2023 年全国大学生信息安全竞赛

作品报告

作品名称： 医疗隐私数据保护：纵向联邦学习场景下差分隐私算法设计

作者姓名： 谢天，郭俊荣，马海越

提交日期： 2023.06.14

目录

摘要

随着医疗服务走向数据驱动的时代，数据共享成为了重要需求。但是，由于医学数据极具隐私性，数据流动成为困难，这就导致单一组织难以获得足够数量的可用样本来训练人工智能算法。本作品旨在利用纵向联邦学习解决此数据孤岛问题，保证各方数据不出本地的情况下安全高效地协同完成模型训练。

在传统的假设中，中央端是可信的第三方，但实际上中央端可能会窃取梯度，进而泄露隐私信息。差分隐私是常用的隐私保护机制，但传统的 DP-SGD 算法会使模型精度下降，且计算开销大。本文提出了一个仅针对隐私数据列的差分隐私随机响应算法 flip 与 flipback，只需进行一次差分隐私操作即可达到隐私保护效果，同时使得模型精度与泛化性保持较高水平。实验结果表明该算法能有效提升模型准确率和收敛速度。此外，我们还探讨了仅有隐私参数设置与模型性能的关系，并在真实医疗数据集上进行了模型构建与实验评估。

随着数据规模的不断扩大和数据隐私的日益受到重视，差分隐私在联邦学习等领域具有广泛的应用前景。采用本文提出的针对隐私数据列的差分隐私随机响应算法 flip 与 flipback，可以有效保护用户隐私数据的安全性，同时保持模型精度不变，因此可应用于医疗、金融、教育等领域需要保护隐私信息的场景中。此外，由于该算法仅需进行一次差分隐私操作即可达到隐私保护效果，且无需对权重矩阵进行裁剪，计算局部敏感度等操作，因此具备较高的可扩展性与实用性，将对未来隐私保护领域的发展产生积极的影响。

关键字：纵向联邦学习　半同态加密　差分隐私　密码学应用

第一章 作品概述

1.1 背景分析

针对 AI 模型的训练，通常需要使用大规模数据以获得较高水平的性能和鲁棒性。但是，随着各种用户隐私保护法规的实施，医疗 AI 所面临的最大挑战之一便是医疗数据的获取。由于医疗数据难以跨机构、跨地域进行分享和应用，医疗 AI 的研究和应用受到了限制。

在宏观层面，目前，欧盟的 GDPR 是全球范围内适用最为广泛的隐私保护法规之一。该法规于 2018 年实施，规定了个人健康信息的隐私保护标准和实施指南，确立了数据主体权利、数据控制方、处理方的权利和义务，并限制了将个人数据转移至第三国家的行为。美国加州也颁布了类似的 CCPA 的法规，在一定程度上授权了消费者了解自己的信息被谁收集和分享的权利。尽管这些法规有效地保护了用户的隐私权益，但也阻碍了医疗 AI 的发展和普及。

而具体到现实生活中，在许多医院里，医疗程序会涉及到多个部门的多个人，如果要使病人的治疗变得有效和高效，部门间医疗人员的决定和行动需要相互协调。例如，社区医院中的全科医生怀疑他的病人可能患有乳腺癌，但是，由于其缺乏相关的知识与经验，也没有资源来证实这一假设，他必须将病人转诊到能够做出明确诊断的专家或者医院中。因此预测性深度学习模型，在辅助医疗诊断方面有着良好的应用前景，它可以辅助全科医生对病人进行判断，对患者是否患病进行初步筛查，减少医院诊断流程，同时也方便了患者。

随着医疗服务进入数据驱动的时代，数据的开放共享成为了一个重要的需求。与此同时，由于医学数据具有极高的隐私性，必须采取高度的保护措施。管理医疗数据的机构通常要求这些数据不得出院、出机构、出市、出省、出国。因此，在医疗数据的利用方面，我们面临着一个不可避免的问题：单一组织很难获得足够数量可用的样本来训练人工智能算法。

目前在医学领域，解决多机构协作的患者数据孤岛问题通常采用将协作机构的患者数据共享至一个集中的数据库的方式。然而，不同医学领域往往拥有各自对应的患者数据库，从而使得协作数据共享无法有效扩展到大量医疗机构之间。此外，由于患者隐私的保护问题以及数据所有权归属问题，协作数据共享对于实现跨机构的协作学习而言仍存在困难。因此，本文旨在利用联邦学习技术来解决医疗机构之间的数

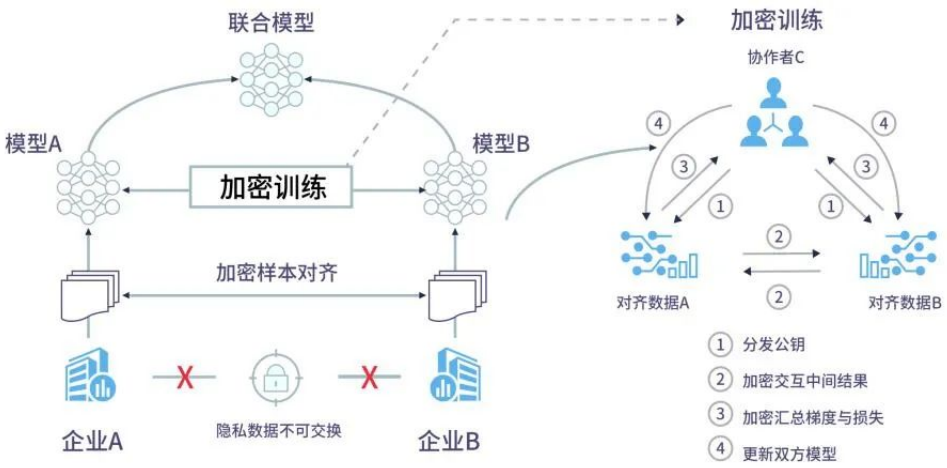

图 1.1　联邦学习模型示意图

据隐私问题。

联邦学习是一种面向安全的大数据的机器学习技术，其应用场景十分广泛。联邦学习系统在解决“数据孤岛”问题中主要有以下优势：

1. 安全性：联邦学习系统采用 RSA,Hash 等加密机制，可确保仅使用交集部分数据进行多方交互，而不会泄露差集部分数据。此外，对梯度和损失计算所需的中间结果进行加密和掩码处理，以确保真实的梯度信息不会向参与方泄露。
2. 无损性：联邦学习模型可采用同态加密技术，在传输过程中不需要传输各方的原始数据，同时这些解密后的数据具有可计算性。
3. 共享性：相比于单独一方，联合建模机制提高了模型的准确性。与数据集中建模相比，联邦学习技术保证了模型质量无损和模型的可解释性。
4. 公平性：联邦学习技术可确保参与方的公平性，使得各方都能在数据独立条件下建立联合训练模型。

综上所述，联邦学习技术对于解决多机构协作的数据孤岛问题具有重要意义。本作品就基于一个真实乳腺癌医疗数据集训练一个联邦模型，完成了分类和预测功能，并提出了一种适用于联邦模型的、符合差分隐私的隐私保护算法，并进行了后续降噪处理，在注重隐私性保证的同时兼顾了模型的准确率。

1.2 相关工作与产品特色

在论文 Federated Machine Learning: Concept and Applications[4] 里，我们总是假定中央端是可信第三方。由于中央端拥有会话的公私钥，GuestA 与 HostB 回传的加密梯度在 C 端均可被解密。

