Advances and Open Problems in Federated Learning

Peter Kairouz[7*] H. Brendan McMahan[7*] Brendan Avent[21] Aurélien Bellet[9]
Mehdi Bennis[19] Arjun Nitin Bhagoji[13] Kallista Bonawitz[7] Zachary Charles[7]
Graham Cormode[23] Rachel Cummings[6] Rafael G.L. D'Oliveira[14]
Hubert Eichner[7] Salim El Rouayheb[14] David Evans[22] Josh Gardner[24]
Zachary Garrett[7] Adrià Gascón[7] Badih Ghazi[7] Phillip B. Gibbons[2]
Marco Gruteser[7,14] Zaid Harchaoui[24] Chaoyang He[21] Lie He [4]
Zhouyuan Huo [20] Ben Hutchinson[7] Justin Hsu[25] Martin Jaggi[4] Tara Javidi[17]
Gauri Joshi[2] Mikhail Khodak[2] Jakub Konečný[7] Aleksandra Korolova[21]
Farinaz Koushanfar[17] Sanmi Koyejo[7,18] Tancrède Lepoint[7] Yang Liu[12]
Prateek Mittal[13] Mehryar Mohri[7] Richard Nock[1] Ayfer Özgür[15]
Rasmus Pagh[7,10] Hang Qi[7] Daniel Ramage[7] Ramesh Raskar[11]
Mariana Raykova[7] Dawn Song[16] Weikang Song[7] Sebastian U. Stich[4]
Ziteng Sun[3] Ananda Theertha Suresh[7] Florian Tramèr[15] Praneeth Vepakomma[11]
Jianyu Wang[2] Li Xiong[5] Zheng Xu[7] Qiang Yang[8] Felix X. Yu[7] Han Yu[12]
Sen Zhao[7]

[1]Australian National University, [2]Carnegie Mellon University, [3]Cornell University,
[4]École Polytechnique Fédérale de Lausanne, [5]Emory University, [6]Georgia Institute of Technology,
[7]Google Research, [8]Hong Kong University of Science and Technology, [9]INRIA, [10]IT University of Copenhagen,
[11]Massachusetts Institute of Technology, [12]Nanyang Technological University, [13]Princeton University,
[14]Rutgers University, [15]Stanford University, [16]University of California Berkeley,
[17] University of California San Diego, [18]University of Illinois Urbana-Champaign, [19]University of Oulu,
[20]University of Pittsburgh, [21]University of Southern California, [22]University of Virginia,
[23]University of Warwick, [24]University of Washington, [25]University of Wisconsin–Madison

摘要

Federated learning (FL) is a machine learning setting where many clients (e.g. mobile devices or whole organizations) collaboratively train a model under the orchestration of a central server (e.g. service provider), while keeping the training data decentralized. FL embodies the principles of focused data collection and minimization, and can mitigate many of the systemic privacy risks and costs resulting from traditional, centralized machine learning and data science approaches. Motivated by the explosive growth in FL research, this paper discusses recent advances and presents an extensive collection of open problems and challenges.

联邦学习 (FL) 是一种机器学习设置，其中许多客户端（例如移动设备或整个组织）在中央服务器（例如服务提供商）的编排下协作训练模型，同时保持训练数据分散。 FL 体现了集中数据收集和最小化的原则，

*Peter Kairouz and H. Brendan McMahan conceived, coordinated, and edited this work. Correspondence to kairouz@google.com and mcmahan@google.com.

可以减轻由传统的集中式机器学习和数据科学方法导致的许多系统性隐私风险和成本。受 FL 研究爆炸性增长的推动，本文讨论了最新进展，并提出了大量未解决的问题和挑战。

目录

1 Introduction

Federated learning (FL) is a machine learning setting where many clients (e.g. mobile devices or whole organizations) collaboratively train a model under the orchestration of a central server (e.g. service provider), while keeping the training data decentralized. It embodies the principles of focused collection and data minimization, and can mitigate many of the systemic privacy risks and costs resulting from traditional, centralized machine learning. This area has received significant interest recently, both from research and applied perspectives. This paper describes the defining characteristics and challenges of the federated learning setting, highlights important practical constraints and considerations, and then enumerates a range of valuable research directions. The goals of this work are to highlight research problems that are of significant theoretical and practical interest, and to encourage research on problems that could have significant real-world impact.

联邦学习（FL）是一种机器学习设置, 其中许多客户端（例如移动设备或整个组织）在中央服务器（例如服务提供商）的协调下协作培训模型, 同时保持培训数据的分散. 它体现了集中收集和数据最小化的原则, 可以减轻传统集中式机器学习带来的许多系统性隐私风险和成本. 从研究和应用的角度来看, 这一领域最近受到了极大的关注. 本文描述了联邦学习环境的定义特征和挑战, 强调了重要的实践约束和注意事项, 然后列举了一系列有价值的研究方向. 这项工作的目标是突出具有重大理论和实践意义的研究问题, 并鼓励对可能产生重大现实影响的问题进行研究.

The term *federated learning* was introduced in 2016 by McMahan et al. [337]: "We term our approach Federated Learning, since the learning task is solved by a loose federation of participating devices (which we refer to as clients) which are coordinated by a central server." An unbalanced and non-IID (identically and independently distributed) data partitioning across a massive number of unreliable devices with limited communication bandwidth was introduced as the defining set of challenges.

术语*federated learning*于2016年由McMahan et al. [337]引入, "我们将我们的方法称为联邦学习, 因为学习任务由参与设备（我们称之为客户端）的松散联盟解决, 这些设备由中央服务器协调." 一个不平衡的非IID在通信带宽有限的大量不可靠设备之间进行（相同且独立分布）数据分区是一组确定的挑战.

Significant related work predates the introduction of the term federated learning. A longstanding goal pursued by many research communities (including cryptography, databases, and machine learning) is to analyze and learn from data distributed among many owners without exposing that data. Cryptographic methods for computing on encrypted data were developed starting in the early 1980s [396, 492], and Agrawal and Srikant [11] and Vaidya et al. [457] are early examples of work that sought to learn from local data using a centralized server while preserving privacy. 重要的相关工作早于联邦学习这一术语的引入. 这是许多研究团体（包括密码学、数据库和机器学习）追求的长期目标是从分布在许多所有者之间的数据中进行分析和学习, 而不公开这些数据. 加密数据的加密计算方法是从1980年代初开始开发的[396, 492], 以及Agrawal and Srikant [11]和Vaidya et al. [457]是早期的工作示例, 这些工作试图使用集中服务器从本地数据中学习, 同时保护隐私.

Conversely, even since the introduction of the term federated learning, we are aware of no single work that directly addresses the full set of FL challenges. Thus, the term federated learning provides a convenient shorthand for a set of characteristics, constraints, and challenges that often co-occur in applied ML problems on decentralized data where privacy is paramount.

