5\}$ ， $S_{\text{non-top}} = \{2, 3, 4\}$ 。

接着进行概率选择，从 n 个梯度值构成的数组中以更高的概率 p 选择属于 S_{top} 的梯度值 $\{g[1], \dots, g[k]\} \in \{0, 1\}^k$ ，在索引对应的梯度上直接添加拉普拉斯噪声：

$$g[i]' = g[i] + \text{Lap}\left(\frac{GS_l}{\epsilon_i}\right)$$

以较低的概率 $1-p$ 选择属于 $S_{\text{non-top}}$ 的索引 $\{g[1], \dots, g[n-k+1]\} \in \{0, 1\}^{n-k}$ ，将

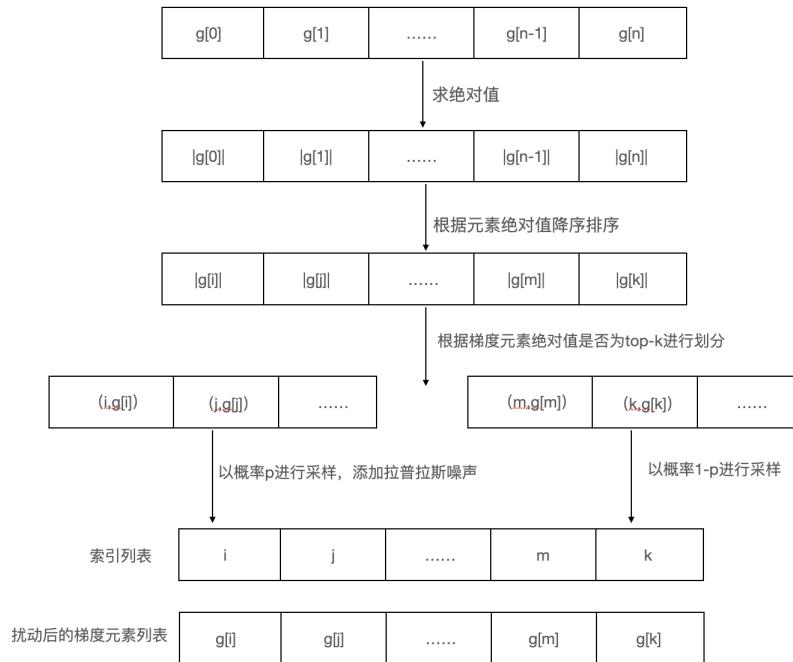


图 4.2: top-K 梯度选择算法流程图

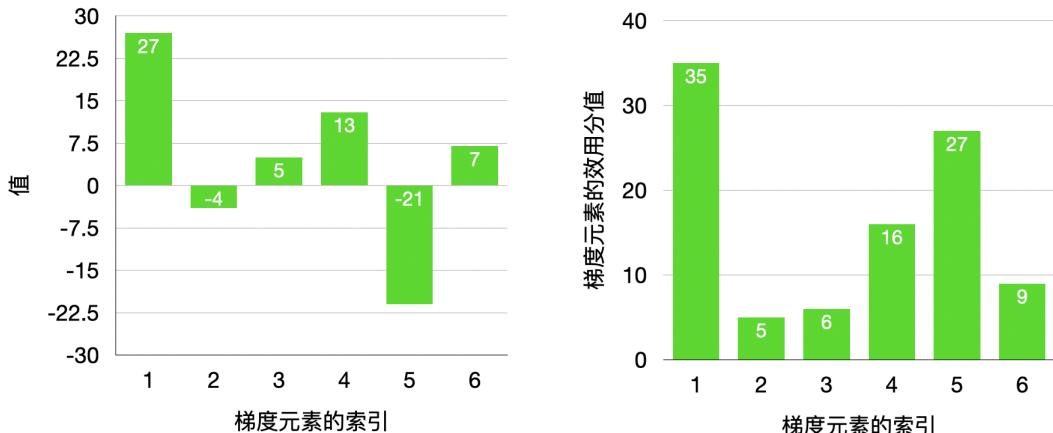


图 4.3: 梯度元素的值及其效用评分

采样扰动后的新的梯度元素及其对应的索引分别存储到两个列表中上传给混洗器，其中， $p = \frac{e^\epsilon \cdot k}{n - k + e^\epsilon \cdot k}$ 。这样的概率选择是满足 ϵ -差分隐私的，如下给出证明。

定理 4.2.1. 对于给定的梯度分值向量 u, u' ，假设算法输出的索引为 j ，根据条件概

率可以给出以下证明：

$$\frac{\Pr[j \mid u]}{\Pr[j \mid u']} \leq \frac{\Pr[j \mid u_j = 1]}{\Pr[j \mid u'_j = 0]} = \frac{p^{\frac{1}{k}}}{(1-p)^{\frac{1}{n-k}}} = e^\epsilon, \text{ where } p = \frac{e^\epsilon k}{n - k + e^\epsilon \cdot k}.$$

Algorithm 6 Top-K 梯度选择算法

```

1: 输入: 梯度向量 g, K, 隐私预算  $\epsilon$ 
2: 初始化: tmp=[], TK=[],  $S_{\text{top}} = []$ ,  $S_{\text{non-top}} = []$  /*tmp 存储梯度绝对值, TK 存储最终输出的梯度
   元素,  $S_{\text{top}}$  和  $S_{\text{non-top}}$  分别存储属于 top-k 元素的索引值和不属于 top-k 元素的索引值 */
3: for  $g[i] \in g$  do
4:   tmp.append(i,abs(g[i])); /*tmp 中的每个元素分别存储数组的索引和对应的元素绝对值 */
5: end for
6: Sort tmp by tmp.values desc; /* 根据梯度绝对值对 tmp 数组进行降序排序 */
7: for tmp[i] in tmp do
8:   if  $i < k$  then
9:      $S_{\text{top}}$ .append(i);
10:  else
11:     $S_{\text{non-top}}$ .append(i);
12:  end if
13: end for
14: for each index  $i \in S_{\text{top}} \cup S_{\text{non-top}}$  do
15:   if  $i$  in  $S_{\text{top}}$  then
16:      $y_{i,j} = \text{tmp}[i].value + \text{Lap}\left(\frac{GS_l}{\epsilon_i}\right)$ ;
17:   else
18:      $y_{i,j} = \omega \mathcal{R} \text{tmp}[i].value$ ;
19:   end if
20: end for
21: 输出:  $\langle index_i, y_i \rangle$ 
  
```

算法6将满足 $\epsilon_1 - LDP$ 的梯度选择和满足 $\epsilon_2 - LDP$ 的拉普拉斯梯度扰动相结合，根据差分隐私的组合定理，整体算法满足 $(\epsilon_1 + \epsilon_2) - LDP$ 。接着分析 Top-K 梯度选择算法的时间复杂度，与非隐私保护的 SGD 相比，LDP-SGD 给本地设备带来了额外的计算成本。对于 Top-K 梯度选择算法，所有梯度元素的效用分数和总和都可以离线初始化。对一个 n 维的向量进行排序需要消耗 $O(d \log d)$ 。将 n 个维度的元素之映射到相应的效用分数需要消耗 $O(n^2)$ ，元素取样需要 $O(n)$ 。因此，每个本地设备都有额外的时间成本 $O(n \log n + n^2 + n) = O(n^2)$ 用于梯度选择和概率选择。由于方案避免了每个维度的梯度扰动，它的计算成本比 LDP 低很多。

4.2.3 拆分混洗算法

McMahan 等人先前的研究工作^[52] 表明，在联邦学习模型中，假如在某个时间段数据是被适当的匿名化，并将数据之间的耦合信息拆分后，模型整体的隐私保障可以得到极大的改善。因此在本章中，我们针对客户端上传的梯度，进行参数的拆分混洗，通过混洗器达到客户端的匿名性，打破从中央服务器接收的数据与特定客户端之间的联系，并在每次迭代中从同一客户端发送的梯度更新中将信息解耦。在安全混洗模型中，利用一个洗牌器来打破用户身份和上传到数据分析器的信息之间的联系。由于需要引入更少的噪音来实现相同的隐私保证，按照这种模式，保护隐私的数据收集的效用得到了改善。