然而论文 Deep Leakage from Gradients[5] 表明，即便加入了高斯噪声作为干扰项，从泄露的梯度中重构整个数据集依旧是可能的，这对隐私数据造成了极大威胁。本作品认为中央端 C 为半诚实端，有从梯度明文中窃取并恢复隐私信息的可能，因此，我们需要使得敏感数据对 C 不可见。

差分隐私 [3] 是一种常用的隐私保护机制。传统的方式 DP-SGD 算法 [1] 是在梯度中添加 Laplace 或 Guassian 噪声，使得数据集混淆，但该方法需要在每轮训练时都对权重矩阵进行裁剪，计算局部敏感度，再向梯度中加噪，这不仅仅增加了计算复杂度，更是使得模型精度大幅下降。

本作品提出了一种仅面向隐私数据列的差分隐私随机响应算法 flip 与 flipback。该算法仅仅需要在预训练前进行一次差分隐私操作，每轮训练时无需再次进行差分隐私, 即可达到隐私保护效果。保护了隐私的同时，该算法也增强了模型的泛化性，使之在测试集上的准确率与未加入隐私保护机制前的模型相当。

我们认为数据集里仅有部分敏感列需要进行差分隐私保护，而对外泄露其他数据是不敏感的。具体而言，在本作品使用的 breast cancer 数据集中，仅有数据列“是否患有癌症”被认为是医疗敏感信息。在模型开始训练前，我们依据 flip 算法对其进行数据值反转，由上文证明可知该算法符合 $\epsilon-$ 差分隐私，又由于 Post-Processing 的后处理性，我们对数据再次进行降噪操作并不会降低隐私参数 ϵ。因此每隔 k 轮进行模型训练时，我们根据 k-1 轮模型可计算出敏感数据先验概率，依此可对模型进行优化，利用 flipback 算法进一步翻转，实验表明该算法使得模型准确率得以提升，并且加快了收敛速度，泛化性更好。

1.3 应用前景

在医学研究和精准医疗方面，该作品的应用可以促进医学研究的发展和精准医疗的实现。由于医疗数据通常存储在不同的机构中，传统的集中式数据分析方法面临数据隐私泄露的风险。而基于差分隐私的纵向联邦模型构建方法能够实现跨机构数据的安全共享和协同研究。

不同医疗机构可以共同建立模型，通过保护个体隐私的同时，共享各自的数据资源，加强数据分析能力，从而加速疾病诊断、治疗方案研究等领域的进展。通过联合多个医疗机构的数据进行模型构建和分析，可以获得更大规模的数据集，从而提高模型的准确性和可靠性。借助差分隐私技术的引入，个体的隐私数据得到有效保护，医疗机构和患者之间的信任也得以增强。

随着数据规模的不断扩大和数据隐私的日益受到重视，差分隐私在联邦学习等

领域具有广泛的应用前景。采用本文提出的针对隐私数据列的差分隐私随机响应算法 flip 与 flipback，可以有效保护用户隐私数据的安全性，同时保持模型精度不变，因此可应用于医疗、金融、教育等领域需要保护隐私信息的场景中。此外，由于该算法仅需进行一次差分隐私操作即可达到隐私保护效果，且无需对权重矩阵进行裁剪，计算局部敏感度等操作，因此具备较高的可扩展性与实用性，将对未来隐私保护领域的发展产生积极的影响。

第二章 作品设计与实现

2.1 算法描述

2.1.1 纵向联邦学习模型基本架构

联邦学习旨在建立一个基于分布数据集的机器学习模型。在模型训练的过程中，模型相关的信息能够在各方之间交换（或者是以加密形式交换），但原始数据不能。这一交换不会暴露每个站点上数据的任何受保护的隐私部分。

设有 N 位参与方 $F_1, F_2 \ldots\ldots F_N$ 协作使用各自的训练数据集 $D_1, D_2 \ldots\ldots D_N$ 来训练机器学习模型。传统的方法是将所有的数据 $D_1, D_2 \ldots\ldots D_N$ 收集起来并且存储在同一个地方，例如存储在某一台云端数据服务器上，从而在该服务器上使用集中后的数据集训练得到一个机器学习模型 M_{SUM}。在传统方法的训练过程中，任何一位参与方会将自己的数据暴露给服务器甚至其他参与方。联邦学习是一种不需要收集各参与方所有的数据便能协作训练一个模型 M_{FED} 的机器学习过程。设 ν_{SUM} 和 ν_{FED} 分别为集中型模型和联邦型模型的性能度量。在使用安全的联邦学习在分布式数据源上构建机器学习模型时，我们允许在保护用户隐私的情况下，联邦学习模型的性能略低于集中型模型的性能。其中 $|\nu_{FED} - \nu_{SUM}| \leq \delta$，$\delta$ 即为允许的性能损失。

而纵向联邦学习（**V**ertical **F**ederated **L**earning）是联邦学习的一大重要分支。当几个参与方拥有同一批用户不同属性的数据时，他们便可使用 VFL 进行协同训练。在 vFL 中，拥有用户特征的参与方（又称为 Guest 方，如下图参与方 A）会持有一个下层网络（Bottom Model），他们将特征输入下层网络，计算得到中间结果（embedding），发送给拥有标签的参与方（简称 Host 方，如下图参与方 B），Host 方使用这些 embedding 和自己持有的标签来训练上层网络（上层网络），再将算得的梯度回传给各个参与方来训练下层网络。由此可见，VFL 不需要任何参与方上传自己的原始数据即可协同训练模型。

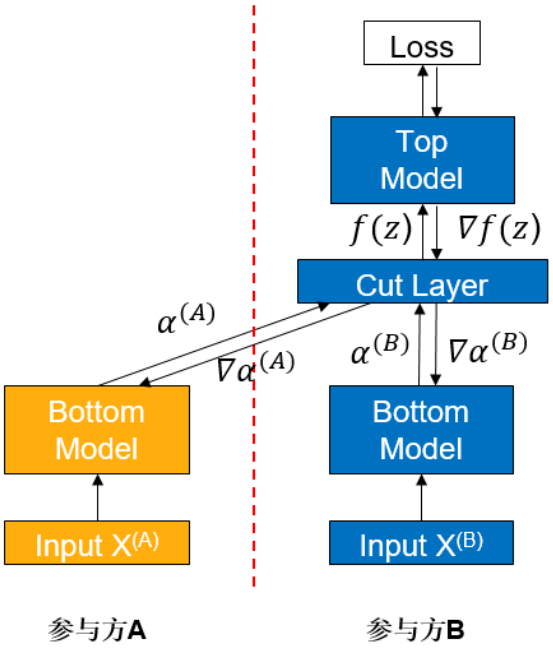

图 2.1　纵向联邦学习模型架构

2.1.2 纵向联邦模型构建算法

本作品的模型采用 logistic 回归实现对 breast cancer 数据集训练和预测功能。

设 y 为二元标签值，取值分布为 {0,1}，表征该用户样本是否患有癌症。而 $X_1, X_2, \ldots X_n$ 表示该数据集的 n 重特征列，如半径、纹理、凹度等等。在 n 维特征列作用下该样本为阳性的概率为

$$P = p(y = 1|X_1, X_2, \ldots X_n)$$

则回归模型可表示为：

$$p = \frac{exp(\beta_0 + \beta_1 X_1 + \beta_2 X_2 \cdots + \beta_n X_n)}{1 + exp(\beta_0 + \beta_1 X_1 + \beta_2 X_2 \cdots + \beta_n X_n)} \tag{2.1}$$

记 $\theta = (\beta_0, \beta_1, \beta_2 \ldots \beta_n)^T$ 经过 logit 变换，即

$$logit(P) = \ln\left(\frac{P}{1-P}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 \cdots + \beta_n X_n = \theta^T X + b$$