相反, 即使引入了联邦学习这一术语, 我们也意识到没有一项工作能够直接解决整个FL挑战. 因此, 联邦

学习这一术语为分散数据上的应用ML问题中经常同时出现的一组特征、约束和挑战提供了方便的速记隐私至上.

This paper originated at the Workshop on Federated Learning and Analytics held June 17–18th, 2019, hosted at Google's Seattle office. During the course of this two-day event, the need for a broad paper surveying the many open challenges in the area of federated learning became clear.[1]

本论文起源于2019年6月17日至18日在谷歌西雅图办公室举办的联邦学习和分析研讨会. 在这两天的活动期间, 需要一篇广泛的论文来调查联邦学习领域中的许多公开挑战, 这一点变得很清楚. [2]

A key property of many of the problems discussed is that they are inherently interdisciplinary — solving them likely requires not just machine learning, but techniques from distributed optimization, cryptography, security, differential privacy, fairness, compressed sensing, systems, information theory, statistics, and more. Many of the hardest problems are at the intersections of these areas, and so we believe collaboration will be essential to ongoing progress. One of the goals of this work is to highlight the ways in which techniques from these fields can potentially be combined, raising both interesting possibilities as well as new challenges.

所讨论的许多问题的一个关键特性是, 它们本质上是跨学科的——解决它们可能不仅需要机器学习, 还需要来自分布式优化、密码学、安全性、差分隐私、公平性、压缩感知、系统、信息论、统计学等领域的技术问题就在这些领域的交叉点上, 因此我们相信合作对于不断取得进展至关重要. 这项工作的目标之一是强调这些领域的技术可以潜在地结合在一起的方式, 既带来有趣的可能性, 也带来新的挑战.

Since the term federated learning was initially introduced with an emphasis on mobile and edge device applications [337, 334], interest in applying FL to other applications has greatly increased, including some which might involve only a small number of relatively reliable clients, for example multiple organizations collaborating to train a model. We term these two federated learning settings "cross-device" and "cross-silo" respectively. Given these variations, we propose a somewhat broader definition of federated learning:

由于联邦学习一词最初是以移动和边缘设备应用为重点引入的[337, 334],将FL应用于其他应用程序的兴趣大大增加, 包括一些可能只涉及少量相对可靠的客户端, 例如多个组织协作培训模型. 我们将这两种联邦学习设置分别称为"跨设备"和"跨竖井". 鉴于这些变化, 我们对联邦学习提出更宽泛的定义：

> ***Federated learning*** *is a machine learning setting where multiple entities (clients) collaborate in solving a machine learning problem, under the coordination of a central server or service provider. Each client's raw data is stored locally and not exchanged or transferred; instead, focused updates intended for immediate aggregation are used to achieve the learning objective.*

Focused updates are updates narrowly scoped to contain the minimum information necessary for the specific learning task at hand; aggregation is performed as early as possible in the service of data minimization. We note that this definition distinguishes federated learning from fully decentralized (peer-to-peer) learning techniques as discussed in Section 2.1.

Although privacy-preserving data analysis has been studied for more than 50 years, only in the past decade have solutions been widely deployed at scale (e.g. [177, 154]). Cross-device federated learning and federated data analysis are now being applied in consumer digital products. Google makes extensive use of federated learning in the Gboard

[1] During the preparation of this work, Li et al. [301] independently released an excellent but less comprehensive survey.

[2] 在这项工作的准备过程中, Li et al. [301]独立发布了一份优秀但不太全面的调查.

mobile keyboard [376, 222, 491, 112, 383], as well as in features on Pixel phones [14] and in Android Messages [439]. While Google has pioneered cross-device FL, interest in this setting is now much broader, for example: Apple is using cross-device FL in iOS 13 [25], for applications like the QuickType keyboard and the vocal classifier for "Hey Siri" [26]; doc.ai is developing cross-device FL solutions for medical research [149], and Snips has explored cross-device FL for hotword detection [298].

Cross-silo applications have also been proposed or described in myriad domains including finance risk prediction for reinsurance [476], pharmaceuticals discovery [179], electronic health records mining [184], medical data segmentation [15, 139], and smart manufacturing [354].

The growing demand for federated learning technology has resulted in a number of tools and frameworks becoming available. These include TensorFlow Federated [38], Federated AI Technology Enabler [33], PySyft [399], Leaf [35], PaddleFL [36] and Clara Training Framework [125]; more details in Appendix 8. Commercial data platforms incorporating federated learning are in development from established technology companies as well as smaller start-ups.

Table 1 contrasts both cross-device and cross-silo federated learning with traditional single-datacenter distributed learning across a range of axes. These characteristics establish many of the constraints that practical federated learning systems must typically satisfy, and hence serve to both motivate and inform the open challenges in federated learning. They will be discussed at length in the sections that follow.

These two FL variants are called out as representative and important examples, but different FL settings may have different combinations of these characteristics. For the remainder of this paper, we consider the cross-device FL setting unless otherwise noted, though many of the problems apply to other FL settings as well. Section 2 specifically addresses some of the many other variations and applications.

Next, we consider cross-device federated learning in more detail, focusing on practical aspects common to a typical large-scale deployment of the technology; Bonawitz et al. [81] provides even more detail for a particular production system, including a discussion of specific architectural choices and considerations.

1.1 The Cross-Device Federated Learning Setting

This section takes an applied perspective, and unlike the previous section, does not attempt to be definitional. Rather, the goal is to describe some of the practical issues in cross-device FL and how they might fit into a broader machine learning development and deployment ecosystem. The hope is to provide useful context and motivation for the open problems that follow, as well as to aid researchers in estimating how straightforward it would be to deploy a particular new approach in a real-world system. We begin by sketching the lifecycle of a model before considering a FL training process.

	Datacenter distributed learning	**Cross-silo federated learning**	**Cross-device federated learning**
Setting	Training a model on a large but "flat" dataset. Clients are compute nodes in a single cluster or datacenter.	Training a model on siloed data. Clients are different organizations (e.g. medical or financial) or geo-distributed datacenters.	The clients are a very large number of mobile or IoT devices.
Data distribution	Data is centrally stored and can be shuffled and balanced across clients. Any client can read any part of the dataset.	**Data is generated locally and remains decentralized.** Each client stores its own data and cannot read the data of other clients. Data is not independently or identically distributed.	
Orchestration	Centrally orchestrated.	**A central orchestration server/service organizes the training**, but never sees raw data.	
Wide-area communication	None (fully connected clients in one datacenter/cluster).	Typically a hub-and-spoke topology, with the hub representing a coordinating service provider (typically without data) and the spokes connecting to clients.	
Data availability	—— All clients are almost always available. ——		Only a fraction of clients are available at any one time, often with diurnal or other variations.
Distribution scale	Typically 1 - 1000 clients.	Typically 2 - 100 clients.	Massively parallel, up to 10^{10} clients.
Primary bottleneck	Computation is more often the bottleneck in the datacenter, where very fast networks can be assumed.	Might be computation or communication.	Communication is often the primary bottleneck, though it depends on the task. Generally, cross-device federated computations use wi-fi or slower connections.
Addressability	Each client has an identity or name that allows the system to access it specifically.		Clients cannot be indexed directly (i.e., no use of client identifiers).
Client statefulness	Stateful — each client may participate in each round of the computation, carrying state from round to round.		Stateless — each client will likely participate only once in a task, so generally a fresh sample of never-before-seen clients in each round of computation is assumed.
Client reliability	—— Relatively few failures. ——		Highly unreliable — 5% or more of the clients participating in a round of computation are expected to fail or drop out (e.g. because the device becomes ineligible when battery, network, or idleness requirements are violated).
Data partition axis	Data can be partitioned / re-partitioned arbitrarily across clients.	Partition is fixed. Could be example-partitioned (horizontal) or feature-partitioned (vertical).	Fixed partitioning by example (horizontal).

