知识联邦数据安全交换(FLEX)白皮书

同盾科技人工智能研究院

2020 年 10 月

目录

名词术语

1. 协调方(coordinator)：也称第三方、仲裁方(arbiter)，是一个中间方，主要完成联邦过程中的辅助计算，不存储数据。本文统一用C表示。
2. 参与方(party)：除第三方外，联邦过程中的其他参与者。参与方可能是数据提供者，也可能是模型(数据)使用者。在很多联邦中，某个参与方可能即是数据提供者，又是模型使用者。本文中某个参与方通常用P_i表示，双方联邦中通常用P_1(或A)和P_2(或B)表示两个参与方。
3. 发起方(guest)：也称模型(数据)使用者，通常是指在数据查询等任务中发起请求任务的一方，或者是指在建模任务中发起建模请求的一方，并在建模任务中会提供标签。联邦中通常用P_1(或A)表示发起方。
4. 服务方(host)：也称数据提供者，提供数据查询、计算或者建模服务的一个参与方。如果参与方既是数据提供者，又是模型使用者，通常认为该参与方是发起方，而不是服务方。双方联邦中通常用P_2(或B)表示服务方。
5. 特征(feature)：数据提供者提供用于训练或计算的特征指标，通常用X表示。
6. 标签(label)：模型使用者提供的用于训练模型的标签，用Y表示。
7. 用户 ID：可以是用户身份证号、手机号、姓名等具有唯一标示的信息，通常用u_{id}表示。
8. 模型(model)：指通过训练得到的机器学习模型，可能是深度模型，也可能是树模型或其它传统机器学习模型，模型参数常用θ表示。
9. 发送方(sender)：指信息交流中，输出信息的一方。
10. 接收方(receiver)：指信息交流中，接收信息的一方。

常用符号

1. A或P_1：发起方
2. B或P_2：服务方
3. C：协调方
4. P_i：第i个参与方
5. X或x：特征数据
6. Y或y：标签数据
7. θ：模型参数；
8. L或l：损失函数
9. T_l和T_r：左右子树
10. $(\cdot)^T$或$(\cdot)'$：矩阵转置
11. $[x]$：x的密文
12. Bl：布隆过滤器
13. Mt：映射表
14. S：集合
15. s：切分点
16. $f(\cdot)$和$f^{-1}(\cdot)$：映射函数和逆映射函数
17. $E(\cdot)$和$D(\cdot)$：加密函数和解密函数
18. ∇和g：梯度或一阶导数
19. h：二阶导数
20. pk和sk：公钥和私钥
21. $(\cdot)^{(i)}$：参与方P_i对应的数据或参数等

1.引言

近年来，联邦学习（federated learning）、安全多方计算（secure MPC）等领域成为学术界和工业界关注的重点，其主要目的是利用多个参与方数据进行安全计算或训练。无论是联邦学习还是安全多方计算都只是知识联邦[1]体系中的一种联邦功能，其相互关系和区别可以参阅同盾科技 AI 研究院在 2020 年 5 月发布的知识联邦白皮书[2]。

1.1 联邦生态

人工智能的发展需要大数据，而大量的数据实际上分散在不同的机构，要充分利用这些数据就需要让更多的机构参与到联邦中。业界对联邦普遍持观望的态度，这种观望其实蕴含了矛盾体的两面：一面是期待；一面是担忧。期待的是可以利用联邦生态中的多样性数据优化自身业务，同时也期待自身数据可以实现价值变现。担忧的是联邦生态是不是有足够大的规模、有多样性的数据，甚至联邦生态中的数据是否能保证安全合规。

联邦表面上看只是将不同的参与方连接起来，作为一个整体共同参与联邦应用，但实际上它并不是简单地连接和通信。除了连通之外，联邦还要保证在交互过程中不会泄漏参与方的数据隐私。所以，联邦的本质是多个参与方之间的数据安全交换。

目前为止还没有一种数据安全交换标准形成，能让各方确保数据交换过程的安全性是有保障的，进而愿意加入到联邦中。一旦参与机构（数据提供者）足够多，联邦规模足够大，数据多样性就有保障，也就会有更多机构（数据使用者）愿意来使用联邦服务，也会有更多科技型机构（模型和应用开发者）来提供丰富的算法、模型和应用。

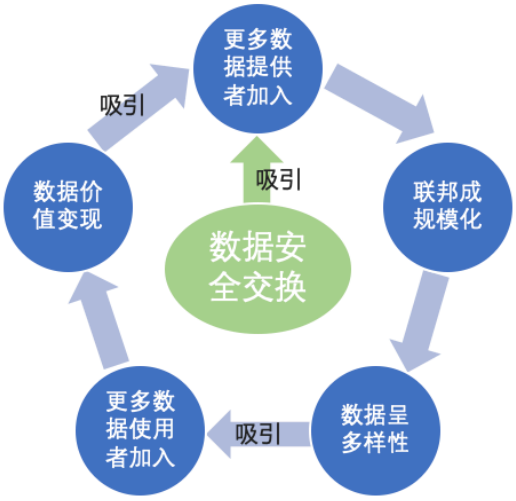


图 1.1 联邦生态构建

从图 1.1 中展现的联邦生态构建过程可知，数据安全交换是整个联邦生态的核心。为了保证联邦生态的顺利建设和良性发展，业界亟需一套统一的数据安全交换标准，支持各种联邦应用落地。

1.2 FLEX 协议

基于这种需要，同盾科技 AI 研究院设计并打造了一套标准化的联邦协议——联邦数据安全交换（**F**ederated **L**earning **EX**change，**FLEX**）协议。**FLEX 协议约定了联邦过程中参与方之间数据交换顺序，以及在交换前后采用的数据加解密方法**。只要参与各方能够遵守这些约定，就可以安全地加入到联邦中提供数据或使用联邦服务，无需担心数据隐私会有泄漏风险。

如图 1.2 所示，FLEX 协议实际上包括两层：

1) 应用协议：这一层协议是面向联邦算法的，为联邦算法提供多方数据交换的应用支撑。协议中会约定多方间数据交换的顺序和采用的具体密码算法。联邦过程中采用的通信协议也会被封装在这里。
2) 公共组件：是上层应用协议所依赖的基础密码算法和安全协议，比如同态加密、秘密分享等。

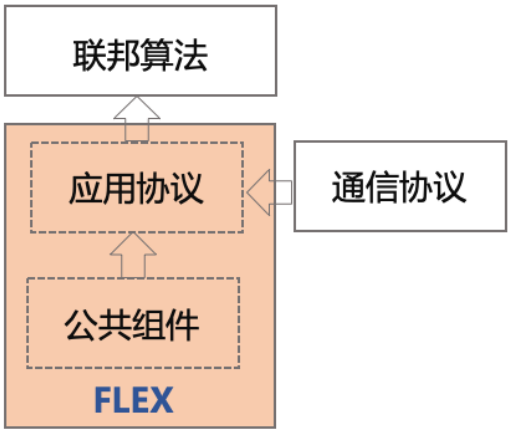
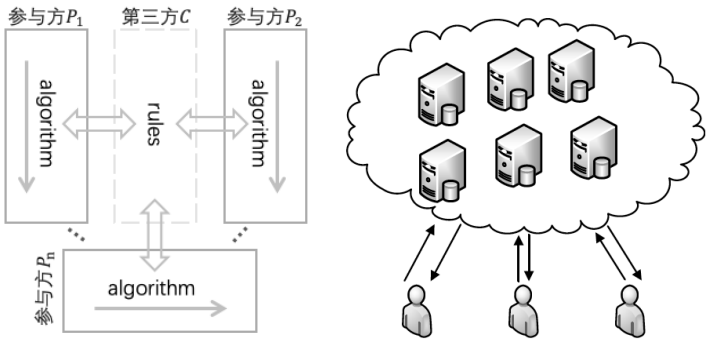


图 1.2 FLEX 协议总览

1.2.1 基础架构约定

联邦应用中一般要包含至少两个参与方和一个可信第三方。参与方提供参与联邦的数据，并主要负责数据的加密和解密工作；第三方负责执行一些约定的规则，并辅助参与方之间进行信息传输交换。此外，第三方还有一个更重要的作用就是可以为监管部门提供数据安全审计服务，这在金融类强监管行业中是很有必要的。



(a)约定架构 (b)虚拟第三方

图 1.2.1 基础架构

FLEX 协议中默认会有一个第三方，参与方都是直接与第三方连接，可以轻松加入联邦，约定架构基本如图 1.2.1(a)所示。这里第三方可以是一个可信的实体机构，也可以是一个虚拟服务器，一般采用公有云服务商提供的云服务器，如图 1.2.1(b)所示。在只有两方参与时，由于双方可以直接进行通信和计算，第三方的职能可以由某个参与方代替完成，因此可以不需要第三方。

1.2.2 通信约定

FLEX 协议不会约定具体采用哪种通信协议。原则上，目前支持联邦应用的通信协议都可以在 FLEX 中使用。在 FLEX 的代码实现中，会把通信模块调用封装进去，用户只需使用上层应用协议，无需关心具体通信过程。

1.2.3 安全约定

根据业务场景的不同，对数据隐私和安全的要求也各有不同。而联邦算法由于实现方式不同，可能面临的数据泄漏风险各不相同。FLEX 协议将会针对不同联邦算法分别提出安全要求，并在协议具体流程介绍完后进行相应的安全分析。

当然所谓“安全”也只是相对的，在模型训练阶段的安全，并不能保证模型预测使用时数据也是安全的。诚实的参与方也可能成为合谋者协作窃取数据，可信第三方是不是足够可信？这些尽管已经超出了 FLEX 协议中安全约定的范围，但却是联邦合作伙伴们关注的重点问题，本文也会在最后进行讨论分析。

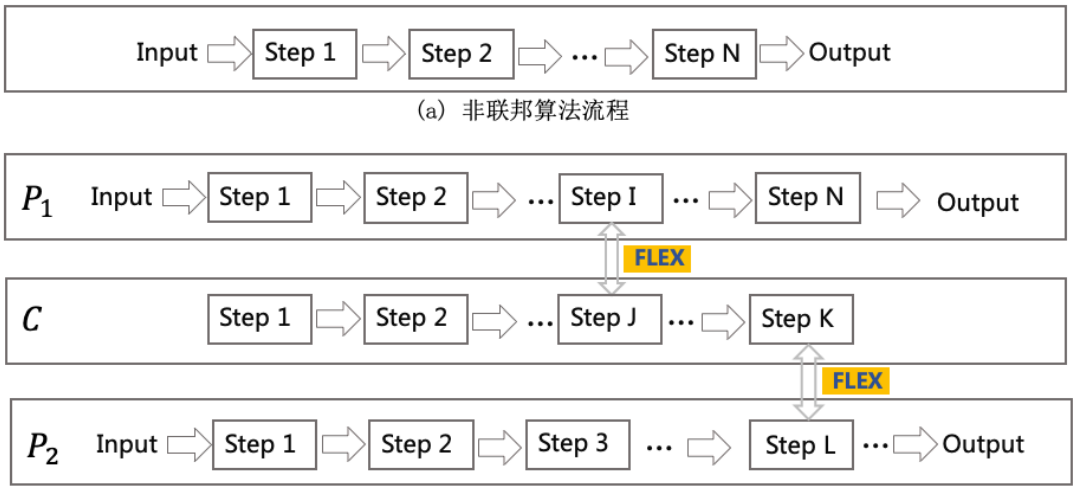


图 1.3 联邦算法与非联邦算法流程对比

1.3 联邦算法

联邦应用一般是通过联邦算法实现。联邦算法基本上是在一般的非联邦算法的基础上演化而来。非联邦算法只有一个参与方，算法步骤只是在本地执行，从输入数据和参数到执行步骤 1,2,…,N，最后输出结果，整个过程不会和外界其它方有交互，如图 1.3(a)所示。

联邦算法则因为联邦过程有多方参与，需要在多方之间进行数据交换，相对会更复杂。图 1.3(b)展示了由两个参与方（P_1和P_2）与第三方（C）一起执行某个联邦算法的过程。参与方P_1在与第三方进行数据交换前实际上执行的仍然是非联邦算法的步骤，只是在某些步骤（比如参数更新）中需要另一方P_2的中间结果时才发生联邦，执行 FLEX 协议，得到中间结果后将继续执行后续步骤直至算法结束。

FLEX 协议封装了数据安全交换的实现，并定义了与联邦算法相关步骤的接口，也就是参数的输入和输出。至于联邦算法中的其它非相关步骤，FLEX 是不关心的。而联邦算法在执行中只在需要多方数据安全交换时才调用 FLEX 协议，提供输入参数并接收输出结果，无需关心 FLEX 内部是如何加解密和通信的，更不需要关心其它参与方是谁、其它数据存储在哪里等问题。

1.4 跨特征与跨样本

参与方的数据分布会影响联邦算法的设计。不同的数据分布会需要不同的联邦算法，呈现出的安全问题也不尽相同。在实际应用中，经常遇到三种联邦需求：跨样本联邦、跨特征联邦和复合型联邦。跨样本联邦是指每个参与方的数据具有相同的特征分布，但各方的用户样本是独立的，而且每个参与方都有与自己样本对应的标签数据。跨特征联邦中参与方数据特征分布不同，但拥有共同的用户样

本，并且一般只有一方有标签数据。复合型联邦是跨样本和跨特征的组合，只有一小部分样本或特征集是参与各方的交集，其余数据无论是特征分布还是样本分布都不相同。复合型联邦一般需要足够规模的数据才会有效果，在联邦生态初期能发挥的作用有限。本文我们更关注的是跨特征和跨样本联邦。

在机构间合作中，跨特征联邦会更常见，毕竟各家机构间特征互补，才能对业务更有帮助。尤其是有些发起方在业务实践中积累了一些标签和少量特征，需要更多外部特征才能得到理想模型。此时不仅仅要保证特征数据的安全，还要防止标签数据的泄漏，毕竟标签数据是发起方的业务机密。而且由于模型需要用多方数据才能训练，模型预测时也同样需要多方数据才能完成，这也就意味着跨特征联邦在生产环境还需要联邦预测。

而跨样本联邦往往发生在同业或个体之间。尽管由于行业竞争的原因，很多大型机构不愿与同业进行联邦，但是很多中小型机构迫于生存压力期待加入联邦来提升行业竞争力。跨样本联邦中，由于用户特征数据和标签都是在同一个参与方内，可以直接在内部计算或训练，因此安全问题主要集中在模型汇集和更新中。此外，跨样本联邦的模型预测过程无需多方参与，因此不存在类似于跨特征联邦预测的安全问题。

FLEX 中提供的联邦协议都是与具体应用场景相关的。在每个协议流程前，我们会阐述该协议的适用场景，用户可以根据应用场景说明判断协议的适用性。

1.5 知识联邦重塑数据生产关系

图 1.5 新型数据生产关系

如图 1.5，未来的社会，数据是生产资料，人工智能是生产力，知识联邦是生产关系。目前，数据作为生产要素驱动了人工智能的发展，人工智能的突破是生产力的突破，它提供了一种提升效率的方法。而知识联邦则是一种新型生产关系，它能够改变我们使用数据的方式，实现数据等生产资料在时间和空间上的价值转换和交易，也能够影响并推动作为生产力的人工智能的再次突破，造就可信 AI 3.0。

联邦尤其适用于开展跨机构的数据资产协作，有助于促进不同主体之间的数据共享和优化业务流程。基于联邦技术，做大数据分析就不需要再收集获取数据，而是直接使用数据即可，数据所有权不会发生变化。数据的拥有者真正实现对数据的所有，最大化数据在多种场景下的多次价值实现，其它机构都是按照联邦协

议使用数据。与过去相比，数据资源的所有权变了，相当于生产关系中的一个重要要素也就改变了。

联邦提供了一种数据安全的分布式计算环境，使数据不用集中到一家机构，也能实现智能计算和分析，降低数据共享阻力。可以说，联邦的应用有望扭转当下数据日益集中化的趋势，避免中心化垄断，重新平衡各方利益。也会进一步推动数字经济向开放共享的方向发展，从而彻底重塑数据的“生产关系”，开创数字经济时代的新模式。

1.6 小结

简单地讲，联邦的本质就是多个参与方之间的数据安全交换。联邦数据安全交换（FLEX）协议约定了联邦过程中参与方之间数据交换顺序，以及在交换前后采用的数据加解密方法。其中包含一系列的约定，根据不同的功能，具体又可以分为联邦共享、联邦预处理、联邦计算、联邦训练、联邦预测、联邦推理、联邦决策等。只要遵守这些约定，参与方就可以安全地加入到联邦中，无需担心数据隐私会有泄漏风险，也满足数据合规要求[3]。

本文第 2～6 章定义了联邦过程中涉及的各种数据安全交换协议，第 7 章详细介绍了协议中依赖的基础密码算法和安全协议等公共组件，第 8 章对联邦安全性做了总结和展望。

