安全多方混合线性回归

1 问题和挑战

近年来, 数据安全和隐私保护的重要性日益增加, 工业界对联合机器学习建模中的隐私问题愈加关注, 而联邦学习被认为是一种保证安全性前提下实现多方建模的重要手段, 不同的联邦学习模型和系统被相继提出[YLCT19, ZPGS19, MZ17], 然而一些问题和挑战仍旧存在。

第一个问题是因选择横向或者纵向模型造成的数据浪费[1]。在联合建模的场景下, 由于数据样本在ID和特征上的限制, 训练和推理时必须选择ID全部对齐或者特征全部对齐, 无法对齐的样本则不能参与训练和推理。这种方式将所有的训练和预测数据限制在能够对齐ID的样本范围之内, 使得在很多场景下大量数据被浪费。试考虑一个两方(A方和B方) 建模的场景, 其中A中的样本ID为1, 2, 3, 4, ... 5000, 特征为$feature_1, feature_2, ..., feature_{10}$; B方样本ID为3000, 3001, 3002, ..., 12000, 特征为$feature_6, feature_7, ..., feature_{25}$。若采用横向联邦学习, 则只有6个特征参与训练和推理; 若采用纵向联邦学习, 则只有5001个样本能够参与训练, 见Fig.1。 此外, 在推理阶段, 无法进行ID对齐(或者特征)对齐的样本则不能进行推理, 导致在一些场景下区分横向纵向的联邦学习方法无法很好地满足场景的需求。仍以以上的数据举例, 假设A方为模型的主要使用方, AB方通过纵向联邦学习方法得到了模型M。在推理阶段, A方的每一条推理请求(query) 都需要和B方的已有样本进行ID对齐, 否则无法得到结果。在很多场景下, 能够对齐的样本往往仅占总体样本的较小一部分, 区分横向纵向的方式将使得很多数据无法进行有效的推理。

第二个是联邦学习训练的效率问题。由于联邦学习分布式的场景和应用其

[1] 当前联邦学习方法[YLCT19] 将联邦学习算法分为横向和纵向两大类, 横向模型用于解决ID不一致但全部一致的情况; 而纵向模型用于解决特征不一致但ID一致的情况。

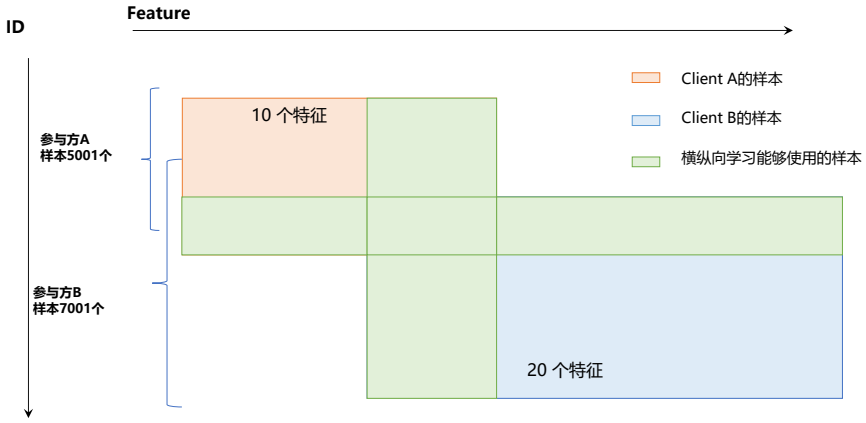

Figure 1: 在很多情况下, 两方联合建模时无论采用横向或者纵向学习, 均会有大量的样本被浪费, 造成精度损失。

中的诸多加密手段, 训练过程中的计算时间消耗和网络传输大大高于传统的分布式学习。一般来说, 在同态加密等手段下训练联邦学习模型的计算量是常规训练方法的百倍以上; 其次, 频繁的密文传输使得网络通信的时间消耗成为限制联邦模型训练效率的重要因素之一。一些系统的训练时间常常多达数十个小时[FAT21], 使得模型无法在较为关注时间效率的场景下使用。

第三个是模型和系统的易用性问题。算法系统的易用性是一个算法产品是否成功的关键之一; 此外, 系统的部署难易程度也直接决定着用户的使用体验。联邦学习算法涉及分布式计算, 多方安全计算和机器学习, 系统一般较为复杂, 缺乏经验的用户往往难以对联邦学习算法的各种参数和算法细节有较为深入的了解, 而分数据横向纵向的算法则进一步对于用户的使用造成了门槛。因此, 易用且可靠的算法和系统是对联邦学习算法设计的一大挑战。