下面进行一些简化操作，将偏置项 b 合并进入权重矩阵 θ 内：

$$\theta^T = \begin{pmatrix} \theta^T \\ b \end{pmatrix}$$

那么相应的，可以得到 y 与 x 的关系：

$$y = \frac{1}{e^{-\theta^T X}}$$

则其损失函数和梯度求导后分别如下所示：

$$J(\theta) = \frac{1}{n}\sum_{i\in S}\log\left(1+e^{-y_i\theta^T x_i}\right) \tag{2.2}$$

$$\frac{dJ}{d\theta} = -\frac{1}{n}\sum_{\in S}\left[y_i\log\left(\frac{1}{1+e^{-\theta^T x_i}}\right)+(1-y_i)\log\left(\frac{1}{1+e^{-\theta^T x_i}}\right)\right]+\frac{\lambda\theta}{n} \tag{2.3}$$

其中 $\frac{\lambda\theta}{n}$ 为正则项，阻止模型过拟合。

2.1.3 同态加密算法引入和梯度函数改造

可以看出，在梯度计算过程中，特征数据（x）和标签数据（y）是必不可少的。因此，一种最直接的数据交换方案是其中一方将自己独有的数据明文发送给对方，对方计算出梯度后再返回。然而，这样的数据交互方式存在信息泄露的风险，因为其中一方会获得全部的信息，这显然是不符合规范的。

为了避免明文传输带来的风险，可以采用将需要的数据以密文形式发送的方法。此时就需要引入同态加密算法。

同态加密算法是一种特殊的加密技术，它具有特殊的性质：在密文域中进行的加、减、乘和除操作等价于在明文域中执行相应的操作。也就是说，如果我们使用同态加密算法对数据进行加密，那么在密文域中进行计算所得到的结果可以被正确地映射回明文域。这使得在不泄露数据的情况下完成计算成为可能。

因此，通过同态加密算法，我们可以将特征数据和标签数据加密后发送给计算方进行模型训练，从而实现隐私保护和计算任务的同时完成。这种方法使得数据拥有者可以安全地将其数据用于联邦学习，并且可以有效地保护数据隐私。

本系统的纵向联邦学习算法将基于 Paillier 算法实现。由上文公式推导可见，损失2.2和梯度2.3的公式中包含着指数运算，因此，如果要用 Paillier 算法进行加密，需要对原公式进行一定的改造，使其仅用加法和乘法来表示。将指数运算改造为加法与乘法运算的一个常用方法就是用泰勒展开来进行近似。利用

$$\log\left(1+e^{-z}\right) = \log 2 - \tfrac{1}{2}z + \tfrac{1}{8}z^2 + O(z^4)$$

改写后的梯度公式如下所示：

$$\frac{dJ}{d\theta} = \begin{bmatrix} X_A^T\left(\frac{1}{4}X_A\theta_A^T + \frac{1}{4}X_B\theta_B^T - y + 0.5\right) + \lambda\theta_A \\ X_B^T\left(\frac{1}{4}X_A\theta_A^T + \frac{1}{4}X_B\theta_B^T - y + 0.5\right) + \lambda\theta_B \end{bmatrix} \times \frac{1}{n}$$

2.1.4 差分隐私机制的引入

差分隐私（Differential Privacy）[3] 是一个用数学语言描述的隐私定义，是算法所具有的属性。

定义 1 给定算法 $M : \mathrm{X}^n \to \mathrm{Y}$, 考察 X^n 中任意两个数据集 X, X', 称满足 $||X - X'||_1 \leq 1$ 的为邻近数据集。我们称如下算法算法 M 是 $\epsilon-$ 差分隐私的, 如果：

对于任意邻近数据集 $X, X' \in \mathrm{X}^n$, 以及任意 $T \in \mathrm{Y}$, 都满足:

$$Pr[M(X) \in T] \leq e^{\epsilon} Pr[M(X') \in T] \tag{2.4}$$

为了衡量该算法对隐私的保护性，我们引入隐私参数 ϵ 的定义。

$$\epsilon = \ln\left(\frac{Pr|M(x) \in T|}{Pr|M(y) \in T|}\right)$$

这意味着：无论输入是否包含任意特定个体的数据，算法 M 的输出的概率之间差别只有 e^{ϵ}。当隐私参数 ϵ 足够小时，M 所引入的随机性就相对的较大，使得观察 M 的输出无法判断输入是 x 还是 x', 由此达到了隐私性

2.1.5 差分隐私两条定理及其安全性证明

在证明我们的算法满足 $\epsilon-$ 差分隐私前，我们先证明 $\epsilon-$ 差分隐私的两个重要定理：

定理 1 (Post-Processing 性) 令 $M : X^n \to Y$ 是 $\epsilon - DP$ 的，令 $F : Y \to Z$ 为任意映射，那么 $F \circ M$ 也是 $\epsilon - DP$ 的

证明 1 对邻进数据集中任意一对 $x, y, \|x - y\|_1 \leq 1$, 取 $S \subseteq Y$,
令 $T = \{r \in R : f(r) \subseteq S\}$，则有：

$$\begin{aligned} Pr(f(M(x)) \in S) &= Pr(M(x) \in T) \\ &\leq exp(\epsilon) Pr(M(y) \in T) \\ &= exp(\epsilon) Pr(f(M(y)) \in S) \end{aligned}$$

差分隐私是一个强大的隐私保护工具，但是要实现强差分隐私约束往往需要增加大量噪声。一般而言，数据的分析包括多个环节，如果在每个环节上都满足差分隐私约束，势必会令数据分析结果失真。差分隐私的传递性给出了一个强有力的断言：只要

在数据分析的任何一个环节，随机算法 M 满足 ϵ-DP。，那么在仅用该环节的结果作为输入的任意之后的数据处理过程都满足 ϵ-DP。

后处理性（Post-Processing）意味着在差分隐私机制的输出结果上执行任意计算也总是安全的。任意计算均不会降低差分隐私机制所提供的隐私保护程度。特别地，我们甚至可以对计算结果进行后处理，以降低噪声量、改善输出结果, 这也是我们 flipback 算法的依据。

定理 2 (Composition 性) *$M = (M_1, M_2, \ldots, M_k)$ 是一系列 $\epsilon - DP$ 函数 (选取是顺序的，自适应的)，那么最终 M 是 $k\epsilon - DP$ 的*

证明 2 *考虑临近数据集 X, X', 对于函数顺序输出的 $y = (y_1, \ldots, y_k)$, 我们有：*

$$\begin{aligned}\frac{Pr(M(X)=y)}{Pr(M(X')=y)} &= \prod_{i=1}^{k} \frac{Pr(M_i(X)=y_i|M_1(X)=y_1,\ldots,M_i(X)=y_i)}{Pr(M_i(X')=y_i|M_1(X')=y_1,\ldots,M_i(X')=y_i}) \\ &\leq \prod_{i=1}^{k} exp(\epsilon) \\ &= exp(k\epsilon)\end{aligned}$$

组合后的隐私预算要高于每个差分隐私机制，即多个差分隐私进行组合后，隐私暴露的可能性增加了。在相同的隐私参数下，为了避免隐私预算的增加，我们需要尽量减少差分隐私算法执行的次数。

2.1.6 差分隐私随机响应算法设计及其隐私证明

我们提出了差分隐私算法 flip 以对数据集进行随机响应操作，使之在训练过程中保持了 ϵ 隐私性；为了降低该算法可能带来的模型精度下降，我们又设计了相关的 flipback 算法进行后处理操作。

Algorithm 1: Flip(Differential Priavcy Random Response).

