《SoK》及相关内容简介

这篇文章是关于多方安全学习和联邦学习相关技术研究现状的综述，同时它对当前已有的一些方案和框架进行了调研。

摘要

目前，从具有隐私保护的多个数据控制器收集高质量的训练数据是训练高质量机器学习模型的一个关键挑战。这些潜在的解决方案可以打破孤立数据集之间的障碍，从而扩大可供处理的数据范围。为此，学术界研究人员和产业界最近提出了两个主流的解决方案：1）安全多方学习（简称MPL）；2）联邦学习（简称FL）。作者从隐私保护程度、通信方式、通信开销、数据格式、训练模型的准确性和应用场景等方面对这两种解决方案进行了全面评估。

为了展示研究进展并探讨未来发展方向，作者还深入研究了MPL和FL的协议和框架。首先，作者定义了在多个数据源上训练具有隐私保护的机器学习模型的问题（简称TMMPP）。然后，作者从技术路线、支持的参与方数量、数据划分方式、威胁模型、支持的机器学习模型等方面比较了TMMPP的研究现状，指出了TMMPP的优势和局限性。接下来，作者介绍了支持多个数据源的在线训练的最新方案和框架。最后，作者讨论了解决TMMPP问题的潜在方向。

文章贡献

•给出了TMMPP问题的定义，并列出了解决TMMPP所面临的技术问题的挑战，包括统计挑战、效率挑战和安全挑战。

•调研了为解决TMMPP问题而做出的最新工作，并根据构建的底层技术对各种框架进行了分类。对于MPL的解决方案，作者将其分为四类：基于HE的MPL框架、基于gc的MPL框架、基于SS的MPL框架和基于混合协议的MPL框架。对于FL的解决方案，我们根据隐私机制的类型将FL框架分为三类：非加密FL框架、基于DP的FL框架和基于SC的FL框架。然后，作者分析了这两条路线之间的差异，以及各自的优势和劣势。

•调研了关于TMMPP解决方案的平台。对于开源平台，作者根据FL的类型、隐私机制等一系列指标对其进行比较。对于封闭源代码平台，作者根据它们的应用场景与其他平台进行比较。

•全面概述了TMMPP的历史，并讨论了其未来发展方向，包括隐私保护、效率优化、支持更多数据分区和类型以及系统实现。

**问题定义**

考虑以下场景：一组可能是半诚实甚至恶意的数据控制器计划在它们所拥有的数据上联合训练一个机器学习模型，并且过程保证安全。在训练期间，这些数据控制器与集中式服务器之间可能会进行通信。但是，它们不会将原始数据上传到集中的一方（或其他数据控制器）。

在这里，我们给出了上述问题的正式定义。在不丧失一般性的情况下，设分别为可能相互不信任的n个数据控制器所持有的原始数据集。我们用表示数据控制器协同训练的机器学习模型。

图1、TMMPP架构图

然后，TMMPP的问题可以表述为：

输入：每个数据控制器以其拥有的原始数据作为输入。

输出：一个由所有数据控制器联合训练的全局模型，在训练过程中不会向其他人公开任何数据控制器的原始数据的任何信息。

具体地说，在MPL的解决方案中，一组数据控制器在其输入上协同计算一个函数而不透露任何信息，包括计算结果之外的中间结果。此外，MPL方案可以潜在地保持模型的私有性。在FL的求解中，目标是在每个数据控制器的数据都在本地存储和处理的约束下学习全局机器学习模型，并且只有模型参数通过一个集中式服务器进行通信。集中式服务器和数据控制器都可能不受信任。

注意，在本文中，MPL和FL两个系列解决方案中涉及的数据控制器都是输入方和计算方。然而，MPL的解决方案中还有其他一些设置，可以将计算外包给少数计算方。而在FL的实际业务场景中，数据控制者可能会将自己的数据保密，但会获得经过训练的模型，即该数据控制器不是计算方。