2.联邦共享

联邦共享(federated sharing)是指多个参与方之间数据安全共享，这种数据的开放共享不是完全的开放，是有一定安全约束的开放。基于联邦共享，可以对外提供数据查询检索、样本对齐等服务。数据查询中查询请求方又会有保护查询 ID 不被其它方知道的需求，为了响应这种需求，这里提供了一种匿踪查询协议。在大规模样本数据对齐中，为了提升对齐效率，需要预先进行样本过滤。基于样本过滤协议，可以安全快速地滤除大量交集外样本，加速样本对齐的进程。

2.1 匿踪查询协议

匿踪查询（invisible inquiry）主要是解决查询过程中如何保护查询请求方用户 ID 信息不为其它参与方所知。匿踪查询主要是采用混淆扩充和不经意传输（OT）两种技术手段来隐匿查询方的用户 ID，让其它参与方无法轻易追踪到具体 ID。

应用场景：

应用于联邦共享或数据查询的场景中。假设参与方P_1为查询请求方，参与方P_2为提供查询服务的服务方。由P_1提供用户 ID 信息向P_2发起查询请求，P_2接收到请求后在本地数据库检索该 ID 信息，并将结果返回给P_1。

用户 ID 可以是用户身份证号、手机号、姓名等具有唯一标示的信息明文或密文。如是密文，则要求各参与方必须采用相同的脱敏加密方法对明文进行加工处理。

安全要求：

1) 查询请求方P_1只能获得查询 ID 对应的查询结果，不能获得其它额外信息；
2) 服务方P_2不能直接知道查询 ID，但允许以某个概率猜出查询 ID。

基本思想：

对P_1的查询 ID 进行混淆扩充，也就是根据 ID 生成规则随机生成$n-1$个 ID 作为混淆信息，将原来的一个查询扩充到n个查询。混淆 ID 与真实的查询 ID 一起，组成一个含有n个 ID 的集合S，将集合S发送至P_2。P_2对集合S中的n个 ID 分别在本地数据库进行检索，得到n个候选查询结果。

匿踪查询的第二个关键就是采用了不经意传输协议进结果传输。也就是P_2作为发送方，在得到候选查询结果后，需要执行$1-n$（n选 1）的 OT 协议把候选查询结果加密后发送给接收方。P_1作为接受方，也执行$1-n$的 OT 协议，将接收到的信息进行解密只得到真实查询 ID 对应的结果。

协议过程：

假设是双方参与的联邦场景。发起方P_1为查询请求方，服务方P_2提供查询服务。匿踪查询(OT-INV)协议的详细描述参见表 2.1。

表 2.1 匿踪查询协议流程

输入：

u_{id}：查询方P_1提供的待查询的用户 ID；

n：用于混淆扩充的 ID 数；

输出：

x：P_1查询得到的与u_{id}对应的结果；

具体步骤：

Step1：P_1提出查询请求，并按照对应 ID 生成规则，将待查询的 ID 随机扩充为n个，组成一个 ID 集合$S = \{u_{id_1}, u_{id_2}, \dots, u_{id_{n-1}}, u_{id}\}$发送给第三方$C$；

Step2：C将收到的所有 ID 转发给服务方P_2；

Step3：P_2对n个 ID 分别进行查询检索，得到n个候选查询结果$(x_1, x_2, \dots, x_n)$；

Step4：P_2运行$1-n$的 OT 协议，将n个候选结果全部发给C；

Step5：C将结果转发给P_1；

Step6：P_1运行对应的$1-n$的 OT 协议，解密出u_{id}对应的结果x。

匿踪查询协议对应的时序图如图 2.1 所示：

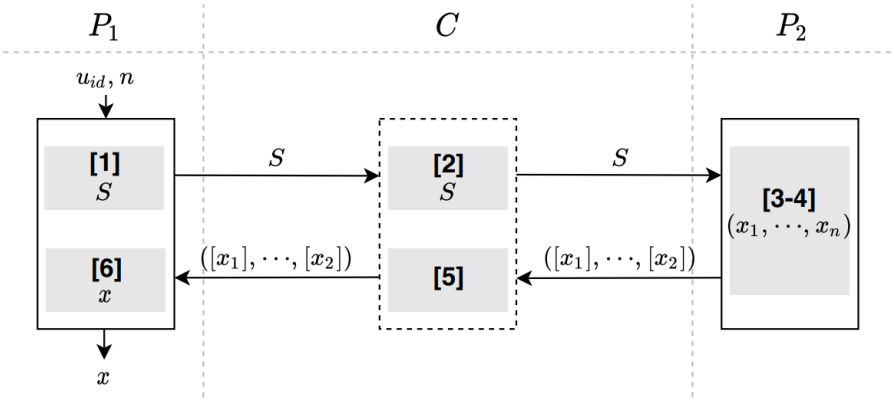


图 2.1 匿踪查询协议时序图

上述协议过程中，用到的 ID 生成规则是模拟的用户 ID 真实设计规则。比如手机号作为 ID，生成规则主要有：

1）总长 11 位；

2）前 3 位为运营商号段（可枚举：138、131 等）；

3）中间 4 位为区域号（可枚举，但比较大）；

4）最后 4 位为随机数。

常用的用户 ID 三要素包括姓名、身份证和手机号，不同的 ID 需要不同的生成规则。

$1-n$的 OT 协议也是匿踪查询协议中的关键一环，保证了查询方只能解密出真实查询 ID 对应的结果，具体介绍可以参考第 7.7 不经意传输节。

协议中第三方C在两方参与联邦查询时，主要起到信息中转的功能；在多方参与提供服务时，C除了转发所有的 ID 给各参与方之外，还要汇总各参与方的查询结果，生成整体结果反馈给查询方。

另外，查询方输出到C的 ID 可以是明文，也可以是加密脱敏后的密文，只要各参与方采用相同的加密脱敏规则即可。

安全分析：

理论上，匿踪查询协议只是降低了服务方获取查询方查询 ID 的概率，从原来100%降低为$1/n$，n是协议中随机扩充的 ID 数量。参与方P_1可以调整输入变量n的值来平衡查询的成本和安全性。n越大，安全性越高，但查询代价越高；n越小，安全性越低，查询代价也越低。

2.2 样本对齐

样本对齐(sample alignment)主要是基于用户 ID 寻找不同参与方之间的用户交集，即共同拥有的用户样本集合。样本对齐类似于传统联合建模中不同参与方之间撞库的操作，但是样本对齐协议对用户隐私保护有更严格的要求。

样本对齐可通过对称加密、非对称加密、OT 等技术实现。对于大规模的样本量（千万以上），则需要先进行样本过滤，快速滤除大部分交集外样本，生成候选样本集合（千万以内），然后再进行安全对齐，确定用户交集。

样本对齐主要用于跨特征联邦前的数据准备过程，以寻找各参与方之间相同的用户样本，用于后续的联邦计算或联邦训练。

2.2.1 样本过滤协议

样本过滤(sample filtering)的目的是在数据样本量大时，快速滤除大量交集外样本，以保证后续样本对齐可以快速完成。当一方数据量达到千万级以上时，通过样本过滤可以得到一个相对较小的候选样本集合，这个候选集合比真实交集“大的不多”，通常是在千万级以内。

应用场景：当至少有一方数据集中包含超过千万的用户样本时，需要先进行样本过滤。

安全要求：

1) 参与方不能知道其它参与方的用户 ID 数据；
2) 第三方不能知道各参与方的用户 ID，以及用户交集的规模。

基本思想：

样本过滤的核心是选用一个随机二值向量作为中间参考，让各方数据与该向量进行比较，进而间接完成相互之间的比较。这样就避免了各方数据直接进行比较，也就降低了泄漏的可能。此外，在比较前，对参与方数据进行了映射、重新排序操作，防止第三方从中间结果中反推参与方数据。

参与方首先将本地的用户 ID 全都映射到一个二值向量上，映射到的位置置 1 并记录映射关系。然后再将该向量按某种规则进行重新排序，同时生成一个长度相同的随机二值向量作为各方的中间参考。随后，分别在各方对随机向量和排序后的向量进行按位相等判断，并将结果发给第三方。第三方对所有的结果再统一执行按位相等判断，然后将判断结果反馈给各参与方，各方最后再将结果逆映射和溯源到原始用户 ID。

协议过程：

假设有n个参与方参与样本过滤，每个参与方P_i提供了一个用户 ID 集合S_i，通过该协议每个参与方可以得到一个新集合$\hat{S}_i$，新集合中已经部分滤除了不同的用户样本，只剩下候选用户样本。样本过滤(BF-SF)协议的具体流程见表 2.2.1。

表 2.2.1 样本过滤协议流程

输入：

S_i：参与方P_i用于样本过滤的用户 ID 集合；

输出：

$\hat{S}_i$：参与方P_i输出的候选子集；

具体步骤：

Step1：参与方分别对初始集合S_i计算一个长度为m的布隆过滤器Bl_i，同时保存Bl_i与S_i的映射表Mt_i；

Step2：参与方调用两次密钥交换协议，得到两个随机数α和β；

Step3：参与方分别使用哈希算法H计算密钥$key = H(\alpha)$，并以key为密钥，采用格式保留加密算法生成过滤器索引$[0, m-1]$到自身的一一映射，记该映射为f；

Step4：参与方利用映射F对布隆过滤器加密得到$V_i = f(Bl_i)$；

Step5：参与方以β为种子生成长度为m的伪随机比特串R；

Step6：参与方执行按位相等判断得到$\tilde{V}_i = (V_i == R)$，将$\tilde{V}_i$发送到第三方；

Step7：第三方对所有$\tilde{V}_i$执行按位相等判断，并生成向量$V_C = (\tilde{V}_1 == \tilde{V}_2 == \cdots == \tilde{V}_n)$，然后将$V_C$发送给各参与方；

Step8：参与方执行按位与运算得到$V_i^* = V_C \& V_i$，并将B_i^*逆映射生成样本过滤后的布隆过滤器$Bl_i^* = f^{-1}(V_i^*)$；

Step9：参与方利用映射表Mt_i将Bl_i^*中所有值为 1 的索引映射为用户 ID，得到候选子集$\hat{S}_i$。

注：当数据量过大时，Step1 和 Step9 需要占用的内存和计算资源较多，建议将这两步移到协议外执行，则协议的输入输出均为布隆过滤器。

上述协议中使用的布隆过滤器参见第 7.6 节，哈希算法H可采用 SHA256 或 SM3 算法，格式保留加密采用了第 7.5 节中介绍的 AES-FF1 算法，生成伪随机比特串采用了第 7.3 节中介绍的安全伪随机数生成算法。

样本过滤协议对应的时序图如图 2.2.1 所示：

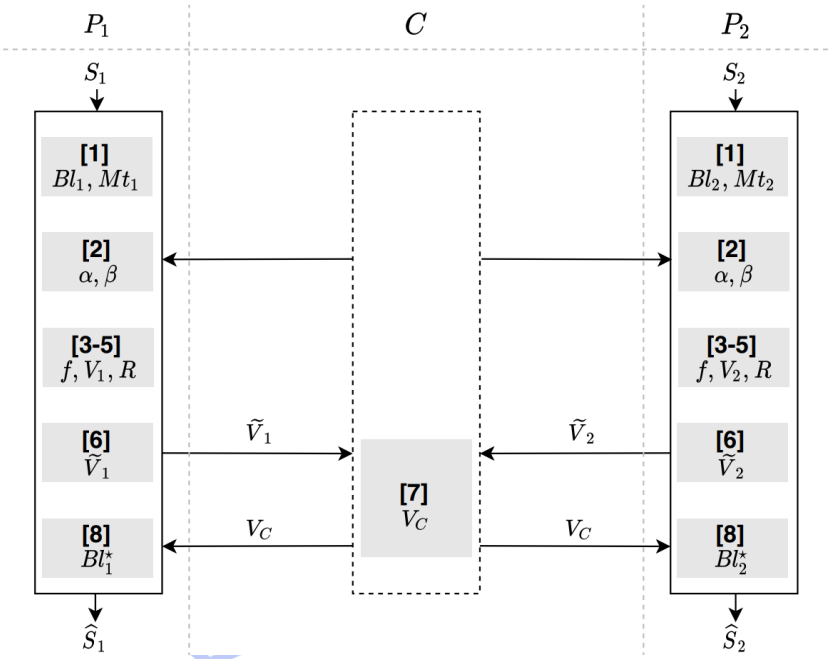


图 2.2.1 样本过滤协议时序图

安全分析：

- 该协议通过将用户 ID 映射到布隆过滤器，避免了用户 ID 直接参与比较判断；
- 格式保留加密方法打乱了布隆过滤器中二进制位的顺序，进一步降低了其它方获取用户 ID 的可能性；
- 通过伪随机比特串作为中间参考，并采用按位相等判断运算，可以保证作为中间方的第三方无法知道各方用户 ID 和数据规模。

2.2.2 安全对齐协议

安全对齐(secure alignment: SAL)是一种简单、高效的样本对齐协议，其目的是计算参与方之间的样本交集。该协议的计算量和传输量会随样本集的变大而增大，因此不适合较大规模的数据集。较大规模集合对齐前，要先执行样本过滤，再执行安全对齐。

应用场景：适用于参与方数据集规模在千万级以内的样本对齐。

安全要求：

1) 参与方不能知道其它方的用户 ID，也不能泄漏交集外的用户样本信息；
2) 第三方不能知道甚至追溯到参与方的用户 ID 数据。

基本思想：

安全对齐是将所有各方的用户信息上传至第三方，借助第三方辅助完成对齐的。为了避免信息会泄漏给第三方，上传信息必须是经过加密的，而且各方采用的加密方式必须是一致的。

协议过程：

假设有n个参与方参与对齐，每个参与方P_i提供了一个用户 ID 集合S_i。为了完成安全对齐，参与方之间必须要先共同协商出一个密钥，然后用该密钥将用户 ID 加密，加密后的密文再发送至第三方。这样第三方就可以在密文上对齐样本，并将对齐结果按顺序返还给各参与方。参与方则可以按对齐顺序提取本方样本子集，也就是样本交集。各方得到的样本交集内的样本排序也要是一致的。安全对齐(DH-SAL)协议的具体流程见表 2.2.2。

表 2.2.2 安全对齐协议流程

输入：
S_i：参与方P_i用于对齐的用户 ID 集合；
输出：
S^*：排序好的用户样本交集；
具体步骤：
Step1：各参与方先调用密钥交换协议，得到共同的随机数r，然后再分别计算密钥 $key = H_1(r)$；
Step2：参与方P_i对本方用户 ID 分别计算哈希值，然后再采用以key为密钥的对称加密算法进行加密$[u_{id}] = E\big(H_2(u_{id})\big)$，得到密文集合$U_i = [u_{id}], \forall u_{id} \in S_i$，最后将$U_i$发送到第三方；
Step3：第三方计算U_i的交集，并对交集内元素排序，同时生成交集元素在集合U_i中对应的索引顺序表I_i，最后将I_i返还给参与方；
Step4：参与方根据索引表I_i从S_i中提取交集S^*。

上述协议中使用的密钥交换协议参见第 7.2 节。H_1和H_2都是哈希算法，H_1可以采用 SHA256 或 SM3 算法，H_2可以采用 MD5 算法。对称加密E_k可以采用 AES 或 SM4 算法。在对齐过程中，各参与方都要采用同一种算法才能保证对齐顺利进行。

安全对齐协议对应的时序图如图 2.2.2 所示：

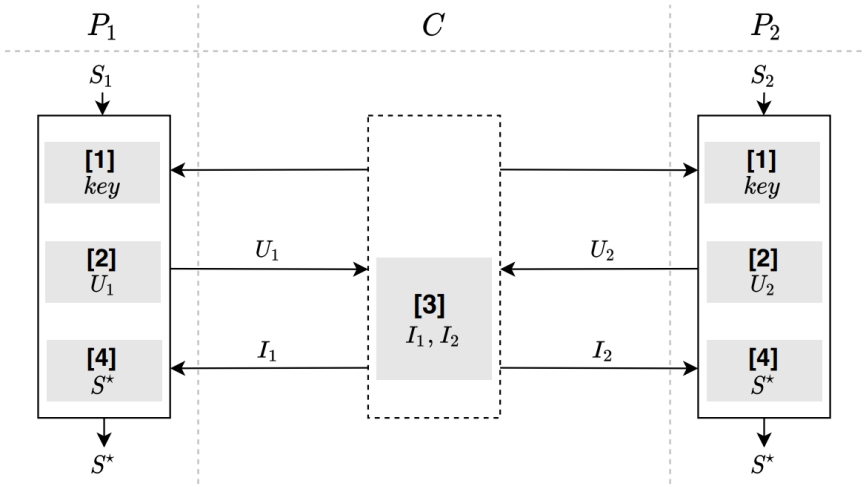


图 2.2.2 安全对齐协议时序图

安全分析：

- 在数据交换过程，各参与方都没有接触到其它方的用户 ID，因此参与方的用户 ID 是不会泄漏到其它参与方的。
- 第三方虽然能接触到各方的数据，但接触到的是加密后的密文数据，而且第三方不知道其密钥和加密算法，也就无法推断出用户 ID 信息，因此参与方的用户 ID 数据也不会泄漏给第三方，是安全的。