针对第一个问题, 在第3节, 我们提出一种新的混合联邦学习方法, 使得模型和输入数据不需要区分横纵。“混合”的意思是指:对于输入数据, 不区分横向纵向, 若数据即包含横向数据和纵向数据, 则将横向数据和纵向数据混合起来作为模型的输入数据。在训练阶段, 能够使所有的样本都可以参与训练, 以达到更好的训练精度; 在推理阶段, 参与推理的样本不需要进行ID对齐, 使得单独一方也能够进行推理; 针对第二个问题, 在第4节, 我们在BFGS的基础上针对联邦学习

场景提出一种新的算法改进, 利用联邦学习中的目标函数为可分函数(seperable function)的性质对不同类型的样本和特征加以区分, 以提高原始算法的训练速度; 其次, 在第5节, 我们引入多方安全计算, 通过秘密分享和效率较高的半同态加密, 为模型的训练和推理提供安全保证。最后, 对于第三个问题, 本文提出的对于输入数据没有横向纵向限制的混合方法, 使得完全没有算法背景知识的用户也能轻松使用算法。结合能够一键部署的系统框架, 提升了产品的易用性, 降低了使用门槛, 优化了用户体验。

2 纵向联邦线性回归

纵向联邦线性回归的方法最先由[YLCT19]提出。在[YLCT19]中, 作者提出了一种基于半同态加密和存在可信第三方的纵向联邦学习算法。在本节, 我们简述[YLCT19]所提出的纵向线性回归算法。

总体来说, 纵向线性回归的损失函数和预测函数与普通的线性回归算法均相同。不同之处在于训练数据分布在不同的参与方, 为了通过梯度下降法进行训练, 梯度的计算需要由多方共同完成。在计算梯度时, 每一个参与方通过利用自己拥有的那部分数据, 计算出梯度的中间结果并在一个可信第三方的参与下完成梯度汇总并得到最终的梯度值, 从而进行参数更新。为了防止梯度和原始数据泄露, 作者对传输到其他参与方的所有数据进行了半同态加密。因此, 在其他参与方上进行的计算将全部在半同态加密下完成。

具体地, 给定学习率η,正则化参数λ, 和p个参与方, 它们的数据集分别记为$\{X_i^{(1)}\}_{i\in D_1}, \{X_i^{(2)}\}_{i\in D_2}, ..., \{X_i^{(p)}\}_{i\in D_p}$, 每个参与方特征空间所对应的线性回归参数分别为$W_1, W_2, ..., W_p$, 总的参数集合为$W = W_1 \cup W_2 \cup ... \cup W_p$, 假设其中某一个参与方, 此处不妨记为参与方$z$,拥有label, 记为$\{y_i\}_{i\in D_z}$, 那么训练损失函数则可表示为:

$$
\begin{aligned}
L(W) &= \frac{1}{n}\sum_{i=1}^{n}(\sum_{j=1}^{p} W_j X_i^{(j)} - y_i)^2 + \frac{\lambda}{2}(\sum_{j=1}^{p}||W_j||^2) \\
&= \frac{1}{n}\sum_{i=1}^{n}(\sum_{j=1}^{p} \hat{y}^{(j)} - y_i)^2 + \frac{\lambda}{2}(\sum_{j=1}^{p}||W_j||^2)
\end{aligned}
$$

训练参数关于损失函数的梯度为:

$$\nabla_W L == \frac{2}{n}\sum_{i=1}^{n}(\sum_{j=1}^{p}\hat{y}^{(j)} - y_i)x + \lambda||W_j||$$