**挑战**

在解决TMMPP问题时，作者总结了三个关键挑战，即统计挑战、效率挑战和安全挑战。

（一）统计挑战：数据控制器保存的数据通常以非IID（非独立同分布）的方式生成或收集的。此外，分布在数据控制器上的数据量也可能有很大的变化，换句话说，数据是不平衡的。非IID和不平衡的数据使得训练一个高质量的机器学习模型变得困难，并且可能增加分析和评估的复杂性。例如，占总数据量很大一部分的数据控制器将在集中式服务器聚合模型参数阶段起决定性作用，这将影响模型的性能。

（二）效率挑战：效率是TMMPP的一个重要瓶颈，包括通信开销和计算复杂度。在多个数据源上训练机器学习模型涉及大量的数据控制器，例如，在FL场景中，可能会有数百万个数据控制器。由于保护每个数据控制器隐私的安全性要求，它们之间的通信比原始数据的本地计算带来了多个数量级的额外开销。此外，在MPL框架中，通信和计算的开销很大程度上取决于底层协议。例如，基于同态加密的协议通常会导致较高的计算复杂度，而基于GC的协议通常会导致昂贵的通信开销。一般来说，为了提高MPL框架的效率，需要在通信开销和计算复杂度之间进行权衡。

（三）安全挑战：不能完全信任每个数据控制器和集中式服务器。他们中的一些人可能是敌手，他们可以以某种方式对私有信息进行攻击或干扰训练算法的正常执行。此外，FL通过交换模型参数（如局部梯度）而不是原始数据来保护每个数据控制器的数据。然而，如果对这些梯度进行合理的操作，这些梯度可能会泄露原始数据的敏感信息，从而导致集中式服务器在聚合这些梯度时出现隐私泄露。

在本文中，作者考虑了两种威胁模型：半诚实模型和恶意模型：（1）半诚实（也称为诚实但好奇或被动）的对手试图从协议执行过程中他们无权获得的数据中获取尽可能多的信息。但它们不会偏离协议规范。（2） 恶意（也称为主动）对手可以任意破坏协议，例如向其他数据控制器发送不正确的消息。在这个模型中实现安全性的协议可以提供非常高的安全性保证。

**基于同态的MPL框架**

同态是一种加密形式，可以直接对密文执行特定的代数运算，而不需要解密它，也不知道任何有关私钥的信息。然后它生成一个加密结果，其解密结果与对明文执行的相同操作的结果完全匹配。

同态可以分为三种类型。单同态加密（PHE）只允许一种类型的操作（加法或乘法），次数不限。一个著名的算法是Paillier方案的加法同态加密（AHE），它只能执行加法运算。为了在密文之间进行加法和乘法操作，可以使用somewhat同态加密（SWHE）和全同态加密（FHE）。somewhat同态加密由于随着操作次数的增加，密文中的噪音也会不断增加，当噪音达到一定程度后会导致无法正确解密，因此它只可以在有限的次数内完成某些类型的操作（目前Gentry提出的Bootstrapping方法通过对密文进行重加密，可以一定程度的缓解噪音的问题），而全同态加密可以不受次数限制地处理所有操作。然而，FHE的计算复杂度要比SWHE和PHE昂贵得多。

图2、同态加密结构

这张图所示的就是基于同态加密的安全的双方计算，一方为同态密码系统生成一个密钥对（，），并将公钥连同他加密之后的消息一起发送给另一方，先将自己要参与运算的消息加密，再利用密码系统的同态属性完成运算操作。最后，将还是密文状态的结果发回，可以使用它的私钥进行解密。由于未加密的数据本身不被传输，也不可能被其他方猜到，因此泄露原始数据的可能性很小。注意，非线性功能，如ReLU，Sigmod激活函数不能被HE支持。