客户端的匿名性可以通过现有的多种机制来实现，这取决于中央服务器在特定场景下如何跟踪客户端。作为一个典型的保护隐私的最佳做法，如果使每个客户对服务器产生一定程度的匿名性，就能使客户的个人身份识别与他们的权重更新无法关联。例如，如果服务器通过 IP 地址追踪客户，每个客户可以通过使用网络代理、VPN 服务、公共 WiFi 接入产生一个无法追踪的 IP 地址。再比如，如果服务器通过软件生成的元数据（如 ID）来追踪客户，每个客户可以在向服务器发送元数据之前将其随机化。

但是，我们认为，客户端的匿名性不足以防止通信链道的攻击。例如，如果客户端在每次迭代中同时上传了大量的权重更新，中央服务器仍然可以将它们连接在一起。因此，我们设计了混洗算法，以打破来自相同客户的模型权重更新之间的联系，并将其放置于客户端上传梯度更新至中央服务器之间，使中央服务器很难结合多个客户端的同步更新来推断任何本地设备的更多信息，具体算法如7所示。

Algorithm 7 混洗器中的拆分混洗算法

- 1: **Input:** 本地客户端上传的索引列表和梯度元素列表 $\langle index_{i,j}, y_{i,j} \rangle$
 - 2: **for** $(index_{i,j}, y_{i,j}) \in \langle index_{i,j}, y_{i,j} \rangle$ **do**
 - 3: 在通信时刻 $(0, T)$ 期间随机采样梯度元素 $t_{id}^s \leftarrow U(0, T)\%$
 - 4: **end for**
 - 5: 在时刻 t_{id}^s 将梯度元素 (id, w_{id}) 发送给中央服务器
-

我们的混洗算法通过以下步骤对客户端上传的梯度参数进行混洗，然后上传给中央服务器：

- 权重分割：每个客户端都对其本地模型的权重进行分割，但给每个分割后的元素分配一个元数据，以表明其在网络结构中的权重位置。
- 权重混洗：对于所有客户端分割后的权重采用随机扰动机制进行混洗。

如图4.4所示，假使现有本地模型 X_1, X_2, X_3 ，每个模型都有相同的结构，但权重值不同。原始的联邦学习框架是将模型在本地训练后得到的参数直接发送到中央服务器。图4.4展示了我们的方案中，首先，对于每个模型，我们分割每个本地模型经过本地训练后所产生的权重。然后，对于每个权重，我们通过随机混洗机制对其进行混洗，并将每个权重及其索引发送到中央服务器。

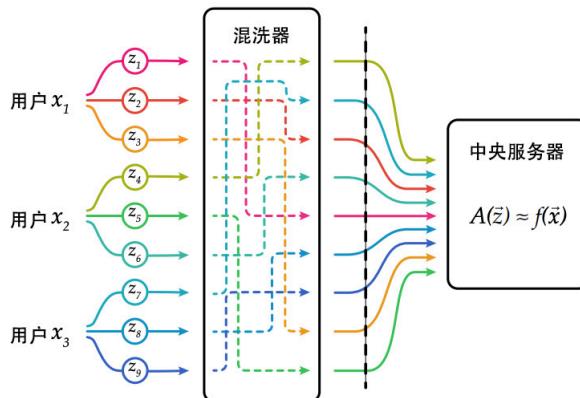


图 4.4: 联邦学习安全混洗模型中执行参数拆分混洗的混洗器

对于 $\epsilon = O(1)$ ，串行组合的 ϵ -LDP 算法 (A_1, \dots, A_n) ，令 $A_{\text{shuffle}}(x_1, \dots, x_n) = A_1(x_{\pi(1)}), A_2(x_{\pi(2)}), \dots, A_n(x_{\pi(n)})$ ，经过混洗操作 $\pi : [n] \rightarrow [n]$ 后表示为 A_{shuffle} ，混洗器输出的梯度结果满足 (ϵ', δ) -DP，其中 $\epsilon' = O\left(\frac{\epsilon \sqrt{\log(1/\delta)}}{\sqrt{n}}\right)$ 。混洗结果并不会改变数据集的统计特性，也不会增加 LDP 的隐私预算。

4.3 隐私性和收敛性证明

4.3.1 隐私性证明

隐私放大 (Privacy Amplification) 是本章所提出的安全框架中混洗器对隐私效果增强的理论分析，基于该理论，可将现有的本地化差分隐私方法直接应用在安全框架上。

在算法5中，每个本地客户端采用满足 $(\epsilon_1 + \epsilon_2) - LDP$ 的 Top-K 梯度选择算法得到梯度元素列表和索引列表，将其上传至混洗器进行拆分混洗后，所获取的数据满足 $\epsilon_c - DP$ 。从 $(\epsilon_c + \epsilon_l)$ 到 ϵ_c 的转变可通过隐私放大约论证明。 $(\epsilon_c + \epsilon_l)$ 对应于较大的数值，表示较低的隐私性； ϵ_c 对应于较小的数值，表示较高的隐私性。因此经过混洗器后，隐私性得到了增强。由差分隐私的强组合性可保证 Top-K 梯度选择算法在每次迭代中对每个样本 d_{ij} 都能保证 $(\epsilon_c + \epsilon_l)$ -本地差异隐私，因此本节只需要分析采样和混洗操作的隐私放大性。

定理 4.3.1. 算法5是满足 (ϵ, δ) - 差分隐私的，当对于任意 $\delta, \delta > 0$ ，并且有：

$$\epsilon = \mathcal{O} \left(\epsilon_0 \sqrt{\frac{qT \log(2qT/\delta) \log(2/\delta)}{n}} \right)$$

假设在联邦学习模型中，需要迭代的次数为 $t \in [T]$ 。 $\mathcal{M}_t(\theta_t, \mathcal{D})$ 表示在时刻 t 对于数据集 \mathcal{D} 和模型参数为 θ_t 的差分隐私机制， θ_{t+1} 表示模型的输出。因此，在数据集 $\mathcal{D} = \bigcup_{i=1}^m \mathcal{D}_i \in \mathfrak{S}^n$ 上的差分隐私机制定义如下：

$$\mathcal{M}_t(\theta_t; \mathcal{D}) = \mathcal{H}_{ks} \circ \text{samp}_{m,k}(\mathcal{G}_1, \dots, \mathcal{G}_m) \quad (4.1)$$

其中， $\mathcal{G}_i = \text{samp}_{r,s}(\mathcal{R}(\mathbf{x}_{i1}^t), \dots, \mathcal{R}(\mathbf{x}_{ir}^t))$ 并且 $\mathbf{x}_{ij}^t = \nabla_{\theta_t} f(\theta_t; d_{ij}), \forall i \in [m], j \in [r]$ 。 \mathcal{H}_{ks} 表示在 ks 个数据样本上进行混洗操作， $\text{samp}_{a,b}$ 表示从有 a 个元素的集合中随机抽样 b 个元素的操作， $\mathcal{R}(\mathbf{x}_{i1}^t)$ 表示本地客户端采用 Top-K 梯度选择算法得到满足 $(\epsilon_1 + \epsilon_2) - LDP$ 的梯度向量。

接下来我们给出 \mathcal{M}_t 的隐私性证明：

假设客户端 $i \in [m]$ 的本地数据集为 $\mathcal{D}_i = \{d_{i1}, d_{i2}, \dots, d_{ir}\} \in \mathfrak{S}^r$, $\mathcal{D} = \bigcup_{i=1}^m \mathcal{D}_i$ 表示总体数据集。根据公式4.1, $\mathcal{Z}(\mathcal{D}^{(t)}) = \mathcal{H}_{ks}(\mathcal{R}(\mathbf{x}_1^t), \dots, \mathcal{R}(\mathbf{x}_{ks}^t))$ 表示在本地客户端进行模型训练得到满足本地差分隐私的 ks 个权重集合上混洗后的权重。任取 $\tilde{\delta} > 0$, 当 $\epsilon_0 \leq \frac{\log(ks/\log(1/\tilde{\delta}))}{2}$ 时, 算法 \mathcal{Z} 满足 $(\tilde{\epsilon}, \tilde{\delta}) - \text{DP}$ 差分隐私, 可得:

$$\tilde{\epsilon} = \mathcal{O} \left(\min \{ \epsilon_0, 1 \} e^{\epsilon_0} \sqrt{\frac{\log(1/\tilde{\delta})}{ks}} \right) \quad (4.2)$$