记$d_i = \sum_{j=1}^{p}\hat{y}^{(j)} - y_i$, 在计算$d_i$时, 需要将每个参与方的$\hat{y}$通过同态加密传至有label的那一方, 因此需要对其他参与方的$\hat{y}_i^{(j)}$进行加密以防止隐私泄露。记$[[\cdot]]$为加法同态加密, $[[d_i]] = \sum_{j=1}^{p}[[\hat{y}_i^{(j)}]] - [[y_i]]$。因此, 在同态加密下, 训练参数关于损失函数的梯度为:

$$[[\nabla_{W_j} L]] = \frac{2}{n}\sum_{i=1}^{n}([[d_i]]x_i^{(j)}) + [[W_j]]$$

2.1 纵向联邦线性回归的训练过程

根据以上的讨论, 在表1中, 我们给出纵向联邦线性回归的训练过程。在训练时, 除了每个参与方之外, 还需要引入一个可信的第三方。第三方负责同态加密公钥私钥的生成和梯度密文的解密。在[YLCT19]中, 为了防止第三方获取到训练参与方的梯度明文, 在解密前, 每个参与方对发送给第三方的待解密数据加入了随机掩码。在半诚实模型的假设下, 这种方式可以防止第三方获取训练参与方的隐私信息。在表1中, 假设所有参与方中只有一个有label, 将其记为参与方B; 由于除B之外的其他参与方的行为均相同, 我们将其中的一个记为参与方A, 并只描述A的步骤。

下面我们简要讨论表1中协议在在半诚实模型的假设下的安全性。在半诚实假设下, 各方严格遵循协议步骤执行算法。在步骤1, 2中, 各参与方传出的数据均为同态加密后的数据, A, B方均无私钥, 所以参与方A, B均无法获得除本方数据以外的任何信息。在步骤3中, 各参与方传输自己的梯度给第三方解密前加入了随机掩码, 所以第三方也无法获得训练参与方的任何信息。故1中协议在半诚实模型下是安全的。

2.2 纵向联邦线性回归的预测过程

纵向联邦线性回归的预测过程同样需要各个参与方协作并由一个可信第三方输出最终结果。表2总结了预测的过程。注意到各参与方直接发送数据给可信第三

	其他参与方 (将其中一个参与方记为A)	含有LABEL 的参与方 (假设为参与方B)	第三方
步骤1	初始化本方训练参数	初始化本方训练参数	生成同态加密的公私钥, 并将公钥发送给训练参与方
步骤2	计算$[[\hat{y}_A]]$	接收其他方$[[\hat{y}]]$,计算$[[d_i]]$并返回给其他参与方; 计算$[[L]]$ 并发给第三方	
步骤3	生成掩码R_A, 计算$[[\nabla_{W_A} L]]$, 加上$[[R_A]]$, 得到$[[\nabla_{W_A} L]] + [[R_A]]$, 并发送给第三方	生成掩码R_B, 计算$[[\nabla_{W_B} L]]$, 加上$[[R_B]]$, 得到$[[\nabla_{W_B} L]] + [[R_B]]$, 并发送给第三方	解密$[[L]]$, $[[\nabla_{W_A} L]] + [[R_A]]$, $[[\nabla_{W_B} L]] + [[R_B]]$ 并将结果分别返回给各个参与方
步骤4	更新本方训练参数	更新本方训练参数	

Table 1: 纵向联邦线性回归的训练过程

方, 在第三方是完全可信的情况下, 各个参与方的数据和结果并不会暴露给其他方。

	参与方 (将其中一个参与方记为A)	第三方
步骤1		将需要预测的数据ID i发送给各个参与方
步骤2	计算$[[\hat{y}_i^A]]$并发给第三方	计算$\sum_{j=1}^{p} \hat{y}_i^{(j)}$ 并返回结果

Table 2: 纵向联邦线性回归的预测过程

3 混合联邦线性回归

在本节我们提出一种新的联邦线性模型线性模型。?????。在本文中, “混合”的意思是指:对于输入数据, 不区分横向纵向, 若样本数据同时包含横向数据和纵向数据, 则将横向数据和纵向数据混合起来作为模型的输入数据。这样设计1) 可以提高模型的准确度; 2) 能够增强算法的易用性。

假设多方样本分布有一定的关联性, 不妨假设每个参与方的样本均为一个高维空间总体在不同子空间内的采样, 将这个高维空间记为S。从直觉出发, 不失一般性地, 假设在这个S中有一组参数W和函数F, 使得$Y = F(W, X)$(对于线性回归, F为线性函数; 对于逻辑斯蒂回归, F为softmax函数)。当样本分散在不同的子空间时, 它们在不同的维度上可能出现互补, 缺失或者交叉的情况。例如, AB两方联合建模, 假设双方样本ID均相同, A方拥有特征$feature_1, feature_2, feature_3$, B方拥有特征$feature_3, feature_4, feature_5$。注意到, 若能通过某种方式对于双方共有的$feature_3$赋权, 则将AB两方其他维度拼接起来, 即可得到在S中的原始数据样本, 则双方数据可以全部用于训练。相似地, 对于ID无法对齐的样本, 则将缺失的特征部分填充为空值, 只对非空部分加权。于是, 对于更一般的情况, 我们假设不同子空间具有不同的权重。混合线性算法的基本思想即是通过将不同参与方的特征加权, 并将各个参与方的样本扩展到S,然后近似S中的原函数$Y = F(W, X)$。