传统的 MPC 技术路线分为两类，分别针对两类不同的电路系统。 一类叫布尔电路，指的是每一个计算单元的表示形式是布尔门（与门、或门、非门）；这类布尔电路，会有一套专门的 MPC 密码协议来处理。另一类叫算术电路，指电路完全由加法和乘法组成。出于性能效率的原因，不同的电路类型任务需要用选用不同的密码协议来实现。

安全两方计算所使用的协议一般为混淆电路（GC）结合不经意传输（OT）；而安全多方计算（三方或者三方以上）所使用的协议一般为秘密分享（SS）结合不经意传输。前者（GC+OT）主要的问题在于计算开销会比较大，而且一般比较适合于做两方之间的隐私计算。它的好处在于需要的通讯轮数比较少。后者的问题在于通常需要迭加多轮 OT，会引入非常高的通讯轮数；它的好处在于计算开销比较小。对于网络要求比较高的场景里面不太适合用这种 SS+OT 的方法。

**混淆电路**

是姚期智最初提出的一种安全两方计算的底层技术，任何可计算问题都可以转换为一个个电路，于是就有了加法电路、比较电路和乘法电路等等。当然，更复杂的计算过程，如深度学习等等，也是可以转换成电路的。一个电路是由一个个门（gate）组成的，比如与门、非门、或门、与非门等等。每个门包括两个输入线（input wire）和一个输出线（output wire）。混淆电路就是通过加密和扰乱这些电路的值来掩盖信息的。在最经典的混淆电路中，加密和扰乱是以门为单位的。GC提供了一个交互协议，供两方（一个加扰器和一个求值器）不经意地计算一个用布尔电路表示的任意函数。

图3,4、OT过程

经典GC的构建包括三个主要阶段：混淆、转移和估值。首先，对于电路的每条线，加扰器生成两个随机字符串标签和，分别表示该线的两个可能的位值“0”和“1”。对于电路中的每个门，加扰器创建一个真值表。真理表的每个输出都使用与其输入相对应的两个标签进行加密。这是由加扰器选择一个密钥派生函数来完成的，该函数使用两个标签生成对称密钥。然后加扰器会重新排列真相表的每一行，也就是混淆操作。在混淆阶段之后，加扰器将被乱码的表以及与其输入位对应的输入线标签一起传送给求值器。求值器通过不经意的传输安全地获取与输入相对应的标签。对于输入线的乱码表和标签，求值器负责迭代地解密乱码表，直到得到函数的最终结果。

**GC实例：以与门为例**

图5,6、与门&真值表

随机生成 6 个密钥{k\_1^0,k\_1^1,k\_2^0,k\_2^1,k\_3^0,k\_3^1}，分别表示𝑤1,𝑤2,𝑤3这三条线为0和1时的两种情况。然后利用对称加密算法En()生成4个密文c\_0,0,c\_0,1,c\_1,0,c\_1,1，〖En〗\_(a,b) (c) 表示用𝑎, 𝑏作为加密秘钥，使用加密算法En()来加密𝑐。

图6,7、混淆过程

Alice将，打乱顺序，在电路门储存这四个乱序的值，记为𝑐1,𝑐2,𝑐3,𝑐4【混淆电路的混淆就是这样来的】，并将该表发送给Bob

Bob的解密过程：Alice已经把自己的输入对应的key发给Bob，比如Alice的输入是0，那就发 ，输入是1就发 。同时把和Bob有关的key都发给Bob，也就是 和 。然后Bob根据自己的输入挑选相关的key。由于Bob收到的这些key都是随机数，所以并没有任何有效信息泄露。假设现在Bob根据收到的 和自己的 ，对上述加密表的每一行尝试解密，最终只有一行能解密成功得到，即为该门的输出值。