3.联邦预处理

联邦预处理(federated preprocessing)是指联邦环境下的数据预处理。这里不仅要完成数据预处理，还要考虑参与方的数据安全和隐私保护。数据预处理是指在主要的处理以前对数据进行的一些处理，比如分箱、特征选择等。有监督的预处理是一种常用的数据预处理方式，它需要使用标签信息。而在跨特征联邦中，只有一方有标签信息，其它方只有特征数据。这就需要联邦预处理的相关协议，基于这些协议可以安全地利用发起方的标签数据。本章介绍的协议主要是面向联邦分箱和联邦特征选择的。

安全要求：

1) 不能泄露参与方的特征数据；
2) 不能泄露发起方的标签数据。

3.1 联邦分箱

在风控建模等实际应用中，通常需要将连续特征离散化，也就是将连续的特征转换或划分为离散的特征。连续特征离散化常采用分箱方法，离散化后的特征，可以让模型有更高的鲁棒性。

联邦分箱(federated binning: FB)是指通过联邦的方式对参与方的特征进行离散化处理的过程。如果联邦分箱需要利用标签信息指导分箱过程，那么它就是一种有监督的联邦分箱。这种有监督的联邦分箱在跨特征联邦中应用较多，因为参与方只能通过联邦的方式才能利用发起方的标签信息进行分箱。但是为了保护隐私，标签和特征指标都不允许被其它参与方知道。

应用场景：适用于无标签的一方需要利用有标签一方的标签信息进行特征离散化处理。

分箱的方法有很多种，这里的联邦分箱采用的是一种基于决策树的方法。下面重点介绍一下基于决策树的联邦分箱协议(DT-FB)。

基本思想：

联邦分箱的核心就是要把标签信息通过同态加密的方式传递到没有标签的一方，这样没有标签的一方就可以利用标签进行分箱。DT-FB 协议使用基尼增益作为选取切分点的标准，因此就需要在候选切分点上统计划分信息。这里需统计的划分信息包括：特征值小于切分点的样本个数(n_l)、特征值大于等于切分点的样本个数(n_{nl})、特征值小于切分点的正样本的个数(p_l)、特征值大于等于切分点的正样本的个数(p_{nl})以及当前切分点所处的位置(pos)。由于划分信息可能会泄漏标签信息，因此划分信息只能以密文形式在参与方出现，这样就需要发起方进行解密和寻找最优切分点。

协议过程：

假设有两个参与方，P_1提供标签，P_2提供待分箱的特征数据，采用 HE-DT-FB 协议可以安全的得到最优切分点s^*和切分出的左右子树T_l和T_r。其具体流程见表 3.1。

表 3.1 HE-DT-FB 协议流程

输入：

Y：发起方P_1提供的标签；

$x^{(2)}$：参与方P_2提供的待分箱的特征数据；

输出：

s^*：分箱的最优切分点；

具体步骤：

Step1：参与方P_1生成一对同态公私钥(pk,sk)，并将标签Y进行同态加密生成标签密文$E_{pk}(Y)$，最后将密文和公钥pk一起发送给参与方P_2；

Step2：参与方P_2接收标签密文，对特征向量$x^{(2)}$使用等频分箱得到初始化的候选切分点$s(i)$，并用标签密文在密文空间上对每个候选切分点统计划分信息$[pr(i)] = (n_l, n_{nl}, [p_l], [p_{nl}], pos)$，最后将$[pr(i)]$发送给参与方$P_1$；

Step3：参与方P_1用私钥对$[pr(i)]$的密文解密得到$pr(i) = (n_l, n_{nl}, p_l, p_{nl}, pos) = D_{sk}([pr(i)])$，并计算基尼增益，选取增益最大的候选切分点作为最优切分点s^*，并将切分点s^*发送给参与方P_2；

Step4：参与方P_2记录当前切分点s^*，并根据切分点划分出左右子树T_l和T_r，并重复 Step2-Step4 直至满足终止条件。

HE-DT-FB 协议对应的时序图如图 3.1 所示：

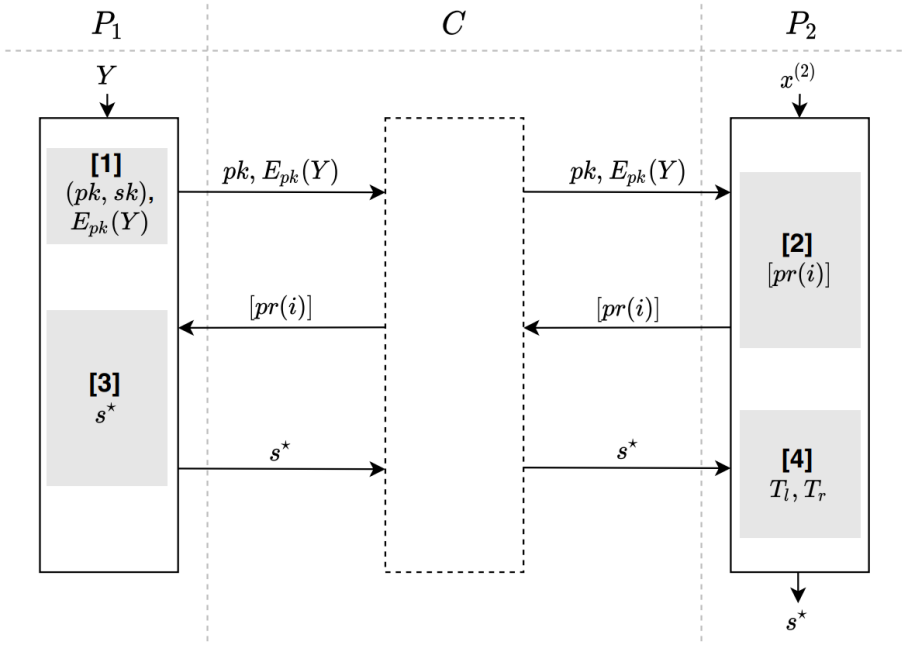


图 3.1 HE-DT-FB 协议时序图

安全分析：

参与方的特征数据是在本地计算，因此特征数据不会发生对外泄漏，是安全的。标签信息是以同态加密形式传输给参与方，参与方无法对标签进行解密，而且统计的划分信息也无法解密，第三方和参与方也就无法反向推导出标签信息，因此发起方的标签信息也不会发生泄漏。

3.2 联邦特征选择

特征选择一般作为模型训练的前置步骤，通过算法筛选掉冗余或建模效果差的特征，从而提高建模效果以及加快训练速度。联邦特征选择(federated feature selection: FFS)是指对所有参与方数据的特征进行联合协同筛选，去除参与方之间重复冗余的特征，只保留更重要的特征指标，以提高训练数据质量并缩短训练时间。联邦特征选择过程需要使用各参与方的特征数据和发起方的标签信息。

由于特征选择的方法很多，不同的算法对应的数据交换过程不一样，因此也就有多种联邦特征选择协议。下面介绍的两种协议，一种是基于逐步回归的思想进行特征选择，其核心是计算联邦矩阵，另一个是基于信息价值进行特征选择，其核心是计算信息价值。

3.2.1 联邦矩阵计算协议

联邦矩阵计算(federated matrix computing: FMC)协议是面向逐步回归特征选择算法的，其算法核心是利用协方差矩阵计算一个统计量，逐步去除统计量最小的特征，直到统计量最小值满足预设要求，特征选择才算完成。这里的模型参数θ是通过最小二乘法来估算的：

$$\theta = (X^T X)^{-1} X^T Y。$$

其中，特征数据矩阵X是由参与方特征数据联合组成，Y是发起方提供的标签。显然，上式计算的关键是如何计算协方差矩阵$X^T X$和$X^T Y$。由于其它参与方的特征数据不允许直接汇聚到一起，而发起方的标签数据也不能直接发送给其它参与方，这就给直接计算协方差矩阵带来了困难。

下面以两方跨特征联邦为例进一步分析如何计算$X^T X$和$X^T Y$。此时，参与方的用户样本一致，特征指标不同。假设，发起方P_1提供部分特征数据X_1和标签Y，P_2提供部分特征数据X_2，因此，上述特征数据矩阵X是由双方特征拼接而成，$X = [X_1, X_2]$。对应的$X^T X$和$X^T Y$就可以写成：

$$X^T X = \begin{bmatrix} X_1^T \\ X_2^T \end{bmatrix} [X_1, X_2] = \begin{bmatrix} X_1^T X_1 & X_1^T X_2 \\ X_2^T X_1 & X_2^T X_2 \end{bmatrix},$$

$$X^T Y = \begin{bmatrix} {X_1}^T \\ {X_2}^T \end{bmatrix} Y = \begin{bmatrix} {X_1}^T Y \\ {X_2}^T Y \end{bmatrix}。$$

显然，每个参与方可以利用本地数据集X_i直接计算$X_i^T X_i$，发起方也可以直接计算$X_1{}^T Y$。而剩余部分$X_1^T X_2$和$X_2{}^T Y$则需要参与方直接协作运算，这里称作联邦矩阵计算，对应的$X_1^T X_2$和$X_2{}^T Y$被称作联邦矩阵。由于X_1和X_2分布在不同的参与方，在无法直接共享数据的前提下，安全地计算联邦矩阵$X_1^T X_2$和$X_2{}^T Y$则成为了重中之重。

应用场景：适用于跨特征联邦中的联邦矩阵计算，常用于特征选择。

基本思想：

本协议利用矩阵秘密分享的思想，首先将各参与方的特征数据矩阵分别拆分成矩阵碎片，拆分要求是矩阵碎片之和等于特征数据矩阵。然后参与方利用矩阵碎片信息生成秘密碎片用于共享和传播。最后在发起方这边根据获取的秘密碎片，计算联邦矩阵$X_1^T X_2$和$X_2{}^T Y$。

协议过程：

假定有两个参与方提供了n个样本，发起方P_1提供部分特征数据$X_1 \in R^{n\times m_1}$和标签$Y \in R^{n\times 1}$，服务方P_2提供部分特征数据$X_2 \in R^{n\times m_2}$，要计算联邦矩阵$X_1^T X_2$和$X_2{}^T Y$。FMC 协议具体流程见表 3.2.1。

表 3.2.1 FMC 协议流程

输入：

$X_i \in R^{n\times m_i}$：参与方P_i提供的特征数据；

输出：

$X_1^T X_2$：联邦矩阵；

具体步骤：

Step1：发起方P_1将参数m_1和n发送给第三方C；

Step2：服务方P_2将参数m_2发送给第三方C；

Step3：第三方基于矩阵秘密分享协议和参数m_1, m_2和n分别生成 4 个随机矩阵：R_1，U_1，R_2和U_2，并将R_1和U_1发给P_1，将R_2和U_2发给P_2；

Step4：发起方P_1根据接收到的随机矩阵生成碎片$\tilde{X}_1$，发送给服务方P_2；

Step5：服务方P_2根据接收到的随机矩阵生成碎片$\tilde{X}_2$，并生成随机矩阵V_2，同时计算$Z = (\tilde{X}_1)^T X_2 + U_2 - V_2$，并把$\tilde{X}_2$，$V_2$和$Z$分别发送给发起方$P_1$；

Step6：发起方P_1计算$V_1 = Z + U_1 - R_1{}^T \tilde{X}_2$，并更新中间结果$X_1{}^T X_2 = V_1 + V_2$。

上述协议中的矩阵R_1和$\tilde{X}_1 \in R^{n\times m_1}$，$R_2$和$\tilde{X}_2 \in R^{n\times m_2}$，$U_1, U_2, V_1$和$V_2 \in R^{m_1\times m_2}$。用到的矩阵秘密分享协议细节可以参见第 7.8 节。

尽管协议中只给出了$X_1{}^T X_2$的计算方式，事实上$X_2{}^T Y$也可以通过上述协议计算。该协议是一个双方的联邦矩阵计算协议，在多方参与的场景下也是以此协议为基础，让发起方分别与其它方进行联邦特征选择，即可实现多方间的联邦特征选择。

FMC 协议对应的时序图如图 3.2.1 所示：

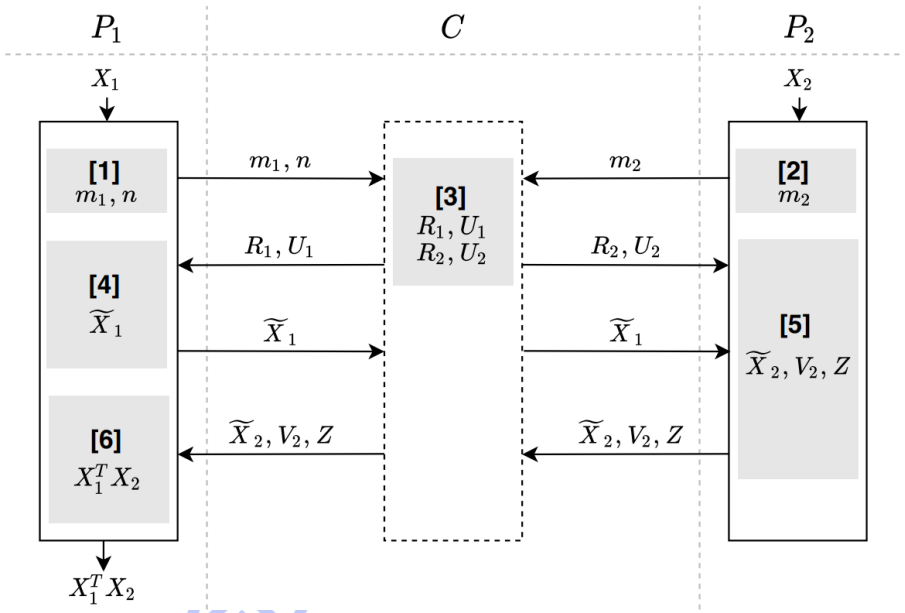


图 3.2.1 FMC 协议时序图

安全分析：

- 在数据交换环节，参与方的特征数据都是通过矩阵碎片以秘密的形式传递，特征数据没有直接参与传递，第三方或者参与方方也无法进行反向求解，因此特征数据不会发生对外泄漏，是安全的。
- 发起方的标签信息也只是作为矩阵一列参与计算，数据交换过程中同上面一样，因此也不会发生标签信息泄漏。

3.2.2 IV-FFS 协议

根据信息价值(information value: IV)的定义可知，IV 的计算是以证据权重(WOE)为基础的。分别定义p_{y_i}是第i个分箱中正样本占所有正样本的比例，p_{n_i}是第i个分箱中负样本占所有负样本的比例。那么第i个分箱的证据权重为$woe_{\mathrm{i}} = \ln(p_{y_i}/p_{n_i})$，相应的 IV 值是：$iv_{\mathrm{i}} = (p_{y_i} - p_{n_i}) * woe_{\mathrm{i}}$。整个特征变量的信息价值是各分箱信息价值的和：$iv = \sum_i iv_{\mathrm{i}}$。显然这里 IV 值计算的核心就是统计分箱中的正负样本数量。

应用场景：适用于跨特征联邦场景下的信息价值的计算，常用于特征选择。

基本思想：

IV-FFS 主要是通过同态加密将发起方的标签信息加密后传给服务方进行分箱信息的统计，也就是每个分箱中正负样本的数量，统计结果再以密文形式发给发起方，让发起方进行汇总计算证据权重和 IV 值，最后再将 IV 值返还给服务方。

协议过程：

假定有两个参与方提供了n个样本，发起方P_1提供标签$Y \in R^{n\times1}$，服务方P_2提供分箱后的特征数据$x^{(2)} \in R^{n\times1}$，分箱后特征的切分点信息$V \in R^{l\times1}$，要计算该特征的信息价值。该协议具体流程见表 3.2.2。