进一步严格描述, 假设一次训练中共有p个参与方, 记p个参与方的所有样本为集合为X, 每个参与方拥有一部分特征和数据, 对于参与方a, 若它有X中的第i个样本, 则记为x_i^a。注意由于各个参与方相同ID样本, 分别对应着同一个数据ID的不同特征, 因此不同参与方的同一个ID的样本对应的数据一般不同。例如x_i^a 和x_i^b 分别对应着同一个样本ID在A方具有的特征下的值和在B方具有的特征下的值。有时一条数据只在部分参与方有值, 那么在其他参与方, 这条数据的值为空。

首先,令S记作所有样本特征组成的空间。我们将每个参与方样本空间的维度扩展到S。由于我们不希望空值部分对预测值有任何影响, 在实际实现时, 对于原样本缺失的特征处直接补0。

接着, 我们对每个参与方的特征赋权。第一步是确定样本特征的重叠方式。对于横纵混合的数据而言, 例如参与方为ABC三方, 有的样本特征可能在AB两方重叠, 有的可能在BC两方重叠, 也有可能ABC三方都重叠或者仅仅在一方存在, 见Fig.2。 对于每一方的每一个样本, 需要确定它和哪些参与方有重叠。参与方数量为p时, 则重叠方式最多有$2^p - 1$种。第二步, 对每一种重叠方式赋予不同的权重变量。具体地, 对于第i个样本, 假设它的特征重叠参与方集合为O_i,

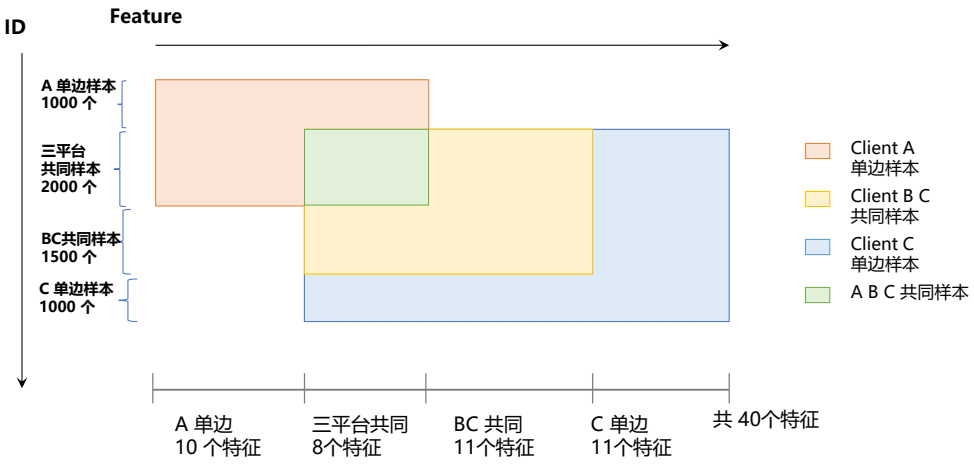

Figure 2: 多方数据的重叠方式：假设参与方为ABC三方, 有的样本特征可能在AB两方重叠, 有的可能在BC两方重叠, 也有可能ABC三方都重叠或者仅仅在一方存在。

令f_α为加权后的预测函数, f为softmax或者线性函数, 则预测值为,

$$\hat{y_i} \triangleq f_\alpha(W, \alpha, x_i) \triangleq \sum_{j \in O_i} f(\alpha_j \hat{x_i}^{(n)}, W)$$

其中α_j是第j个重叠参与方的权重。

由以上讨论, 再对权重总和的大小加入二范数正则, 以MSE作为Loss,可得混合线性回归的损失函数为:

$$L(W, \alpha) = \frac{1}{n} \sum_{i=1}^{n} (f_\alpha(W, \alpha, x_i) - y_i)^2 + (\sum_{j=1}^{p} \alpha_{p_j} - p)^2$$

