(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 111581663 B
(45)授权公告日 2022.05.03

(21)申请号 202010360559.0

(22)申请日 2020.04.30

(65)同一申请的已公布的文献号
申请公布号 CN 111581663 A

(43)申请公布日 2020.08.25

(73)专利权人 电子科技大学
地址 611731 四川省成都市高新区(西区)西源大道2006号

(72)发明人 李洪伟 徐婕妤 徐国文

(74)专利代理机构 电子科技大学专利中心 51203
代理人 邹裕蓉

(51)Int.Cl.
G06F 21/62(2013.01)
G06N 3/04(2006.01)
G06N 3/08(2006.01)
H04L 9/08(2006.01)
H04L 9/40(2022.01)

(56)对比文件
CN 109684855 A,2019.04.26
CN 110719158 A,2020.01.21
CN 110443063 A,2019.11.12
CN 109543445 A,2019.03.29
CN 110874491 A,2020.03.10
CN 109359588 A,2019.02.19
CN 110796267 A,2020.02.14
US 2015049163 A1,2015.02.19
US 2019227980 A1,2019.07.25
EP 3477527 A1,2019.05.01
Guowen Xu.Data Security Issues in Deep Learning:Attacks, Countermeasures, and Opportunities.《IEEE Communications Magazine》.2019,第57卷(第11期),第116-122页.
贾延延等.联邦学习模型在涉密数据处理中的应用.《中国电子科学研究院学报》.2020,(第01期),
张铭凯等.多数据源下机器学习的隐私保护研究.《网络空间安全》.2020,(第04期), (续)

审查员 陈玲

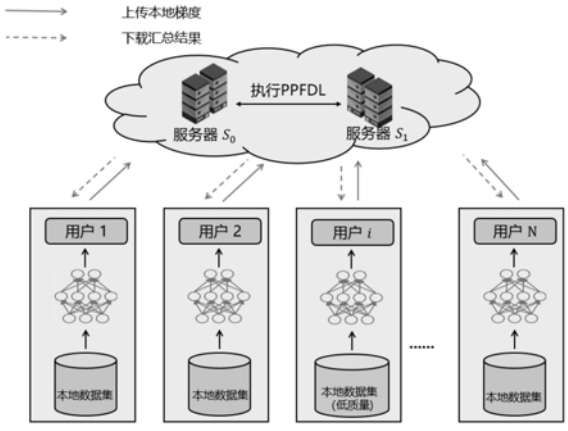
权利要求书3页 说明书5页 附图1页

(54)发明名称

面向非规则用户的保护隐私的联邦深度学习方法

(57)摘要

本发明提供一种面向非规则用户的保护隐私的联邦深度学习方法,包括步骤:1)系统设置步骤;2)迭代初始化步骤;3)更新加密的用户的可靠性;4)更新加密的汇总信息:服务器利用更新得到的加密的用户的可靠性信息为汇总结果的权重来更新各梯度的加密的汇总结果使得可靠性越低的用户对汇总结果的影响越小。本发明保护所有用户相关信息的隐私,并且减少用户在训练过程中因使用低质量数据的影响,同时确保用户相关信息的真实性。由服务器完成大部分计算,对于计算能力有限的终端用户来说非常友好,且对用户在整个训练过程中由于各种不可预知的原因而中途退出也具有鲁棒性。



CN 111581663 B

[转续页]

[接上页]

(56)对比文件

Meng Hao.Towards Efficient and Privacy-preserving Federated Deep Learning.《ICC 2019》.2019,全文.

刘峰.一种隐私保护关联规则挖掘的混合算法.《计算机应用研究》.2012,第29卷(第3期),第1107-1110页.

Wenbo Jiang.PTAS: Privacy-preserving Thin-client Authentication Scheme in Blockchain-based PKI.《Future Generation Computer Systems》.2019,全文.

Guowen Xu.Achieving efficient and privacy-preserving truth discovery in crowd sensing systems.《http://dx.doi.org/doi: 10.1016/j.cose.2016.11.014》.2017,第1-36页.

1.面向非规则用户的保护隐私的联邦深度学习方法,其特征在于,云服务器方执行以下步骤:

1)系统设置步骤:被指定的两台服务器S_0、S_1,服务器S_1保存有第三方为其生成一对非对称密钥(pk_1,sk_1),pk_1为公钥,sk_1为私钥;

服务器S_0用于接收用户发送的使用随机值与公钥pk_1进行加密处理后的梯度信息$(x_i^m-\sigma_i^m, Enc_{pk_1}(\sigma_i^m))$;其中,$x_i^m$为用户i第m次迭代的梯度,i为用户序号i=1,...,N,N为系统内用户总数,σ_i^m为用户i第m次迭代时生成的用户随机数,$Enc_{pk_1}(A)$表示使用公钥pk_1对A加密,m表示迭代次数序号,m∈[1,M],M表示完成一代训练epoch时所进行的迭代Iteration的总次数;服务器S_1用于接收用户发送的随机数σ_i^m;