表 3.2.2 IV-FFS 协议流程

输入：

$Y \in R^{n\times1}$：发起方P_1提供的标签；

$x^{(2)} \in R^{n\times1}$：服务方P_2提供的分箱后的特征数据；

$V \in R^{l\times1}$：分箱后特征数据$x^{(2)}$的切分点信息；

输出：

iv：服务方P_2收到该特征对应的信息价值；

具体步骤：

Step1：发起方P_1生成一对同态公私钥(pk,sk)，并将标签Y进行同态加密生成标签密文$E_{pk}(Y)$，然后将密文发送给服务方P_2；

Step2：服务方P_2在密文空间上统计分箱中正负样本数，并将密文$[n_p]$和$[n_n]$发送给P_1；

Step3：P_1用私钥sk解密每个分箱的统计信息：$D_{sk}([n_p])$和$D_{sk}([n_n])$，并计算iv值发给P_2；

Step4：P_2接收相应的iv值。

该协议对应的时序图如图 3.2.2 所示：

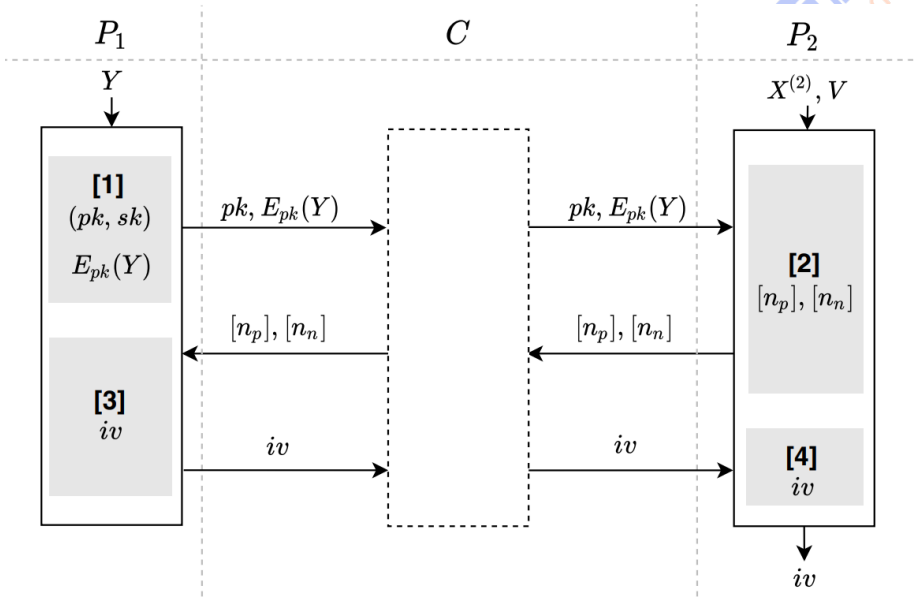


图 3.2.2 IV-FFS 协议时序图

安全分析：

- 服务方的特征向量是在本地统计，因此特征数据不会发生对外泄漏，是安全的。

- 标签信息是以同态加密形式传输给服务方，服务方无法对标签进行解密，而且也不能解密统计结果，因此数据交换过程中第三方和服务方无法反向推导出标签信息，发起方的标签信息也不会发生泄漏。

4.联邦计算

联邦计算(federated computing)是指利用多个参与方的数据按照某个规则进行计算。计算规则是预先设定好的，与场景强相关。联邦计算的难点是各方数据在计算过程中不能离开本地，并且不能泄漏出去。

下面针对金融场景中经常遇到的多头共债问题，介绍两种多头共债的联邦计算协议。

4.1 多头共债

所谓多头共债(multi-loan)是指某个用户与多家金融机构发生借贷关系，一旦用户资金发生问题，会产生很大的金融风险。解决多头问题最有效的方法就是直接评估该用户在多家机构里的累积借贷是否超出其授信指标或实际收入水平。然而，由于合规要求，各家机构的数据不能直接汇集到一起进行计算，因此需要一种新的方式来解决这个问题，就是本节介绍的多头共债协议。

简单地讲，多头共债协议就是利用多家参与机构构的数据来联合计算和评估某用户的借贷风险，同时要保证该用户在各机构的借贷数据不会离开本地。

应用场景：适用于金融借贷中进行多头共债风险控制。评估用户借贷风险的核心是计算用户的已经发生的贷款总额是否超出其偿还能力。偿还能力这里假定是已知的，通常是其固定收入相关。

安全要求：

1) 不能泄露用户在服务方的贷款额；
2) 不能泄露用户在服务方的贷款之和。

4.1.1 HE-ML 协议

基本思想：

该协议是基于同态加密进行多头风险计算，计算过程中需要对每个参与方的数据进行加密后再上传至第三方，第三方对密文数据进行汇总。而第三方在汇总时不能进行解密。

协议过程：

假设有n个参与方作为服务方P_i提供用户贷款额，由第三方向发起方提供查询服务。协议中第三方会向服务方查询对应用户 ID 的贷款额，并进行汇总和风险计算，只是所有的运算是在密文空间上进行的，最后将结果反馈给发起方解密。

该协议的具体流程如表 4.1.1 所示：

表 4.1.1 HE-ML 协议流程

输入：

u_{id}和r：分别是发起方提供的用户 ID 和该用户的偿还能力；

L_i：该用户在服务方P_i的贷款额；

输出：

$Risk$：0 或 1，发起方得到的风险结果；

具体步骤：

Step1：发起方P_1生成一对同态公私钥(pk,sk)，并将r进行同态加密生成密文$[r]$，最后将$[r]$，pk和u_{id}一起发送给第三方；

Step2：第三方将pk和u_{id}发放给服务方P_i；

Step3：服务方P_i查询该用户在本方贷款额L_i，并将其加密成密文$[L_i]$；然后将$[L_i]$发送给第三方；

Step4：第三方在密文空间上计算所有贷款额总和与偿还能力的差，得到密文结果$[T]$，并将结果发送给发起方；

Step5：发起方将结果$[T]$解密后判断T是否大于 0，得到最后结果：如果$T \geq 0$，$Risk$为 1；否则$Risk$为0。

协议对应的时序图如 4.1.1 所示：

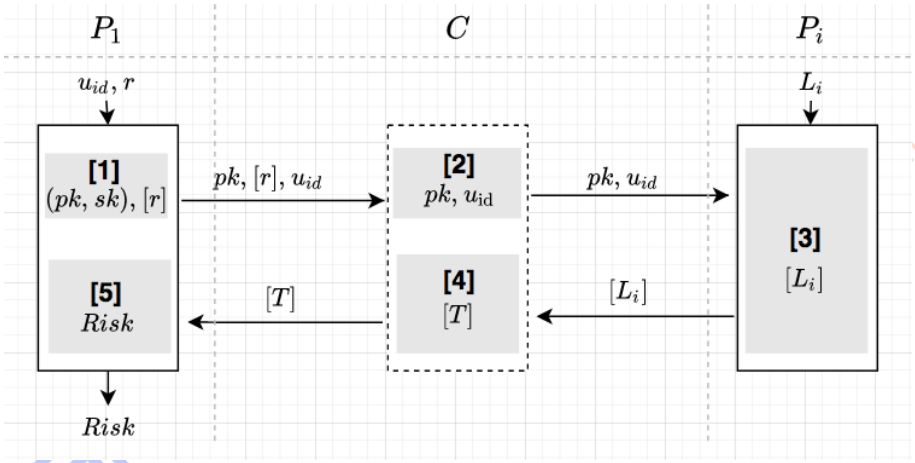


图 4.1.1 HE-ML 协议时序图

安全分析：

发起方提供的偿还能力信息和服务方的贷款信息都是以密文形式发到第三方，第三方只能进行计算，无法解密，因此用户的借贷信息和偿还能力都不会发生泄漏，是安全的。

4.1.2 SS-ML 协议

基本思想：

利用秘密分享的原理将用户的偿还能力和贷款额拆分成碎片，并在服务方之间分享和计算。第三方对计算结果进行汇总统计，并将结果反馈给发起方。

协议过程：

该协议中，同样是由发起方向第三方发起查询服务，并向第三方提供用户 ID 和其偿还能力，而第三方则根据用户 ID 分别向服务方发起查询和判断。

协议过程中，第三方会首先根据服务方数量将偿还能力（金额）拆分成碎片，并将偿还能力秘密碎片分享给所有服务方；然后服务方会将贷款额拆分成多个碎片，并将贷款额碎片分享给其它服务方；同时服务方会接收发送过来的贷款额和偿还能力碎片，对接收到的碎片进行计算，并将计算结果上传至第三方；随后第三方根据汇集的结果进行统计，得到查询判断结果，并将结果反馈给发起方。该协议的具体流程如表 4.1.2 所示。

表 4.1.2 SS-ML 协议流程

输入：

u_{id}和r：分别是发起方提供的用户 ID 和该用户的偿还能力；

L_i：该用户在服务方P_i的贷款额；

输出：

$Risk$：0 或 1，发起方得到的风险结果；

具体步骤：

Step1：发起方根据秘密分享的原理将r拆分成n个碎片r_i，与u_{id}分别发送给对应的服务方P_i；

Step2：服务方P_i查询该用户在本方贷款额L_i，并将L_i拆分成n个碎片$L_{i,j}$，$j = 1, \dots, n$。将碎片分享到其它服务方，第j个服务方得到$L_{i,j}$；

Step3：服务方P_i计算其接收到的所有贷款额碎片之和$L_i' = \sum_j L_{i,j}$，然后用L_i'减去其收到的偿还能力碎片得到：$L_i^* = L_i' - r_i$，并将L_i^*发送至第三方

Step4：第三方生成判断结果$Risk$，并满足如下条件：如果$\sum_{i=1}^{n} L_i^* \geq 0$，$Risk$值为 1；否则$Risk$值为0；

Step5：第三方将结果$Risk$反馈给发起方。

协议对应的时序图如图 4.1.2 所示：

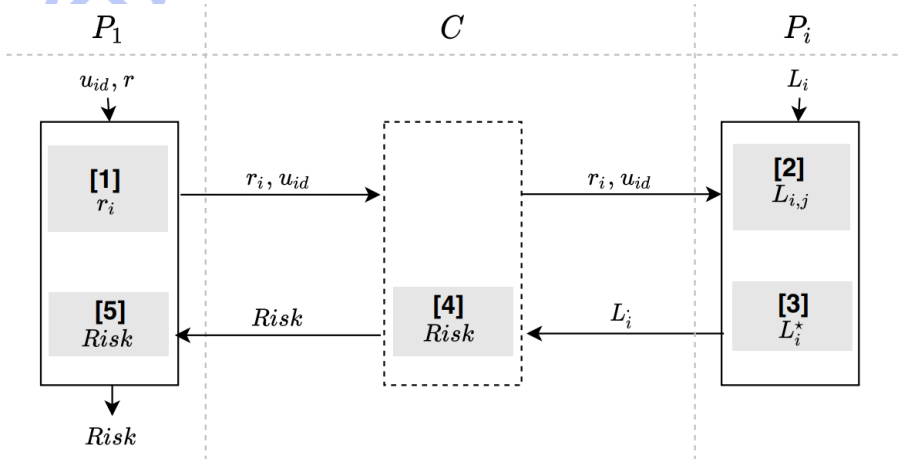


图 4.1.2 SS-ML 协议时序图

安全分析：

发起方提供的偿还能力信息和服务方的贷款信息都是以碎片形式出现在其它服务方中，各方都只能得到部分碎片，无法得到整体信息，而第三方在进行汇总时，汇总对象是计算的中间结果，无法反推，因此用户的借贷信息和偿还能力都不会发生泄漏，是安全的。

5.联邦训练

联邦训练(federated training)的目的是要从多个参与方数据中学习一个模型，如果是有监督学习，则必须有一方提供标签数据(Y)。在跨特征联邦中，不失一般性，通常假设P_1是提供标签的一方，也就是发起方；而在跨样本联邦中，每一方都会有相应的标签。

为了描述方便，在这里将多个参与方简化为两个参与方的情况。下面根据联邦方式不同，对参与联邦的数据分别描述如下：

- 跨特征联邦(cross-feature federation)：参与方P_1和P_2提供了N个对齐后的样本，每个样本分别有k_1和k_2个特征指标。
  - P_1提供标签数据$Y = y_{i_{i=1}}^{N}$，和特征数据集：$X^{(1)} = \{x_i^{(1)}\}_{i=1}^{N}$，其中$x_i^{(1)} = (x_i^{(1)}(1), x_i^{(1)}(2), \dots, x_i^{(1)}(k_1))^T$；
  - P_2提供特征数据集：$X^{(2)} = \{x_i^{(2)}\}_{i=1}^{N}$，其中$x_i^{(2)} = (x_i^{(2)}(1), x_i^{(2)}(2), \dots, x_i^{(2)}(k_2))^T$。
- 跨样本联邦(cross-sample federation)：参与方P_1和P_2分别提供N_1和N_2个样本，每个样本有相同的k个特征指标。
  - P_1提供标签数据$Y^{(1)} = \{y_i\}_{i=1}^{N_1}$，和特征数据集：$X^{(1)} = \{x_i^{(1)}\}_{i=1}^{N_1}$，其中$x_i^{(1)} = (x_i^{(1)}(1), x_i^{(1)}(2), \dots, x_i^{(1)}(k))^T$；
  - P_2提供标签数据$Y^{(2)} = y_{i_{i=1}}^{N_2}$，和特征数据集：$X^{(2)} = \{x_i^{(2)}\}_{i=1}^{N_2}$，其中$x_i^{(2)} = (x_i^{(2)}(1), x_i^{(2)}(2), \dots, x_i^{(2)}(k))^T$。

联邦训练是利用参与方提供的数据计算模型参数θ，使函数$f(X^{(1)}, X^{(2)}, \theta)$逼近标签$Y$。对于线性回归、逻辑回归等算法，跨样本联邦的数据交换过程相对比较简单，而跨特征联邦会比较复杂。尽管联邦训练中数据交换过程会因采用的模型不同而变化，但是联邦训练对数据安全的要求基本是一致的。

安全要求：

1) 不能泄露参与方的训练数据；
2) 不能泄漏参与方的模型参数；
3) 不能泄露标签数据。

下面将分别介绍适用于不同模型的数据安全交换协议。本章除了第 5.4.2 和 5.5 节是针对跨样本联邦的外，其余协议都是面向跨特征联邦的。

5.1 线性回归

线性回归就是利用目标函数$L(\theta)=\sum_{i=1}^{N}(l_i)^2=\sum_{i=1}^{N}(y_i-\theta^T x_i)^2$求解模型参数$\theta$。假设$\theta^{(1)}$和$\theta^{(2)}$分别是模型参数$\theta$中对应参与方$P_1$和$P_2$的部分，令$u_i^{(1)}=\theta^{(1)^T}x_i^{(1)}$，$u_i^{(2)}=\theta^{(2)^T}x_i^{(2)}$。那么$l_i=u_i^{(1)}+u_i^{(2)}-y_i$，联邦线性回归的目标函数就可以写成：

$$L(\theta)=\sum_{i=1}^{N}(l_i)^2=\sum_{i=1}^{N}\left(u_i^{(1)}+u_i^{(2)}-y_i\right)^2=\sum_{i=1}^{N}\left(\theta^{(1)^T}x_i^{(1)}+\theta^{(2)^T}x_i^{(2)}-y_i\right)^2$$

$$=\sum_{i=1}^{N}\left(\left(\theta^{(1)^T}x_i^{(1)}-y_i\right)^2+2\left(\theta^{(2)^T}x_i^{(2)}\right)\left(\theta^{(1)^T}x_i^{(1)}-y_i\right)+\left(\theta^{(2)^T}x_i^{(2)}\right)^2\right)。$$

已知，在实际计算过程中，参与方P_1和P_2模型参数迭代更新策略为：

$$\theta^{(1)}\leftarrow\theta^{(1)}-\eta\nabla_{\theta^{(1)}}L,$$

$$\theta^{(2)}\leftarrow\theta^{(2)}-\eta\nabla_{\theta^{(2)}}L。$$

其中

$$\nabla_{\theta^1}L=\sum_{i=1}^{N}2l_i x_i^{(1)},$$

$$\nabla_{\theta^{(2)}}L=\sum_{i=1}^{N}2l_i x_i^{(2)}。$$

从上式可知，计算模型参数更新的关键是计算损失函数l_i。而由于只有发起方P_1有标签数据，因此需要将各方数据和标签加密后汇总计算。

应用场景：在跨特征联邦训练中，如果模型采用线性回归模型，各方之间协同计算损失函数时可采用该协议。

基本思想：由于跨特征联邦线性回归中关键是计算损失函数，因此不同参与方可以先利用本地数据计算，将得到的中间结果加密后上传至第三方，由第三方汇总后解密，再反馈给各参与方。

协议过程：

假定两个参与方，一个是发起方P_1，一个是服务方P_2，分别利用本地数据训练模型。由于该协议基于同态加密实现，因此称为 HE-Linear-FT 协议。该协议的具体步骤见表 5.1。

表 5.1 HE-Linear-FT 协议流程

输入：

$u^{(i)}$：参与方P_i提供的中间结果；

y：发起方P_1提供的标签；

输出：

l：参与方收到的损失函数；

具体步骤：

Step1：第三方生成一对同态公私钥(pk,sk)，并将公钥pk发送给P_1和P_2；

Step2：P_1用公钥加密生成密文$[u^{(1)}-y]=E_{pk}(u^{(1)}-y)$，并发送给第三方；

Step3：P_2用公钥加密生成密文$[u^{(2)}]=E_{pk}(u^{(2)})$，并发送给第三方；

Step4：第三方计算l的密文$[l]=[u^{(1)}-y]+[u^{(2)}]$，并用私钥$sk$解密得到$l=D_{sk}([l])$，最后将$l$分别发给参与方$P_1$和$P_2$。