值得注意的是, 在实际应用过程中, 我们有时可以获得各个参与方特征权重的先验知识。例如在不少情况下, 我们可以知晓数据采集方式, 预处理方式以及涉及特征重要性的相关专家经验等。则可以凭借经验对各种重叠方式人为设置权重, 此时混合线性方法退化为一般的线性回归。例如最简单地, 将α按照特征重合的次数按比例设置时, 在这种情况下, 对于完全纵向的三方建模, 则设置所

有的α均为1。在没有先验的情况下, 参与方的权重作为变量的一部分, 由训练得到。在下一章节, 我们介绍训练过程中用到的优化算法及其相关改进。

4 对一二阶导数计算的改进

为了加速训练, 对于有先验的情况(普通线性回归, 损失函数为凸函数), 我们采用二阶优化方法BFGS, 同时对其按照联邦学习的实际情况进行分布式改造；对于没有先验的情况, 损失函数为非凸函数, 为防止α和W相互影响造成无法收敛的情况, 我们采用最速下降法, 对W使用固定步长, 对α的步长采取回溯搜索的策略。对于步长回溯和BFGS的详细介绍, 可参考[NW06], 这里暂不赘述。为避免叙述过于冗长, 以下仅以普通线性回归为例进行介绍, 即α不作为训练参数。当α也作为训练参数时, 仍可根据类似的思想对α的求导过程加以改进。

区别于常用的机器学习中的分布式优化算法, 联邦学习在训练时的时间空间开销更加巨大, 原因主要在于：1)多方通信交互时需要同态加密, 单次乘法加法的时间消耗可达普通操作的百倍以上, 在同态加密下的运算成为了整个系统的时间瓶颈; 2)同态加密下的数据量呈指数级别增大, 对网络传输带来了挑战。对于这个问题, 我们希望对传统的一二阶优化方法进行改进：力求使尽量多的计算可以在不加密的情况下利用本地数据在本地完成, 最后将加密后的少量结果传输至master, 仅仅通过简单的同态加密运算步骤即可得到全局结果。

在第3节中, 我们将每个参与方的样本按照重叠模式进行了分类。由于线性函数的线性可加性, 对于完全没有重叠的样本, 可以将它们在本地计算后与有重叠样本的结果相加。因此, 我们将每一方的样本分为私有样本和非私有样本两部分。私有样本指各方完全没有特征重叠的样本；非私有样本则是除去私有样本以外的其他样本。通过线性映射, 将私有样本部分完全在本地计算。然后针对不可分的部分, 将相关中间结果传输到master上进行计算。

对于横纵混合的联邦学习目标函数, 令$l \triangleq MSE$, 令全空间S的维数为m, 参与方p_i的样本子空间为S_i的维数为m_{p_i}, X为全体样本的集合, 该集合的大小为n, X_{priv_i} 为私有样本的集合, $X - X_{priv}$为非私有样本的集合, 则损失函数可以写

成如下的加性可分函数(additive separable function):

$$L(W) \triangleq \sum_{x \in X_{priv_i}} l(f_{p_i}(x, U_{p_i}^T W), t) + \sum_{x \in X - X_{priv}} l(f(x, W), t)$$

其中f_{p_i} 是本地预测函数, 定义域为p_i方样本特征构成的子空间, 即是从S_i到R的函数; f 是全局预测函数, 定义域为S, 即f是从S到R的函数; U_{p_i}是从R^m到$R^{m_{p_i}}$的线性映射, 在实际算法实现时, 由于我们对于缺失的样本直接补0, U_{p_i}即为从R^m到$R^{m_{p_i}}$的indicator matrix, 若U_{p_i}的第i行第j列为1, 则表示S_i中的第j个特征位于S的第i个位置; $l(W)$为R^m上的损失函数; $l_{p_i}(Z)$为$R^{m_{p_i}}$(子空间)上的损失函数; t是需要拟合的目标值。令$Z = U_{p_i}^T W$, 对于上式求梯度和Hessian矩阵, 应用链式法则:

$$\begin{aligned}
n\nabla_W L &= \sum_{x \in X_{priv_i}} U_{p_i} \nabla_{f_{p_i}} l(f_{p_i}) \nabla_z f_{p_i}(z) + \sum_{x \in X - X_{priv}} \nabla_f l(f) \nabla_W f(W) \\
&= \sum_{x \in X_{priv_i}} U_{p_i} \nabla_Z l_{p_i}(Z, t) + \sum_{x \in X - X_{priv}} \nabla_f l(f) \nabla_W f(W) \\
n\nabla_W^2 L &= \sum_{x \in X_{priv_i}} U_{p_i} \nabla_W (\nabla_Z l_{p_i}(Z, t)) + \sum_{x \in X - X_{priv}} \nabla_f l(f) \nabla_W^2 f(W) \\
&= \sum_{x \in X_{priv_i}} U_{p_i} \nabla_Z (\nabla_Z l_{p_i}(Z, t)) \nabla_Z W + \sum_{x \in X - X_{priv}} \nabla_f l(f) \nabla_W^2 f(W) \\
&= \sum_{x \in X_{priv_i}} U_{p_i} \nabla_Z^2 l_{p_i}(Z, t)) U_{p_i}^T + \sum_{x \in X - X_{priv}} \nabla_f l(f) \nabla_W^2 f(W)
\end{aligned}$$

注意到$l_{p_i}(Z, t)$ 为本地函数, 故可将上式拆分在多个client计算。

4.1 对于私有样本: $x \in X_{priv_i}$

由上式可知, 私有样本的一二阶导数可完全在本地计算得到, 本地结果通过$U_{p_i}^T$映射到全空间R^m, 最后上传master通过一次reduce-sum操作得到最终结果。同理, 在使用拟牛顿法近似L对W的Hessian矩阵$\nabla_W^2 L$时, 可在每个参与方本地直接对$\nabla_{W_{p_i}}^2 L_{p_i}$进行近似, 然后将近似结果映射到全空间。既减小了计算复杂度, 同时也降低了近似稀疏高维矩阵时产生的误差。

由以上讨论, 可得私有数据部分的一阶导数和Hessian的逆的近似矩阵伪代

码Alg.1。

Algorithm 1 私有数据部分的一二阶导数计算(以参与方p, 第k 轮迭代为例)

输入: 第$k-1$轮的优化参数W_{k-1}^{p}和本方数据所得损失函数的梯度g_p和Hessian矩阵的逆矩阵H_p, 第p个参与方的训练数据Z_p和特征转换矩阵U_p

输出: 梯度值参与方p 的本地梯度g_p和Hessian矩阵的逆矩阵H_p

function GETGRADINVHESSPRIV(W_p)
  $g_{p_{last}} = g_p$
  $g_p = \nabla_w L(Z_p, W_p)$
  // 根据BFGS中的方法更新W
  $W_p = \text{UPDATEW}(W_p, H_p)$
  $d = g_p - g_{p_{last}}$
  $s = W_p - W_{k-1}^{p}$
  $\rho = 1/(d^T s)$
  // 根据BFGS中的方法更新H
  $H_p = \text{UPDATEH}(d, s, \rho)$
  return g_p, H_p
end function

4.2 对于ID重合的数据：$x \in X - X_{priv}$

4.2.1 梯度

损失函数对训练参数w_i的导数为:

$$\begin{aligned}\frac{\partial L}{\partial w_i} &= \frac{\partial(\sum_p L^{(p)})}{\partial f}\frac{\partial(\sum_p f^{(p)})}{\partial w_i} \\ &= (\sum_p \frac{\partial L^{(p)}}{\partial f})(\sum_p \frac{\partial f^{(p)}}{\partial w_i}) = \sum_p [\frac{\partial f^{(p)}}{\partial w_i}\sum_p \frac{\partial L^{(p)}}{\partial f}]\end{aligned}$$

由上式可总结除梯度的计算过程如下: 参与方在本地计算$\frac{\partial L_{(p)}}{\partial f}$, $\frac{\partial f^{(p)}}{\partial w_i}$并传给Master, Master聚合各方$\frac{\partial L^{(p)}}{\partial f}$得到$\frac{\partial L}{\partial f}$并发送给所有client; 然后每个client计算$\frac{\partial L}{\partial f}\frac{\partial f^{(p)}}{\partial w_i}$并发送给Master; 最后Master聚合各方结果得到一阶导数$\frac{\partial L}{\partial w_i}$。