表 1: Typical characteristics of federated learning settings vs. distributed learning in the datacenter (e.g. [150]). Cross-device and cross-silo federated learning are two examples of FL domains, but are not intended to be exhaustive. The primary defining characteristics of FL are highlighted in bold, but the other characteristics are also critical in determining which techniques are applicable.

	Datacenter distributed learning	**Cross-silo federated learning**	**Cross-device federated learning**
设置	在大型但“扁平”的数据集上训练模型。 客户端是单个集群或数据中心中的计算节点。	在孤立数据上训练模型。 客户是不同的组织 （例如医疗或金融）或地理分布式数据中心。	客户是大量的移动或物联网设备。
数据分布	数据集中存储， 可以在客户端之间进行混洗和平衡。 任何客户端都可以读取数据集的任何部分。	**数据在本地生成并保持去中心化**。每个客户端存储自己的数据，无法读取其他客户端的数据。数据不是独立或同分布的。	
编排	集中编排。	**中央编排服务器/服务组织培训**，但从未看到原始数据。	
广域通信	无 （一个数据中心/集群中的完全连接的客户端）。	通常是一个中心辐射型拓扑， 中心代表协调服务提供者 （通常没有数据），而辐射连接到客户端。	
数据 可用性	———— 所有客户端几乎总是可用的。 ————		任何时候只有一小部分客户可用， 通常有昼夜或其他变化。
分销规模	通常为 1 - 1000 个客户端。	通常有 2 - 100 个客户。	大规模并行，最多 10^{10} 客户。
主要瓶颈	计算通常是数据中心的瓶颈，可以假设非常快的网络。	可能是计算或通信。	沟通通常是主要瓶颈， 尽管这取决于任务。通常，跨设备联合计算使用 wi-fi 或更慢的连接。
可寻址性	每个客户端都有一个身份或名称，允许系统专门访问它。		客户端不能直接索引 （即， 不使用客户端标识符）。
客户端状态	Stateful — 每个客户端都可以参与每一轮计算，从一轮到一轮携带状态。		无状态—每个客户端可能只参与一次任务， 因此通常在每轮计算中假设一个从未见过的客户端的新样本。
客户端可靠性	———————— 相对较少的失败。 ————————		高度不可靠—参与一轮计算的客户端中有 5% 或更多预计会失败或退出 （例如，因为在违反电池、 网络或空闲要求时设备变得不合格）。
数据分区轴	数据可以跨客户端任意分区/重新分区。	分区是固定的。可以是样本分区 （水平）或特征分区 （垂直）。	固定分区样本 （水平）。

表 2: 联邦学习设置的典型特征与数据中心中的分布式学习 （例如 [150]）。跨设备和跨孤岛联合学习是 FL 域的两个示例，但并非详尽无遗。 FL 的主要定义特征以粗体突出显示，但其他特征对于确定哪些技术适用也是至关重要的。

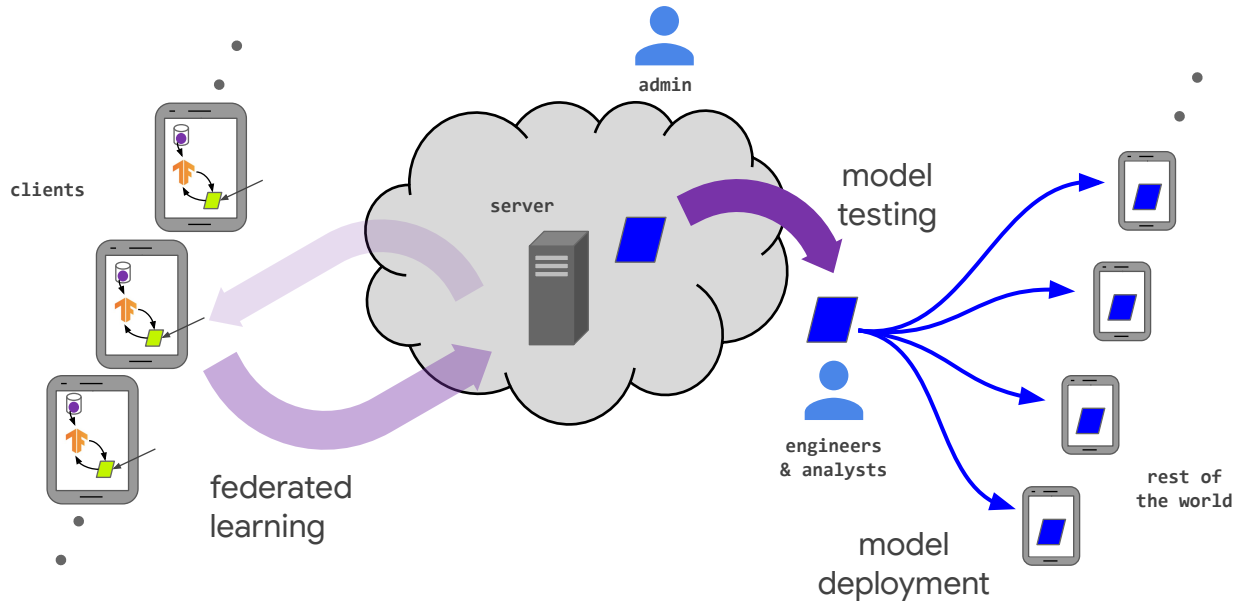


图 1: The lifecycle of an FL-trained model and the various actors in a federated learning system. This figure is revisited in Section 4 from a threat models perspective.