当 $\epsilon_0 = \mathcal{O}(1)$ 时, 有 $\tilde{\epsilon} = \mathcal{O} \left(\epsilon_0 \sqrt{\frac{\log(1/\tilde{\delta})}{ks}} \right)$ 。

令 $\mathcal{T} \subseteq \{1, \dots, m\}$ 表示在时刻 t 选取的 k 个客户端。对于 $i \in \mathcal{T}$, $\mathcal{T}_i \subseteq \{1, \dots, r\}$ 表示在时刻 t 客户端 i 所抽样的 s 条数据样本。对于任意的 $\mathcal{T} \in \binom{[m]}{k}$ 和 $\mathcal{T}_i \in \binom{[r]}{s}$, $i \in \mathcal{T}$, 有 $\bar{\mathcal{T}} = (\mathcal{T}, \mathcal{T}_i, i \in \mathcal{T})$, $\mathcal{D}^{\mathcal{T}_i} = \{d_j : j \in \mathcal{T}_i\}$ for $i \in \mathcal{T}$, and $\mathcal{D}^{\bar{\mathcal{T}}} = \{\mathcal{D}^{\mathcal{T}_i} : i \in \mathcal{T}\}$ 。 \mathcal{T} 和 $\mathcal{T}_i, i \in \mathcal{T}$ 为抽样产生的任意子集, 其中的随机性由客户端抽样和数据集抽样所决定。算法 \mathcal{M}_t 可以等价的表示为 $\mathcal{M}_t = \mathcal{Z}(\mathcal{D}^{\bar{\mathcal{T}}})$ 。

假设现有数据集: $\mathcal{D}' = (\mathcal{D}'_1) \bigcup (\cup_{i=2}^m \mathcal{D}_i) \in \mathfrak{S}^n$, 其中数据集 $\mathcal{D}'_1 = \{d'_{11}, d'_{12}, \dots, d'_{1r}\}$ 和 \mathcal{D}_1 为相邻数据集, 它们的第 d_{11} 条和第 d'_{11} 条数据样本不同。如果 \mathcal{M}_t 是满足 $(\bar{\epsilon}, \bar{\delta}) - \text{DP}$ 差分隐私的, 那么对于算法 \mathcal{M}_t 所选的任意子集 \mathcal{S} 都应该满足:

$$\Pr[\mathcal{M}_t(\mathcal{D}) \in \mathcal{S}] \leq e^{\bar{\epsilon}} \Pr[\mathcal{M}_t(\mathcal{D}') \in \mathcal{S}] + \bar{\delta} \quad (4.3)$$

$$\Pr[\mathcal{M}_t(\mathcal{D}') \in \mathcal{S}] \leq e^{\bar{\epsilon}} \Pr[\mathcal{M}_t(\mathcal{D}) \in \mathcal{S}] + \bar{\delta} \quad (4.4)$$

由于式4.3和4.4是对称的, 因此只需要证明其中一条。下文给出式4.3的证明:

令 $q = \frac{ks}{mr}$, 我们给出条件概率的定义:

$$\begin{aligned} A_{11} &= \Pr \left[\mathcal{Z} \left(\mathcal{D}^{\bar{T}} \right) \in \mathcal{S} \mid 1 \in \mathcal{T} \text{ and } 1 \in \mathcal{T}_1 \right] \\ A'_{11} &= \Pr \left[\mathcal{Z} \left(\mathcal{D}'^{\bar{T}} \right) \in \mathcal{S} \mid 1 \in \mathcal{T} \text{ and } 1 \in \mathcal{T}_1 \right] \\ A_{10} &= \Pr \left[\mathcal{Z} \left(\mathcal{D}^{\bar{T}} \right) \in \mathcal{S} \mid 1 \in \mathcal{T} \text{ and } 1 \notin \mathcal{T}_1 \right] = \Pr \left[\mathcal{Z} \left(\mathcal{D}'^{\bar{T}} \right) \in \mathcal{S} \mid 1 \in \mathcal{T} \text{ and } 1 \notin \mathcal{T}_1 \right] \\ A_0 &= \Pr \left[\mathcal{Z} \left(\mathcal{D}^{\bar{T}} \right) \in \mathcal{S} \mid 1 \notin \mathcal{T} \right] = \Pr \left[\mathcal{Z} \left(\mathcal{D}'^{\bar{T}} \right) \in \mathcal{S} \mid 1 \notin \mathcal{T} \right] \end{aligned} \tag{4.5}$$

令 $q_1 = \frac{k}{m}$, $q_2 = \frac{s}{r}$, 那么 $q = q_1 q_2$, 然后可以得到:

$$\Pr [\mathcal{M}_t(\mathcal{D}) \in \mathcal{S}] = q A_{11} + q_1 (1 - q_2) A_{10} + (1 - q_1) A_0 \tag{4.6}$$

$$\Pr [\mathcal{M}_t(\mathcal{D}') \in \mathcal{S}] = q A'_{11} + q_1 (1 - q_2) A_{10} + (1 - q_1) A_0 \tag{4.7}$$

因此, 我们可以得到:

$$A_{11} \leq e^{\tilde{\epsilon}} A'_{11} + \tilde{\delta} \tag{4.8}$$

$$A_{11} \leq e^{\tilde{\epsilon}} A_{10} + \tilde{\delta} \tag{4.9}$$

式4.7成立, 因此混洗器 \mathcal{M}_t 是满足 ε_c -差分隐私的。

4.3.2 模型收敛性分析

在本节中, 我们分析采用采样和混洗算法后模型的收敛性。

回顾第二章的基础知识, 在随机梯度下降算法的每次迭代中, 中央服务器将当前的参数向量发送给所有本地客户端, 客户端收到后在本地数据集上进行模型训练, 计算本地模型的梯度并上传给中央服务器, 然后中央服务器计算收到的梯度的平均值/平均数并更新全局模型。

在算法5中, 在每一轮迭代过程中, 中央服务器聚合上传的 ks 个加噪后的梯度, 如算法5的第 15 行所示, 中央服务器进行聚合后得到结果: $\bar{\mathbf{g}}_t \leftarrow \frac{1}{ks} \sum_{i \in \mathcal{U}_t, j \in \mathcal{S}_{it}} \langle y_{i,j} \rangle$, 然后通过随机梯度下降算法更新全局模型参数: $\theta_{t+1} \leftarrow \Pi_C (\theta_t - \eta_t \bar{\mathbf{g}}_t)$ 。

既然随机扰动机制是无偏的，那么平均梯度 $\bar{\mathbf{g}}_t$ 也是无偏的，也就是说，我们有 $\mathbb{E}[\bar{\mathbf{g}}_t] = \nabla_{\theta_t} F(\theta_t)$ ，其中期望是相对于客户端和数据点的随机抽样以及扰动机制的随机性而言的。

令 $F(\theta)$ 为凸函数，考虑这样一个随机梯度下降算法： $\theta_{t+1} \leftarrow \Pi_C(\theta_t - \eta_t \mathbf{g}_t)$ ， \mathbf{g}_t 满足 $\mathbb{E}[\mathbf{g}_t] = \nabla_{\theta_t} F(\theta_t)$ 并且 $\mathbb{E}\|\mathbf{g}_t\|_2^2 \leq G^2$ 。当确定 $\eta_t = \frac{D}{G\sqrt{t}}$ ，可以得到：

$$\mathbb{E}[F(\theta_T)] - F(\theta^*) \leq 2DG \frac{2 + \log(T)}{\sqrt{T}} = \mathcal{O}\left(DG \frac{\log(T)}{\sqrt{T}}\right) \quad (4.10)$$