2)迭代初始化步骤:服务器S_0初始化各次迭代的使用随机值与公钥pk_1进行加密处理后的汇总信息$(x_{*(0)}^m-\sigma_{*(0)}^m, Enc_{pk_1}(\sigma_{*(0)}^m))$,其中,$x_{*(0)}^m$为汇总结果$x_*^m$的初始值,$Enc_{pk_1}(\sigma_{*(0)}^m)$为加密后的第m次迭代中生成的汇总随机数$\sigma_*^m$的初始值;服务器$S_1$使用与服务器$S_0$相同的方式设置第m次迭代中生成的汇总随机数$\sigma_*^m$的初始值$\sigma_{*(0)}^m$;

3)更新加密的用户的可靠性信息:服务器S_0利用给定的加密处理后的梯度信息$(x_i^m-\sigma_i^m, Enc_{pk_1}(\sigma_i^m))$、加密处理后的汇总信息$(x_*^m-\sigma_*^m, Enc_{pk_1}(\sigma_*^m))$和服务器$S_1$给定的汇总随机数$\sigma_*^m$以及保存的私钥$sk_1$一起进行各用户i的可靠性更新:由服务器$S_0$生成加密的用户的可靠性信息$(T_i-\sigma_i^T, Enc_{pk_1}(\sigma_i^T))$,其中,$T_i$为用户i的可靠性,$T_i$与用户i本地梯度与汇总结果之间的距离呈负相关,$\sigma_i^T$为生成的用户i的可靠性随机数;服务器$S_1$获得用户i的可靠性随机数$\sigma_i^T$;

4)更新加密的汇总信息:服务器S_0利用更新得到的加密的用户的可靠性信息$(T_i-\sigma_i^T, Enc_{pk_1}(\sigma_i^T))$作为权重,以及服务器$S_1$获得每个用户的可靠性随机数$\sigma_i^T$和用户随机数$\sigma_i^m$来更新各梯度m的加密的汇总信息$(x_*^m-\sigma_*^m, Enc_{pk_1}(\sigma_*^m))$使得可靠性越低的用户对汇总结果的影响越小,可靠性越高的用户对汇总结果的影响越大,并将更新的各梯度m的加密的汇总信息发送至各用户。

2.如权利要求1所述方法,其特征在于,步骤2)具体包括以下步骤:

服务器S_0利用所有用户梯度与用户随机数之差的和的平均值初始化$x_{*(0)}^m-\sigma_{*(0)}^m$,$x_{*(0)}^m-\sigma_{*(0)}^m=\sum_{i=1}^{i=N}(x_i^m-\sigma_i^m)/N$;利用所有用户产生的随机数初始化加密后的汇总随机数$Enc_{pk_1}(\sigma_{*(0)}^m)=Enc_{pk_1}(\sum_{i=1}^{i=N}\sigma_i^m)^{-N}$;服务器$S_1$利用所有用户梯度之和的平均值初始化随机数$\sigma_{*(0)}^m=\sum_{i=1}^{i=N}\sigma_i^m/N$。

3.如权利要求1所述方法,其特征在于,步骤3)的具体方法是:

3-1)服务器S_0首先计算

$((x_i^m-\sigma_i^m)-(x_*^m-\sigma_*^m)=(x_i^m-\sigma_i^m)-(\sigma_i^m-\sigma_*^m)=\mu_i^m-(\sigma_i^m-\sigma_*^m)$,其中,$\mu_i^m$为用户i的第m个梯度与的与第m个汇总结果之间的距离;接着计算

$Enc_{pk_1}((\mu_i^m)^2-(\sigma_i^m-\sigma_*^m)^2)=(Enc_{pk_1}((\mu_i^m-(\sigma_i^m-\sigma_*^m))(\mu_i^m-(\sigma_i^m-\sigma_*^m)))Enc_{pk_1}((\sigma_i^m-\sigma_*^m))^{\mu_i^m-(\sigma_i^m-\sigma_*^m)}Enc_{pk_1}((\sigma_i^m-\sigma_*^m))^{\mu_i^m-(\sigma_i^m-\sigma_*^m)}$，通过同态性质计算$Enc_{pk_1}(\mu_i-\sum_{m=1}^{m=M}(\sigma_i^m-\sigma_*^m)^2)$，其中$\mu_i$，为用户i梯度与汇总结果之间的距离，$\mu_i=\sum_{m=1}^{m=M}(x_i^m-x_*^m)^2$；

3-2)服务器S_1计算$\sum_{m=1}^{m=M}(\sigma_i^m-\sigma_*^m)^2$，接着发送$Enc_{pk_1}(\sum_{m=1}^{m=M}(\sigma_i^m-\sigma_*^m)^2)$给服务器$S_0$；

3-3)服务器S_0计算

$Enc_{pk_1}(\mu_i-\sum_{m=1}^{m=M}(\sigma_i^m-\sigma_*^m)^2)\cdot Enc_{pk_1}(\sum_{m=1}^{m=M}(\sigma_i^m-\sigma_*^m)^2)=Enc_{pk_1}(\mu_i)$；