1.1.1 The Lifecycle of a Model in Federated Learning

The FL process is typically driven by a model engineer developing a model for a particular application. For example, a domain expert in natural language processing may develop a next word prediction model for use in a virtual keyboard. Figure 1 shows the primary components and actors. At a high level, a typical workflow is:

1. **Problem identification:** The model engineer identifies a problem to be solved with FL.
2. **Client instrumentation:** If needed, the clients (e.g. an app running on mobile phones) are instrumented to store locally (with limits on time and quantity) the necessary training data. In many cases, the app already will have stored this data (e.g. a text messaging app must store text messages, a photo management app already stores photos). However, in some cases additional data or metadata might need to be maintained, e.g. user interaction data to provide labels for a supervised learning task.
3. **Simulation prototyping (optional):** The model engineer may prototype model architectures and test learning hyperparameters in an FL simulation using a proxy dataset.
4. **Federated model training:** Multiple federated training tasks are started to train different variations of the model, or use different optimization hyperparameters.
5. **(Federated) model evaluation:** After the tasks have trained sufficiently (typically a few days, see below), the models are analyzed and good candidates selected. Analysis may include metrics computed on standard datasets in the datacenter, or federated evaluation wherein the models are pushed to held-out clients for evaluation on local client data.

Total population size	10^6–10^{10} devices
Devices selected for one round of training	50 – 5000
Total devices that participate in training one model	10^5–10^7
Number of rounds for model convergence	500 – 10000
Wall-clock training time	1 – 10 days

表 3: Order-of-magnitude sizes for typical cross-device federated learning applications.

6. **Deployment:** Finally, once a good model is selected, it goes through a standard model launch process, including manual quality assurance, live A/B testing (usually by using the new model on some devices and the previous generation model on other devices to compare their in-vivo performance), and a staged rollout (so that poor behavior can be discovered and rolled back before affecting too many users). The specific launch process for a model is set by the owner of the application and is usually independent of how the model is trained. In other words, this step would apply equally to a model trained with federated learning or with a traditional datacenter approach.

One of the primary practical challenges an FL system faces is making the above workflow as straightforward as possible, ideally approaching the ease-of-use achieved by ML systems for centralized training. While much of this paper concerns federated training specifically, there are many other components including federated analytics tasks like model evaluation and debugging. Improving these is the focus of Section 3.4. For now, we consider in more detail the training of a single FL model (Step 4 above).

1.1.2 A Typical Federated Training Process

We now consider a template for FL training that encompasses the Federated Averaging algorithm of McMahan et al. [337] and many others; again, variations are possible, but this gives a common starting point.

A server (service provider) orchestrates the training process, by repeating the following steps until training is stopped (at the discretion of the model engineer who is monitoring the training process):

1. **Client selection:** The server samples from a set of clients meeting eligibility requirements. For example, mobile phones might only check in to the server if they are plugged in, on an unmetered wi-fi connection, and idle, in order to avoid impacting the user of the device.

2. **Broadcast:** The selected clients download the current model weights and a training program (e.g. a TensorFlow graph [2]) from the server.

3. **Client computation:** Each selected device locally computes an update to the model by executing the training program, which might for example run SGD on the local data (as in Federated Averaging).

4. **Aggregation:** The server collects an aggregate of the device updates. For efficiency, stragglers might be dropped at this point once a sufficient number of devices have reported results. This stage is also the integration point for many other techniques which will be discussed later, possibly including: secure aggregation for added privacy, lossy compression of aggregates for communication efficiency, and noise addition and update clipping for differential privacy.

5. **Model update:** The server locally updates the shared model based on the aggregated update computed from the clients that participated in the current round.

Table 3 gives typical order-of-magnitude sizes for the quantities involved in a typical federated learning application on mobile devices.

The separation of the client computation, aggregation, and model update phases is not a strict requirement of federated learning, and it indeed excludes certain classes of algorithms, for example asynchronous SGD where each client's update is immediately applied to the model, before any aggregation with updates from other clients. Such asynchronous approaches may simplify some aspects of system design, and also be beneficial from an optimization perspective (though this point can be debated). However, the approach presented above has a substantial advantage in affording a separation of concerns between different lines of research: advances in compression, differential privacy, and secure multi-party computation can be developed for standard primitives like computing sums or means over decentralized updates, and then composed with arbitrary optimization or analytics algorithms, so long as those algorithms are expressed in terms of aggregation primitives.

It is also worth emphasizing that in two respects, the FL training process should not impact the user experience. First, as outlined above, even though model parameters are typically sent to some devices during the broadcast phase of each round of federated training, these models are an ephemeral part of the training process, and not used to make "live" predictions shown to the user. This is crucial, because training ML models is challenging, and a misconfiguration of hyperparameters can produce a model that makes bad predictions. Instead, user-visible use of the model is deferred to a rollout process as detailed above in Step 6 of the model lifecycle. Second, the training itself is intended to be invisible to the user — as described under client selection, training does not slow the device or drain the battery because it only executes when the device is idle and connected to power. However, the limited availability these constraints introduce leads directly to open research challenges which will be discussed subsequently, such as semi-cyclic data availability and the potential for bias in client selection.

1.2 Federated Learning Research

The remainder of this paper surveys many open problems that are motivated by the constraints and challenges of real-world federated learning settings, from training models on medical data from a hospital system to training using hundreds of millions of mobile devices. Needless to say, most researchers working on federated learning problems will likely not be deploying production FL systems, nor have access to fleets of millions of real-world devices. This leads to a key distinction between the practical settings that motivate the work and experiments conducted in simulation which provide evidence of the suitability of a given approach to the motivating problem.

本文的其余部分调查了许多受现实世界联邦学习环境约束和挑战驱动的开放问题, 从医院系统医疗数据的培训模型到使用数亿移动设备的培训. 不用说, 大多数研究联邦学习问题的研究人员很可能不会部署生产FL系统, 也不会访问数以百万计的真实世界设备. 这导致了激励工作的实际设置与模拟中进行的实验之间的关键区别, 模拟中进行的实验为激励问题的给定方法的适用性提供了证据.

This makes FL research somewhat different than other ML fields from an experimental perspective, leading to additional considerations in conducting FL research. In particular, when highlighting open problems, we have attempted, when possible, to also indicate relevant performance metrics which can be measured in simulation, the characteristics of datasets which will make them more representative of real-world performance, etc. The need for

simulation also has ramifications for the presentation of FL research. While not intended to be authoritative or absolute, we make the following modest suggestions for presenting FL research that addresses the open problems we describe:

这使得FL研究从实验的角度与其他ML领域有所不同, 导致在进行FL研究时需要额外考虑. 特别是, 在强调开放性问题时, 我们已尝试在可能的情况下, 也指出可在模拟中测量的相关性能指标、数据集的特征, 这些特征将使其更能代表真实世界的性能, 对模拟的需求也对外语研究的呈现产生了影响. 虽然不是为了权威性或绝对性, 但我们提出以下温和的建议, 以展示解决我们描述的开放性问题的外语研究：

- As shown in Table 1, the FL setting can encompass a wide range of problems. Compared to fields where the setting and goals are well-established, it is important to precisely describe the details of the particular FL setting of interest, particularly when the proposed approach makes assumptions that may not be appropriate in all settings (e.g. stateful clients that participate in all rounds).
- Of course, details of any simulations should be presented in order to make the research reproducible. But it is also important to explain which aspects of the real-world setting the simulation is designed to capture (and which it is not), in order to effectively make the case that success on the simulated problem implies useful progress on the real-world objective. We hope that the guidance in this paper will help with this.
- Privacy and communication efficiency are always first-order concerns in FL, even if the experiments are simulations running on a single machine using public data. More so than with other types of ML, for any proposed approach it is important to be unambiguous about *where computation happens* as well as *what is communicated*.