由 Nesterov 等人在文献^[50] 中的证明可知，算法5的输出 θ_T 满足：

$$\mathbb{E}[F(\theta_T)] - F(\theta^*) \leq \mathcal{O}\left(\frac{LD \log(T) \max\left\{d^{\frac{1}{2}-\frac{1}{p}}, 1\right\}}{\sqrt{T}} \left(1 + \sqrt{\frac{cd}{qn}} \left(\frac{e^{\epsilon_0} + 1}{e^{\epsilon_0} - 1}\right)\right)\right) \quad (4.11)$$

其中，存在 $\sqrt{1 + \frac{cd}{qn} \left(\frac{e^{\epsilon_0} + 1}{e^{\epsilon_0} - 1}\right)^2} \leq \left(1 + \sqrt{\frac{cd}{qn}} \left(\frac{e^{\epsilon_0} + 1}{e^{\epsilon_0} - 1}\right)\right)$ 。

当 $\sqrt{\frac{cd}{qn}} \left(\frac{e^{\epsilon_0} + 1}{e^{\epsilon_0} - 1}\right) \geq \Omega(1)$ 时，可以推导出：

$$\mathbb{E}[F(\theta_T)] - F(\theta^*) \leq \mathcal{O}\left(\frac{LD \log(T) \max\left\{d^{\frac{1}{2}-\frac{1}{p}}, 1\right\}}{\sqrt{T}} \sqrt{\frac{cd}{qn}} \left(\frac{e^{\epsilon_0} + 1}{e^{\epsilon_0} - 1}\right)\right) \quad (4.12)$$

如果我们在算法5中设置学习率为 $\eta_t = \frac{D}{G\sqrt{t}}$ ，其中 $G^2 = L^2 \max\left\{d^{1-\frac{2}{p}}, 1\right\} \left(1 + \frac{cd}{qn} \left(\frac{e^{\epsilon_0} + 1}{e^{\epsilon_0} - 1}\right)^2\right)$ 。那么：

$$\mathbb{E}[F(\theta_T)] - F(\theta^*) \leq \mathcal{O}\left(\frac{LD \log(T) \max\left\{d^{\frac{1}{2}-\frac{1}{p}}, 1\right\}}{\sqrt{T}} \sqrt{\frac{cd}{qn}} \left(\frac{e^{\epsilon_0} + 1}{e^{\epsilon_0} - 1}\right)\right) \quad (4.13)$$

其中，当 $p \in \{1, \infty\}$ 时， $c = 4$ 否则 $c = 14$ 。

定理 4.3.2 (随机梯度下降算法的收敛性). 假使有凸函数 $F(\theta)$ ，数据集 D 的维度为 C ，在模型训练过程中采用随机梯度下降算法 $\theta_{t+1} \leftarrow \Pi_C(\theta_t - \eta_t \mathbf{g}_t)$ ，其中 \mathbf{g}_t 满足 $\mathbb{E}[\mathbf{g}_t] = \nabla_{\theta_t} F(\theta_t)$ 并且 $\mathbb{E}\|\mathbf{g}_t\|_2^2 \leq G^2$ 。当 $\eta_t = \frac{D}{G\sqrt{t}}$ ， $\mathbb{E}[F(\theta_T)] - F(\theta^*) \leq 2DG \left(\frac{2+\log(T)}{\sqrt{T}}\right)$ 成立。

根据文献^[50] 中已有的标准随机梯度下降算法收敛结果中使用的定理4.3.2对 G^2 的约束条件，证明了混洗算法可在 $G^2 = L^2 \max \left\{ d^{1-\frac{2}{p}}, 1 \right\} \left(1 + \frac{cd}{qn} \left(\frac{e^\epsilon + 1}{e^{\epsilon-1}} \right)^2 \right)$ 时达到全局最优解。

4.4 实验评估

4.4.1 实验准备

在本节中，我们进行实验来评估混洗器的性能。所有的实验都是用 PYTHON 语言编译的，其中每个用户都由配备 6GB 内存、四核 2.36GHz Cortex A73 处理器和四核 Cortex A53 1.8GHz 处理器的华为 nova3 安卓手机代替。中央服务器是用两台联想服务器模拟的，这两台服务器有 2 个英特尔 (R) 至强 (R) E5-2620 2.10GHZ CPU，32GB 内存，512SSD，2TB 机械硬盘，运行于 Ubuntu 18.04 操作系统。

在实验过程中，我们选择了深度学习中常用的两个经典数据集-MNIST 手写体数字识别数据集、FMNIST 和 CIFAR-10 数据集进行实验，评估所提出的安全混洗框架。此外，我们让所有用户离线训练一个统一的卷积神经网络，以获得本地用户的梯度。在我们的实验中采用的模型网络结构为 CNN，包括 2 个卷积层，两个池化层层和一个全连接层（32 个神经元）。模型的激活函数为 Softmax，并引入了 DropOut 正则以提高模型的泛化能力。下表展示了 CNN 的网络结构。

神经层	参数
卷积层	8×8 的 16 个滤波器，步长为 2
池化层	2×2
卷积层	4×4 的 32 个滤波器，步长为 2
池化层	2×2
全连接层	32 个神经元
Softmax	10 个神经元

表 4.1: 安全混洗框架实验的模型网络结构

4.4.2 实验设计

在我们模拟的联邦学习环境中，我们设置本地客户端的总数为 1000 个，其中每个客户有一个本地数据集，每个客户都对梯度 $\mathbf{g}_t(d_{ij}) \leftarrow \nabla_{\theta_t} f(\theta_t; d_{ij})$ 进行剪裁，梯度裁剪参数 $C=1/100$ 。之后，运行 Top-K 梯度选择和拆分混洗算法，混洗器将参数上传至中央服务器进行安全聚合，更新全局参数。我们的算法运行了 100 个历时，在前 80 个历时中，我们将学习率设置为 0.3，在剩余的历时中，将其降低到 0.18。在每一次训练迭代过程中，本地客户端与混洗器、中央服务器进行交互计算得到安全聚合之后的全局梯度向量。我们设置了本地隐私参数 $\sigma=2$ ，而中央隐私参数 ϵ 的计算则是由我们来完成的。我们首先使用文献中^[72] 的定理 5.3 通过洗牌数值计算隐私放大率。然后，我们通过 4.3.1 中提出的子抽样计算隐私放大；最后，我们使用差分隐私的强组合性质来获得中央隐私参数 ϵ 。

我们的实验主要分为两个部分：

- (1) 在 MNIST、FMNIST 和 CIFAR 上评估安全混洗算法，评估参数：客户端数量 N 、梯度选择的比率、客户端采样比 f_r 和最大聚合次数对于模型分类准确率的影响。
- (2) 将本文的安全混洗方案与基准方案（非隐私保护的联邦学习方案）、前人提出的隐私保护联邦学习方案（如表 4.2 所示）进行对比，评估指标为模型分类准确率和通信性能。
- (3) 在基于梯度选择和安全混洗的联邦学习模型上应用生成对抗网络攻击进行实验，评估模型的隐私保护效用。

基准方案名称	具体算法
FL	没有添加隐私保护机制的联邦学习模型
PS-FL ^[67]	通过选择性的参数更新实现隐私保护的分布式学习框架
DP-FL	基于中央差分隐私的联邦学习模型
LDP-FL	基于本地差分隐私的联邦学习模型
KSA-FL	本文的隐私保护方案，基于梯度选择和安全混洗的联邦学习模型

表 4.2: 安全混洗框架的比较方案

4.4.3 结果分析

实验一（分析各个参数对模型准确率的影响）

在本文所设计的联邦学习安全混洗算法中，本地客户端的数量、梯度选择的比率、客户端采样比都是影响联邦学习模型分类准确率的因素。为了准确的反映每种参数对于准确率的影响，我们控制变量的进行实验。参考值如下：总客户端数量 N 为 1000，梯度选择的比率分别为 1%、5%、10%、50%、100%，总通信回合为 100，在前 80 个历时中，我们将学习率设置为 0.3，在剩余的历时中，将其降低到 0.18。隐私预算 ϵ 分别为 50、60、100。对于每一种参数组合进行模型训练，具体的实验结果如下文所示。