Input: Dataset $\mathcal{D}$ with labels(the Host side);Privacy Parameter ϵ

Output: A new dataset $\mathcal{D}'$ with random process.

Given $\mathcal{D}$ which only the label column is sensitive,suppose there are k labels differently,

Given $\mathcal{Y}$ as the set of all labels, and $\mathcal{Y}_i$ as i-th label;

Flip_prob = $Pr(Flip(y_i) \neq y_i) = \frac{e^\epsilon}{k-1+e^\epsilon}$;

num_flip = Size($\mathcal{D}$) $\times$ Flip_prob

Indices = Random Choose num_flip samples from $\mathcal{D}$

foreach *sample with label* $\mathcal{Y}_i$ *in Indice* **do**

1.Let $\mathcal{Y}_i$' = $\mathcal{Y} - \mathcal{Y}_i$

2.Uniformly choose one label $\mathcal{Y}_j$ from $\mathcal{Y}_i$'

3.Let i-th label change into $\mathcal{Y}_j$

定理 3 *Flip 算法是满足 $\epsilon - DP$ 的算法*

证明 3 针对本作品中采用的二元标签集，y_i 取值仅有 0 与 1，证明是显然的

$$Pr(F(y_i) = y_i) = \frac{e^\epsilon}{1+e^\epsilon}$$

$$Pr(F(y_i) \neq y_i) = \frac{1}{1+e^\epsilon}$$

$$\therefore \ln\left(\frac{Pr(F(y_i) = y_i)}{Pr(F(y_i) \neq y_i)}\right) = \frac{\frac{e^\epsilon}{1+e^\epsilon}}{\frac{1}{1+e^\epsilon}} = \epsilon$$

故而该算法是 $\epsilon-$ 差分隐私的，隐私性大小由隐私参数 ϵ 控制。

该算法实现了训练前对标签数据集的扰动，以一定概率输出原标签，而以概率 $\frac{1}{1+e^\epsilon}$ 将二元标签翻转。带来的隐私性由隐私参数 ϵ 衡量，若 ϵ 越小，相应的翻转概率越大，数据集置乱程度增加，隐私保护程度越高，但是相应的模型准确率收到了干扰。

为了消减 flip 操作对模型精度的影响，我们提出了一个后处理算法 flipback。根据定理1的后处理性，在差分隐私机制的输出结果上执行任意计算也总是安全的。任意计算均不会降低差分隐私机制所提供的隐私保护程度。于是我们可以在模型训练过程中对计算结果进行后处理，以改善输出结果。

Algorithm 2: Flipback

Input: model weight θ_{i-1} from last epoch $i-1$; hyperparameter k;

Output: update new model weight θ_i

foreach *sample i with label y_i* **do**

  Compute y_predict = $1/(1+e^{-\theta_{i-1}^T x})$

  Compute distance between y & y_predict

  Return distance

Sort distance matrix by descending order

foreach *dist_i in distance matrix* **do**

  If dist_i rank top k:

    execute flip algorithm on label_i

Compute current weight θ_i

该算法旨在利用了每一轮训练的结果作为先验知识，提高下一轮训练的准确率。

在上一轮模型训练完后我们得到一个权重矩阵 θ_{i-1}，根据它可以计算出本轮标签值的先验概率。我们可以依据此先验概率对模型做一些降噪处理，优化输出。

由于训练初始阶段部分标签遭到有意的污染，我们不妨认为在执行预测任务时，该样本会和其他样本产生较大差异，具体的可能表现为预测值与真实值之间的差值较大。于是选取差异最大的前 k 个标签进行 flip 翻转操作。k 为预设的超参值，经实验验证，以 1% $\sim$ 4% 为宜。

根据 Composition 性2, 再次翻转会使得总体 ϵ 增加，在本作品中，我们可以设置 flipback 算法执行的次数 t 来控制 ϵ 的增量，如历经 t 轮训练后才执行一次 flipback。

实际上，翻转的样本很可能是训练前未经过 flip 的样本，那么 flipback 操作相当于进一步置乱模型，增强了隐私性；同时起到与正则项的作用，对原梯度加入了一个惩罚项，阻止过拟合，提高模型泛化能力。在后续作品测试中，我们定量对 flipback 的正则项作用进行探究分析。

2.2 系统架构

在本作品中，将有标签（是否患癌症）的一方设为 HostB，可为模型训练提供 label, 不含标签仅持有特征数据列的另一方设为 GuestA, 再引入中央端 SeverC 实现密钥分发，实现了一个基本的纵向联邦学习架构。

由于 GuestA 端不含有标签列，无法单独进行预测任务；而 HostB 虽然可独立建模，但含有特征列较少。现在 AB 端联合建模，效果显然会超过 B 端单方建模。

在构建纵向联邦模型的过程中，三方 GuestA、HostB 和 ServerC 分别进行如下操

作：

1. A,B 先进行模型初始化，初始化参数参照 Kaiming 初始化 [2] 配置，Host 方 B 利用差分隐私算法 flip 进行敏感数据列的隐私保护
2. 中央端 C 进行公私钥产生与分发
3. A，B 端计算自己的模型信息，A 计算 $\frac{1}{4}\mathrm{X_A}{\theta_\mathrm{A}}^T$ 并半同态加密后发送给 B，B 计算 $\frac{1}{4}\mathrm{X_B}{\theta_\mathrm{B}}^T - y + 0.5$ 半同态加密后发送给 A
4. AB 将各自梯度发送给 C 进行中央端操作
5. 中央端 C 接受双方梯度进行计算，返回相关数据给 AB，完成一轮模型训练
6. 每隔 k 轮（k 为预设超参) 进行一次 flipback 后处理降噪算法，重复以上过程直至模型收敛，评估模型性能

本作品的系统架构如下表所示：

	GuestA	HostB	SeverC
Step0	Kaiming 初始化	Kaiming 初始化并执行 flip 算法	
Step1			Paillier 算法生成公私钥分发
Step2	$\mathrm{E}[\frac{1}{4}\mathrm{X_A}{\theta_\mathrm{A}}^T] \to B$	$\mathrm{E}[\frac{1}{4}\mathrm{X_B}{\theta_\mathrm{B}}^T - y + 0.5] \to A$ $\mathrm{E}[\mathcal{L}] \to C$	
Step3	半同态加密梯度	半同态加密梯度	
Step4	$\mathrm{E}[\frac{\mathrm{d}J}{\mathrm{d}\theta_A}] \to C$	$\mathrm{E}[\frac{dJ}{\mathrm{d}\theta_\mathrm{B}}] \to C$	
Step5			解密 $\mathcal{L}$ 发送 $\frac{dJ}{d\theta} \to A, B$
Step6	更新本地模型	更新本地模型	
Step7	每 k 轮执行一次 flipback 算法 循环执行 Step1-Step7 直至收敛		

表 2.1　作品基本架构

2.3 具体实现

2.3.1 model: 三个参与方的定义

1. 设置参与方的父类，各参与方都需要保存模型的参数、一些中间计算结果以及与其他参与方的连接状况。

```
class Client:
    def __init__(self, config):...
    def connect(self, client_name, target_client):...
    def send_data(self, data, target_client):...
```

2. GuestA 在训练过程中仅提供特征数据。成员函数包括计算 A 的梯度、加密梯度，自身参数的更新；在训练过程中与 B，C 之间的信息传送。

```
class GuestA(Client):
    def __init__(self, X, config):...
    def compute_z_a(self):...
    def compute_encrypted_dJ_a(self, encrypted_u):...
    def update_weight(self, dJ_a):...
    def task_1(self, Host_B_name):...
    def task_2(self, Sever_C_name):...
    def task_3(self):...
```