GC协议的计算和通信开销仅取决于电路中与门的总数，因为XOR门的局部计算可以忽略。不管电路的深度和功能如何，GC都会以固定的轮数执行。然而，基于GC的协议会采用复杂的机制将函数转换为布尔电路。这些协议逐位计算一个函数，这会导致多个操作的通信开销，尤其是乘法操作。近年来，研究者们提出了许多不同的原始GC协议来提高GC的性能。Beaver等人[49]提出的点和置换机制减少了加密次数。Kolesnikov[50]提出的自由异或方法提高了异或门的计算效率，并减小了混淆电路的规模。Pinkas等人[51]提出了一种称为4-2 GRR的混淆行缩减方法，将乱码表的大小从4个密文缩减为2个密文，这与自由异或不兼容。半门使用与门中的自由异或将密文的数量从4个减少到2个。

**不经意传输OT**

我们假设 Alice 有两个数值和，Bob 想知道其中的一个 。通过执行 OT 协议，Bob 知道了，但不知道；同时，Alice 不知道。

图7、OT2过程

除了上述实现，OT 还有多种实现方法。

**通过构建**

图8、OT2到n过程

安全性分析：Bob 的安全性可直接继承自的安全性，即保证了 Alice 无从知晓 Bob 在每一轮选择了前者还是后者，所以 Alice 无从知晓的真实值。Alice的安全性可以通过分类讨论的方法来论证。在第轮中获取的是，从而不会获得任何的信息。在第轮获得的是和到依次异或的值，也不会泄露的信息。

秘密分享：SS方案是密码学中的重要工具，在许多安全协议中被用作构建块。基于SS的协议，如Additive SS[6]、Shamir的SS[60]、GMW（Goldreich Micali Wigderson）[61]、BGW（Ben Or Goldwasser Wigderson）[62]和SPDZ[63]，适用于多方。秘密共享（SS）将秘密以适当的方式分为n 份，每一份由不同的管理者持有，每个参与者无法单独恢复秘密，只有达到指定数目的参与者才能恢复秘密。

**GMW方案**

GWM是基于混淆电路构建的，

图9,10、GMW过程

**Shamir’s SS**

图11、Shamir过程

**BGW**

Shamir(k, n)协议可以看成是在有限域上用k个方程解k个未知数，那BGW协议是将Shamir(k, n)和电路计算进行了结合。

在Shamir(k, n)协议中当所有参与者的输入都通过 Shamir(k, n) 门限秘密共享机制分享后，每个参与者都掌握了协议输入的子秘密。假设一个门的输入分别为𝑎和𝑏，秘密𝑎和𝑏已经分别由秘密分配函数

参与者𝑃𝑖掌握𝑎和b的子秘密𝑎𝑖, 𝑏𝑖。在布尔电路上，可将异或门和与门分别看成在有限域𝐹2 上的加法和乘法。将异或用模为 2 的加法进行计算，与用模为 2 的乘法进行计算。

对于异或门：由于 Shamir SS具有加法同态性，因此

图12,13、异或门&加法

假设，则，而和都由𝑃𝑖掌握，因此𝑃𝑖可以本地计算出。当所有的计算完成后，每个参与者𝑃𝑖公布自己计算出的，即可恢复出和。

图13,14、与门&乘法

对于与门，每个参与者𝑃𝑖计算𝑑𝑖=𝑎𝑖𝑏𝑖，接着每个𝑃𝑖独自选取次数为 次的随机多项式ℎ𝑖(𝑥)，且满足ℎ𝑖(0)=𝑑𝑖,1≤𝑖≤𝑛， <𝑛/2。

向各个参与者分配𝑑𝑖，且𝑐𝑗𝑖= ℎ𝑖(𝑗),1≤𝑖,𝑗≤𝑛。所有参与者分配结束后，𝑃𝑖掌握了信息𝑐1𝑖,𝑐2𝑖,…,𝑐𝑛𝑖，（这一步会产生较大的通信开销）然后再利用公开的重组向量𝜆1,…,𝜆𝑛，𝑃𝑖计算