首先分析安全混洗模型中参与混洗的本地客户端数量对联邦模型分类精度的影响，如图4.5所示，通过 Top-K 梯度选择算法和拆分混洗算法，我们的安全混洗模型（下文简称 KSA-FL）能够以较低的隐私成本实现较高的准确性。在训练中增加客户端数量 N 的同时，KSA-FL 能达到的模型精度与不添加噪声的联邦学习几乎接近。与 MNIST、FMNIST 相比，CIFAR-10 需要更多的客户端，这表明对于一个具有较大神经网络模型的更复杂的任务，当在更多的本地数据和更多的客户端上添加扰动之后，需要更多的通信回合才能使联邦学习模型达到更高的精度。

如图4.6所示，当固定总客户端数量为 100，在不同的隐私预算参数 $\epsilon=50、70、100$ 和不添加隐私保护的联邦学习模型中，混洗器在每次迭代过程中随机采样部分客户端的梯度进行混洗。图4.6展示了全局损失函数值随客户端采样比的变化情况，

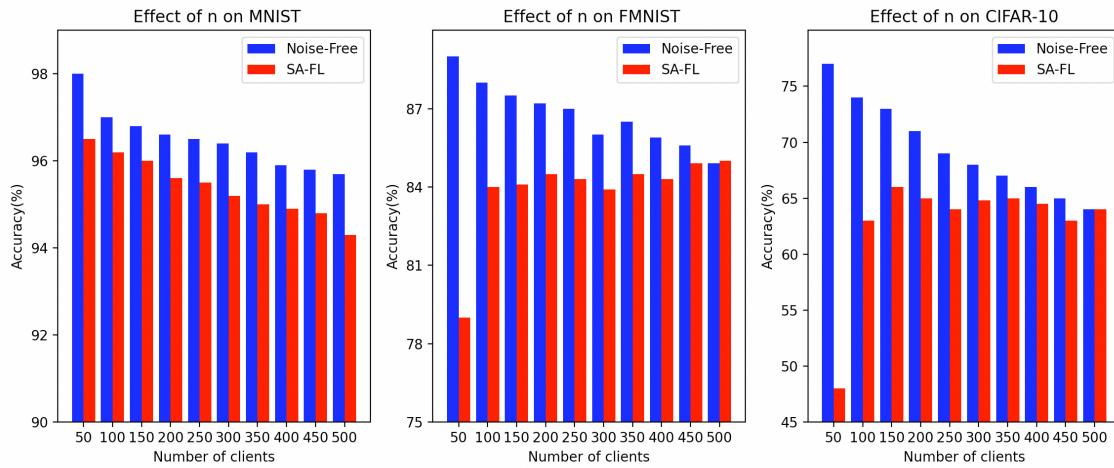


图 4.5: 安全混洗模型中本地客户端数量对联邦学习模型训练精度的影响

总体来说，当客户端采样比为 0.5 左右时，损失函数值最小。联邦学习的隐私保护难点在于使用较低的隐私预算维持较高的模型精度，而选择适宜的客户端数量参与安全聚合会大大提升模型的收敛速度和通信性能。

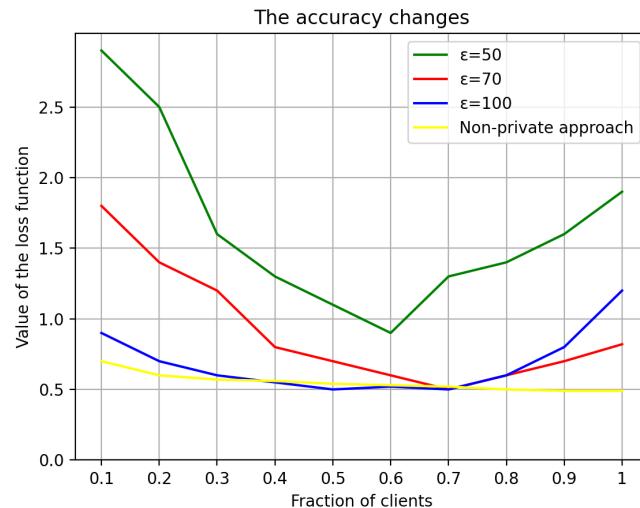


图 4.6: 安全混洗模型中客户端采样比对联邦学习模型训练精度的影响

如图4.7展示了不同的梯度选择比率对联邦学习模型训练精度的影响，X 轴表示联邦模型迭代次数。从图中的曲线变化情况可以看出，各个梯度选择比率的模型收敛速度接近，在 70 个训练回合后基本达到收敛，因此梯度选择并不会影响模型的收敛速度。1% 的梯度选择在历时 100 个训练回合后能达到 83% 的准确率，100%

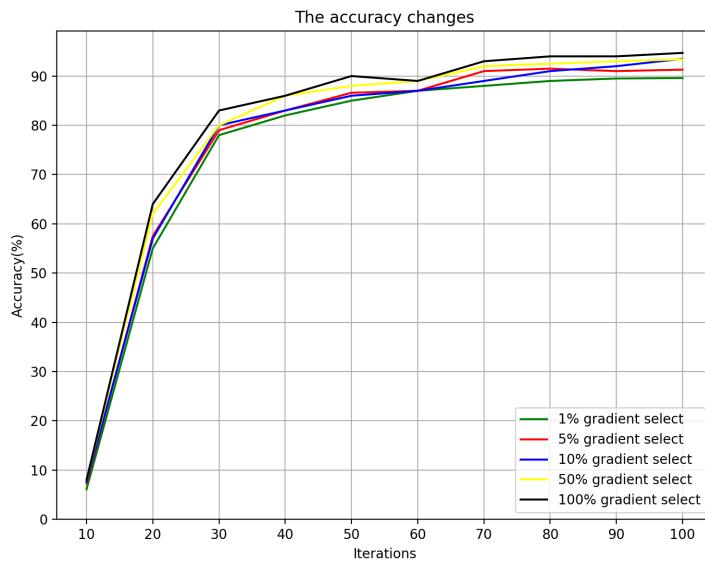


图 4.7: 安全混洗模型中梯度选择比率对联邦学习模型训练精度的影响

的梯度选择比率所能达到的模型准确率为 95%，相差了近 10% 的准确率，可见部分梯度的丢弃确实丢失了部分的训练信息。当梯度选择比率低于 10% 时，随着梯度选择比率的增加，模型的准确率也增加了接近 3%-5% 个百分点，然而当梯度选择比率为 50% 时，模型在 30 个训练回合后的准确率基本与 100% 的梯度选择比率所能达到的模型准确率接近。

在实验中，每一轮的迭代训练都需要由中央服务器和本地客户端交互计算，我们分别计算了不同的梯度选择比率在一次训练迭代中的通信开销，实验结果如表4.3所示。当梯度选择比率下降时，通信开销也基本呈比例下降。当 Top-K 梯度选择比率从 100% 优化至 50% 时，通信性能优化了大约 1.99 倍，而根据图4.7所示，在梯度选择比率从 100% 降低至 50% 时，模型的训练准确率并不会受到太大影响，由此可见一定范围内的梯度选择可以较好的优化联邦学习模型的通信性能。

Top-K/%	通信开销/KB
1	167.022×2
5	863.317×2
10	1684.271×2
50	3502.151×2
100	6995.548×2

表 4.3: 不同梯度选择比率的通信开销

实验二（与前人的隐私保护方案进行对比实验）

我们将本文设计的基于梯度选择与安全混洗的联邦学习隐私保护方案 (KSA-FL) 的与 FL、PS-FL、DP-FL、LDP-FL 方案进行对比实验，选取的数据集为 MNIST，网络模型为 CNN5。我们比较了不同方案在给定相同的隐私预算 ($(0.2, 5e - 6) - DP$) 情况下，模型分类准确率变化情况。