3. Host 方 B 在训练过程中既提供特征数据，又提供标签数据。成员函数包括计算 B 的梯度、加密梯度，自身参数的更新；在训练过程中与 A，C 之间的信息传送，以及 B 执行 flip 算法。

```
class HostB(Client):
    def __init__(self, X, y, config):...
    def compute_u_b(self):...
    def compute_encrypted_dJ_b(self, encrypted_u):...
    def update_weight(self, dJ_b):...
    def task_0(self):...
    def task_1(self, Guest_A_name):...
    def task_2(self,Sever_C_name):...
    def task_3(self):...
    def task_flipback(self,Guest_A_name):...
```

4. 参与方 C 在整个训练过程中主要的作用就是分发秘钥，以及最后的对 A 和 B 加密梯度的解密。

```
class SeverC(Client):
    ......
```

2.3.2 utils: 数据集加载与分配，以及计算，绘图

1. 数据集加载

- 加载乳腺癌数据集
- 返回处理好的训练集、训练集标签、测试集、测试集标签
- 数据拆分，将数据集随机划分为训练集和测试集，其中`random_state`为随机数种子，保证每次划分结果一致
- 数据标准化，使用`StandardScaler`进行数据标准化（z-score 标准化）

```
from sklearn.datasets import load_breast_cancer
def load_data():
    ......
```

2. 将样本进一步拆分，分配给 guest 方 A 和 host 方 B，A 获得部分特征数据，B 获得部分特征数据与标签数据。

```
def vertically_partition_data(X, X_test, A_idx, B_idx):
    ......
```

3. 预测函数`sigmoid`激活函数：

$$y = \frac{1}{1 + e^{-w^T x}}$$

输入测试集与权重矩阵，计算预测值。

```
def predict_sigmoid(weights, X_test):
    ......
```

4. metrics：计算评估函数：输入真实标签和预测值，返回值 accuracy, precision, recall 和 F1 score

```
def calculate_metrics(y_true, y_pred):
    ......
```

5. 训练效果展示：绘制随着 eposh 轮次增加，每轮 loss 和 accuracy 的变化曲线。

```
import matplotlib.pyplot as plt
def loss_acc_fig(loss_A, accuracy_A, loss_B, accuracy_B):
    ......
```

2.3.3 main: 垂直逻辑回归做预测

```
def vertical_logistic_regression(X, y, X_test, y_test, config):
    ......
```

下面对该函数具体执行过程进行详细解释：

1. 使用垂直逻辑回归算法做预测，输入处理好的训练集、训练集标签、测试集、测试集标签，以及训练过程中所需的参数；将样本进一步拆分后，进行各参与方的初始化，建立连接：

```
## 获取数据
XA, XB, XA_test, XB_test = vertically_partition_data(X, X_test,
    config['A_idx'], config['B_idx'])
print('XA:',XA.shape, ' XB:',XB.shape)

## 各参与方的初始化
Guest_A = GuestA(XA, config)
print("Guest_A successfully initialized.")
Host_B = HostB(XB, y, config)
print("Host_B successfully initialized.")
Sever_C = SeverC(XA.shape, XB.shape, config)
print("Sever_C successfully initialized.")

## 各参与方之间连接的建立
Guest_A.connect("B", Host_B)
Guest_A.connect("C", Sever_C)
Host_B.connect("A", Guest_A)
Host_B.connect("C", Sever_C)
Sever_C.connect("A", Guest_A)
Sever_C.connect("B", Host_B)
```

2. B 首先执行 flip 算法，而后进行迭代训练`n_iter`次，在训练过程中，双方的具体通信过程如表2.1，具体函数执行：

```
for i in range(1,config['n_iter']+1):
```

- C 进行 Paillier 算法生成公私钥分发给 A 和 B；
- 参与方 A 计算，使用公钥加密后发送给 B。参与方 B 计算，使用公钥加密后发送给 A 但是 B 在每进行五轮训练，要进行一次 flipback 后处理降噪算法；
- 此时 A 和 B 能各自同态计算加密后的梯度；
- A 和 B 需要加密的梯度发送给 C 来进行解密，但为了避免 C 直接获得梯度信息，A 和 B 可以将梯度加上一个随机数 R_A 与 R_B 再发送给 C。C 获得加密梯度进行后进行解密再返还 A 和 B；
- A 和 B 只需要再减去之间加的随机数就能获得真实的梯度，更新其参数。

```
        print(f"**********epoch{i}**********")
        Sever_C.task_1("A", "B")
        Guest_A.task_1("B")
        if i%5 == 0:
            Host_B.task_flipback("A")
        else:
            Host_B.task_1("A")
        Guest_A.task_2("C")
        Host_B.task_2("C")
        Sever_C.task_2("A", "B")
        Guest_A.task_3()
        Host_B.task_3()
```

3. 每轮训练完成，A 和 B 都进行一次 *sigmoid* 预测，记录损失和预测准确率：

```
        ### A做预测
        yA_pred = predict_sigmoid(Guest_A.weights, XA_test,
            np.zeros_like(XA_test.shape[0]))
        yA_accuracy, yA_precision, yA_recall, yA_f1 = calculate_metrics(y_test,
            yA_pred)
        print(f"yA_accuracy:{yA_accuracy}, yA_precision:{yA_precision},
            yA_recall:{yA_recall}, yA_f1:{yA_f1}")
        accuracy_A.append(yA_accuracy)

        ### B做预测
        yB_pred = predict_sigmoid(Host_B.weights, XB_test,
            np.zeros_like(XB_test.shape[0]))
        yB_accuracy, yB_precision, yB_recall, yB_f1 = calculate_metrics(y_test,
            yB_pred)
        print(f"yB_accuracy:{yB_accuracy}, yB_precision:{yB_precision},
            yB_recall:{yB_recall}, yB_f1:{yB_f1}")
        accuracy_B.append(yB_accuracy)
```

4. 所有训练过程完成，展示训练效果展，即绘制随着 eposh 轮次增加，每轮 loss 和 accuracy 的变化曲线。

```
    print("All process done.")
    loss_acc_fig(Sever_C.loss, accuracy_A, Sever_C.loss, accuracy_B)
    return True
```

2.3.4 运行函数

在config.json中设置本次训练所需要的参数:

```
{
    "n_iter": 30,
    "lambda": 0.1,
    "lr": 0.005,
    "A_idx": [10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28,
        29],
    "B_idx": [0, 1, 2, 3, 4, 5, 6, 7, 8, 9],
    "epsilon": 8.0
}
```

然后执行main.py:

```
X, y, X_test, y_test = load_data()
config = json.load(open('.\Code\config.json'))
vertical_logistic_regression(X, y, X_test, y_test, config)
```

第三章 作品测试与分析

3.1 测试环境

- Python3.7
- 同态加密库：phe
- 机器学习库：sklearn
- 基本数据分析与可视化库：numpy,matplotlib

运行 Code 文件夹下 main.py 即可。更改模型数据集可根据配置 config.json 和 utils.py 完成。

3.2 测试方案

本作品测试方案如下：

1. 在`config.json`中，保持正则化参数 λ 不变，多次改变隐私参数 ϵ，以尝试观测在不同的隐私保护强度下，模型 accuracy 的变化，来评估模型性能；
2. 在`config.json`中，保持隐私参数 ϵ 不变，多次改变正则化参数 λ，观测验证差分隐私机制是否相当于在模型内增添了正则项以提高模型泛化能力，
3. 对比上述实验组，观察改变隐私参数 ϵ 和正则化参数 λ 对 accuracy、loss 和模型收敛速度的影响，得出测试结论，进行算法评估。