𝑐𝑖=

此时𝑃𝑖掌握的子秘密𝑐𝑖即为𝑎𝑏的子秘密。当所有计算完成后，每个参与者公开自己的子秘密，再根据之前叙述的 Shamir(k, n) 门限秘密重构算法即可获得𝑎𝑏。

**基于混合协议的MPL框架**

除了上述的单协议MPL框架外，一些常用的框架通常采用混合协议，它们将两个或多个协议结合起来，以发挥各自的优势，避免各自的缺点。例如，将HE和GC结合在一起的混合协议背后的基本思想是使用HE运算高效地表示算术电路（例如，加法和乘法）的运算，以及使用GC高效地表示布尔电路（例如，比较）的运算。然而，不同方案之间的转换也并非易事，成本也相对昂贵。此外，还有一些框架将MPC与差分隐私相结合（简称DP）

**联邦学习FL**

虽然数据控制器的本地数据不在FL框架中交换，但是服务器和数据控制器之间传输的参数可能会泄露敏感信息。为了保护数据控制器的本地数据不被泄露，以及在训练过程中保护中间数据的隐私，在FL框架中应用了一些隐私技术，以实现数据控制器与服务器交互时隐私地交换参数。本文根据FL框架中的隐私保护机制，将FL框架分为非加密FL框架、基于DP的FL框架和基于SC（安全计算）的FL框架。

**（一）非加密FL框架**：有很多FL框架专注于提高效率或解决统计异质性的挑战，而忽略了交换明文参数带来的潜在风险，即遵循Google提出的原始隐私保护思想。

**（二）基于差分隐私DP的FL框架**：如果我们知道A医院，今天就诊的100个病人，其中有10个肺病，并且我们知道了其中99个人的患病信息，就可以推测剩下一个人是否患有肺病。这种窃取隐私的行为叫做差分攻击。差分隐私是防止差分攻击的方法，通过添加噪声，使得对手获得的数据对于推测同一个数据集中的其他数据几乎毫无用处。通过在原始数据或模型参数中加入随机噪声，DP为单个记录提供统计隐私保证，从而使数据无法恢复，从而保护数据控制者的隐私。

那么对于一个随机化算法A（所谓随机化算法，是指对于特定输入，该算法的输出不是固定值，而是服从某一分布），其分别作用于两个相邻数据集得到的两个输出分布难以区分。



对于两个差别只有一条记录的数据集D,D′，通过随机算法A，输出为结果集合O的概率满足上面公式，ϵ为差分隐私预算，δ 为扰动， ϵ,δ越小，K在D,D′上输出的数据分布越接近。

如果该算法作用于任何相邻数据集，得到一个特定输出 O的概率应差不多，那么我们就说这个算法能达到差分隐私的效果。也就是说，观察者通过观察输出结果很难察觉出数据集一点微小的变化，从而达到保护隐私的目的。

**基于差分隐私DP的FL框架分类**

在FL框架中，DP的应用可分为两类：DP的中心模型（简称CDP）和DP的局部模型（LDP）。如图3所示，在CDP设置中，每轮生成的模型更新参数首先由集中式服务器聚合，然后由集中式服务器进行扰动。在这种结构下模型参数对除集中式服务器之外的所有其他数据控制器都是私有的，即服务器必须是可信的。在LDP设置中，每个数据控制器在与集中式服务器共享数据之前，对其数据进行差异私有化转换。因此，模型参数对其他数据控制器和服务器都是私有的，允许服务器是不诚实的。与CDP相比，LDP提供了更强的隐私保障，但在保持效用的同时实现这一点是一个挑战。此外，在隐私性和模型精度之间存在一种折衷，因为添加更多的噪声可以提供更大的隐私保障，但可能会严重损害模型精度。