该协议对应的时序图如图 5.1 所示：

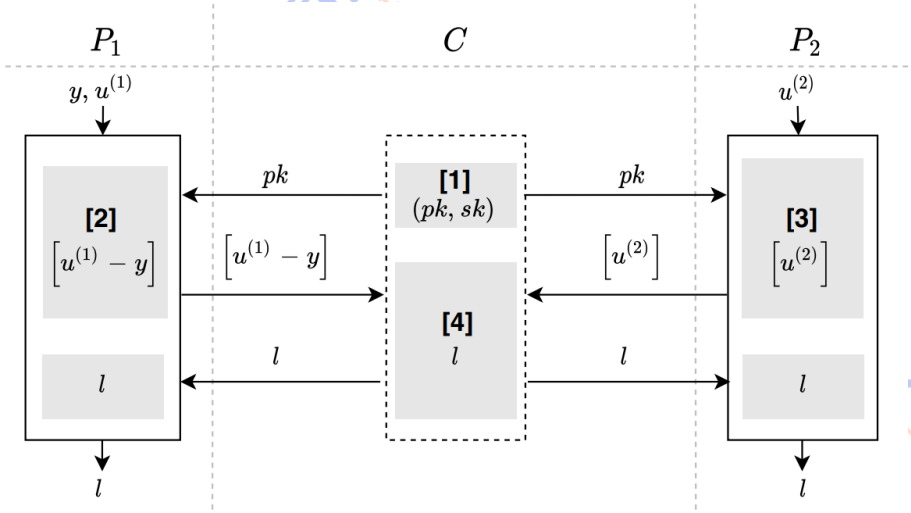


图 5.1 HE-Linear-FT 协议时序图

安全分析：

根据上述协议，各方参与交换的数据都是用本地模型计算后的中间结果，并且是加密后发送至第三方的，训练数据和模型参数都没有离开本地，不会直接泄漏给第三方或其它参与方。按照协议过程，第三方尽管拥有私钥具备解密的能力，但是解密出来也只能得到中间结果或损失大小，无法回溯到训练数据或模型参数，因此该协议不会泄漏隐私信息，符合安全要求。

5.2 逻辑回归

逻辑回归(logistic regression)是一种常用的有监督机器学习算法，其实现简单，应用广泛。逻辑回归是在线性函数$\theta^T x$输出预测值的基础上，寻找一个激活函数$h_\theta(x)=g(\theta^T x)$，将实际值映射到 0,1 之间。选择 Sigmoid 函数为激活函数，则

$$h_\theta(x)=\frac{1}{1+e^{-\theta^T x}}$$

继续采用上一节中相关符号的定义，那么联邦逻辑回归的目标函数是：

$$L = -\frac{1}{N}\sum_{i=1}^{n}(y_i \log h_\theta(x_i) + (1-y_i)\log(1-h_\theta(x_i)))。$$

这里的$y_i = 0$或1。在实际计算过程中，参与方P_1和P_2的模型参数迭代更新策略是：

$$\theta^{(1)} \leftarrow \theta^{(1)} - \eta\nabla_{\theta^{(1)}}L,$$
$$\theta^{(2)} \leftarrow \theta^{(2)} - \eta\nabla_{\theta^{(2)}}L。$$

由于$\theta^T x = {\theta^{(1)}}^T x_i^{(1)} + {\theta^{(2)}}^T x_i^{(2)} = u_i^{(1)} + u_i^{(2)}$，$\frac{\partial u_i^{(1)}}{\partial \theta^{(1)}} = x_i^{(1)}$，$\frac{\partial u_i^{(2)}}{\partial \theta^{(2)}} = x_i^{(2)}$，

所以梯度$\nabla_{\theta^{(1)}}L$和$\nabla_{\theta^{(2)}}L$可以用下式计算：

$$\nabla_{\theta^{(1)}} = \frac{\partial L}{\partial \theta^{(1)}} = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i - h_\theta(x_i)\right)x_i^{(1)},$$

$$\nabla_{\theta^{(2)}} = \frac{\partial L}{\partial \theta^{(2)}} = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i - h_\theta(x_i)\right)x_i^{(2)}。$$

由于联邦场景中参与方数据$x_i^{(1)}$和$x_i^{(2)}$在本地，令$h_\theta(x_i) = \hat{y}_i$，则梯度计算的关键为计算$y_i - \hat{y}_i$。

联邦逻辑回归使用同态加密(homomorphic encryption: HE)和一次一密(one-time pad)实现数据安全交换，下面介绍的面向联邦逻辑回归的两种协议，HE-OTP-LR-FT1 协议和 HE-OTP-LR-FT2 协议，分别是该协议的无第三方版本和有第三方版本。

应用场景：在跨特征联邦训练中，如果模型采用逻辑回归模型，参与方之间协同计算模型参数更新时可以采用该协议。在二分类中，该协议要求标签信息取值为 0 或 1。

5.2.1 HE-OTP-LR-FT1 协议

基本思想：由于只有发起方P_1有标签数据y_i，P_1拿到${\theta^{(2)}}^T x_i^{(2)}$后便可计算中间结果$y_i - \hat{y}_i$，从而完成$P_1$方的梯度计算，所以协议的核心是如何完成$P_2$方的梯度计算。在 HE-OTP-LR-FT1 协议中，由发起方P_1生成私钥，P_2在密文上运算梯度并盲化后交给P_1方解密。

协议过程：

假定两个参与方，一个是发起方P_1，一个是服务方P_2，分别利用本地数据训练模型。该协议的流程见表 5.2.1。

表 5.2.1 HE-OTP-LR-FT1 协议流程

输入：

$x^{(i)}$：参与方P_i提供的训练数据；

$\theta^{(i)}$：参与方P_i对应的模型参数；

$y \in \{0,1\}$：发起方P_1提供的标签；

输出：

$h_\theta(x)$：激活函数计算结果，用于计算损失函数；

$\frac{\partial L}{\partial \theta^{(1)}}$：发起方$P_1$端的梯度，用于更新$\theta^{(1)}$；

$\frac{\partial L}{\partial \theta^{(2)}}$：服务方$P_2$端的梯度，用于更新$\theta^{(2)}$。

具体步骤：

Step1：发起方P_1生成一对公私钥(pk, sk)用于加法同态加密，并将公钥pk发送给服务方P_2；

Step2：服务方P_2计算$u^{(2)} = {\theta^{(2)}}^T x^{(2)}$，然后将$u^{(2)}$发给$P_1$；

Step3：发起方P_1计算$\theta^T x = {\theta^{(1)}}^T x^{(1)} + {\theta^{(2)}}^T x^{(2)}$，生成$\hat{y_i} = h_\theta(x)$并输出，计算梯度$\frac{\partial L}{\partial \theta^{(1)}} = (\mathrm{y} - \hat{\mathrm{y}}) x^{(1)}$用于更新$\theta^{(1)}$。同时，$P_1$将$\mathrm{y} - \hat{\mathrm{y}}$加密，将$[\mathrm{y} - \hat{\mathrm{y}}]$发送给$P_2$；

Step4：服务方P_2计算$\left[\frac{\partial L}{\partial \theta^{(2)}}\right] = [\mathrm{y} - \hat{\mathrm{y}}] x^{(2)}$，选择随机数$R_2$进行加密，将密文$\left[\frac{\partial L}{\partial \theta^{(2)}}\right] + [R_2]$发送给$P_1$；

Step5：发起方P_1替P_2解密得$\frac{\partial L}{\partial \theta^{(2)}} + R_2$，发送给$P_2$；

Step6：服务方P_2得到梯度$\frac{\partial L}{\partial \theta^{(2)}}$用于更新$\theta^{(2)}$。

协议时序图见图 5.2.1。

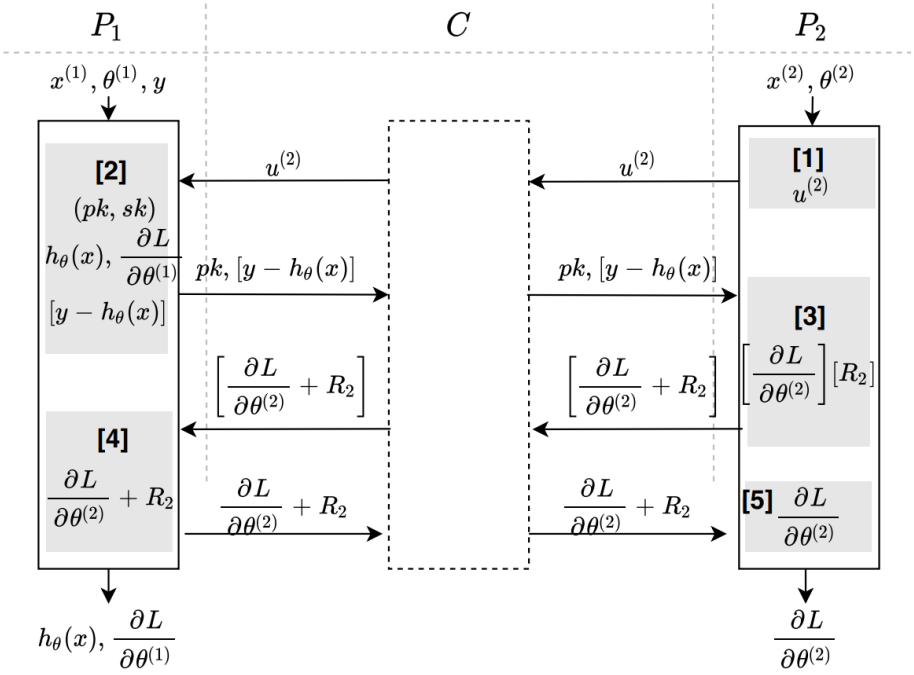


图 5.2.1 HE-OTP-LR-FT1 协议时序图

安全分析：

从协议流程可知，在 step2 和 step4，服务方P_2向发起方P_1传输了数据$u^{(2)} = \theta^{(2)^T} x^{(2)}$和$[\frac{\partial L}{\partial \theta^{(2)}} + R_2]$。在一定边界条件下，$P_1$无法从$\theta^{(2)^T} x^{(2)}$中求解$\theta^{(2)}$或$x^{(2)}$，亦无法从盲化数据$\frac{\partial L}{\partial \theta^{(2)}} + R_2$中求解$x^{(2)}$，即$P_2$的隐私数据$x^{(2)}, \theta^{(2)}$是安全的。在 step3 和 step5，发起方$P_1$向服务方$P_2$传输数据$[\mathrm{y} - \hat{\mathrm{y}}]$和$\frac{\partial L}{\partial \theta^{(2)}} + R_2$。由于参与方$P_2$只有公钥，无法解密得$\mathrm{y} - \hat{\mathrm{y}}$，在一定边界条件下，$P_2$亦无法从$\frac{\partial L}{\partial \theta^{(2)}}$中求解$\mathrm{y} - \hat{\mathrm{y}}$，无法进一步从$\hat{\mathrm{y}}$中求解$\theta^{(1)^T} x^{(1)}$，即$P_1$的隐私数据$x^{(1)}, \theta^{(1)}$和标签数据$\mathrm{y}$是安全的。因此数据交换过程是安全的，不会产生数据隐私泄漏。

5.2.2 HE-OTP-LR-FT2 协议

基本思想：如 5.2.2 所述，协议的核心是如何完成P_2方的梯度计算。在 HE-OTP-LR-FT2 协议中，由第三方生成私钥，发起方P_1为服务方P_2在密文上运算梯度并使用P_2的随机数盲化，计算结果交给第三方解密。

协议过程：

假定三个参与方，发起方P_1和服务方P_2分别利用本地数据训练模型，第三方参与运算。该协议的流程见表 5.2.2。

表 5.2.2 HE-OTP-LR-FT2 协议流程

输入：

$x^{(i)}$：参与方P_i提供的训练数据；

$\theta^{(i)}$：参与方P_i对应的模型参数；

$y \in \{0,1\}$：发起方P_1提供的标签；

输出：

$h_\theta(x)$：激活函数计算结果，用于计算损失函数；

$\frac{\partial L}{\partial \theta^{(1)}}$：发起方$P_1$端的梯度，用于更新$\theta^{(1)}$；

$\frac{\partial L}{\partial \theta^{(2)}}$：服务方$P_2$端的梯度，用于更新$\theta^{(2)}$。

具体步骤：

Step1：第三方生成一对公私钥(pk, sk)用于加法同态加密，并将公钥pk发送给服务方P_2；

Step2：服务方P_2计算$u^{(2)} = \theta^{(2)^T} x^{(2)}$，然后将$u^{(2)}$发给$P_1$；

Step3：发起方P_1计算$\theta^T x = {\theta^{(1)}}^T x^{(1)} + {\theta^{(2)}}^T x^{(2)}$，生成$\hat{y_i} = h_\theta(x)$并输出，计算梯度$\frac{\partial L}{\partial \theta^{(1)}} = (\mathrm{y} - \hat{\mathrm{y}})x^{(1)}$用于更新$\theta^{(1)}$。同时，$P_1$将$\mathrm{y} - \hat{\mathrm{y}}$加密，将$[\mathrm{y} - \hat{\mathrm{y}}]$发送给$P_2$；

Step4：服务方P_2计算$\left[\frac{\partial L}{\partial \theta^{(2)}}\right] = [\mathrm{y} - \hat{\mathrm{y}}]x^{(2)}$，选择随机数$R_2$进行加密，将密文$\left[\frac{\partial L}{\partial \theta^{(2)}}\right] + [R_2]$发送给第三方；

Step5：第三方替P_2解密得$\frac{\partial L}{\partial \theta^{(2)}} + R_2$，发送给$P_2$；

Step6：服务方P_2得到梯度$\frac{\partial L}{\partial \theta^{(2)}}$用于更新$\theta^{(2)}$。

协议时序图见图 5.2.2。

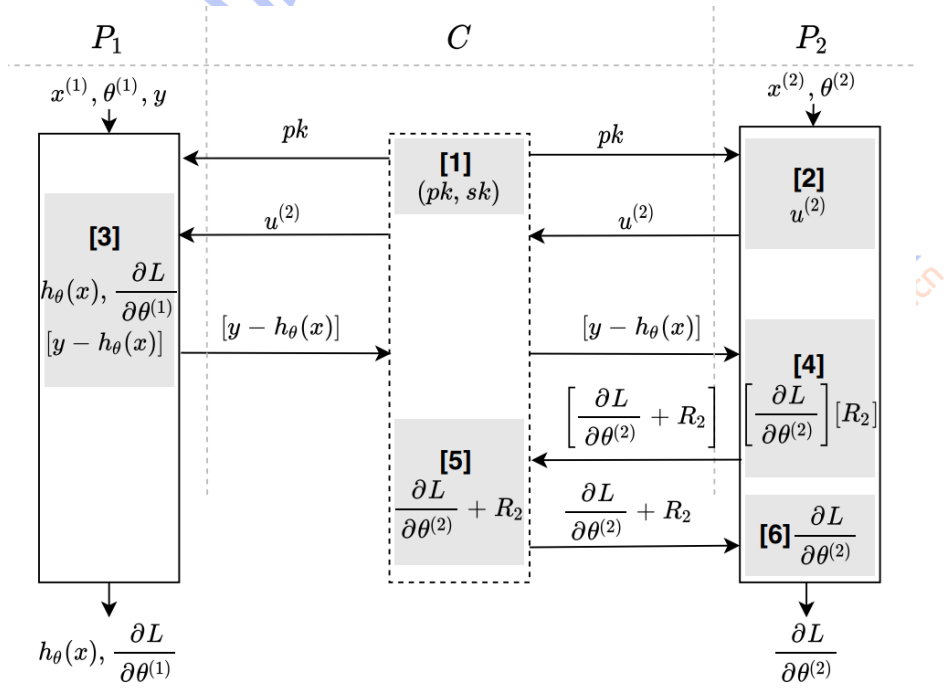


图 5.2.2 HE-OTP-LR-FT2 协议时序图

安全分析：

从协议流程可知，虽然与 HE-OTP-LR-FT1 协议相比，服务方P_2在 step4 中将加密数据$[\frac{\partial L}{\partial \theta^{(2)}} + R_2]$发送给第三方而不再是发起方$P_1$，但是同样地，第三方亦无法从盲化数据$\frac{\partial L}{\partial \theta^{(2)}} + R_2$中求解$x^{(2)}$，即$P_2$的数据是安全的，其他步骤安全性与 HE-OTP-LR-FT1 协议相同。

5.3 神经网络

神经网络模型是一种表达能力很强的模型，相比传统的机器学习模型，在数据量越多的情况下越容易达到更高的性能。这里说的数据量包含两种含义，一是样本数量够多，需要用跨样本的方式来实现联邦；二是特征维度要多样，这就需要用跨特征联邦。针对不同联邦形式，会采用不同的联邦策略，也因此会有不同的数据交换协议，跨样本联邦过程可以采用第 5.5 节安全聚合的协议，这里介绍的协议是面向跨特征联邦的。

首先假设参与联邦的各方之间都采用相同的神经网络模型，在跨特征联邦的场景中，由于输入特征不同，因此需要对参与方缺少的特征进行填充，填充有很多种形式，最简单的就是直接填 0。根据随机梯度下降(SGD)方法可知模型参数更新的过程如下：

$$\theta \leftarrow \theta - \eta \frac{\partial L}{\partial \theta}。$$

显然参数更新的核心是计算模型参数的梯度。由于只有发起方有标签，其它参与方没有标签信息，也就没法直接计算对应的梯度，因此无标签的参与方只能遍历标签可能值计算出所有候选梯度，并将候选梯度发给有标签的发起方，让发起方挑选真实的梯度进行汇总。