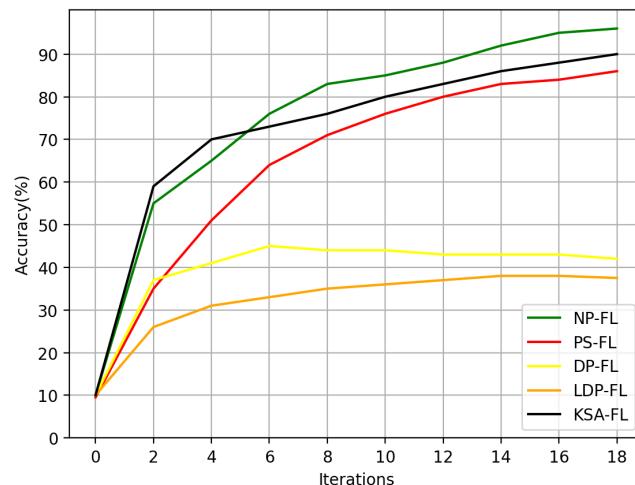


图 4.8: 不同隐私保护方案在 MNIST 数据集上训练的模型分类准确率变化情况

在 $((0.2, 5e - 6) - DP)$ 的隐私预算下，无添加隐私保护的联邦学习基准模型在 18 个训练轮次后能达到 94% 左右的准确率，其余四种实现了联邦学习隐私保护的方案中，本文所设计的 Top-K 安全混洗方案所能达到的准确率最高，在 18 个训练轮次后能达到 90% 的准确率，而在训练开始阶段，模型的准确率提升速度甚至超

过了无添加隐私保护的联邦学习基准模型，这表明 Top-K 梯度选择算法能有效的加速模型的学习速率。基于中央差分隐私的联邦学习隐私保护模型在 18 个训练轮次后能达到 41% 左右的准确率，在 6 个训练轮次过后，模型就接近收敛。基于本地差分隐私的联邦学习隐私保护模型的分类准确率在这几种方案中的表现最低，正如上文分析的，本地差分隐私由于局部噪声带来的误差会随着维度系数的增加而加剧，从而大大降低模型的精度。

实验三（针对攻击模型，分析该方案的隐私保护效用）

4.5 本章总结

本章节我们针对联邦学习模型的整体框架进行了改进，提出了安全混洗模型，在本地客户端和中央服务器之间加设混洗器，通过对本地客户端进行随机抽样，将上传的梯度进行拆分混洗，增加隐私放大效果。然后发送给中央服务器进行聚合。并对方案进行了隐私性证明，表明此安全混洗算法可以保证 ϵ_c 差分隐私，然后对此方案在中央服务器上的随机梯度下降算法进行了收敛性的分析，证明在凸函数上，梯度 \mathbf{g}_t 满足 $\mathbb{E}[\mathbf{g}_t] = \nabla_{\theta_t} F(\theta_t)$ 时模型能达到全局收敛。最后，通过在三种基准数据集上进行对象，证明本章所提出的方案能在保证模型收敛性的情况下，减少隐私预算。

第五章 总结与展望

5.1 论文总结

随着人工智能深度学习的快速发展，出现了越来越多的复杂模型和算法，能够有效的解决各类难题。基于人工智能的产品给医疗、教育、金融、工业等各个领域带来了新一波的发展热潮，也使人们的生活水平大大提高。然而，用户在享受深度学习模型所带来的便利时，也带了许多隐私问题。由于人工智能的基础是基于大数据，许多模型和算法的学习必须基于真实的用户行为数据。而很多深度学习服务的提供方不能有效的保护用户的数据隐私。随着隐私泄露事件越来越多，数据的安全和隐私问题也逐步引起了人们的关注。

与此同时，各类智能设备也在不断发展，用户产生的数据也越来越多，智能设备的算力不断增强。用户不愿意向商业公司或商业机构提供个人隐私数据。之后，产生了联邦学习框架，它解决了分布式终端用户在本地更新模型的问题，目标是保障大数据共享信息时的数据安全、保护本地数据和个人隐私，在多计算节点之间高效的训练机器学习模型。它不是将数据上传到中央服务器进行集中训练，而是参与者在本地进行模型训练并与参数服务器共享模型更新。参数服务器对来自多个参与者的权重进行聚合，并组合创建一个改进的全局模型，这有助于保障用户的数据隐私，降低通信成本。

虽然联邦学习解决了传统集中式深度学习所面临的大规模数据收集等问题，节省了传输数据所占用的通信资源。但是，联邦学习中的共享参数以及传输数据的无线链路仍然可能泄露数据隐私。各类攻击模型阻碍了联邦学习技术的发展，也会极大地威胁到人们的隐私敏感信息。

本文主要研究针对分布式联邦学习系统的隐私安全问题。通过研究神经网络的前向传播算法和差分隐私的相关性质，提出了一套分布式联邦系统中针对梯度和通信信道攻击的隐私安全方案。本文的主要工作和贡献如下：

- (1) 基于本地差分隐私的自适应干扰算法：在客户端本地训练的神经网络模型中，通过分析前向传播算法，计算每个属性类对于模型输出的贡献比，然后，我们设计了一个自适应噪声添加的方案，根据贡献率注入不同隐私预算的噪声。与传统的注入噪声的方法相比，我们在相同的隐私保护程度下最大限度地提高了模型的准确性，减少噪声对模型输出结果的影响，提高模型精度。之后，我们采用动量组合机制计算对梯度施加的噪声量大小，分析了模型整体的隐私预算。最后通过多组真实数据集验证了本地自适应扰动机制的性能，证明了其在相同条件下要优于现有的固定差分隐私随机梯度下降算法。
- (2) 本文提出了安全混洗框架，混洗器对客户端上传的梯度进行采样后，然后拆分混洗，再将混洗模型和自适应本地差分隐私保护方法结合在分布式系统中，提高系统学习效果，实现了数据隐私性与模型可用性之间的更好平衡。最后，基于本文提出的隐私保护框架，我们在三个基准数据集的进行了实验和讨论，并与之前的差分隐私联邦学习框架进行对比实验，证明了方案的有效性和可行性。

综上所述，本文的研究充分证明了所提出框架的有效性，可以极大的联邦学习模型的隐私性和可用性，从而进一步推进了联邦学习在安全领域的应用和发展。

5.2 论文展望

在可预见的未来，大规模、大数据、分布式的深度学习将得到快速发展。物联网、5G、边缘计算等技术也将迅速普及。人类将彻底步入人工智能时代。在此我将对未来的研方向做出几点展望：

- (1) 本文提出的基于本地自适应混洗差分隐私深度学习算法是一种基础算法，在

模型学习的过程中，它的总体隐私预算会随着通信回合的增加而大幅上升，因此后续可以研究其在大型数据集与复杂模型结构中的表现。

- (2) 差分隐私对于数据的保护是基于数学证明的，但是缺乏一定的可解释性，如果能够在差分隐私保护联邦学习模型上建立更加有效的隐私风险评估和隐私成本评估指标，那么将来应该能更好的应用差分隐私，推动联邦学习机制下的差分隐私保护的研究发展。
- (3) 现实生活中，联邦学习的参与方数量可能有百万、千万的级别。当客户端的量级大大增加时，由于本地设备在通信、计算和存储等各个方面的能力大有不同，因此之后关于实际应用中的通信成本、设备异构等方面也需要大量的研究。

参考文献

- [1] Pouyanfar S, Sadiq S, Yan Y, et al. A survey on deep learning: Algorithms, techniques, and applications[J]. ACM Computing Surveys (CSUR), 2018, 51(5): 1-36.
- [2] Voigt P, Von dem Bussche A. The eu general data protection regulation (gdpr)[J]. A Practical Guide, 1st Ed., Cham: Springer International Publishing, 2017, 10: 3152676.
- [3] Kendall A, Gal Y, Cipolla R. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 7482-7491.
- [4] Hu R, Dollár P, He K, et al. Learning to segment every thing[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018: 4233-4241.
- [5] 张仕良. 基于深度神经网络的语音识别模型研究 [D]. 合肥: 中国科学技术大学, 2017.
- [6] Sardianos C, Tsirakis N, Varlamis I. A survey on the scalability of recommender systems for social networks[M]//Social Networks Science: Design, Implementation, Security, and Challenges. Springer, Cham, 2018: 89-110.
- [7] Shen D, Wu G, Suk H I. Deep learning in medical image analysis[J]. Annual review of biomedical engineering, 2017, 19: 221-248.