3.3 测试数据与图表分析

3.3.1 测试 1：改变隐私保护强度观测模型准确率变化

保持正则参数 $\lambda = 2$ 不变，分别令隐私参数 ϵ =2、4、8，测试模型的准确率，得到以下结果：

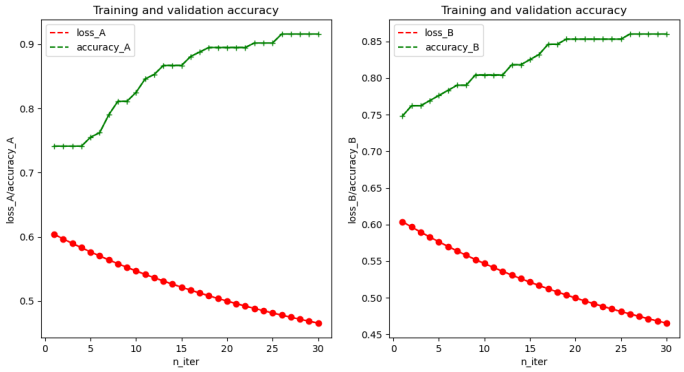

图 3.1　未加隐私保护算法时 A 和 B 的 bottom model 的 loss 和 accuracy 随训练轮次变化图

A model accuracy	B model accuracy
0.916083916083916	0.8601398601398601

表 3.1　$\epsilon = \infty$ 时模型最终准确率

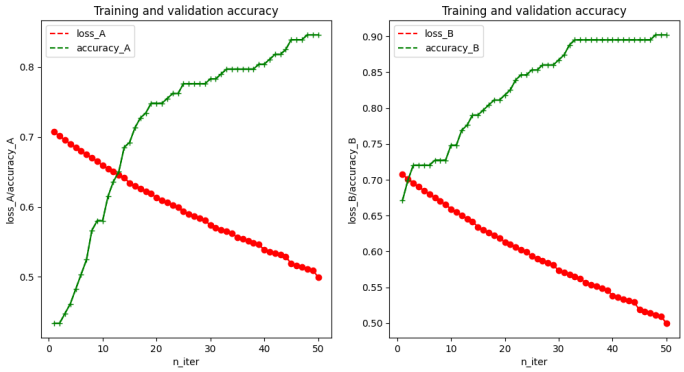

图 3.2　$\epsilon = 2$ 时 A 和 B 的 bottom model 的 loss 和 accuracy 随训练轮次变化图

A model accuracy	B model accuracy
0.8461538461538461	0.9020979020979021

表 3.2　$\epsilon = 2$ 时模型最终准确率

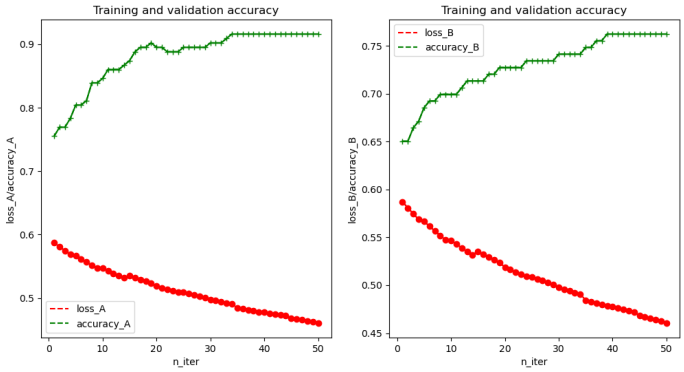

图 3.3　$\epsilon = 4$ 时 A 和 B 的 bottom model 的 loss 和 accuracy 随训练轮次变化图

A model accuracy	B model accuracy
0.916083916083916	0.7622377622377622

表 3.3　$\epsilon = 4$ 时模型最终准确率

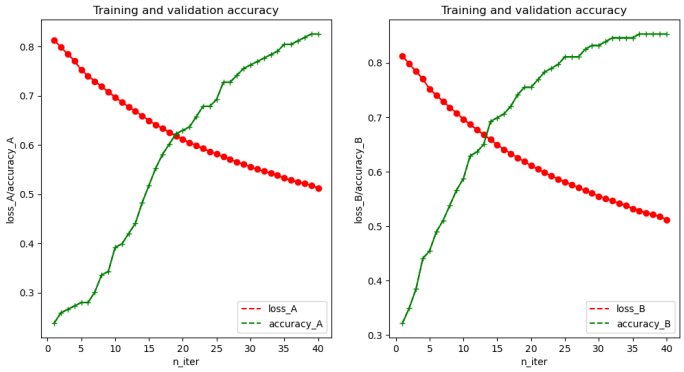

图 3.4　$\epsilon = 8$ 时 A 和 B 的 bottom model 的 loss 和 accuracy 随训练轮次变化图

A model accuracy	B model accuracy
0.8251748251748252	0.8531468531468531

表 3.4　$\epsilon = 8$ 时模型最终准确率

根据以上结果，可做出下图：

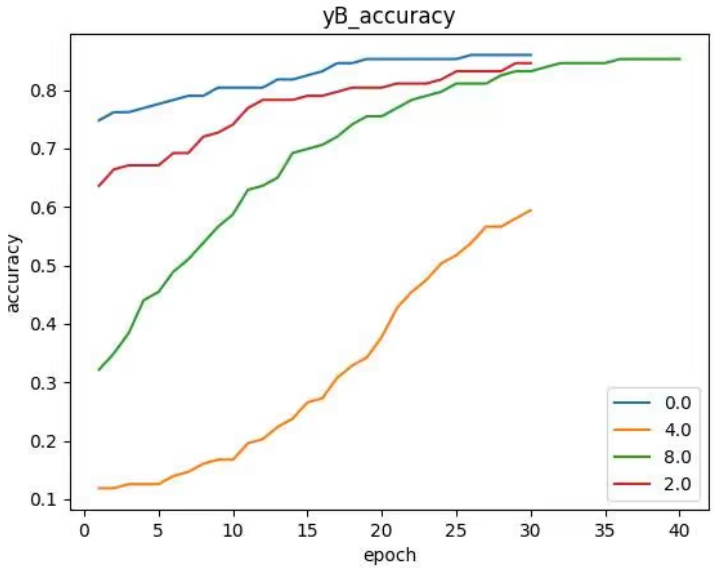

图 3.5　改变隐私参数 ϵ 对模型 accuracy 的影响

虽然本模型架构为基础的 logistic 回归模型，且为了贴合半同态加密算法的限制，优化器选择了自主实现的 SGD 算法，但是最终模型的 accuracy 依然保持了较高水准。在经过 30 轮训练后，无隐私算法干扰组已收敛，使得原 Guest 方 A 在原本并无标签列的情况下，经过和 B，C 方的加密梯度传输通信后，得以对新数据样本做出预测，且预测准确率高达 91.6%。而 Host 方 B 在纳入学习了来自 A 的特征列后，模型经过迭代也达到了 86% 的准确率。

根据差分隐私的数学定义，隐私参数 ϵ 表征两个数据集输出概率的差异，隐私参数预设值越小，算法的隐私保护程度越高。

我们选取 $\epsilon = \infty$ 作为对照组，即不进行任何隐私保护，不对数据集进行任何干扰，直接参与模型训练, 如图3.5，可以看到在不受隐私算法扰动后，模型准确率相对其他设置时一直最高, 并且在前 20 轮时就已经收敛。