应用场景：适用于基于神经网络对跨特征联邦训练中，并且各参与方之间需要采用相同的网络模型。

基本思想：该协议借用不经意传输的思想，把各种可能的标签值都遍历一次来计算参数梯度，并加入些不影响统计结果的随机噪声，然后再将梯度加密后发给发起方，让其解密后来选择真实的梯度进行统计汇总。

协议过程：

假定两个参与方，一个是发起方P_1，一个是服务方P_2，对齐后有n个样本，网络模型参数共有m个。发起方P_1提供标签，标签共有K种可能的离散取值。服务方P_2因为没有标签，所以会计算所有可能的标签取值。由于该协议采用一次一密实现，因此称为 OTP-NN-FT 协议。该协议的流程见表 5.3。

表 5.3 OTP-NN-FT 协议流程

输入：

$Y \in \mathcal{R}^{1\times n}$：发起方$P_1$提供的标签，以及本方输入对应的平均梯度$\bar{g}^{(1)}$；

$g_k^{(2)} \in \mathcal{R}^{m\times n}$：参与方$P_2$中对应标签第$k$个取值时的梯度；

输出：

$g \in \mathcal{R}^{m\times 1}$：两个参与方的平均梯度；

具体步骤：

Step1：参与方P_2生成一个随机矩阵$r \in \mathcal{R}^{m\times n}$，满足每行和为 0 的条件，然后对所有标签取值分别计算$\hat{g}_i^{(2)} = r + g_i^{(2)}$，最后将$\hat{g}_i^{(2)}$发送给$P_1$；

Step2：参与方P_1根据标签实际取值选择相应的梯度组成新的梯度矩阵$\hat{g}^{(2)} \in \mathcal{R}^{m\times n}$，按行取平均计算平均梯度$\bar{g}^{(2)}$；

Step3: 参与方P_1计算$g = \frac{\bar{g}^{(1)}+\bar{g}^{(2)}}{2}$，并将$g$发送给$P_2$。

注：在实现时，通常将模型中对应不同标签的所有梯度同时作为输入，则输入输出为梯度的列表。

协议时序图见图 5.3。

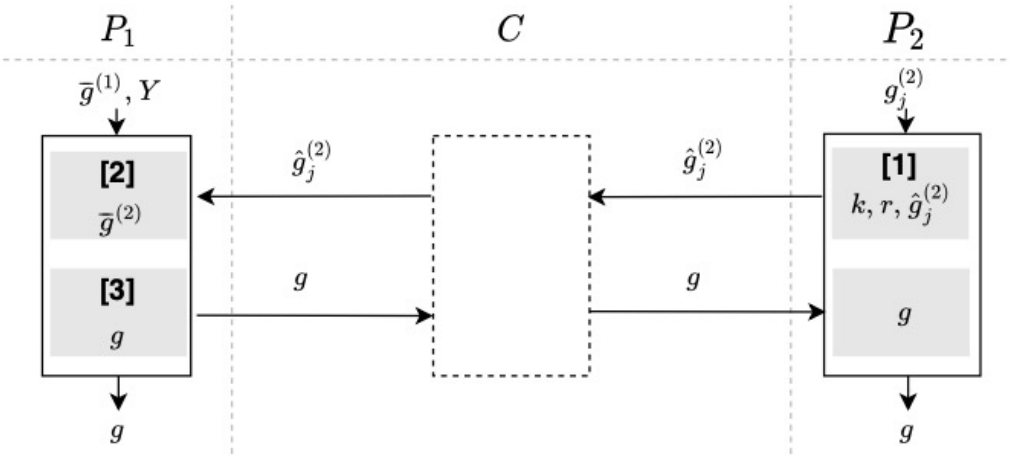


图 5.3 OTP-NN-FT 协议时序图

安全分析：

上述协议中只有参与方的梯度作为交换数据进行传输，而且梯度经过噪声干扰，其它方无法获取真实的梯度信息，因此不会泄漏模型参数，也不存在泄漏样本特征值和标签的可能。

5.4 树模型

树模型是一种应用广泛的机器学习模型，模型效果好并且有较强的可解释性。在决策树的生成中，常用 ID3、C4.5、Gini 指数等指标去选择最优分裂特征、切分点，XGBoost 同样定义了特征选择和切分点选择的指标：

$$Gain = \frac{1}{2}\left[\frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{(G_L+G_R)^2}{H_L+H_R+\lambda}\right] - \gamma。$$

其中，γ表示切分后模型复杂度的增加量，λ表示惩罚系数，$\frac{G_L^2}{H_L+\lambda}$和$\frac{G_R^2}{H_R+\lambda}$分别表示在某个节点按条件切分后左、右子节点的得分，$\frac{(G_L+G_R)^2}{H_L+H_R+\lambda}$表示切分前的得分。此外，$G_L = \sum_{i\in L} g_i$ 和$H_L = \sum_{i\in L} h_i$ 分别表示左子节点上所有样本的一阶g_i和二阶导数h_i之和：

$$g_i = \frac{\partial l\left(y_i, \hat{y}_i^{(t-1)}\right)}{\partial\ \hat{y}_i^{(t-1)}},$$

$$h_i = \frac{\partial^2 l\left(y_i, \hat{y}_i^{(t-1)}\right)}{\partial\ \hat{y}_i^{(t-1)}}。$$

这里$l\left(y_i, \hat{y}_i^{(t-1)}\right)$表示前$t-1$次迭代的损失函数。

XGBoost 中使用上面的指标判断切分增益，增益值$Gain$越大，说明分裂后能使目标函数减少越多，就越好。在 XGBoost 类方法中，其核心就是通过判断增益查找最优特征以及最优切分点，而计算增益的关键就是计算按条件切分后左右子节点的得分。

5.4.1 HE-GB-FT 协议

在跨特征联邦的场景中，由于标签只在发起方存在，其它参与方没有标签信息，也就没有一阶g_i和二阶导数h_i，因此需要在发起方和其它参与方之间进行数据(g_i和h_i)交换。

应用场景：适用于 XGBoost 跨特征联邦训练中。

基本思想：该协议采用同态加密的方式将一阶导数和二阶导数加密后传输给其它参与方。其它参与方只能在密文空间上计算切分增益，并回传给发起方来确定增益最大的特征和切分点。发起方将选择特征和切分点同步给各参与方后，参与方再生成相应左右子树。

协议过程：

假定两个参与方，一个是发起方P_1，一个是服务方P_2，各方已经对特征进行分箱，并且发起方P_1提供相应的一阶和二阶导数。该协议最终会选择最优特征和切分点，进而生成左右子树。由于该协议基于同态加密实现，因此称为 HE-GB-FT 协议。该协议的具体流程见表 5.4.1。

表 5.4.1 HE-GB-FT 协议流程

输入：

$b^{(i)}$：参与方P_i在该节点的所有特征的分箱信息；

g_i和h_i：发起方P_1提供的第i个样本对应的一阶和二阶导数；

输出：

t：该节点是否为叶子节点；

s：该节点的最优切分点信息；

具体步骤：

Step1：发起方P_1生成一对同态公私钥(pk, sk)，并用公钥pk加密所有$\{g_i\}$和$\{h_i\}$得到$\{[g_i]\}$和$\{[h_i]\}$，加密后发送给服务方P_2；

Step2：P_2统计本方特征对应分箱内的$[g_i]$和$[h_i]$，P_2将统计后的密文结果$\{[G_j]\}$和$\{[H_j]\}$发送给P_1，P_1根据本方分箱信息得到明文$\{G_k\}$和$\{H_k\}$；

Step3：P_1解密接收到的$\{[G_j]\}$和$\{[H_j]\}$，与本地计算得到的明文$\{G_k\}$和$\{H_k\}$合并得到$\{G_l\}$和$\{H_l\}$，分别计算每种切分的增益，并判断该节点是否为叶子节点，若为非叶子节点，则计算最优切分点信息s；

Step4：P_1将当前节点是否为叶子节点t以及最优切分点信息s同步给服务方P_2。

协议时序图见图 5.4.1。

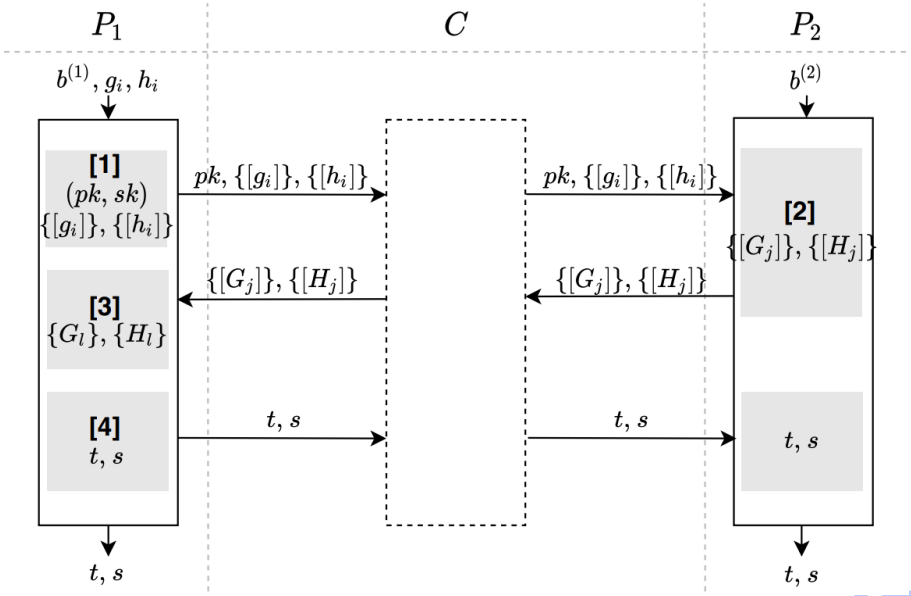


图 5.4.1 HE-GB-FT 协议时序图

安全分析：

由于多方之间传递的核心数据是一阶导数和二阶导数，而且是以同态密文的形式，所以不存在样本的训练数据和标签泄漏的可能。至于切分点和子树列表的传输都不会涉及到样本的训练数据和标签。

5.4.2 CS-GB-FT 协议

在跨样本联邦的场景中，各参与方都有自有样本的标签，也就可以得到一阶和二阶导数，但是由于样本分布在不同方，最优切分产生的左右子树得分却无法直接计算，需要借助第三方进行统计汇总。此外，左右子结点的权重也需要依靠第三方进行计算：$w_l = \frac{G_L}{H_L+\lambda}$和$w_r = \frac{G_R}{H_R+\lambda}$。

应用场景：适用于 XGBoost 跨样本联邦训练。

基本思想：这里主要是借助第三方聚合候选切分点和统计切分信息，并寻找最优切分点和分裂特征。也就是各参与方将候选切分点都放在第三方一起汇总，然后各方根据候选点统计本地数据的一阶和二阶导数，再交给第三方进一步汇总。最后由第三方对比选择最优切分点和分裂特征，划分出左右子节点，并计算切分后生成的左右子节点的权重。

协议过程：

假定有多个参与方P_i，各方都有一定数量的样本和标签。参与方P_i提供当前节点深度，样本特征，特征类型（离散、连续），并已计算好样本相应的一阶和二阶导数。该协议最终会选择最优切分点生成左右子节点，并计算左右子节点的权重。该协议的具体流程见表 5.4.2。

表 5.4.2 CS-GB-FT 协议流程

输入：

$f^{(i)}$和$d^{(i)}$：参与方P_i在当前节点的样本特征信息及节点深度；

$g_k^{(i)}$和$h_k^{(i)}$：参与方P_i中第k个样本对应的一阶导数和二阶导数；

t：节点内特征的类型列表；

输出：

若节点为叶子节点：

w：节点的权重；

若节点为非叶子节点：

$T_l^{(i)}$和$T_r^{(i)}$：参与方P_i分布在生成的左子节点和右子节点上的样本列表；

I：切分点信息，包括切分特征 id，切分值和 gain 值；

具体步骤：

Step1：参与方P_i将本地的节点数量及节点深度$n^{(i)}$和$d^{(i)}$上传至第三方；

Step2：第三方汇总所有的节点数量，判断节点是否为叶子节点，若节点为叶子节点，则计算节点权重w，将以上信息返回给各参与方P_i；

Step3：若节点为叶子节点，则参与方P_i返回计算节点权重；若节点为非叶子节点，参与方P_i利用本地$\{g_k^{(i)}\}$和$\{h_k^{(i)}\}$生成本地候选切分点信息$s^{(i)}$，发送给第三方；

Step4：第三方汇总接收到的切分点信息$s^{(i)}$，并将汇总的信息s发送给各参与方；

Step5：参与方P_i根据切分点列表s统计所有分箱的$G_j^{(i)}$和$H_j^{(i)}$信息，然后将统计结果发送给第三方；

Step6：第三方对接收到的$\{G_j^{(i)}\}$和$\{H_j^{(i)}\}$进一步汇总，得到每个分箱的$\{G_j\}$和$\{H_j\}$，计算最优切分点s^*并判断当前节点是否为叶子节点，若是叶子节点，则计算节点权重w，将以上信息发送给各参与方；

Step7：若节点为叶子节点，则参与方P_i返回计算节点权重；若节点为非叶子节点，参与方P_i根据切分点信息分别统计本方分布在左右子节点的样本，生成$T_l^{(i)}$和$T_r^{(i)}$，并将左右子节点样本数量$n_l^{(i)}, n_r^{(i)}$发送给第三方；

Step8：第三方对接收到的左右子节点样本进行汇总，并再次判断当前节点是否为叶子节点，若是，则计算节点权重w，并将结果发给各参与方；

Step9：若节点为叶子节点，则参与方P_i返回节点权重w；若节点为非叶子节点，各参与方返回切分点信息I和左右子节点上的样本列表$T_l^{(i)}$和$T_r^{(i)}$。

其中，Step2、6、8 中计算节点权重需要参与方与第三方交互完成，其步骤如下：a)参与方P_i对本地节点内的$g_k^{(i)}$和$h_k^{(i)}$求和，得到$g^{(i)}$和$h^{(i)}$，发送给第三方；b）第三方根据接收到的$g^{(i)}$和$h^{(i)}$计算节点权重w，发送到每个参与方；c)参与方P_i返回权重w。

协议时序图见图 5.4.2。

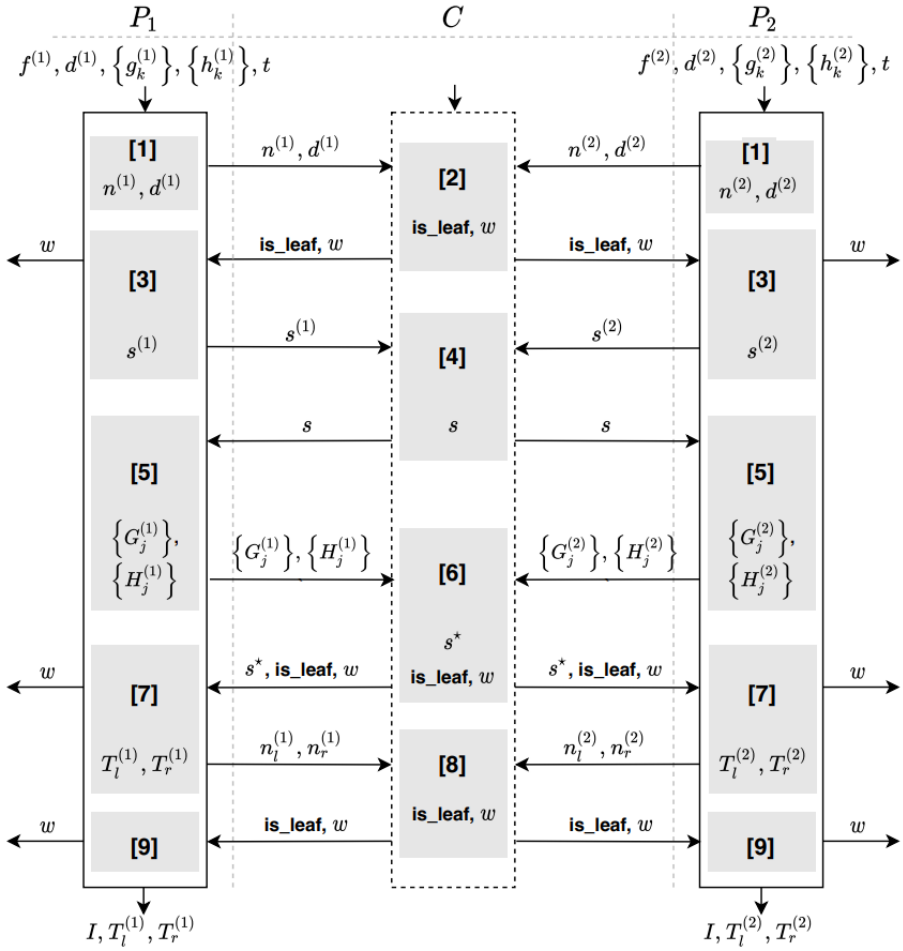


图 5.4.2 CS-GB-FT 协议时序图

安全分析：

该协议中，参与方与第三方交换的主要是分箱内的一阶导数和二阶导数的和，所以不存在样本的训练数据和标签泄漏的可能。至于切分点和子节点权重的传输都不会涉及到样本的训练数据和标签。