Software libraries for federated learning simulation as well as standard datasets can help ease the challenges of conducting effective FL research; Section 8 summarizes some of the currently available options. Developing standard evaluation metrics and establishing standard benchmark datasets for different federated learning settings (cross-device and cross-silo) remain highly important directions for ongoing work.

1.3 Organization

Section 2 builds on the ideas in Table 1, exploring other FL settings and problems beyond the original focus on cross-device settings. Section 3 then turns to core questions around improving the efficiency and effectiveness of federated learning. Section 4 undertakes a careful consideration of threat models and considers a range of technologies toward the goal of achieving rigorous privacy protections. As with all machine learning systems, in federated learning applications there may be incentives to manipulate the models being trained, and failures of various kinds are inevitable; these challenges are discussed in Section 5. Finally, we address the important challenges of providing fair and unbiased models in Section 6.

Section 4对威胁模型进行了仔细的考虑, 并考虑了一系列旨在实现严格隐私保护目标的技术. 与所有机器学习系统一样, 在联邦学习应用程序中, 可能存在操纵正在训练的模型的动机, 各种各样的失败是不可避免的; 这些挑战将在Section 5中讨论. 最后, 我们讨论了在Section 6中提供公平无偏模型的重要挑战.

2 Relaxing the Core FL Assumptions: Applications to Emerging Settings and Scenarios

In this section, we will discuss areas of research related to the topics discussed in the previous section. Even though not being the main focus of the remainder of the paper, progress in these areas could motivate design of the next generation of production systems.

在本节中, 我们将讨论与上一节中讨论的主题相关的研究领域. 尽管不是本文其余部分的主要重点, 但这些领域的进展可能会激发下一代生产系统的设计.

2.1 Fully Decentralized / Peer-to-Peer Distributed Learning

In federated learning, a central server orchestrates the training process and receives the contributions of all clients. The server is thus a central player which also potentially represents a single point of failure. While large companies or organizations can play this role in some application scenarios, a reliable and powerful central server may not always be available or desirable in more collaborative learning scenarios [459]. Furthermore, the server may even become a bottleneck when the number of clients is very large, as demonstrated by Lian et al. [305] (though this can be mitigated by careful system design, e.g. [81]).

在联邦学习中, 中央服务器协调训练过程并接收所有客户端的贡献.因此, 服务器是一个中心参与者, 它也可能代表单点故障.虽然大型公司或组织可以在某些应用场景中扮演这个角色, 但在更具协作性的学习场景 [459] 中, 可靠且强大的中央服务器可能并不总是可用或不可取.此外, 当客户端数量非常大时, 服务器甚至可能成为瓶颈, 如 Lian et al. [305] 所示（尽管这可以通过仔细的系统设计来缓解, 例如 [81]）.

The key idea of fully decentralized learning is to replace communication with the server by peer-to-peer communication between individual clients. The communication topology is represented as a connected graph in which nodes are the clients and an edge indicates a communication channel between two clients. The network graph is typically chosen to be sparse with small maximum degree so that each node only needs to send/receive messages to/from a small number of peers; this is in contrast to the star graph of the server-client architecture. In fully decentralized algorithms, a round corresponds to each client performing a local update and exchanging information with their neighbors in the graph[3]. In the context of machine learning, the local update is typically a local (stochastic) gradient step and the communication consists in averaging one's local model parameters with the neighbors. Note that there is no longer a global state of the model as in standard federated learning, but the process can be designed such that all local models converge to the desired global solution, i.e., the individual models gradually reach consensus. While multi-agent optimization has a long history in the control community, fully decentralized variants of SGD and other optimization algorithms have recently been considered in machine learning both for improved scalability in datacenters [29] as well as for decentralized networks of devices [127, 459, 443, 59, 278, 291, 173]. They consider undirected network graphs, although the case of directed networks (encoding unidirectional channels which may arise in real-world scenarios such as social networks or data markets) has also been studied in [29, 226].

完全去中心化学习的关键思想是通过单个客户端之间的点对点通信代替与服务器的通信.通信拓扑表示

[3]Note, however, that the notion of a round does not need to even make sense in this setting. See for instance the discussion on clock models in [85].

	Federated learning	**Fully decentralized (peer-to-peer) learning**
Orchestration	A central orchestration server or service organizes the training, but never sees raw data.	No centralized orchestration.
Wide-area communication	Typically a hub-and-spoke topology, with the hub representing a coordinating service provider (typically without data) and the spokes connecting to clients.	Peer-to-peer topology, with a possibly dynamic connectivity graph.

表 4: A comparison of the key distinctions between federated learning and fully decentralized learning. Note that as with FL, decentralized learning can be further divided into different use-cases, with distinctions similar to those made in Table 1 comparing cross-silo and cross-device FL.

为一个连通图, 其中节点是客户端, 边表示两个客户端之间的通信通道.网络图通常选择为稀疏且最大度数较小, 以便每个节点只需要向/从少数对等方发送/接收消息;这与服务器-客户端架构的星形图形成对比.在完全去中心化的算法中, 一轮对应于每个客户端执行本地更新并与图中的邻居交换信息[4] 中关于时钟模型的讨论.在机器学习的上下文中, 局部更新通常是局部（随机）梯度步骤, 通信包括将一个人的局部模型参数与邻居进行平均.请注意, 不再有标准联邦学习中模型的全局状态, 但可以设计该过程, 使所有局部模型收敛到所需的全局解决方案, 即各个模型逐渐达成共识.虽然多智能体优化在控制社区有着悠久的历史, 但最近在机器学习中考虑了 SGD 的完全分散的变体和其他优化算法, 以提高数据中心的可扩展性 [29] 以及分散的设备网络 citepColin2016, Vanhaesebrouck2017, Tang2018, Bellet2018a, Koloskova2019, Lalitha2019, elgabligadmm.他们考虑了无向网络图, 尽管 [29, 226] 也研究了有向网络的情况（编码可能出现在现实世界场景中的单向通道, 如社交网络或数据市场）.

It is worth noting that even in the decentralized setting outlined above, a central authority may still be in charge of setting up the learning task. Consider for instance the following questions: Who decides what is the model to be trained in the decentralized setting? What algorithm to use? What hyperparameters? Who is responsible for debugging when something does not work as expected? A certain degree of trust of the participating clients in a central authority would still be needed to answer these questions. Alternatively, the decisions could be taken by the client who proposes the learning task, or collaboratively through a consensus scheme (see Section 2.1.2).