随着 ϵ 从 8 开始降低，隐私算法扰动逐渐增强，模型的收敛速度和准确率开始受到波动。ϵ 为 8 时，扰动较小，可以看到模型收敛速度相比对照组放缓不少，不过由于隐私性较低，最终经过 40 轮迭代后收敛 accuracy 与对照组相当。

相比之下，ϵ 为 4 时隐私性保护增强了一个数量级，扰动加剧，AB 模型收敛时的准确率分别为 96% 和 76%，并且收敛速度大幅降低。分析比较可见，在隐私性增强的同时，Guest 方的准确率反而达到了新高，可能是隐私算法对 B 数据集做扰动的同时强化了 A 的 bottom model 的泛化能力，使其在预测能力上更为突出；而相对的，B 的 label 在训练初期被大量随机翻转，而 flipback 算法使得翻转进一步加剧，B 的

bottom model 被扰动后造成了预测 acuuracy 的下降。

但是 ϵ 为 2 时的模型，相较于前两个隐私参数的设置，这次的隐私性更强，但同时模型收敛速度却达到了三者最快，最终 B 的 acuuracy 达到了最高 90.3%，这可能是由于 flipback 算法设置时二次翻转概率系数的设置，使得训练后期部分错误标签得以复原，从而提高了 B 的 bottom model 的 accuracy。然而 A 已经接受了来自 B 的干扰梯度参与训练，后续无法得以复原，故 A 的 bottom model 准确率略有降低，为 84%。

3.3.2 测试 2：探究隐私参数强度设置是否增强了模型鲁棒性

保持正则参数 $\epsilon = 4$ 不变，分别令隐私参数 ϵ =0.1、0.2、0.8、2，测试模型的准确率，得到以下结果：

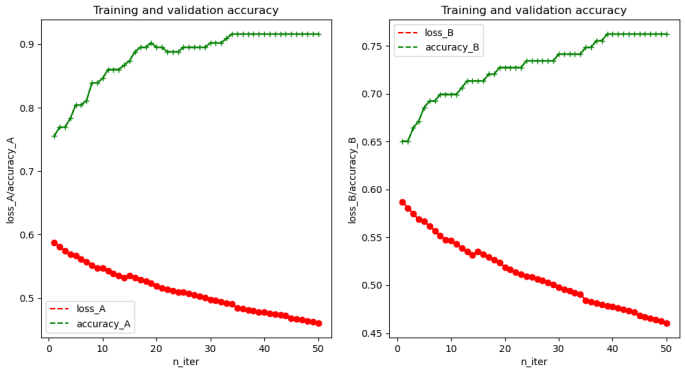

图 3.6　$\lambda = 2$ 时 A 和 B 的 bottom model 的 loss 和 accuracy 随训练轮次变化图

A model accuracy	B model accuracy
0.916083916083916	0.7622377622377622

表 3.5　$\lambda = 2$ 时模型最终准确率

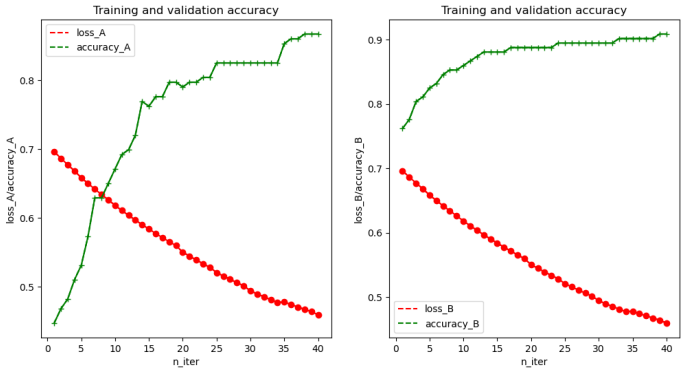

图 3.7　$\lambda = 0.8$ 时 A 和 B 的 bottom model 的 loss 和 accuracy 随训练轮次变化图

A model accuracy	B model accuracy
0.8671328671328671	0.9090909090909091

表 3.6　$\lambda = 0.8$ 时模型最终准确率

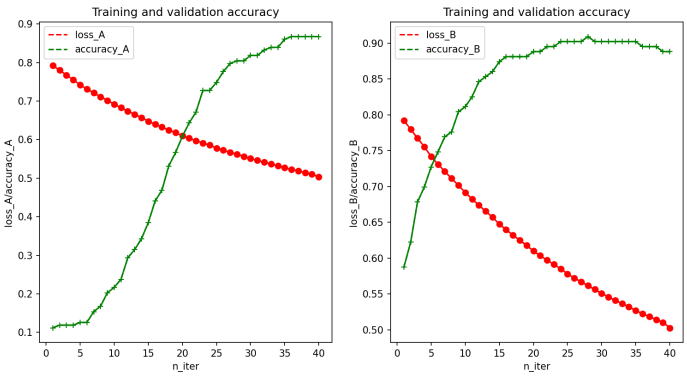

图 3.8　$\lambda = 0.2$ 时 A 和 B 的 bottom model 的 loss 和 accuracy 随训练轮次变化图

A model accuracy	B model accuracy
0.8671328671328671	0.8881118881118881

表 3.7　$\lambda = 0.2$ 时模型最终准确率

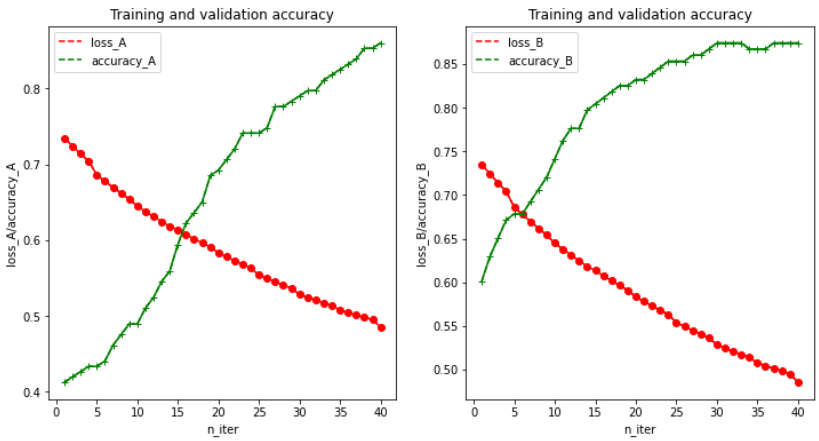

图 3.9　$\lambda = 0.1$ 时 A 和 B 的 bottom model 的 loss 和 accuracy 随训练轮次变化图

A model accuracy	B model accuracy
0.8601398601398601	0.8741258741258742

表 3.8　$\lambda = 0.1$ 时模型最终准确率

根据以上结果，可做出下图：

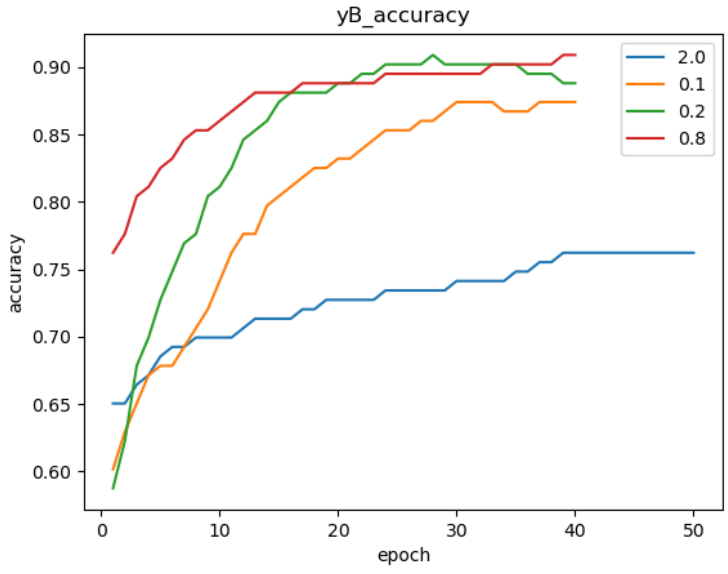

图 3.10　改变正则化参数 λ 对模型 accuracy 的影响

为了探究 ϵ 设置对模型性能的影响，我们继续进一步实验。隐私算法可能起到了类似正则项的作业，添加随机扰动以阻止模型过拟合，从而提高模型的预测能力。那

么在隐私算法扰动的前提下，削减正则项系数应该有利于提升模型性能，下面是实验验证：

我们选取 $\lambda = 2$ 作为对照组，这是在不进行隐私保护时，反复得出的最优正则项系数。我们在此标准值上进行上下随机采样选取几个值，比对模型性能作出分析。