3-4)服务器S_0与服务器S_1共同执行SecDiv协议得到用户i的可靠性T_i，服务器S_0获得加密处理后的用户i的可靠性信息$(T_i-\sigma_i^T,Enc_{pk_1}(\sigma_i^T))$，服务器$S_1$获得的用户i的可靠性随机数$\sigma_i^T$，其中同执行SecDiv协议的具体方法为：

3-4-1)服务器S_0随机选取2个整数h_1，h_2并预设一个可靠性系数C，计算中间值x'_1和x'_2，$x_1'=Enc_{pk_1}(C)\cdot Enc_{pk_1}(h_1),x_2'=Enc_{pk_1}(\mu_i)\cdot Enc_{pk_1}(h_2)$，再将$x'_1$和$x'_2$的值发送给服务器$S_1$；

3-4-2)服务器S_1收到中间值x'_1和x'_2后，首先对接收到的中间值利用私钥sk_1进行解密得到$d_j=Dec_{sk_1}(x_j'),\ j=1,2$，接着随机选择整数$\sigma_i^T$，根据$d_1$，$d_2$，$\sigma_i^T$构造混乱电路GC，再使用公钥$pk_1$加密随机选择的整数$\sigma_i^T$得到$Enc_{pk_1}(\sigma_i^T)$，最后将GC和混淆的$d_1$、$d_2$、$\sigma_i^T$和$Enc_{pk_1}(\sigma_i^T)$发送到服务器$S_0$；

3-4-3)服务器S_0和S_1共同执行不经意传输协议OT协议使得服务器S_1来获得d_1、d_2的混淆值；

3-4-4)服务器S_0运行GC来得到$\frac{C}{\mu_i}-\sigma_i^T$，其中$T_i=\frac{C}{\mu_i}$；再计算得到加密后的用户i的可靠性$Enc_{pk_1}(T_i)=Enc_{pk_1}\left(\frac{C}{\mu_i}-\sigma_i^T\right)\cdot Enc_{pk_1}(\sigma_i^T)$，最后计算得到加密处理后的用户i的可靠性信息$(T_i-\sigma_i^T,Enc_{pk_1}(\sigma_i^T))$。

4.如权利要求1所述方法，其特征在于，步骤4)具体包括以下步骤：

4-1)服务器S_0计算 $Enc_{pk_1}(T_i\cdot x_i^m-\sigma_i^m\cdot\sigma_i^T)=(Enc_{pk_1}((T_i-\sigma_i^m)(x_i^m-\sigma_i^T))Enc_{pk_1}(\sigma_i^m)^{x_i^m-\sigma_i^T}Enc_{pk_1}(\sigma_i^T)^{T_i-\sigma_i^m}$，再计算$Enc_{pk_1}(\sum_{i=1}^{i=N}T_i\cdot x_i^m-\sigma_i^m\cdot\sigma_i^T)=\prod_{i=1}^{i=N}Enc_{pk_1}(T_i\cdot x_i^m-\sigma_i^m\cdot\sigma_i^T)$；

4-2)服务器S_1计算$\sum_{i=1}^{i=N}\sigma_i^m\cdot\sigma_i^T$后发送$Enc_{pk_1}(\sum_{i=1}^{i=N}\sigma_i^m\cdot\sigma_i^T)$给$S_0$；

4-3)服务器S_0计算：

$Enc_{pk_1}(\sum_{i=1}^{i=N}T_i)=Enc_{pk_1}(\sum_{i=1}^{i=N}(T_i-\sigma_i^T))\cdot\ Enc_{pk_1}(\sum_{i=1}^{i=N}\sigma_i^T)$；

$$Enc_{pk_1}\left(\sum_{i=1}^{i=N} T_i \cdot x_i^m\right) = Enc_{pk_1}\left(\sum_{i=1}^{i=N} T_i \cdot x_i^m - \sigma_i^m \cdot \sigma_i^T\right) \cdot Enc_{pk_1}\left(\sum_{i=1}^{i=N} \sigma_i^m \cdot \sigma_i^T\right)$$

；

4-4)服务器S_0和服务器S_1共同执行SecDiv协议，使得服务器S_0获得更新的梯度m对应的加密的汇总结果$(x_*^m - \sigma_*^m, Enc_{pk_1}(\sigma_*^m))$，服务器$S_1$获得更新的汇总随机数$\sigma_*^m$，其中同执行SecDiv协议的具体方法为：

4-4-1)服务器S_0随机选取2个整数h_3，h_4，并计算中间值x'_3和x'_4，$x'_3 = Enc_{pk_1}(\sum_{i=1}^{i=N} T_i x_i^m) \cdot Enc_{pk_1}(h_3), x'_4 = Enc_{pk_1}\left(\sum_{i=1}^{i=N} T_i\right) \cdot Enc_{pk_1}(h_4)$，然后将中间值$x'_3$和$x'_4$的值发送给服务器$S_1$；