4.2.2 Hessian矩阵

由于Hessian矩阵的计算量过大, 不直接计算, 而是通过BFGS中的方法对Hessian的

逆矩阵进行近似。令近似的Hessian逆矩阵记做H, 在BFGS中H的迭代公式为:

$$H_{k+1} = (I - \rho_k s_k d_k^T) H_k (I - \rho_k d_k s_k^T) + \rho_k s_k s_k^T \tag{1}$$

其中

$$\begin{aligned} s_k &= w_{k+1} - w_k \\ d_k &= \nabla f_W(W)^{(k+1)} - \nabla f_W(W)^{(k)} \\ \rho_k &= \frac{1}{s_k d_k^T} \end{aligned}$$

观察可知, 精确计算H时, 需要client传输自己的s, master则需要汇总并计算$\nabla_W f(W), s, d, \rho$,并求出最终值。

此处将精确求解H和g的过程归纳如下:

	Master	Client
步骤1	计算$\frac{\partial L_{(p)}}{\partial f}$, $\frac{\partial f^{(p)}}{\partial w_i}$	计算$\frac{\partial l_{(p)}}{\partial f}$, $\frac{\partial f^{(p)}}{\partial w_i}$并发送至Master
步骤2	Master聚合各方$\frac{\partial L^{(p)}}{\partial f}$得到$\frac{\partial L}{\partial f}$并发送给所有client	
步骤3	计算$\frac{\partial L}{\partial f}\frac{\partial f^{(p)}}{\partial w_i}$	计算$\frac{\partial L}{\partial f}\frac{\partial f^{(p)}}{\partial w_i}$ 并发送给Master
步骤4	Master聚合各方结果得到一阶导数$\frac{\partial L}{\partial w_i}$	
步骤5	在Master上计算H, 方法类似Alg.1	

Table 3: 精确求解H和g的过程

5 加入安全计算协议

在之前的章节, 我们介绍了本方法使用的损失函数和优化方法, 在本节我们将安全多方计算加入上述的算法的计算过程, 以保证在计算过程中的数据隐私性。此处基于半诚实模型, 即假设被攻陷方(currpted party)只是从协议中收集信息, 而没有偏离协议规范。出于安全性和公平性考虑, 我们采用去中心化的架构。这意味着秘钥生成和解密需要由所有参与方共同完成, 每个参与方的秘钥不能相同, 且解密时需要全体参与方共同参与才能正确解密。在训练和推理过程中,

我们采用半同态加密通过在密文状态下进行运算以保证隐私性和较高的计算效率。出于实际诸多场景的业务安全和运算速度的需求, 本方案可以保证原始数据以及运算过程中的一阶导数不被泄露。以下先介绍我们用到的两个基本组件: 1）秘密分享和2) 半同态加密。

5.1 基于秘密分享的秘钥生成

基于多项式差值的秘密分享(Shamir's Secret Sharing) 由Shamir等人提出[Sha79]。它将一个明文分解成多个部分, 在解密时通过多个部分结合才能正确恢复明文。由于本文采用了Paillier加密算法, 秘钥的生成基于两个大质数。在进行多方秘钥生成时, 需要在每个参与方的参与下生成两个大质数p和q并计算它们的乘积N, 每个参与方保有大质数的一部分, 但均不知道这两个大质数具体是多少。我们基于[BF01]提出的方法, 通过BGW [BOGW88] 和秘密分享实现以上的功能。秘钥生成过程大致可以分为三个阶段：1）每个参与方生成随机的p_i和q_i, 并通过BGW协议收集pi,qi并计算N;2）素数检测, 参与方检测生成的N是否为两个素数的乘积。如果是, 则进入下一个阶段, 否则丢弃生成的N, 从第一个阶段重新开始。3）得到N后, 参与方通过秘密分享生成秘钥并把秘钥的各个部分(shares)返回给各个参与方。同时生成公钥返回给所有参与方。注意到第一和第二个阶段一般会重复较多的次数, 但实验证明它的时间开销仍然在可接受的范围之内[MWB99]。

5.2 基于阈值解密的半同态加密

同态加密是实现安全多方计算的有效手段之一[YLCT19, ZPGS19]。它允许人们对密文进行特定形式的代数运算得到仍然是加密的结果, 将其解密所得到的结果与对明文进行同样的运算结果一样。换言之, 它令人们可以在加密的数据中进行诸如检索、比较等操作, 得出正确的结果, 而在整个处理过程中无需对数据进行解密。全同态加密可以实现加法和乘法在密文下的计算而速度往往较慢；半同态加密只能实现加法或者乘法运算, 但往往速度较快。联邦学习的诸多场景往往对时间有较高的要求, 因此, 我们采用了较为高效的Paillier [Pai99]进行密文计算。由于无法计算密文乘法, 对非线性函数诸如logistic function, 我们采用了类似[]的方法, 将非线性函数近似为分段的线性函数来求解。基于5.1所