- [8] Papernot N, Abadi M, Erlingsson U, et al. Semi-supervised knowledge transfer for deep learning from private training data[J]. arXiv preprint arXiv:1610.05755, 2016.
- [9] Dwork C, McSherry F, Nissim K, et al. Calibrating noise to sensitivity in private data analysis[C]//Theory of cryptography conference. Springer, Berlin, Heidelberg, 2006: 265-284.
- [10] Rivest R L, Adleman L, Dertouzos M L. On data banks and privacy homomorphisms[J]. Foundations of secure computation, 1978, 4(11): 169-180.
- [11] Wu X, Fredrikson M, Jha S, et al. A methodology for formalizing model-inversion attacks[C]//2016 IEEE 29th Computer Security Foundations Symposium (CSF). IEEE, 2016: 355-370.
- [12] Hitaj B, Ateniese G, Perez-Cruz F. Deep models under the GAN: information leakage from collaborative deep learning[C]//Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. 2017: 603-618.
- [13] Shokri R, Stronati M, Song C, et al. Membership inference attacks against machine learning models[C]//2017 IEEE Symposium on Security and Privacy (SP). IEEE, 2017: 3-18.
- [14] Dwork C. Differential privacy[C]//International Colloquium on Automata, Languages, and Programming. Springer, Berlin, Heidelberg, 2006: 1-12.
- [15] Alfeld S, Zhu X, Barford P. Data poisoning attacks against autoregressive models[C]//Proceedings of the AAAI Conference on Artificial Intelligence. 2016, 30(1).
- [16] Yao A C. Protocols for secure computations[C]//23rd annual symposium on foundations of computer science (sfcs 1982). IEEE, 1982: 160-164.

- [17] Meng X, Bradley J, Yavuz B, et al. Mllib: Machine learning in apache spark[J]. *The Journal of Machine Learning Research*, 2016, 17(1): 1235-1241.
- [18] Wang X, Han Y, Wang C, et al. In-edge ai: Intelligentizing mobile edge computing, caching and communication by federated learning[J]. *IEEE Network*, 2019, 33(5): 156-165.
- [19] Li T, Sahu A K, Talwalkar A, et al. Federated learning: Challenges, methods, and future directions[J]. *IEEE Signal Processing Magazine*, 2020, 37(3): 50-60.
- [20] Tran N H, Bao W, Zomaya A, et al. Federated learning over wireless networks: Optimization model design and analysis[C]//*IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2019: 1387-1395.
- [21] McMahan B, Moore E, Ramage D, et al. Communication-efficient learning of deep networks from decentralized data[C]//*Artificial intelligence and statistics*. PMLR, 2017: 1273-1282.
- [22] Zhu L, Han S. Deep leakage from gradients[M]//*Federated learning*. Springer, Cham, 2020: 17-31.
- [23] Aono Y, Hayashi T, Wang L, et al. Privacy-preserving deep learning via additively homomorphic encryption[J]. *IEEE Transactions on Information Forensics and Security*, 2017, 13(5): 1333-1345.
- [24] Ma C, Li J, Ding M, et al. On safeguarding privacy and security in the framework of federated learning[J]. *IEEE network*, 2020, 34(4): 242-248.
- [25] 曹志义, 牛少彰, 张继威. 基于半监督学习生成对抗网络的人脸还原算法研究[J]. *电子与信息学报*, 2018, 40(2): 323-330. Distributed differential privacy via shuffling. In *Eurocrypt*. Springer, 2019.

- [26] Goodfellow I, Pouget-Abadie J, Mirza M, et al. Generative adversarial nets[J]. Advances in neural information processing systems, 2014, 27.
- [27] Radford A, Metz L, Chintala S. Unsupervised representation learning with deep convolutional generative adversarial networks[J]. arXiv preprint arXiv:1511.06434, 2015.
- [28] Salimans T, Goodfellow I, Zaremba W, et al. Improved techniques for training gans[J]. Advances in neural information processing systems, 2016, 29: 2234-2242.
- [29] Goodfellow I, Bengio Y, Courville A. Deep learning[M]. MIT press, 2016.
- [30] Johnson R, Zhang T. Accelerating stochastic gradient descent using predictive variance reduction[J]. Advances in neural information processing systems, 2013, 26: 315-323.
- [31] Zhang T. Solving large scale linear prediction problems using stochastic gradient descent algorithms[C]//Proceedings of the twenty-first international conference on Machine learning. 2004: 116.
- [32] Dwork C, Kenthapadi K, McSherry F, et al. Our data, ourselves: Privacy via distributed noise generation[C]//Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, Berlin, Heidelberg, 2006: 486-503.
- [33] McSherry F, Talwar K. Mechanism design via differential privacy[C]//48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07). IEEE, 2007: 94-103.
- [34] LBengio Y. Learning deep architectures for AI[M]. Now Publishers Inc, 2009.

- [35] Dwork C, Roth A. The algorithmic foundations of differential privacy[J]. Found. Trends Theor. Comput. Sci., 2014, 9(3-4): 211-407.
- [36] Bassily R, Smith A, Thakurta A. Private empirical risk minimization: Efficient algorithms and tight error bounds[C]//2014 IEEE 55th Annual Symposium on Foundations of Computer Science. IEEE, 2014: 464-473.
- [37] Acs G, Melis L, Castelluccia C, et al. Differentially private mixture of generative neural networks[J]. IEEE Transactions on Knowledge and Data Engineering, 2018, 31(6): 1109-1121.
- [38] Su D, Cao J, Li N, et al. Differentially private k-means clustering and a hybrid approach to private optimization[J]. ACM Transactions on Privacy and Security (TOPS), 2017, 20(4): 1-33.
- [39] Salakhutdinov R, Mnih A, Hinton G. Restricted Boltzmann machines for collaborative filtering[C]//Proceedings of the 24th international conference on Machine learning. 2007: 791-798.
- [40] Bassily R, Smith A, Thakurta A. Private empirical risk minimization: Efficient algorithms and tight error bounds[C]//2014 IEEE 55th Annual Symposium on Foundations of Computer Science. IEEE, 2014: 464-473.
- [41] McSherry F D. Privacy integrated queries: an extensible platform for privacy-preserving data analysis[C]//Proceedings of the 2009 ACM SIGMOD International Conference on Management of data. 2009: 19-30.
- [42] Thakurta A G. Differentially private convex optimization for empirical risk minimization and high-dimensional regression[M]. The Pennsylvania State University, 2013.

- [43] Lee J, Kifer D. Concentrated differentially private gradient descent with adaptive per-iteration privacy budget[C]//Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery Data Mining. 2018: 1656-1665.
- [44] Balle B, Wang Y X. Improving the Gaussian mechanism for differential privacy: Analytical calibration and optimal denoising[C]//International Conference on Machine Learning. PMLR, 2018: 394-403.
- [45] Shokri R, Shmatikov V. Privacy-preserving deep learning[C]//Proceedings of the 22nd ACM SIGSAC conference on computer and communications security. 2015: 1310-1321.
- [46] 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.
- [47] Song S, Chaudhuri K, Sarwate A D. Stochastic gradient descent with differentially private updates[C]//2013 IEEE Global Conference on Signal and Information Processing. IEEE, 2013: 245-248.
- [48] Geyer R C, Klein T, Nabi M. Differentially private federated learning: A client level perspective[J]. arXiv preprint arXiv:1712.07557, 2017.
- [49] Truex S, Baracaldo N, Anwar A, et al. A hybrid approach to privacy-preserving federated learning[C]//Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security. 2019: 1-11.
- [50] Nesterov Y. Introductory lectures on convex optimization: A basic course[M]. Springer Science Business Media, 2003.
- [51] M, Lantz E, Jha S, et al. Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing[C]//23rd USENIX Security Symposium (USENIX Security 14). 2014: 17-32.