4-4-2)服务器S_1收到中间值x'_3和x'_4后，首先对接收到的中间值利用私钥sk_1进行解密得到$d_j = Dec_{sk_1}(x'_j)$，j＝3，4；接着随机选择整数作为更新的汇总随机数σ_*^m，根据d_3，d_4，σ_*^m构造混乱电路GC，最后将GC和混淆的d_3、d_4、σ_*^m和$Enc_{pk_1}(\sigma_*^m)$的值发送到服务器S_0；

4-4-3)S_0和S_1共同执行OT协议来获得d_3、d_4的混淆值；

4-4-4)S_0运行GC来得到$\frac{\sum_{i=1}^{i=N} T_i x_i^m}{\sum_{i=1}^{i=N} T_i} - \sigma_*^m$，更新加密后的汇总结果

$Enc_{pk_1}(x_*^m) = Enc_{pk_1}(\frac{\sum_{i=1}^{i=M} T_i \cdot x_i^m}{\sum_{i=1}^{i=M} T_i} - \sigma_*^m) \cdot Enc_{pk_1}(\sigma_*^m)$，最后计算得到加密处理后的用户i的汇总信息$(x_*^m - \sigma_*^m, Enc_{pk_1}(\sigma_*^m))$并发送至用户i。

面向非规则用户的保护隐私的联邦深度学习方法

技术领域

[0001] 本发明具体涉及一种在云环境下面向非规则用户的保护隐私的联邦深度学习方法，属于信息安全技术领域。

技术背景

[0002] 传统的集中式深度学习通常需要一个数据处理中心(如云服务器)来收集大量的用户数据，并训练深度神经网络。深度学习作为人工智能的一个有前途的分支，已经渗透到电子健康的各个领域，如疾病预测、分析、预防和远程医疗等。此外，日益流行的网络链接设备提供了大量的电子健康记录，如可穿戴设备和智能手机等，在这些记录上进行深度学习模型可以显著提高医疗服务的质量。虽然将深度学习应用到电子健康有显而易见的好处，但收集大量的电子健康记录可能会泄露参与者隐私敏感信息，如疾病历史和医疗记录。为了解决这个问题，联邦学习被广泛应用于电子健康，因为它只通过在神经网络之间共享参数来支持神经网络优化，而不是共享用户的原始数据。用户只需要将本地完成一次训练iteration之后的训练模型(深度神经网络)梯度发送给云服务器，云服务器收集梯度计算汇总结果再下发至用户，用户根据接收到的汇总结果对本地的下一次的训练时使用的模型参数进行调整，直至完成训练。

[0003] 然而，最新的研究表明，对手(如云服务器)仍然可以利用共享的梯度和汇总结果恢复目标数据(如数据标签、成员关系等)。为了解决联合训练过程中的隐私泄露问题，人们提出了许多出色的解决方案，并将其应用于各种场景。

[0004] 在真实的场景中，每个用户所持有的原始数据的质量通常是不均匀的。拥有高级专业知识或终端设备的用户通常生成高质量的数据，而其他人可能持有低质量的数据。在本文中，我们将这些低质量数据的用户视为不规则用户。显然，在联合训练过程中，不规则用户共享的参数可能会影响训练的准确性，甚至导致最终模型的无用性。

[0005] 综上，目前的联邦学习机制存在以下不足:1)当模型的准确性是可接受的时，对手仍然可以很容易地恢复用户的敏感数据。2)要求成功完成私有训练任务的服务器是可信的，这与许多实际场景中不可信服务器设置的假设相矛盾。3)服务器可以访问每个用户的可靠性，即每个用户的“数据质量”信息(称为用户的可靠性)未进行保密。

发明内容

[0006] 本发明所要解决的技术问题是，提供一种考虑到不规则用户的可靠性并能保留数据隐私的联邦深度学习方法。

[0007] 本发明为解决上述技术问题所采用的技术方案是，面向非规则用户的保护隐私的联邦深度学习方法包括以下步骤：

[0008] 1)系统设置步骤：被指定的两台服务器S_0、S_1，服务器S_1保存有第三方为其生成一对非对称密钥(pk_1,sk_1)，pk_1为公钥，sk_1为私钥；

[0009] 服务器S_0用于接收用户发送的使用随机值与公钥pk_1进行加密处理后的梯度信息

$(x_i^m-\sigma_i^m, Enc_{pk_1}(\sigma_i^m))$;其中,$x_i^m$为用户i第m次迭代的梯度,i为用户序号i=1,…,N,N为系统内用户总数,σ_i^m为用户i第m次迭代时生成的用户随机数,$Enc_{pk_1}(A)$表示使用公钥pk_1对A加密,m表示迭代次数序号,m∈[1,M],M表示完成一代训练epoch时所进行的迭代Iteration的总次数;服务器S_1用于接收用户发送的随机数σ_i^m;