述的方法, 我们利用秘密分享生成Paillier秘钥, 使每个参与方持有秘钥的一部分, 并用它来进行同态加密计算。在解密时, 通过所有参与方合作解密得到最终的值。

5.3 联邦安全训练

基于以上介绍的的多方秘钥生成5.1和阈值半同态加密5.2, 我们得以构造一个联邦学习的安全训练方法。出于在绝大多数业务中的实际情况, 我们假设参加联邦学习的各个参与方是满足半诚实假设的。即, 各个参与方一定会忠实执行密码学协议, 但会试图从密码学协议执行过程产生的中间结果中提取隐私数据。相比较而言, 对于更强的安全性假设–恶意模型(参与者可以完全不遵守密码学协议, 并会采取任何手段对密码学协议进行攻击从而提取隐私信息) 半诚实模型在时间效率上显著更高。值得注意的是, 本方法可以进一步通过引入加密认证和零知识证明来构造基于恶意模型的方法。

基于半诚实假设, 我们的安全训练需要达到的目标是：除了输入数据(各个参与方的数据和训练参数)和最后结果(各个参与方的模型参数), 各个参与方无法获得其余的任何信息。需要指出的是, 在本方案中, 出于时间性能考虑, master会知道近似的Hessian矩阵的逆矩阵H。由于Master整个训练过程中只能获得到H,除H以外的所有信息均对Master加密, Master无法获得任何具体的一阶导数值, 更无法获知原始数据, 训练参数等。这是能够满足绝大多数业务场景的安全需求的。

整个安全训练过程分为大致可分为三个步骤:1)为各个参与方生成公钥和私钥, 这一步已经在5.1中叙述, 此处不再赘述; 2）参与方用Paillier对数据进行加密, 并在加密的状态下计算梯度g。根据4, g的密文由master负责计算。Master得到g的密文结果后, 将对应特征的值发送给各个参与方。参与方随之进行解密5.2, 得到更新后的梯度g。得到g后, 为了进一步计算BFGS中的s和t, 见Equation.1, 各个参与方先计算自己的si和ti(注意到对于参与方i, 某个特征可能不存在, 则另其为0), 然后通过BGW [BOGW88] 中的方法生成shares发送到各方计算出H矩阵的share, 最后发送给Master进行解密, 得到近似的Hessian的逆矩阵H。之后master将H带入(1)式中求出w的密文并发送给各个client。3)每个参与方对w的密文进行解密得到更新后的w。

References

[BF01] Dan Boneh and Matthew Franklin. Efficient generation of shared rsa keys. *J. ACM*, 48(4):702–722, July 2001.

[BOGW88] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, STOC '88, page 1–10, New York, NY, USA, 1988. Association for Computing Machinery.

[FAT21] Webankfintech,2021. https://github.com/FederatedAI/FATE, 2021.

[MWB99] Michael Malkin, Thomas Wu, and Dan Boneh. Experimenting with shared generation of rsa keys. In *In Proceedings of the Internet Society's 1999 Symposium on Network and Distributed System Security (SNDSS*, pages 43–56, 1999.

[MZ17] Payman Mohassel and Yupeng Zhang. Secureml: A system for scalable privacy-preserving machine learning. Cryptology ePrint Archive, Report 2017/396, 2017. https://eprint.iacr.org/2017/396.

[NW06] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, New York, NY, USA, second edition, 2006.

[Pai99] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Proceedings of the 17th International Conference on Theory and Application of Cryptographic Techniques*, EUROCRYPT'99, page 223–238, Berlin, Heidelberg, 1999. Springer-Verlag.

[Sha79] Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, November 1979.

[YLCT19] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Trans. Intell. Syst. Technol.*, 10(2), January 2019.

[ZPGS19] Wenting Zheng, Raluca Ada Popa, Joseph E. Gonzalez, and Ion Stoica. Helen: Maliciously secure coopetitive learning for linear models. *CoRR*, abs/1907.07212, 2019.