- [52] McMahan H B, Ramage D, Talwar K, et al. Learning differentially private recurrent language models[J]. arXiv preprint arXiv:1710.06963, 2017.
- [53] Geyer R C, Klein T, Nabi M. Differentially private federated learning: A client level perspective[J]. arXiv preprint arXiv:1712.07557, 2017.
- [54] Bhowmick A, Duchi J, Freudiger J, et al. Protection against reconstruction and its applications in private federated learning[J]. arXiv preprint arXiv:1812.00984, 2018.
- [55] Truex S, Liu L, Chow K H, et al. LDP-Fed: Federated learning with local differential privacy[C]//Proceedings of the Third ACM International Workshop on Edge Systems, Analytics and Networking. 2020: 61-66.
- [56] Comiter M. Attacking artificial intelligence[J]. Belfer Center Paper, 2019: 2019-08.
- [57] Abadi M, Chu A, Goodfellow I, et al. Deep learning with differential privacy[C]//Proceedings of the 2016 ACM SIGSAC conference on computer and communications security. 2016: 308-318.
- [58] Papernot N, Abadi M, Erlingsson U, et al. Semi-supervised knowledge transfer for deep learning from private training data[J]. arXiv preprint arXiv:1610.05755, 2016.
- [59] Xie L, Lin K, Wang S, et al. Differentially private generative adversarial network[J]. arXiv preprint arXiv:1802.06739, 2018.
- [60] Jordon J, Yoon J, Van Der Schaar M. PATE-GAN: Generating synthetic data with differential privacy guarantees[C]//International conference on learning representations. 2018.
- [61] Zhang J, Zheng K, Mou W, et al. Efficient private ERM for smooth objectives[J]. arXiv preprint arXiv:1703.09947, 2017.

- [62] Wang D, Ye M, Xu J. Differentially private empirical risk minimization revisited: Faster and more general[J]. arXiv preprint arXiv:1802.05251, 2018.
- [63] Wang D, Chen C, Xu J. Differentially private empirical risk minimization with non-convex loss functions[C]//International Conference on Machine Learning. PMLR, 2019: 6526-6535.
- [64] Abadi M, Chu A, Goodfellow I, et al. Deep learning with differential privacy[C]//Proceedings of the 2016 ACM SIGSAC conference on computer and communications security. 2016: 308-318.
- [65] Bach S, Binder A, Montavon G, et al. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation[J]. PloS one, 2015, 10(7): e0130140.
- [66] Sashank J Reddi, Ahmed Hefny, Suvrit Sra, Barnabas Poczos, and Alex Smola. Stochastic variance reduction for nonconvex optimization. In International conference on machine learning, pages 314–323, 2016.
- [67] Shokri R, Shmatikov V. Privacy-preserving deep learning[C]//Proceedings of the 22nd ACM SIGSAC conference on computer and communications security. 2015: 1310-1321.
- [68] Krizhevsky A, Hinton G. Learning multiple layers of features from tiny images[J]. 2009.
- [69] Melis L, Song C, De Cristofaro E, et al. Exploiting unintended feature leakage in collaborative learning[C]//2019 IEEE Symposium on Security and Privacy (SP). IEEE, 2019: 691-706.
- [70] Shokri R, Stronati M, Song C, et al. Membership inference attacks against machine

learning models[C]//2017 IEEE Symposium on Security and Privacy (SP). IEEE, 2017: 3-18.

- [71] Bonawitz K, Ivanov V, Kreuter B, et al. Practical secure aggregation for privacy-preserving machine learning[C]//proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. 2017: 1175-1191.
- [72] Balle B, Bell J, Gascón A, et al. The privacy blanket of the shuffle model[C]//Annual International Cryptology Conference. Springer, Cham, 2019: 638-667.
- [73] Young T, Hazarika D, Poria S, et al. Recent trends in deep learning based natural language processing[J]. ieee Computational intelligenCe magazine, 2018, 13(3): 55-75.
- [74] Aono Y, Hayashi T, Wang L, et al. Privacy-preserving deep learning via additively homomorphic encryption[J]. IEEE Transactions on Information Forensics and Security, 2017, 13(5): 1333-1345.
- [75] Dwork C. A firm foundation for private data analysis[J]. Communications of the ACM, 2011, 54(1): 86-95.
- [76] Bach S, Binder A, Montavon G, et al. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation[J]. PloS one, 2015, 10(7): e0130140.
- [77] Binder A, Montavon G, Lapuschkin S, et al. Layer-wise relevance propagation for neural networks with local renormalization layers[C]//International Conference on Artificial Neural Networks. Springer, Cham, 2016: 63-71.
- [78] Aji A F, Heafield K. Sparse communication for distributed gradient descent[J]. arXiv preprint arXiv:1704.05021, 2017.

致 谢

时光荏苒，岁月如梭，研究生的日子过得飞快，转眼间我的硕士研究生学习生涯即将接近尾声。在华东师范大学读研的这两年时光，我不但学习到了很多知识，也结识了许多良师益友，此时此刻，我的内心充满了无限的感慨。所谓饮水思源，在此我要向每位陪伴我，鼓励我，教导我的人表示由衷的感谢。

从 2019 年收到华东师范大学的研究生录取通知书，我满怀憧憬和抱负的来到华师大，来到上海可信计算实验室，有幸成为曹珍富老师的学生。感谢实验室的各位老师们，他们不但为我们提供了优质教学环境和资源，还创造了良好的学习氛围，通过一流的科研实力和丰富的科研热情带领我们学习最前沿的科研成果。为了充实我们的研究生生活，学院定期举办各种学术会议和活动，邀请到国内外知名学者给我们做讲座，让我们有机会接触到最新的科研成果。而且，无论是在科研还是生活上遇到问题，老师们都会耐心的给我们提建议，鼓励帮助我们一起克服这些困难。

研究生的时光是轻快而稍纵即逝的，和实验室同学、室友的朝夕相处是我最难忘的回忆。因为有室友高圆圆、陈少敏、冯世玲，宿舍的氛围一直是欢快的，我们早晨共同早起去图书馆自习，下课了去实验室读论文，空闲时间一起在操场打篮球，欢声笑语，常伴我们。三年时光里，我们彻夜未睡，通宵准备数模竞赛；早出晚归，一起在理科楼度过日日夜夜，都将成为我的学生时代美好的回忆。

同门情谊似手足之情，感谢实验室的各位同窗好友，吴楠、汤琦、陆鹏皓、李翔宇、任城东、李明冲等，是有你们的互励互助，我才得以开心努力而充实的度过了这段美好的研究生生活，希望以后仍然

有机会共同努力、共同奋斗。

最后，非常感谢我的父母和家人一直以来对我的鼓励与陪伴。在研究生生涯的这两年，我更加深刻体会到未来自己身上所担负的责任，希望我在未来的工作中能兢兢业业，踏实负责，实现我的社会价值；在未来的生活中，希望我能多多陪伴我的父母以回报养育之恩。

在这篇论文完成之际意味着三年的硕士生涯即将告一段落，而自己也将踏上人生的下一段旅程。回顾硕士三年的时光，非常有幸能成为华东师范大学的学子。非常庆幸能成为曹珍富老师的学生，非常庆幸能和实验室的大家成为朋友，这是人生中可遇而不可求的经历。最后，也感谢各位评审和答辩的专家在百忙之中对我论文的指导，谢谢你们。

何慧娴

二零二壹年九月

攻读硕士学位期间发表论文、参与科研和获得荣誉情况

■ 已完成学术论文

- [1] 第一作者, 第二作者. Adaptive Privacy-preserving and Shuffling Aggregation in Federated-learning[C]. 2021 The 11th International Workshop on Computer Science and Engineering, Shanghai, China.[第一作者]