[0010] 2)迭代初始化步骤:服务器S_0初始化各次迭代的使用随机值与公钥pk_1进行加密处理后的汇总信息$(x_{*(0)}^m-\sigma_{*(0)}^m, Enc_{pk_1}(\sigma_{*(0)}^m))$,其中,$x_{*(0)}^m$为汇总结果$x_*^m$的初始值,$Enc_{pk_1}(\sigma_{*(0)}^m)$为加密后的第m次迭代中生成的汇总随机数$\sigma_*^m$的初始值;服务器$S_1$使用与服务器$S_0$相同的方式设置第m次迭代中生成的汇总随机数$\sigma_*^m$的初始值$\sigma_{*(0)}^m$;

[0011] 3)更新加密的用户的可靠性:服务器S_0利用给定的加密处理后的梯度信息$(x_i^m-\sigma_i^m, Enc_{pk_1}(\sigma_i^m))$、加密处理后的汇总信息$(x_*^m-\sigma_*^m, Enc_{pk_1}(\sigma_*^m))$和服务器$S_1$给定的汇总随机数$\sigma_*^m$以及保存的私钥$sk_1$一起进行各用户i的可靠性更新:由服务器$S_0$生成加密的用户的可靠性信息$(T_i-\sigma_i^T, Enc_{pk_1}(\sigma_i^T))$,其中,$T_i$为用户i的可靠性,$T_i$与用户i本地梯度与汇总结果之间的距离呈负相关,$\sigma_i^T$为生成的用户i的可靠性随机数;服务器$S_1$获得用户i的可靠性随机数$\sigma_i^T$;

[0012] 4)更新加密的汇总结果:服务器S_0利用更新得到的加密的用户的可靠性信息$(T_i-\sigma_i^T, Enc_{pk_1}(\sigma_i^T))$作为权重,以及服务器$S_1$获得每个用户的可靠性随机数$\sigma_i^T$和用户随机数$\sigma_i^m$来更新各梯度m的加密的汇总信息$(x_*^m-\sigma_*^m, Enc_{pk_1}(\sigma_*^m))$使得可靠性越低的用户对汇总结果的影响越小,可靠性越高的用户对汇总结果的影响越大,并将更新的各梯度m的加密的汇总信息发送至各用户。

[0013] 本发明的有益效果是:提出的不规则用户中保留隐私的联邦学习方法,保护所有用户相关信息的隐私,包括每个用户的梯度、可靠性以及汇总结果,在将本地梯度上传到云中之前,每个用户都使用附加的同态加密对其进行加密,并且减少用户在训练过程中因使用低质量数据的影响,同时确保用户相关信息的真实性。本发明方法的大部分计算都是由服务器完成的,对于计算能力有限的终端用户来说非常友好,且对用户在整个训练过程中由于各种不可预知的原因而中途退出也具有鲁棒性。

附图说明

[0014] 图1为联邦学习系统的示意图。

[0015] 如图1所示,在系统模型中包括服务器S_0、S_1和N个用户共同实现保护隐私的联合训练。所有参与的用户首先同意一个统一的深度神经网络DNN。然后,在每一代训练epoch的每一次训练iteration中,每个用户使用其本地数据集训练DNN,并计算训练样本对应的梯度。为了加快收敛速度和提高训练的准确性,每个用户都对其本地梯度加密并将其提交给云。接下来,两个没有勾结的云服务器S_0和S_1交互执行本发明参数传递方法以获取密文(梯度的汇总结果)并将汇总结果返回给所有用户。最后,每个用户对密文进行解密,并更新本地DNN

的参数。为了获得满意的网络结构，两个服务器和所有用户反复执行上述操作，直到DNN满足预定义的优化条件。

[0016] 本发明方法，包括以下步骤：

[0017] 1)系统设置步骤：被指定的两台服务器S_0、S_1，服务器S_1保存有第三方为其生成一对非对称密钥(pk_1,sk_1)，pk_1为公钥，sk_1为私钥；

[0018] 服务器S_0用于接收用户发送的使用随机值与公钥pk_1进行加密处理后的梯度信息$(x_i^m-\sigma_i^m, Enc_{pk_1}(\sigma_i^m))$；其中，$x_i^m$为用户i第m次迭代的梯度，i为用户序号i＝1，…，N，N为系统内用户总数，σ_i^m为用户i第m次迭代时生成的用户随机数，$Enc_{pk_1}(A)$表示使用公钥pk_1对A加密，m表示迭代次数序号，m∈[1,M]，M表示完成一代训练epoch时所进行的迭代Iteration的总次数；服务器S_1用于接收用户发送的随机数σ_i^m；

[0019] 2)迭代初始化步骤：服务器S_0初始化各次迭代的使用随机值与公钥pk_1进行加密处理后的汇总信息$(x_{*(0)}^m-\sigma_{*(0)}^m, Enc_{pk_1}(\sigma_{*(0)}^m))$，其中，$x_{*(0)}^m$为汇总结果$x_*^m$的初始值，$Enc_{pk_1}(\sigma_{*(0)}^m)$为加密后的第m次迭代中生成的汇总随机数$\sigma_*^m$的初始值；服务器$S_1$使用与服务器$S_0$相同的方式设置第m次迭代中生成的汇总随机数$\sigma_*^m$的初始值$\sigma_{*(0)}^m$；