将 λ 从 2 开始降低，正则化项的权重减小，模型准确率和收敛速度提升了很多。λ 为 0.8 时，将原有的正则化项的权重减少近一半，可以看到 AB 模型收敛时的准确率分别为 87% 和 91%，B 模型的准确率提升了 20%，并且收敛速度大幅加快。

但是将 λ 继续减小到 0.2，比 λ 为 2 时正则化项的权重减小了一个数量级，模型的训练误差减弱，AB 模型收敛时的准确率分别为 86% 和 88%，λ 为 0.8 时相差不大，但在第 28 轮训练时模型准确率达到最高点后开始下降。其收敛速度也比 λ 为 0.8 慢些。

我们尝试将 λ 继续减小到 0.1，相较于前两个正则化参数的设置，该模型相较前两个实验组泛化能力更弱，在图3.10体现为模型准确率较 $\lambda = 0.8$ 和 $\lambda = 0.2$ 时准确率下降，收敛速度减慢。但仍比对照组 $\lambda = 2$ 准确率高，收敛速度快。

分析比较可见，将对照组的 λ 减小后，实验组模型的准确率和收敛速度都有提高。但若 λ 太低，会使模型的泛化能力减弱，造成过拟合。因此，应选择合适的正则化参数 λ，注意隐私算法对正则化参数 λ 调整的影响，从而平衡模型的训练误差和模型复杂度之间的关系。

综上，隐私算法确实起到了类似正则项的作用，在提高隐私性的同时也加强了模型的泛化能力，使之更具鲁棒性，在测试集上预测效果表现更加优异。

3.4 算法评估与模型总结

在联邦学习中加入差分隐私随机响应机制的实验中，观察到差分隐私机制在一定程度上起到了正则项的作用，从而提高了模型的泛化能力。通过降低过拟合风险，差分隐私机制帮助模型在未见数据上表现更好。

同时，引入差分隐私机制也有效地保护了参与方的隐私信息。差分隐私技术通过添加噪声或扰动来随机化数据，成功地保护了个体隐私，并对隐私泄露的概率进行了量化分析。该机制对隐私攻击具有一定的鲁棒性，并能够实现一定程度的隐私保护。

在调节差分隐私参数 ϵ 和正则化参数 λ 方面，需要根据具体场景进行权衡和调优。不同的差分隐私参数 ϵ 设置会对模型性能和隐私保护产生不同的影响，而正则化参数 λ 则影响模型的泛化能力，同时差分隐私机制在一定程度上起到了正则项的作用，因此在调节差分隐私参数 ϵ 和正则化参数 λ 时，需要综合考虑。

然而，值得注意的是，在实际应用中，差分隐私机制可能面临一些限制和挑战。例如，计算开销的增加和性能下降等问题需要进一步研究和解决。因此，未来的工作可以探索优化差分隐私参数选择算法和研究更有效的差分隐私机制，以进一步提升模型性能和隐私保护的效果。并且，未来可以选择一些更复杂的神经网络模型以实现模型优化。

第四章 创新性说明

4.1 面向联邦学习场景下半诚实中央端进行隐私保护

本作品在假定纵向联邦学习里中央端也为半诚实端的场景下，设计隐私算法使得传递给中央端的梯度被混淆，从而无法通过差分攻击等方式重构敏感数据。

我们认为数据集里仅有部分敏感列需要进行差分隐私保护，而对外泄露其他数据是不敏感的。在训练前期即对敏感数据列进行隐私保护操作，利用差分隐私的数学保证实现整个通信过程中的敏感数据安全。

4.2 定量化处理隐私性和模型评估指标

隐私性与准确率是一对矛盾的命题，二者不可兼得。本作品通过引入差分隐私中隐私参数 ϵ 的定义，从而定量衡量算法的隐私性；同时，从模型的准确率，精确率, 召回率，F1score4 个评价指标给出了模型的衡量标准，便于进行测试评估。

4.3 设计了基于差分隐私的随机响应机制

传统的差分隐私方式 DP-SGD 算法是在梯度中添加 Laplace 或 Guassian 噪声，使得数据集混淆，但该方法需要在每轮训练时都对权重矩阵进行裁剪，计算局部敏感度，再向梯度中加噪，这不仅仅增加了计算复杂度，更是使得模型精度大幅下降。

本作品提出了一种仅面向隐私数据列的差分隐私随机响应算法 flip，该算法仅仅需要在预训练前进行一次差分隐私操作，每轮训练时无需再次进行差分隐私, 即可达到隐私保护效果。

4.4 提出了一种与隐私保护算法配套的后处理算法

由于 flip 差分隐私算法给数据集带来了不可逆的随机扰动，对模型精度产生了负面效果，基于此本作品提出了一种后处理算法 flipback，来进行后续降噪处理。在注重隐私性的同时，增强模型鲁棒性，提高模型的预测能力。

第五章 总结

本文研究了联邦学习中的隐私问题，并提出了一种针对隐私数据列的差分隐私随机响应算法 flip 与 flipback。实验结果表明，该算法能够有效地保护隐私数据的安全性，同时保持模型精度和泛化性能在较高水平。同时，由于该算法无需对权重矩阵进行裁剪等计算复杂度高的操作，具有较高的可扩展性和实用性。我们在 breast cancer 数据集上进行了实验验证，并探讨了仅有部分敏感列需要进行差分隐私保护的情况。结果表明，flip 与 flipback 算法可以使得模型准确率得以提升，并且加快了模型的收敛速度，同时保护了隐私数据的安全性。

总之，本文对于联邦学习中的隐私保护问题提出了一种新颖的解决方案，有效处理了传统 DP-SGD 算法计算复杂度高、精度下降等问题，具有一定的创新性。在实验验证中，该算法取得了显著的性能改善，不仅提升了模型精度和收敛速度，同时保证了隐私数据的安全性，展示了良好的应用前景。未来，该算法可以在医疗、金融、教育等领域需要保护隐私信息的场景中应用。从更广泛的角度来看，本文提出的解决方案对于隐私保护领域的发展会产生积极的影响，并为未来的研究提供了一定的参考价值。

参考文献

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, oct 2016.

[2] Wadii Boulila, Maha Driss, Mohamed Al-Sarem, Faisal Saeed, and Moez Krichen. Weight initialization techniques for deep learning algorithms in remote sensing: Recent trends and future perspectives, 2021.

[3] Zhanglong Ji, Zachary C. Lipton, and Charles Elkan. Differential privacy and machine learning: a survey and review, 2014.

[4] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications, 2019.

[5] Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients, 2019.