5.5 安全聚合

安全聚合(secure aggregation)是指在联邦训练中由第三方对各参与方产生的中间结果进行聚合的过程。这里的中间结果，可以是模型训练产生的梯度或模型参数等。聚合可以有很多种形式，最简单的方式可以采用取平均值作为聚合运算。

安全聚合可以提升模型的鲁棒性，但前提条件是各参与方之间进行交换的模型参数必须一致。

安全聚合可使用多种技术实现，如同态加密、一次一密等。同态加密方案的计算复杂度较高，尤其是在模型参数较多时，其加解密过程会非常耗时。而且，由于训练迭代过程中需要经常进行聚合，通信成本也会非常高。相比之下，一次一密计算相对简单，其计算量和通信量都较小，但也可以达到同样的安全要求。

应用场景：安全聚合适用于跨样本联邦训练，各参与方将参数变量的中间结果发送到第三方，由第三方对其聚合并返还聚合结果。

5.5.1 OTP-SA-FT 协议

基于一次一密实现的安全聚合协议(OTP-SA-FT)是一种效率很高的安全聚合方式。它主要使用安全密钥交换、安全伪随机数生成和一次一密的基础协议。

基本思想：该协议核心是将参与方待交换的中间结果按位运算加上一个随机数，而所有参与方使用的随机数总和为 0，这样在第三方就可以直接进行中间结果的聚合平均计算。协议的重点是如何生成多个随机数，并在各方不知其它方随机数的情况下能保证总和为 0，这也是使用安全密钥交换和安全伪随机数生成协议的原因。

协议过程：

假定有两个参与方P_1和P_2，各方都有一定数量的样本和标签用于训练。利用本地数据和标签可以训练某个模型，假设模型参数由n个变量组成，那么得到模型参数的中间结果$\theta^{(1)}$和$\theta^{(2)} \in \mathcal{R}^{1\times n}$。该协议最终会安全地生成模型参数聚合后的中间结果。协议的具体流程见表 5.5.1。

表 5.5.1 OTP-SA-FT 协议流程

输入： $\theta^{(i)}$：参与方P_i产生的中间结果； **输出：** θ：聚合后的中间结果；
具体步骤： Step1：参与方P_i调用安全密钥交换协议，得到相同的随机数r并以r为种子使用安全伪随机数生成算法生成与$\theta^{(i)}$相同类型和大小的随机数序列R，同时将中间结果$\theta^{(i)}$编码为整数类型，记为$\theta^{(i)^*}$； Step2：参与方P_1计算$\hat{\theta}^{(1)} = \theta^{(1)^*} + R$，并发送给第三方； Step3：参与方$P_2$计算$\hat{\theta}^{(2)} = \theta^{(2)^*} - R$，并发送给第三方； Step4：第三方计算聚合结果$\hat{\theta} = \left(\hat{\theta}^{(1)} + \hat{\theta}^{(2)}\right)/2$，并分别发送给参与方； Step5：参与方分别解码$\hat{\theta}$得到聚合后的中间结果θ。

在本协议中，编码是指将 32 位浮点数转换为 64 位无符号整数，其中的加减运算为模2^{64}的加减运算，对应的解码为将 64 位无符号整数转换为 32 位浮点数的运算。

协议时序图见图 5.5.1。

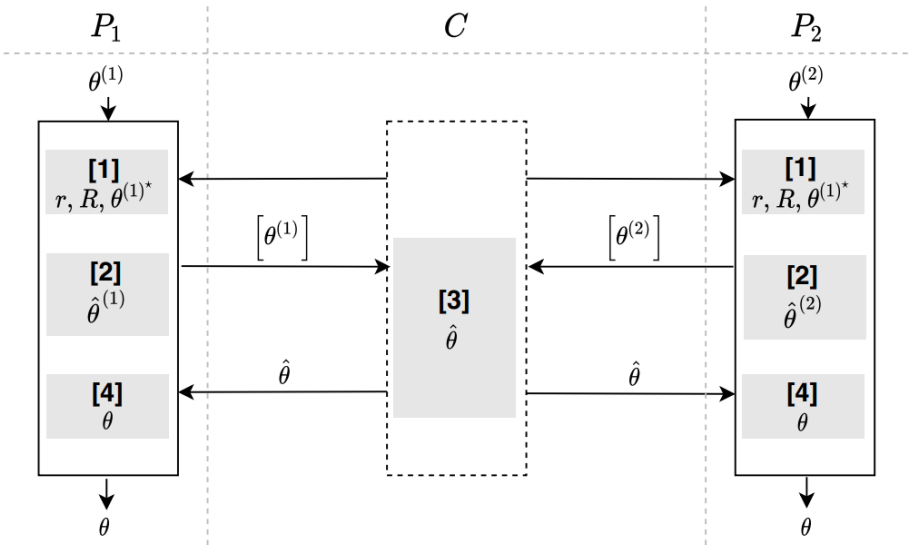


图 5.5.1 OTP-SA-FT 协议时序图

安全分析：

该协议是以模型参数的中间结果作为交换数据进行多方交换，不涉及训练数据和标签信息，因此不会造成训练数据和标签的泄漏。此外，模型参数在传输到第三方前先做了整型编码，然后加入了随机数作为干扰，第三方也无法知道实际的模型参数中间结果，所以模型参数也不会产生泄漏，是安全的。

5.5.2 HE-SA-FT 协议

基本思想：基于同态加密安全聚合的核心就是将参与方待交换的中间结果预先进行同态加密，加密后再上传至第三方，而第三方不经解密，直接在密文空间上进行聚合平均计算，计算结果再以密文形式反馈给各方，让参与方自行解密。

协议过程：

假定有两个参与方P_1和P_2，各方都有一定数量的样本和标签用于训练。利用本地数据和标签可以训练某个模型，假设模型参数由n个变量组成，那么得到模型参数的中间结果$\theta^{(1)}$和$\theta^{(2)} \in \mathcal{R}^{1\times n}$。该协议最终会安全地生成模型参数聚合后的中间结果。协议的具体流程见表 5.5.2。

表 5.5.2 HE-SA-FT 协议流程

输入：
$\theta^{(i)}$：参与方P_i产生的中间结果；
输出：

θ：聚合后的中间结果；

具体步骤：

Step1：参与方P_i调用安全密钥交换协议，得到相同的随机数r，并用随机数作为种子生成同态公私钥对(pk, sk)；

Step2：参与方P_i将中间结果$\theta^{(i)}$用公钥加密得到$[\theta^{(i)}] = E_{pk}(\theta^{(i)})$，并发送给第三方；

Step3：第三方直接在密文空间上计算中间结果的平均值$[\theta] = \frac{1}{2}\Sigma_i[\theta^{(i)}]$，并分别发送给参与方；

Step4：参与方用私钥解码得到聚合后的中间结果$\theta = D_{sk}([\theta])$。

对应的协议时序图如图 5.5.2 所示：

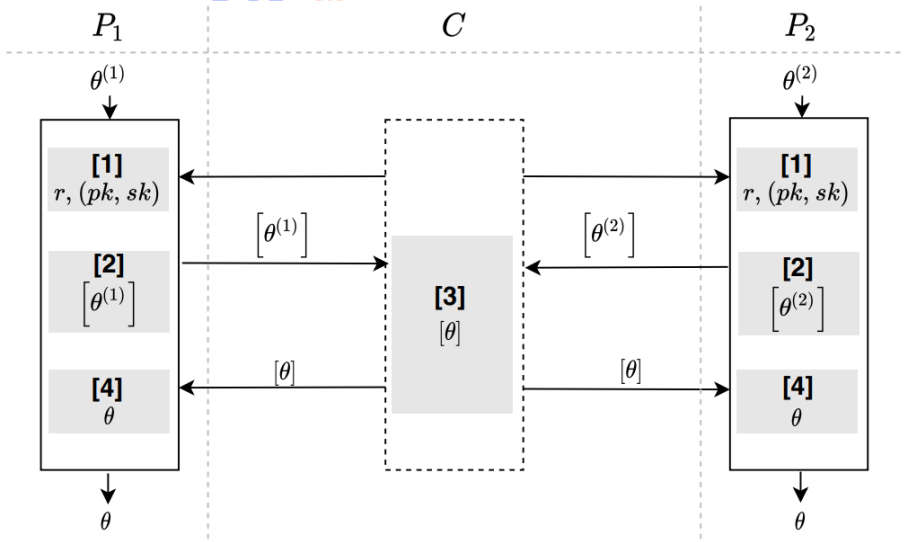


图 5.5.2 HE-SA-FT 协议时序图

安全分析：

该协议同样是以模型参数的中间结果作为交换数据进行多方交换，不涉及训练数据和标签信息，因此不会造成训练数据和标签的泄漏。此外，模型参数在传输到第三方前已经做了同态加密，第三方不需要也无法对中间结果解密，所以模型参数也不会产生泄漏，是安全的。

6.联邦预测

联邦训练生成模型后就可以投入使用进行预测了。跨样本联邦，只需要本地特征数据和本地模型及可以直接预测，不会涉及数据交换。而跨特征联邦中，由于训练过程用到了多方特征，预测时模型还会用到多方特征，会涉及到数据交换，因此这里的联邦预测(federated prediction)是面向跨特征联邦的。

逻辑回归联邦预测的核心是计算$u = \theta^T x$，因为x分布在不同参与方，需要分别在各方计算一个中间结果$u^{(i)} = \theta^{(i)^T} x^{(i)}$，然后再对中间结果汇总求和。神经网络模型则需要利用本地模型分别预测一个中间结果$u^{(i)}$，然后将它们汇总平均。因此可以将各方计算的中间结果输入第三方，让第三方帮助完成计算，这样既可以保证预测数据，也可以保证模型参数不会被泄漏。

安全要求：

1) 不能泄露参与方的预测数据；
2) 不能泄漏参与方的模型参数。

应用场景：当联邦逻辑回归或神经网络模型进行预测时，在各参与方之间协同计算需要该协议。

基本思想：联邦预测将各方的中间结果，通过同态加密的方式加密后发送给第三方，第三方将结果汇总后返回给发起方，进行后续运算。

协议过程：

为简化起见，依然假定有两个参与方，一个是发起方P_1，一个是服务方P_2，分别利用本地模型计算了中间结果并提供给第三方。由于该协议采用了同态加密思想，因此协议简称为 HE-LR-FP 协议。该协议的具体流程见表 6.1：

表 6.1 HE-LR-FP 协议流程

输入： $u^{(i)}$：参与方P_i提供的中间结果； **输出：** u：发起方P_1收到的汇总结果；
具体步骤： Step1：发起方P_1生成一对同态公私钥(pk,sk)，并将公钥pk和$u^{(1)}$加密后一起发给第三方； Step2：第三方公钥pk发送给服务方P_2； Step3：服务方P_2接收公钥，然后将$u^{(2)}$加密后发给第三方； Step4：第三方在密文空间上计算$[u] = E_{pk}(u^{(1)} + u^{(2)})$，并将结果发给$P_1$； Step5：$P_1$用私钥$sk$解密后得到$u = D_{sk}([u])$。

协议时序图见图 6.1。

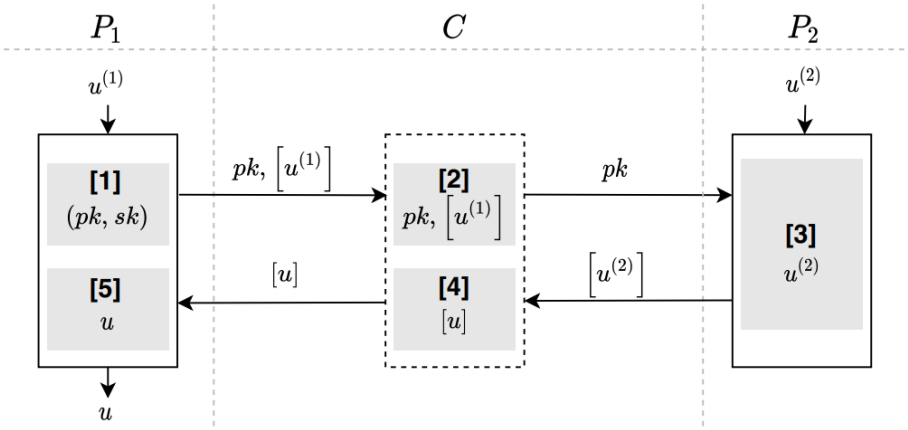


图 6.1 HE-LR-FP 协议时序图

安全分析：

从协议流程可知，数据交换过程中只涉及中间结果，不涉及训练数据或模型参数，而且是加密后传输至第三方。第三方上的运算也是在密文空间上发生，不会解密，因此数据交换过程是安全的，不会产生数据隐私泄漏。

7.公共组件

前面讲述的联邦数据安全交换过程会经常使用一些基础性的密码算法或安全协议，比如同态加密、秘密分享等。这一章，我们将这些基础算法或协议单独整理在一起，作为底层的公共组件，支撑联邦数据安全交换过程。

7.1 同态加密

一般的加密方案关注的都是数据存储安全。没有密钥的用户，不可能从加密结果中得到有关原始数据的任何信息，也不能对加密结果做任何操作的，只能进行存储、传输。

同态加密(homomorphic encryption: HE)关注的则是数据处理安全。同态加密提供了一种对加密数据进行处理的功能。也就是说，其他人可以对加密数据进行处理，但是处理过程不会泄露任何原始内容。同时，拥有密钥的用户对处理过的数据进行解密后，得到的正好是处理后的结果。

同态加密根据加密函数的不同可以分为加法同态、乘法同态和全同态加密。加法同态只能进行加减法运算，如 Paillier 算法[4]。乘法同态只能进行乘除法运算，如 RSA 算法。全同态可以进行各种运算，包括加减乘除、多项式求值、指数、对数、三角函数等，Gentry 算法。

目前 FLEX 协议中主要使用的是 Paillier 算法和 EC-ElGamal 算法。EC-ElGamal 算法满足加法同态，是基于椭圆曲线的 ElGamal 加密算法，而原始的 ElGamal 算法[5]则满足乘法同态。

7.2 密钥交换协议

密钥交换协议解决了对称密码体制中的密钥分发问题，使得通信双方可以通过公开信道安全地交换共享的密钥或随机数。

Diffie–Hellman(简称 DH)密钥交换[6]是密码学领域内最早付诸实践的密钥交换方法之一。DH 算法可以让双方在完全缺乏对方(私有)信息的前提条件下通过不安全的信道达成一个共享的密钥，并可用于对后续信息交换进行对称加密。

FLEX 协议中采用的是经典的 DH 密钥交换算法，其安全性是依赖于计算离散对数的困难程度。针对不同等级的安全要求，可以选择使用不同长度的大素数 p，常用的有 2048、3072、4096、6144、8192 位五种长度。通常，位数越大安全性越高，但计算时间就越长。

7.3 安全伪随机数生成

随机数被广泛用于密钥产生、初始化向量、时间戳、认证挑战码、密钥协商、大素数产生等方面。随机数生成器就是用于生成随机数的算法或函数，它的安全性对密码系统的安全性有重要影响。随机数生成器包括：真随机（非确定性）数生成器和伪随机（确定性）数生成器两类。

真随机数生成器随机性非常好，完全不可预测和回溯，但一般需要依靠特定的物理系统环境条件才能获取，所以应用范围有限。一般系统使用的随机数生成器都是伪随机的。伪随机数生成器之所以被称为确定性随机数生成器，是因为在确定输入（种子）的情况下，它的输出也就确定了，相同的输入必然导致相同的输出，这类随机数生成器一般只依赖软件算法实现，对系统要求较低，应用范围广泛。

联邦数据交换过程中经常需要生成随机数和伪随机数，一些可由参与方独立生成的随机数应使用系统提供的真随机数生成函数生成。安全的伪随机数生成算法是除了满足统计学伪随机性外，还需要满足密码学安全伪随机性，即不能以显著大于 50%的概率在多项式时间内推算出序列的其它任何部分。

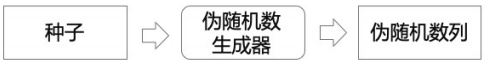


图 7.3 伪随机生成器结构

如图 7.3，安全伪随机数生成器(pseudo-random number generator: PRNG)通过输入一个种子，生成密码学安全的伪随机数列。伪随机数的种子是一串随机的比特序列，根据种子就可以生产出专属于自己的伪随机数列。伪随机数生成器是公开的，但种子是需要自己保密的，这就好像密码算法是公开的，但密钥只能自己保密。由于种子不可以被攻击者知道，因此不可以使用容易被推测的值，例如不可以用当前时间作为种子。