[0020] 具体的，服务器S_0利用所有用户梯度与用户随机数之差的和的平均值初始化$x_{*(0)}^m-\sigma_{*(0)}^m$，$x_{*(0)}^m-\sigma_{*(0)}^m=\sum_{i=1}^{i=N}(x_i^m-\sigma_i^m)/N$；利用所有用户产生的随机数初始化加密后的汇总随机数$Enc_{pk_1}(\sigma_{*(0)}^m)=Enc_{pk_1}(\sum_{i=1}^{i=N}\sigma_i^m)^{-N}$；服务器$S_1$利用所有用户梯度之和的平均值初始化随机数$\sigma_{*(0)}^m=\sum_{i=1}^{i=N}\sigma_i^m/N$；

[0021] 3)更新加密的用户的可靠性：服务器S_0利用给定的加密处理后的梯度信息$(x_i^m-\sigma_i^m, Enc_{pk_1}(\sigma_i^m))$、加密处理后的汇总信息$(x_*^m-\sigma_*^m, Enc_{pk_1}(\sigma_*^m))$和服务器$S_1$给定的汇总随机数$\sigma_*^m$以及保存的私钥$sk_1$一起进行各用户i的可靠性更新：由服务器$S_0$生成加密的用户的可靠性信息$(T_i-\sigma_i^T, Enc_{pk_1}(\sigma_i^T))$，其中，$T_i$为用户i的可靠性，$T_i$与用户i本地梯度与汇总结果之间的距离呈负相关，$\sigma_i^T$为生成的用户i的可靠性随机数；服务器$S_1$获得用户i的可靠性随机数$\sigma_i^T$；

[0022] 具体的：

[0023] 3-1)服务器S_0首先计算$((x_i^m-\sigma_i^m)-(x_*^m-\sigma_*^m)=(x_i^m-\sigma_i^m)-(\sigma_i^m-\sigma_*^m)=\mu_i^m-(\sigma_i^m-\sigma_*^m)$，其中，$\mu_i^m$为用户i的第m个梯度与的与第m个汇总结果之间的距离；接着计算$Enc_{pk_1}((\mu_i^m)^2-(\sigma_i^m-\sigma_*^m)^2)=(Enc_{pk_1}((\mu_i^m-(\sigma_i^m-\sigma_*^m))(\mu_i^m-(\sigma_i^m-\sigma_*^m)))Enc_{pk_1}((\sigma_i^m-\sigma_*^m))^{\mu_i^m-(\sigma_i^m-\sigma_*^m)}Enc_{pk_1}((\sigma_i^m-\sigma_*^m))^{\mu_i^m-(\sigma_i^m-\sigma_*^m)}$，通过同态性质计算$Enc_{pk_1}(\mu_i-\sum_{m=1}^{m=M}(\sigma_i^m-\sigma_*^m)^2)$，其中$\mu_i$，为用户i梯度与汇总结果之间的距离，$\mu_i=\sum_{m=1}^{m=M}(x_i^m-x_*^m)^2$；

[0024] 3-2)服务器S_1计算$\sum_{m=1}^{m=M}(\sigma_i^m-\sigma_*^m)^2$,接着发送$Enc_{pk_1}(\sum_{m=1}^{m=M}(\sigma_i^m-\sigma_*^m)^2)$给服务器$S_0$;

[0025] 3-3)服务器S_0计算$Enc_{pk_1}(\mu_i-\sum_{m=1}^{m=M}(\sigma_i^m-\sigma_*^m)^2)\cdot Enc_{pk_1}(\sum_{m=1}^{m=M}(\sigma_i^m-\sigma_*^m)^2)=Enc_{pk_1}(\mu_i)$;

[0026] 3-4)服务器S_0与服务器S_1共同执行SecDiv协议得到用户i的可靠性T_i,服务器S_0获得加密处理后的用户i的可靠性信息$(T_i-\sigma_i^T,Enc_{pk_1}(\sigma_i^T))$,服务器$S_1$获得的用户i的可靠性随机数$\sigma_i^T$,其中同执行SecDiv协议的具体方法为:

[0027] 3-4-1)服务器S_0随机选取2个整数h_1,h_2并预设一个可靠性系数C,计算中间值x'_1和x'_2,$x_1'=Enc_{pk_1}(C)\cdot Enc_{pk_1}(h_1),x_2'=Enc_{pk_1}(\mu_i)\cdot Enc_{pk_1}(h_2)$,再将$x'_1$和$x'_2$的值发送给服务器$S_1$;