值得注意的是, 即使在上述分散设置中, 中央机构仍可能负责设置学习任务.例如, 考虑以下问题：谁决定在去中心化环境中要训练的模型是什么？使用什么算法？什么超参数？当某些事情没有按预期工作时, 谁负责调试？仍然需要参与客户对中央机构的一定程度的信任才能回答这些问题.或者, 决策可以由提出学习任务的客户做出, 或者通过共识方案协作（参见 Section 2.1.2）.

Table 4 provides a comparison between federated and peer-to-peer learning. While the architectural assumptions of decentralized learning are distinct from those of federated learning, it can often be applied to similar problem domains, many of the same challenges arise, and there is significant overlap in the research communities. Thus, we consider decentralized learning in this paper as well; in this section challenges specific to the decentralized approach are explicitly considered, but many of the open problems in other sections also arise in the decentralized case.

Table 4 提供了联合学习和点对点学习之间的比较. 虽然分散学习的架构假设与联邦学习的架构假设不同,

[4]但是请注意, 轮的概念在此设置中甚至不需要有意义.例如, 参见 [85].

但它通常可以应用于类似的问题领域, 出现许多相同的挑战, 并且在研究社区中有显着的重叠. 因此, 我们在本文中也考虑了去中心化学习; 本节明确考虑了分散式方法特有的挑战, 但其他节中的许多未决问题也出现在分散式情况下.

2.1.1 Algorithmic Challenges

A large number of important algorithmic questions remain open on the topic of real-world usability of decentralized schemes for machine learning. Some questions are analogous to the special case of federated learning with a central server, and other challenges come as an additional side-effect of being fully decentralized or trust-less. We outline some particular areas in the following.

关于机器学习去中心化方案在现实世界中的可用性这一主题, 许多重要的算法问题仍未解决. 有些问题类似于具有中央服务器的联邦学习的特殊情况, 而其他挑战则是完全去中心化或无信任的额外副作用. 我们在下面概述了一些特定的领域.

Effect of network topology and asynchrony on decentralized SGD Fully decentralized algorithms for learning should be robust to the limited availability of the clients (with clients temporarily unavailable, dropping out or joining during the execution) and limited reliability of the network (with possible message drops). While for the special case of generalized linear models, schemes using the duality structure could enable some of these desired robustness properties [231], for the case of deep learning and SGD this remains an open question. When the network graph is complete but messages have a fixed probability to be dropped, Yu et al. [498] show that one can achieve convergence rates that are comparable to the case of a reliable network. Additional open research questions concern non-IID data distributions, update frequencies, efficient communication patterns and practical convergence time [443], as we outline in more detail below.

完全分散的学习算法应该对客户端的有限可用性（客户端在执行过程中暂时不可用, 退出或加入）和网络的有限可靠性（可能的消息丢失）具有鲁棒性.而对于广义线性模型的特殊情况, 使用对偶结构的方案可以实现一些期望的鲁棒性属性, 对于深度学习和SGD, 这仍然是一个悬而未决的问题. 当网络图完成但消息具有固定的丢弃概率时, Yu et al. [498]表明可以实现与可靠网络情况相当的收敛速度. 其他开放性研究问题涉及非IID数据分布, 更新频率, 有效通信模式和实际收敛时间[443], 我们将在下文更详细地概述.

Well-connected or denser networks encourage faster consensus and give better theoretical convergence rates, which depend on the spectral gap of the network graph. However, when data is IID, sparser topologies do not necessarily hurt the convergence in practice: this was analyzed theoretically in [357]. Denser networks typically incur communication delays which increase with the node degrees. Most of optimization-theory works do not explicitly consider how the topology affects the runtime, that is, wall-clock time required to complete each SGD iteration. Wang et al. [469] propose MATCHA, a decentralized SGD method based on matching decomposition sampling, that reduces the communication delay per iteration for any given node topology while maintaining the same error convergence speed. The key idea is to decompose the graph topology into matchings consisting of disjoint communication links that can operate in parallel, and carefully choose a subset of these matchings in each iteration. This sequence of subgraphs results in more frequent communication over connectivity-critical links (ensuring fast error convergence) and less frequent communication over other links (saving communication delays).

连接良好或密度更高的网络鼓励更快的一致性, 并提供更好的理论收敛速度, 这取决于网络图的谱隙.然

而, 当数据是IID时, 稀疏拓扑在实践中并不一定会影响收敛：这是在[357]中从理论上分析的.密集网络通常会导致通信延迟, 而通信延迟随着节点度的增加而增加.大多数优化理论工作没有明确地考虑拓扑如何影响运行时间, 也就是说, 完成每个SGD迭代所需的壁时钟时间.Wang et al. [469]提出了MATCHA, 这是一种基于匹配分解采样的分散SGD方法, 在保持相同错误收敛速度的同时, 减少了任何给定节点拓扑每次迭代的通信延迟.其关键思想是将图拓扑分解为由可并行运行的不相交通信链路组成的匹配, 并在每次迭代中仔细选择这些匹配的子集.这种子图序列导致在连接关键链路上进行更频繁的通信（确保快速错误收敛）, 而在其他链路上进行更频繁的通信（节省通信延迟）.

The setting of decentralized SGD also naturally lends itself to asynchronous algorithms in which each client becomes active independently at random times, removing the need for global synchronization and potentially improving scalability [127, 459, 59, 29, 306].

分散式SGD的设置自然也适用于异步算法, 其中每个客户机在随机时间独立活动, 消除了全局同步的需要, 并可能提高可伸缩性[127, 459, 59, 29, 306].

Local-update decentralized SGD The theoretical analysis of schemes which perform several local update steps before a communication round is significantly more challenging than those using a single SGD step, as in mini-batch SGD. While this will also be discussed later in Section 3.2, the same also holds more generally in the fully decentralized setting of interest here. Schemes relying on a single local update step are typically proven to converge in the case of non-IID local datasets [278, 279]. For the case with several local update steps, [467, 280] recently provided convergence analysis. Further, [469] provides a convergence analysis for the non-IID data case, but for the specific scheme based on matching decomposition sampling described above. In general, however, understanding the convergence under non-IID data distributions and how to design a model averaging policy that achieves the fastest convergence remains an open problem.

在一轮通信之前执行多个局部更新步骤的方案的理论分析比使用单个SGD步骤的方案（如在小型批量SGD中）具有更大的挑战性.虽然这也将在后面的Section 3.2中讨论, 但在这里感兴趣的完全分散的环境中, 同样的情况也更普遍.依赖于单个本地更新步骤的方案通常在非IID本地数据集[278, 279]的情况下被证明是收敛的.对于具有多个局部更新步骤的情况, [467, 280]最近提供了收敛性分析.此外, [469]为非IID数据情况提供了收敛性分析, 但针对基于上述匹配分解采样的特定方案.然而, 一般来说, 理解非IID数据分布下的收敛性以及如何设计实现最快收敛的模型平均策略仍然是一个有待解决的问题.