**（三）基于安全计算SC的FL框架**

安全计算方法如HE和MPC在FL框架中得到了广泛的应用，它们只向参与方透露计算结果，而不透露任何额外的信息。

同态应用于FL框架的方法与MPL框架的方法类似，只是有些细节稍有区别。在FL框架中，同态用于保护参与方和服务器之间交互的模型参数（如梯度）的隐私，而不是在MPL框架中应用的参与方和服务器之间交互的数据。

MPC涉及多方，并且在非常高的安全保证下保持了最初的准确性。它保证了每一方除了结果外什么都不知道，并且可以应用到FL模型中，以实现安全聚合和保护本地模型。在基于MPC的FL框架中，集中式服务器不能获得任何本地信息和本地更新，只能在每一轮中观察精确的聚合结果。然而，应用于FL框架的MPC技术将产生大量额外的通信和计算成本。目前，SS作为MPC中的一种基本协议，在FL框架中应用最为广泛，尤其是Shamir的SS。

**技术路线比较**

首先，从隐私保护、通信方式、通信开销、数据格式、训练模型的准确性和应用场景等方面讨论了MPL和FL的优缺点。在表一中总结了TMMPP不同技术路线的特点。

•隐私保护。MPL框架中使用的MPC协议为各方提供了很高的安全保证。而非加密FL框架在数据控制器和服务器之间交换的模型参数是明文的，敏感信息也可能泄露。

•沟通方式。MPL中的数据控制器之间的通信通常是对等的，没有可信的第三方，而FL通常是客户机-服务器的结构，有一个集中的服务器。换句话说，MPL中的每个数据控制器在状态上是相等的，而FL中的数据控制器和集中式服务器是不相等的。

•通信开销。由于FL结构中数据控制器之间的通信可以由一个集中的服务器来协调，因此通信开销小于MPL的对等形式，特别是当数据控制器的数量非常大时。

•数据格式。目前，在MPL的解决方案中，没有考虑非IID设置。然而，在FL的求解中，由于每个数据控制器都对模型进行局部训练，因此更容易采用非IID设置。

•训练模型的准确性。在MPL中，全局模型的精度通常没有损失。但是如果FL使用DP来保护隐私，那么全局模型通常会有一定的准确性损失。

•应用场景。结合以上分析，我们可以发现MPL更适合于安全性和准确性更高的场景，而FL更适合对更多数据控制器性能要求更高的场景。

**方案比较**

作者研究了14个MPL框架和11个FL框架，它们都实现了具有隐私保护的多方训练机器学习模型。作者将这些MPL框架进行了分组。

通过观察和对比这些框架我们可以发现：

•MPL框架通常使用定制的协议，只支持一定数量的参与者，即可扩展性差。此外，目前框架中涉及的参与方一般为两方或三方，当参与方较多时，甚至数十方，框架的效率会急剧下降。

•非加密FL框架交换原始模型参数时没有任何防止泄漏敏感信息的安全保证。在机器学习模型的训练过程中，如果中间结果受到强大的攻击，数据控制器的隐私就会受到威胁。

•现有的框架（包括MPL框架和FL框架）通常考虑水平数据分区的情况，而很少有研究考虑垂直数据分区的情况。我们认为，在数据处理框架中，数据的横向分割比纵向分割相对容易。此外，那些声称支持垂直数据分区的框架只能训练简单的机器学习模型，如线性回归。考虑到垂直数据分区在现实世界中也很常见和重要，因此在垂直数据分区中有效地训练复杂的机器学习模型仍然是有希望的。

•目前的框架大多是针对参数模型的训练，如线性回归、logistic回归、神经网络等，少数框架是针对非参数模型的训练，如决策树等。在现实世界中，这些非参数模型也得到了广泛的应用，因此扩展这些框架以支持更多的非参数模型也是未来一个有趣的方向。

**框架比较**

作者还详细地介绍了八个平台。5个开源、3个闭源。

（一）FATE：据我们所知，FATE（federed AI Technology Enable）是WeBank人工智能部门于2019年2月提出的第一个开源工业级FL框架。它使多个公司和机构能够有效地协作，训练符合数据安全和数据保护法规的机器学习模型。