[0028] 3-4-2)服务器S_1收到中间值x'_1和x'_2后,首先对接收到的中间值利用私钥sk_1进行解密得到$d_j=Dec_{sk_1}(x_j'),\ j=1,2$,接着随机选择整数$\sigma_i^T$,根据$d_1$,$d_2$,$\sigma_i^T$构造混乱电路GC,再使用公钥$pk_1$加密随机选择的整数$\sigma_i^T$得到$Enc_{pk_1}(\sigma_i^T)$,最后将GC和混淆的$d_1$、$d_2$、$\sigma_i^T$和$Enc_{pk_1}(\sigma_i^T)$发送到服务器$S_0$;

[0029] 3-4-3)服务器S_0和S_1共同执行不经意传输协议OT协议使得服务器S_1来获得d_1、d_2的混淆值;

[0030] 3-4-4)服务器S_0运行GC来得到$\frac{C}{\mu_i}-\sigma_i^T$,其中$T_i=\frac{C}{\mu_i}$;再计算得到加密后的用户i的可靠性$Enc_{pk_1}(T_i)=Enc_{pk_1}\left(\frac{C}{\mu_i}-\sigma_i^T\right)\cdot Enc_{pk_1}(\sigma_i^T)$,计算得到加密处理后的用户i的可靠性信息$(T_i-\sigma_i^T,Enc_{pk_1}(\sigma_i^T))$;

[0031] 4)更新加密的汇总结果:服务器S_0利用更新得到的加密的用户的可靠性信息$(T_i-\sigma_i^T,Enc_{pk_1}(\sigma_i^T))$作为权重,以及服务器$S_1$获得每个用户的可靠性随机数$\sigma_i^T$和用户随机数$\sigma_i^m$来更新各梯度m的加密的汇总信息$(x_*^m-\sigma_*^m,Enc_{pk_1}(\sigma_*^m))$使得可靠性越低的用户对汇总结果的影响越小,可靠性越高的用户对汇总结果的影响越大,并将更新的各梯度m的加密的汇总信息发送至各用户;

[0032] 具体的:

[0033] 4-1)服务器S_0计算$Enc_{pk_1}(T_i\cdot x_i^m-\sigma_i^m\cdot\sigma_i^T)=(Enc_{pk_1}((T_i-\sigma_i^m)(x_i^m-\sigma_i^T))Enc_{pk_1}(\sigma_i^m)^{x_i^m-\sigma_i^T}Enc_{pk_1}(\sigma_i^T)^{T_i-\sigma_i^m}$,再计算$Enc_{pk_1}(\sum_{i=1}^{i=N}T_i\cdot x_i^m-\sigma_i^m\cdot\sigma_i^T)=\prod_{i=1}^{i=N}Enc_{pk_1}(T_i\cdot x_i^m-\sigma_i^m\cdot\sigma_i^T)$;

[0034] 4-2)服务器S_1计算$\sum_{i=1}^{i=N}\sigma_i^m\cdot\sigma_i^T$后发送$Enc_{pk_1}(\sum_{i=1}^{i=N}\sigma_i^m\cdot\sigma_i^T)$给$S_0$;

[0035] 4-3)服务器S_0计算:

[0036] $Enc_{pk_1}\left(\sum_{i=1}^{i=N} T_i\right) = Enc_{pk_1}\left(\sum_{i=1}^{i=N}(T_i - \sigma_i^T)\right) \cdot Enc_{pk_1}\left(\sum_{i=1}^{i=N} \sigma_i^T\right)$；

[0037] $$Enc_{pk_1}\left(\sum_{i=1}^{i=N} T_i \cdot x_i^m\right) = Enc_{pk_1}\left(\sum_{i=1}^{i=N} T_i \cdot x_i^m - \sigma_i^m \cdot \sigma_i^T\right) \cdot Enc_{pk_1}\left(\sum_{i=1}^{i=N} \sigma_i^m \cdot \sigma_i^T\right);$$

[0038] 4-4)服务器S_0和服务器S_1共同执行SecDiv协议，使得服务器S_0获得更新的梯度m对应的加密的汇总结果$(x_*^m - \sigma_*^m, Enc_{pk_1}(\sigma_*^m))$，服务器$S_1$获得更新的汇总随机数$\sigma_*^m$，其中同执行SecDiv协议的具体方法为：

[0039] 4-4-1)服务器S_0随机选取2个整数h_3，h_4，并计算中间值x'_3和x'_4，$x'_3 = Enc_{pk_1}(\sum_{i=1}^{i=N} T_i x_i^m) \cdot Enc_{pk_1}(h_3), x'_4 = Enc_{pk_1}\left(\sum_{i=1}^{i=N} T_i\right) \cdot Enc_{pk_1}(h_4)$，然后将中间值$x'_3$和$x'_4$的值发送给服务器$S_1$；

[0040] 4-4-2)服务器S_1收到中间值x'_3和x'_4后，首先对接收到的中间值利用私钥sk_1进行解密得到$d_j = Dec_{sk_1}(x'_j)$，$j = 3,4$；接着随机选择整数作为更新的汇总随机数σ_*^m，根据d_3，d_4，σ_*^m构造混乱电路GC，最后将GC和混淆的d_3、d_4、σ_*^m和$Enc_{pk_1}(\sigma_*^m)$的值发送到服务器S_0；