Personalization, and trust mechanisms Similarly to the cross-device FL setting, an important task for the fully decentralized scenario under the non-IID data distributions available to individual clients is to design algorithms for learning collections of personalized models. The work of [459, 59] introduces fully decentralized algorithms to collaboratively learn a personalized model for each client by smoothing model parameters across clients that have similar tasks (i.e., similar data distributions). Zantedeschi et al. [504] further learn the similarity graph together with the personalized models. One of the key unique challenges in the decentralized setting remains the robustness of such schemes to malicious actors or contribution of unreliable data or labels. The use of incentives or mechanism design in combination with decentralized learning is an emerging and important goal, which may be harder to achieve in the setting without a trusted central server.

与跨设备 FL 设置类似, 在个人客户端可用的非 IID 数据分布下完全去中心化场景的一项重要任务是设计用于学习个性化模型集合的算法. [459, 59] 的工作引入了完全去中心化的算法, 通过平滑具有相似任务（即相

似数据分布）的客户端之间的模型参数来协作学习每个客户端的个性化模型. Zantedeschi et al. [504] 进一步学习相似图和个性化模型. 分散设置中的关键独特挑战之一仍然是此类方案对恶意行为者或不可靠数据或标签的贡献的稳健性. 将激励或机制设计与去中心化学习相结合是一个新兴且重要的目标, 在没有可信中央服务器的情况下可能更难实现.

Gradient compression and quantization methods In potential applications, the clients would often be limited in terms of communication bandwidth available and energy usage permitted. Translating and generalizing some of the existing compressed communication schemes from the centralized orchestrator-facilitated setting (see Section 3.5) to the fully decentralized setting, without negatively impacting the convergence is an active research direction [278, 391, 444, 279]. A complementary idea is to design decentralized optimization algorithms which naturally give rise to sparse updates [504].

在潜在的应用中, 客户机通常在可用的通信带宽和允许的能源使用方面受到限制.将一些现有的压缩通信方案从集中式协调器促进的设置（参见Section 3.5）转换并推广到完全分散的设置, 而不会对融合产生负面影响, 这是一个积极的研究方向 [278, 391, 444, 279]. 一个补充想法是设计分散优化算法, 该算法自然会产生稀疏更新.

Privacy An important challenge in fully decentralized learning is to prevent any client from reconstructing the private data of another client from its shared updates while maintaining a good level of utility for the learned models. Differential privacy (see Section 4) is the standard approach to mitigate such privacy risks. In decentralized federated learning, this can be achieved by having each client add noise locally, as done in [239, 59]. Unfortunately, such local privacy approaches often come at a large cost in utility. Furthermore, distributed methods based on secure aggregation or secure shuffling that are designed to improve the privacy-utility trade-off in the standard FL setting (see Section 4.4.3) do not easily integrate with fully decentralized algorithms. A possible direction to achieve better trade-offs between privacy and utility in fully decentralized algorithms is to rely on decentralization itself to amplify differential privacy guarantees, for instance by considering appropriate relaxations of local differential privacy [146].

完全去中心化学习的一个重要挑战是防止任何客户端从共享更新中重建另一个客户端的私有数据, 同时保持学习模型的良好效用水平.差分隐私（参见 Section 4）是减轻此类隐私风险的标准方法.在分散的联邦学习中, 这可以通过让每个客户端在本地添加噪声来实现, 如 [239, 59] 中所做的那样.不幸的是, 这种本地隐私方法通常会在效用方面付出巨大代价.此外, 基于安全聚合或安全改组的分布式方法旨在改善标准 FL 设置（参见 Section 4.4.3）中的隐私-效用权衡, 不容易与完全去中心化的算法集成.在完全去中心化的算法中实现隐私和效用之间更好的权衡的一个可能方向是依靠去中心化本身来放大差分隐私保证, 例如通过考虑适当放松局部差分隐私 [146].

2.1.2 Practical Challenges

An orthogonal question for fully decentralized learning is how it can be practically realized. This section outlines a family of related ideas based on the idea of a distributed ledger, but other approaches remain unexplored.

完全分散学习的一个正交问题是如何实际实现它.本节概述了一系列基于分布式账本思想的相关思想, 但其他方法尚未探索.

A blockchain is a distributed ledger shared among disparate users, making possible digital transactions, including

transactions of cryptocurrency, without a central authority. In particular, smart contracts allow execution of arbitrary code on top of the blockchain, essentially a massively replicated eventually-consistent state machine. In terms of federated learning, use of the technology could enable decentralization of the global server by using smart contracts to do model aggregation, where the participating clients executing the smart contracts could be different companies or cloud services.

区块链是在不同用户之间共享的分布式账本, 使数字交易成为可能, 包括加密货币交易, 而无需中央授权.特别是, 智能合约允许在区块链上执行任意代码, 本质上是大规模复制的最终一致状态机.就联邦学习而言, 使用该技术可以通过使用智能合约进行模型聚合实现全局服务器的分散, 其中执行智能合约的参与客户可以是不同的公司或云服务.

However, on today's blockchain platforms such as Ethereum [478], data on the blockchains is publicly available by default, this could discourage users from participating in the decentralized federated learning protocol, as the protection of the data is typically the primary motivating factor for FL. To address such concerns, it might be possible to modify the existing privacy-preserving techniques to fit into the scenario of decentralized federated learning. First of all, to prevent the participating nodes from exploiting individually submitted model updates, existing secure aggregation protocols could be used. A practical secure aggregation protocol already used in cross-device FL was proposed by Bonawitz et al. [80], effectively handling dropping out participants at the cost of complexity of the protocol. An alternative system would be to have each client stake a deposit of cryptocurrency on blockchain, and get penalized if they drop out during the execution. Without the need of handling dropouts, the secure aggregation protocol could be significantly simplified. Another way of achieving secure aggregation is to use confidential smart contract such as what is enabled by the Oasis Protocol [119] which runs inside secure enclaves. With this, each client could simply submit an encrypted local model update, knowing that the model will be decrypted and aggregated inside the secure hardware through remote attestation (though see discussion of privacy-in-depth in Section 4.1).

然而, 在以太坊 [478]等当今区块链平台上, 区块链上的数据默认情况下是公开的, 这可能会阻止用户参与分散式联邦学习协议, 因为数据保护通常是FL的主要激励因素.为了解决这些问题, 有可能修改现有的隐私保护技术, 以适应分散联邦学习的场景.首先, 为了防止参与节点利用单独提交的模型更新, 可以使用现有的安全聚合协议.已在跨设备FL中使用的实用安全聚合协议由Bonawitz et al. [80]提出, 以协议的复杂性为代价有效地处理退出参与者.另一个替代系统是让每个客户在区块链上持有加密货币存款, 如果他们在执行期间退出, 就会受到惩罚.无需处理辍学, 安全聚合协议可以大大简化.实现安全聚合的另一种方法是使用机密智能合约, 如在安全飞地内运行的Oasis协议 [119]所启用的合约.这样, 每个客户端都可以简单地提交一个加密的本地模型更新, 知道该模型将通过远程认证在安全硬件内解密和聚合（尽管请参阅Section 4.1中的隐私深入讨论）.