FATE提供算法级api详细的安装和使用文档，供用户直接使用。它提供了一个安全的计算框架来支持各种机器学习算法，如线性回归、logistic回归、boosting-tree[15]、NNs等，这些算法支持异构和同质样式。FATE还提供了各种模型评估，包括二进制分类、多重分类、回归评估和局部与联邦比较。目前，FATE支持所有三种联邦学习体系结构，包括垂直联邦学习、水平联邦学习和联邦迁移学习。此外，FATE支持独立部署和集群部署。

（二）TFF：TFF（tensorflowfederated）由Google开发，是一个用于联邦机器学习和其他分散数据计算的开源框架。TFF为所有TensorFlow用户提供了一个灵活、开放的框架，用于本地模拟分散计算。它使开发人员能够在他们的模型和数据上模拟所包含的FL算法，以及对新算法进行实验。TFF还支持非学习计算，例如分散数据上的聚合分析。TFF提供了两个不同层次的API：联邦学习（FL）API和联邦核心（FC）API。FL API提供了一组高级接口，将现有的Keras或非Keras机器学习模型插入到TFF框架中。通过FL-API，用户可以对现有的TensorFlow模型进行FL评估，而不必研究FL算法的细节。FC API具有一组较低级别的接口，是框架的核心，也是构建FL的基础。接口通过在强类型函数编程环境中将TensorFlow与分布式通信操作符相结合来简洁地表达自定义联合算法。

使用TFF，开发人员可以声明性地表示联邦计算，因此可以将它们部署到不同的运行时环境中。然而，目前发布的TFF的最新版本只支持水平FL，没有底层隐私技术（如HE、MPC和DP）来保护数据安全。因此，TFF只适用于实验测试和仿真，不能在实际环境中部署。此外，TFF只支持多机单机模拟训练模型，不支持集群部署。

（三）PaddleFL：PaddleFL是在Baidu开发的机器学习框架PaddlePaddle的基础上开发的一个开源FL框架。它主要是为深度学习而设计的，在计算机视觉、自然语言处理、推荐等领域提供了多种外语学习策略和应用。

PadleFL为研究人员提供了一个基本的编程框架，并封装了一些公共FL数据集，研究人员可以轻松地复制和比较不同的FL算法。借助于丰富的模型库和预先训练的模型，能够很容易地在分布式集群中部署联邦学习系统。在PadleFL的设计中，它基于ABY3实现了安全的训练和推理任务，并使用DP作为隐私机制之一。目前，PadleFL支持Kubernetes轻松地部署它，并开放一个相对完整的水平联邦学习版本，但是垂直联合学习和转移联邦学习仍然处于早期阶段。

（四）PySyft公司：PySyft 是一个开源的Python库，用于FL和隐私保护，它的第一个版本实现由领先的分散式人工智能平台OpenMined领导。

PySyft是一个灵活、易于使用的库，能够在深度学习模型上执行私有和安全的计算。PySyft为开发人员提供实现算法的接口。它使用FL、安全计算技术（如MPC和HE）以及隐私保护技术（如DP）在不同的深度学习框架（如asiPyTorch、Keras和TensorFlow）中，将私有数据与模型训练分离。

（五）coMind：coMind是一个开源项目，旨在基于TensorFlow联合训练具有隐私保护功能的机器学习模型。

它开发了一个自定义优化器，实现了张量流的联合平均，可以方便地训练神经网络。此外，它还提供了一系列的教程和例子来帮助用户如何使用TensorFlow和FL。它提供了两种类型的例子，包括三个基本例子和三个高级例子，分别介绍了如何以本地、分布式和联邦的方式训练和评估TensorFlow机器学习模型。在这个项目中，消息传递接口和python sockets都可以实现联邦平均。但是，与TFF类似，coMind不提供任何加密方法。