[0041] 4-4-3)S_0和S_1共同执行OT协议来获得d_3、d_4的混淆值；

[0042] 4-4-4)S_0运行GC来得到$\frac{\sum_{i=1}^{i=N} T_i x_i^m}{\sum_{i=1}^{i=N} T_i} - \sigma_*^m$，更新加密后的汇总结果$Enc_{pk_1}(x_*^m) = Enc_{pk_1}(\frac{\sum_{i=1}^{i=M} T_i \cdot x_i^m}{\sum_{i=1}^{i=M} T_i} - \sigma_*^m) \cdot Enc_{pk_1}(\sigma_*^m)$，由于$\frac{\sum_{i=1}^{i=N} T_i x_i^m}{\sum_{i=1}^{i=N} T_i} = x_*^m$，最后计算得到加密处理后的用户i的汇总信息$(x_*^m - \sigma_*^m, Enc_{pk_1}(\sigma_*^m))$并发送至用户i；

[0043] 5)用户利用接收到的服务器S_0发送的加密处理后的用户i的汇总信息对训练模型的参数进行调整后再次进行训练，得到当前的第m次迭代训练的训练模型的梯度x_i^m并生成对应此次迭代训练的用户随机数σ_i^m，对x_i^m与σ_i^m进行加密处理得到加密处理后的梯度信息$(x_i^m - \sigma_i^m, Enc_{pk_1}(\sigma_i^m))$并发送至$S_0$，同时将$\sigma_i^m$发送至$S_1$，重复步骤3)至步骤5)直至用户完成训练。

[0044] 整个过程中两个服务器分工不同，只有服务器S_1拥有私钥，只有服务器S_0有用户的可靠性以及更新后的加密的汇总结果，以防止服务器恶意解密用户数据。

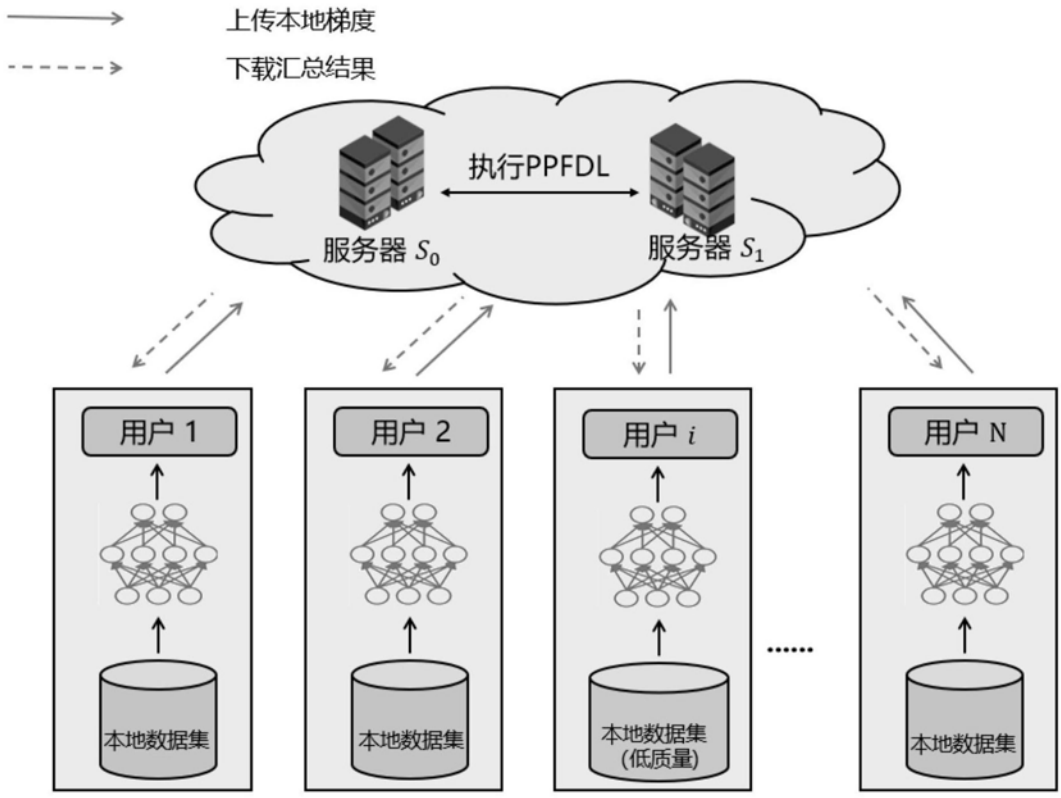

上传本地梯度
下载汇总结果
执行PPFDL
服务器 S_0
服务器 S_1
用户 1
用户 2
用户 i
用户 N
本地数据集
本地数据集
本地数据集
(低质量)
本地数据集

图1