FLEX 协议中多处使用了该算法，如在基于一次一密的安全聚合中，加密所需的密钥是通过安全伪随机数生成算法生成；样本过滤中，使用伪随机数生成算法生成替换所需的比特串。NIST.SP.800-90 标准中规定了四种安全伪随机数生成算法 Hash_DRBG，HMAC_DRBG，CTR_DRBG 和 Dual_EC_DRBG。目前 FLEX 协议中采用的是 HMAC_DRBG[7]。

7.4 一次一密

一次一密(one-time pad)是一种理想的加密方案，该方案中每个消息用不同的密钥加密，每个密钥只使用一次。一次一密会使用乱码本记录一个大的不重复的真随机密钥集。实际应用中，发送方会对所发消息加密，然后销毁乱码本中用过部分。接收方有一个同样的乱码本，并依次使用乱码本上的每个密钥去解密密文，

接收方在解密消息后销毁乱码本中用过的部分。新的消息则用乱码本的新的密钥加解密。

FLEX 协议中使用一次一密时，会先将待加密的数值转换为整数，然后与密钥相加或相减，解密时需要再将整数转换为浮点数。由于密钥需要参与方在线生成，所以一次一密通常要与 7.2 节的密钥交换协议和 7.3 节的安全伪随机生成算法结合使用。

7.5 格式保留加密

格式保留加密(format preserving encryption: FPE)是一种可以保证密文与明文具有相同的格式与长度的加密方式。FPE 常用于数据去标识化或脱敏中，能保持明文和密文的格式相同，如英文加密后仍为英文，数字加密后仍为数字。而常规的分组密码，如 AES、SM4 算法，则不能保证密文与明文具有相同的格式与长度。身份证号码经过 AES 加密后长度会变长，并且不能保证密文保持身份证号的格式。

FLEX 中的格式保留加密采用了 NIST.SP.800-38G 标准中的 AES-FF1 算法[8]，将其当作一种随机置换算法，来生成布隆过滤器内的随机置换。为提高计算效率，选用了 base 为 2 的特例，来支持比特长度为 64 位以下的明文的加密。

7.6 布隆过滤器

布隆过滤器(bloom filter: BF)，是一种空间效率很高的随机数据结构，以比特数组的形式表示，用来判断一个元素是否存在集合内。布隆过滤器具有运行速度快、占用内存小的特点，因此常应用于海量数据的处理。

布隆过滤器并不适用“零错误”的场景，因为它是一个基于概率的数据结构，只能判断某一元素可能存在该集合中或肯定不存在该集合中。

样本过滤协议使用了布隆过滤器来表示用户 ID 集合，并且支持布隆过滤器之间按位与(&)和按位相等判断(==)运算。

7.7 不经意传输

不经意传输(oblivious transfer: OT)协议，是一种可保护隐私的双方通信协议，能使通信双方以一种选择模糊化的方式传输消息。协议有两个参与方，发送方和接收方。发送方拥有的消息数为n，公开的有安全参数p，安全坐标参数B和随机预言模型H，加法和乘法运算都基于特定的椭圆曲线进行。

尽管发送方有n条消息，但是执行协议后，接收方只能得到他想要得到的其中一条或多条消息。在整个过程中，发送方不能控制接收方的选择，发送方不知道接收方得到了哪几条消息，接收方也不能得到除了选择之外的其它消息。

OT 协议具有多种形式，根据收发消息数量的不同可以分为：$1-2$（2 选 1），$1-n$（n选 1）和$k-n$（n选k）的 OT 协议。OT 协议应用广泛，在样本对齐、安全多方计算等领域均有应用。FLEX 中的匿踪查询使用了$1-n$的 OT 协议，具体参见文献[9]。

7.8 秘密分享

秘密分享(secret sharing: SS)是一种共享秘密的技术，通过某种方法将秘密拆分，从 N 个信道同时发送，即使有信道存在恶意者，也无法恢复秘密。秘密分享能在计算前后始终保持秘密在参与方之间分享，并且在计算过程中不会泄漏参与方的敏感数据。

秘密分享在 FLEX 中主要应用在多头共债和联邦矩阵计算协议中。这里提供的秘密分享支持任意数量参与方之间的秘密分享，还支持加法、乘法、点乘、比较等常用运算。

7.9 其它密码算法

FLEX 的底层还使用了许多其它经典的密码算法，如对称密码、非对称密码、Hash 函数等。具体地，

- 对称密码主要使用 AES[10]、SM4[11]；
- 非对称密码中的椭圆曲线密码算法实现参考 ANSI X9.63[12]和 SM2[13]；
- Hash 函数主要使用 MD5[14]、SHA1[15]、SHA256[16]、SM3[17]。

7.10 小结

公共组件为 FLEX 上层应用协议提供了基础支撑，前面章节介绍的应用协议都部分采用了这里的公共组件，具体对应关系可以参照表 7.10。

表 7.10 FLEX 应用协议与公共组件的对应

章节	协议	同态加密	密钥交换	伪随机数	一次一密	格式保留	布隆过滤器	不经意传输	秘密分享	其它
2.1	OT-INV							√		√
2.2.1	BF-SF			√		√	√			√
2.2.2	SAL		√							√
3.1	HE-DT-FB	√								
3.2.1	FMC								√	
3.2.1	IV-FFS	√								
4.1.1	HE-ML	√								
4.1.2	SS-ML								√	

5.1	HE-Linear-FT	√								
5.2.1	HE-OTP-LR-FT1	√			√					
5.2.2	HE-OTP-LR-FT2	√			√					
5.3	OTP-NN-FT				√					
5.4.1	HE-GB-FT	√								
5.4.2	CS-GB-FT									√
5.5.1	OTP-SA-FT		√	√	√					√
5.5.2	HE-SA-FT	√								
6.1	HE-LR-FP	√								

8.联邦安全性

在对现代密码方案的安全性进行评估时，有信息论安全和计算安全两种安全性级别。虽然信息论安全的安全级别看似比计算安全要高，但是考虑到方案的可应用性时，人们往往选择了具有计算安全性的方案。信息论安全是指既使攻击者拥有无限的时间和计算能力，他们也没有足够的“信息”实现攻击。

计算安全指从理论上可以破解，但攻击者在有限的攻击时间和计算能力下无法成功实施攻击。当然，如果超出方案所限定的攻击时间或计算能力，攻击者是可以攻破方案的；但是，此时的攻击时间和计算能力往往是攻击者所无法承受的。本书中介绍的 FLEX 协议具有计算安全性。

事实上，联邦安全性不仅仅包括联邦协议的安全性，还包括联邦平台的安全性。联邦协议只是集成在联邦算法中的一部分，而实际的应用中会对算法进一步封装形成联邦平台。联邦平台的安全性会涉及到更多方面，从数据导入到平台，到数据存储到模型训练，以及模型在生产环境进行预测，每一个环节都存在安全隐患。而且应用行业不同，数据敏感度分级不同，隐私保护的要求也不同，尤其是金融、医疗行业安全要求最高。总之，联邦安全性应包括联邦平台和联邦协议的安全性，同时还要充分考虑行业规范要求。

8.1 联邦安全矩阵

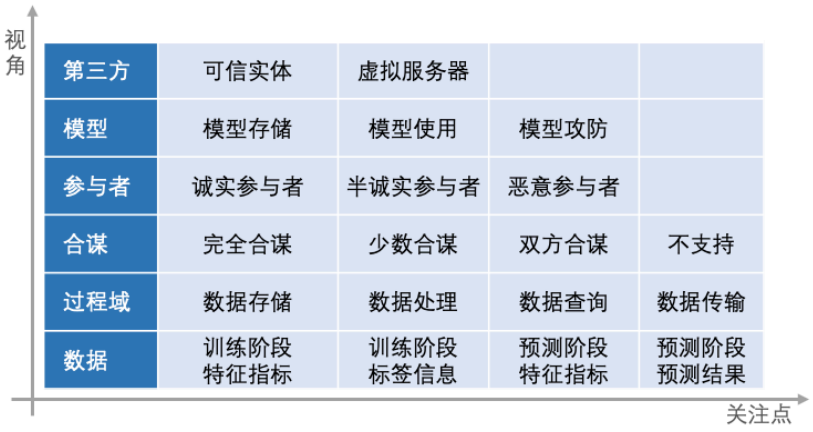


图 8.1 联邦安全矩阵

从不同的角度看联邦安全，关注的问题点会完全不同。这里我们提供了一个联邦安全矩阵(图 8.1)，总结了在不同视角下联邦安全的关注点。

- 数据：从数据性质角度，用户普遍关心的训练阶段特征指标和标签信息是否安全，预测阶段特征指标和预测结果是否存在泄漏；
- 过程域：在数据应用过程中，主要关注数据存储、数据处理、数据查询、数据传输过程的安全性；

- 参与者：从参与者角度，联邦更关心合作方是属于诚实参与者、半诚实参与者或恶意参与者；
- 合谋：针对恶意参与者，联邦模型是否能抵抗完全合谋、少数合谋、双方合谋还是完全不支持合谋；
- 模型：从模型维度，会涉及到模型存储、模型使用、模型攻防中的安全性；
- 第三方：从第三方的角度，联邦参与方更关心可信第三方到底是一个怎样的实体机构还只是一个虚拟服务器。

基于联邦安全矩阵分析，可知 FLEX 协议解决的是数据处理、数据查询和数据传输过程中特征指标、标签和预测结果的安全和隐私保护问题，应用协议均假设参与者为半诚实参与者，并支持抗合谋攻击。FLEX 中的第三方可以是一个虚拟服务器，而出于安全考虑，虚拟服务器的安全性以及管理控制方式就至关重要；而可信实体机构一般会由行业内有社会责任感和公信力的企业机构担任。而其它安全问题诸如数据存储、模型存储、模型使用等则应该是联邦平台关注的重点，与 FLEX 无关。

8.2 联邦协议安全性

借鉴 Goldreich 的安全性定义[18]，将联邦参与者分为诚实参与者、半诚实参与者和恶意参与者。在整个协议执行过程中，

- 诚实参与者：对协议完全“遵纪守法”，不存在提供虚假数据、泄漏、窃听和中止协议的行为；
- 半诚实参与者：虽然会按照要求执行各个步骤，不存在提供虚假数据、中止协议等行为，但是他们会保留所有收集到的信息以便推断出其他参与者的秘密信息；
- 恶意参与者：完全无视协议执行要求，他们可能存在提供虚假数据、泄漏他们收集到的所有信息、窃听甚至中止协议等行为。

根据参与者的不同，联邦参与者模型分为半诚实模型和恶意模型。半诚实模型下，协议的参与者仅包含诚实参与者和半诚实参与者。如果恶意参与者参与协议的执行，则此类模型称为恶意模型。

联邦协议安全性可以直观地理解为，对于一个半诚实参与者，如果可以利用自己的输入与协议的输出通过单独模拟整个协议的执行过程而得到在执行协议过程他所能得到的任何信息，那么协议就能保证输入的隐私性；对于一个恶意参与者，如果可以直接利用协议的输出通过单独模拟整个协议的执行过程而得到协议过程中他所能得到的任何信息，那么协议就能保证输入的隐私性。

如果一个联邦协议能被这样模拟，参与者就不能从协议的执行过程中得到有价值的信息，这样的联邦过程就是安全的。和加密方案的安全性定义类似，不论是半诚实模型还是恶意模型，攻破指的是攻击者利用他/她所得到输出信息和中间信息推导出其他参与者的输入隐私数据。

作为一种联邦协议，FLEX 中的大部分应用协议也都是可以保障半诚实参与者的数据安全的，即：执行 FLEX 协议后，除了协议的执行结果外没有任何信息泄露。不过，FLEX 中有两个协议，匿踪查询和样本过滤，不能严格满足半诚实参与者的数据安全。匿踪查询实际上是将待查询的用户 ID 信息隐藏在n个查询 ID 中，将 ID 信息的暴露概率从 1 降低为$1/n$。样本过滤是用于过滤大部分的非交集 ID，从参与方的大规模数据中挑选出部分可用于后续样本对齐的数据。样本过滤可能会泄漏部分能对齐的 ID 信息，但是泄漏概率比较小。

8.3 安全与效率的平衡

如上所述，计算安全在理论上仍然是可以破解的，只是要求攻击方具有非常强的计算和存储能力，攻击代价是巨大的。同样，安全性的提升也会导致数据加解密的速度变慢，进而影响联邦算法的效率。

与非联邦算法相比，影响联邦算法效率的因素更多。值得关注的几个因素包括：

- 加解密的速度：取决于算法和密钥的选择；
- 密文的处理速度：尤其是同态加密需要在密文上运算；
- 密文的数据规模：加密后的数据存储量会发生变化；
- 网络和通信协议：参与方之间的信息传输方式。

FLEX 中的协议在选取密码算法前已经进行过大量的实验验证，能有效地支持联邦算法实施。协议中部分参数比如密钥长度等也可以根据需要灵活调整，以保证算法稳定高效的运行。此外，为了提升联邦效率，我们专门设计了一套的轻量级的联邦通讯框架——**Ionic Bond**。该框架支持联邦环境下点到点和域通讯等通讯模式，具有极高的网络并发性能；还支持超大数据一次性传输，不受硬盘读写性能影响。Ionic Bond 部署简单，开箱即用，适合容器化、k8s 部署，支持水平扩展，相关内容我们也会开放出来。

8.4 小结

本文阐述了联邦应用中涉及的数据安全交换协议。参与交换的敏感数据会随着联邦技术的发展而变化，协议也会逐步迭代升级，我们将在后续的版本中更新。随着 FLEX 协议的应用深入，为适应不同应用场景和效率要求，还需要设计不同安全等级的协议。甚至在面对恶意参与者或合谋攻击时，能提供更安全的标准保障各参与方的数据安全。

参考文献

[1] Hongyu Li, Dan Meng, Hong Wang, and Xiaolin Li. Knowledge Federation: A Unified and Hierarchical Privacy-Preserving AI Framework[J]. 2020 IEEE International Conference on Knowledge Graph (ICKG), Nanjing, China, 2020, pp. 84-91.

[2] 同盾科技人工智能研究院，《知识联邦白皮书》[R]，2020.

[3] 中华人民共和国，个人信息保护法(草案) [Z]，2020.10.

[4] Paillier P. Public-key cryptosystems based on composite degree residuosity classes[C]. International conference on the theory and applications of cryptographic techniques. Springer, Berlin, Heidelberg, 1999: 223-238.

[5] ElGamal T. A public key cryptosystem and a signature scheme based on discrete logarithms[J]. IEEE transactions on information theory, 1985, 31(4): 469-472.

[6] Gillmor D. Negotiated Finite Field Diffie-Hellman Ephemeral Parameters for Transport Layer Security (TLS)[J]. IETF RFC 7919, 2016.

[7] NIST Special Publication 800-90A. Recommendation for Random Number Generation Using Deterministic Random Bit Generators [OL]. https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-90a.pdf. Last accessed on 10/21/2020.

[8] NIST Special Publication 800-38G. Recommendation for Block Cipher Modes of Operation: Methods for Format-Preserving Encryption[OL]. https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-38G.pdf. Last accessed on 10/21/2020.

[9] Chou T, Orlandi C. The simplest protocol for oblivious transfer[C]. International Conference on Cryptology and Information Security in Latin America. Springer, Cham, 2015: 40-58.

[10] Daemen J, Rijmen V. Reijndael. The Advanced Encryption Standard[J]. Dr. Dobb's Journal: Software Tools for the Professional Programmer, 2001, 26(3): 137-139.

[11] GM/T 0002-2012: SM4 分组密码算法[OL]. http://www.gmbz.org.cn/main/viewfile/20180108015408199368.html. Last accessed on 10/21/2020.

[12] ANSI, "Public Key Cryptography for the Financial Services Industry: Key Agreement and Key Transport using Elliptic Curve Cryptography"[S], ANSI X9.63, 2001.

[13] GB/T 32918.(1-5)-2016: 信息安全技术 SM2 椭圆曲线公钥密码算法[OL]. http://www.gb688.cn/bzgk/gb/std_list?p.p1=0&p.p90=circulation_date&p.p91=desc&p.p2=32918. Last accessed on 10/21/2020.

[14] Rivest R. RFC1321: The MD5 message-digest algorithm(1992)[OL]. https://tools.ietf.org/html/rfc1321. Last accessed on 10/21/2020.

[15] FIPS PUB 180-1: Secure Hash Standard(1995)[OL]. https://nvlpubs.nist.gov/nistpubs/Legacy/FIPS/fipspub180-1.pdf. Last accessed on 10/21/2020.

[16] FIPS PUB 180-2: Secure hash standard (2002)[OL]. https://csrc.nist.gov/CSRC/media/Publications/fips/180/2/archive/2002-08-01/documents/fips180-2.pdf. Last accessed on 10/21/2020.

[17] GM/T 0004-2012: SM3 密码杂凑算法[OL]. http://www.gmbz.org.cn/main/viewfile/20180108023812835219.html. Last accessed on 10/21/2020.

[18] Goldreich O. Secure multi-party computation (manuscript version 1.4)[OL]. http://www.wisdom.weizmann.ac.il/~oded/PSX/prot.pdf. Last accessed on 10/21/2020.