In order to prevent any client from trying to reconstruct the private data of another client by exploiting the global model, client-level differential privacy [338] has been proposed for FL. Client-level differential privacy is achieved by adding random Gaussian noise on the aggregated global model that is enough to hide any single client's update. In the context of blockchain, each client could locally add a certain amount of Gaussian noise after local gradient descent steps and submit the model to blockchain. The local noise scale should be calculated such that the aggregated noise on blockchain is able to achieve the same client-level differential privacy as in [338]. Finally, the aggregated global model on blockchain could be encrypted and only the participating clients hold the decryption key, which protects the model from the public.

为了防止任何客户端试图利用全局模型重构另一个客户端的私有数据, 客户端级别的差分隐私[338]已针对FL提出.客户端级别的差分隐私是通过在聚合的全局模型上添加随机高斯噪声来实现的, 该噪声足以隐藏任

何单个客户端的更新.在区块链的上下文中, 每个客户可以在局部梯度下降步骤后局部添加一定量的高斯噪声, 并将模型提交给区块链.计算局部噪声等级时, 应确保区块链上的聚集噪声能够实现与[338]中相同的客户端级差分隐私.最后, 区块链上的聚合全局模型可以加密, 并且只有参与的客户端持有解密密钥, 从而保护模型不受公众影响.

2.2 Cross-Silo Federated Learning

In contrast with the characteristics of cross-device federated learning, see Table 1, cross-silo federated learning admits more flexibility in certain aspects of the overall design, but at the same time presents a setting where achieving other properties can be harder. This section discusses some of these differences.

与跨设备联邦学习的特点相比，参见Table 1，跨孤岛联邦学习在整体设计的某些方面承认了更大的灵活性，但同时也提出了一个设置，在实现其他属性时可以更难。本节讨论其中一些差异。

The cross-silo setting can be relevant where a number of companies or organizations share incentive to train a model based on all of their data, but cannot share their data directly. This could be due to constraints imposed by confidentiality or due to legal constraints, or even within a single company when they cannot centralize their data between different geographical regions. These cross-silo applications have attracted substantial attention.

如果许多公司或组织共享基于其所有数据训练模型的激励，但不能直接共享其数据，则跨孤岛设置可能是相关的。这可能是由于保密性或法律限制所施加的限制，或者甚至是在单个公司内无法在不同地理区域之间集中数据时。这些跨筒仓的应用引起了广泛关注。

Data partitioning In the cross-device setting the data is assumed to be partitioned by examples. In the cross-silo setting, in addition to partitioning by examples, partitioning by features is of practical relevance. An example could be when two companies in different businesses have the same or overlapping set of customers, such as a local bank and a local retail company in the same city. This difference has been also referred to as horizontal and vertical federated learning by Yang et al. [490].

数据分区 在跨设备设置中，假设数据按示例进行分区。在跨筒仓设置中，除了通过示例进行分区之外，通过特征进行分区也具有实际意义。一个例子可能是当两个不同业务的公司拥有相同或重叠的客户集时，例如同一城市的本地银行和本地零售公司。这种差异也被 Yang et al. [490] 称为横向和纵向联邦学习。

Cross-silo FL with data partitioned by features, employs a very different training architecture compared to the setting with data partitioned by example. It may or may not involve a central server as a neutral party, and based on specifics of the training algorithm, clients exchange specific intermediate results rather than model parameters, to assist other parties' gradient calculations; see for instance [490, Section 2.4.2]. In this setting, application of techniques such as secure multi-party computation or homomorphic encryption have been proposed in order to limit the amount of information other participants can infer from observing the training process. The downside of this approach is that the training algorithm is typically dependent on the type of machine learning objective being pursued. Currently proposed algorithms include trees [118], linear and logistic regression [490, 224, 316], and neural networks [317]. Local updates similar to Federated Averaging (see Section 3.2) has been proposed to address the communication challenges of feature-partitioned systems [316], and [238, 318] study the security and privacy related challenges inherent in such systems.

数据按特征分区的跨筒仓 FL，与数据按示例分区的设置相比，采用了非常不同的训练架构。它可能会也可能不会涉及中央服务器作为中立方，根据训练算法的具体情况，客户端交换特定的中间结果而不是模型参数，以协助其他方的梯度计算；参见例如 [490, Section 2.4.2]。在这种情况下，已经提出了安全多方计算或同态加密等技术的应用，以限制其他参与者可以通过观察训练过程推断出的信息量。这种方法的缺点是训练算法通常取决于所追求的机器学习目标的类型。目前提出的算法包括树[118]、线性和逻辑回归[490, 224, 316]和神经网络[317]。类似于联邦平均（见 Section 3.2）的本地更新已经被提出来解决特征分区系统 [316] 的通信挑战，并且 [238, 318] 研究安全和隐私相关的挑战此类系统中固有的。

Federated transfer learning [490] is another concept that considers challenging scenarios in which data parties share only a partial overlap in the user space or the feature space, and leverage existing transfer learning techniques [365] to build models collaboratively. The existing formulation is limited to the case of 2 clients.

联合迁移学习 [490] 是另一个概念，它考虑了具有挑战性的场景，其中数据方仅共享用户空间或特征空间中的部分重叠，并利用现有的迁移学习技术 [365] 协作构建模型。现有的公式仅限于 2个客户的情况。

Partitioning by examples is usually relevant in cross-silo FL when a single company cannot centralize their data due to legal constraints, or when organizations with similar objectives want to collaboratively improve their models. For instance, different banks can collaboratively train classification or anomaly detection models for fraud detection [476], hospitals can build better diagnostic models [139], and so on.

当单个公司由于法律限制而无法集中其数据时，或者当具有相似目标的组织希望协作改进其模型时，通过示例进行分区通常与跨孤岛 FL 相关。例如，不同银行可以协同训练分类或异常检测模型用于欺诈检测[476]，医院可以构建更好的诊断模型 [139]等等。

An open-source platform supporting the above outlined applications is currently available as *Federated AI Technology Enabler (FATE)* [33]. At the same time, the IEEE P3652.1 Federated Machine Learning Working Group is focusing on standard-setting for the Federated AI Technology Framework. Other platforms include [125] focused on a range of medical applications and [321] for enterprise use cases. See Section 8 for more details.

支持上述应用程序的开源平台目前以 *Federated AI Technology Enabler (FATE)* [33] 的形式提供。与此同时，IEEE P3652.1 联合机器学习工作组正专注于联合人工智能技术框架的标准制定。其他平台包括专注于一系列医疗应用的 [125] 和用于企业用例的 [321]。有关更多详细信息，请参阅 Section 8。

Incentive mechanisms In addition to developing new algorithmic techniques for FL, incentive mechanism design for honest participation is an important practical research question. This need may arise in cross-device settings (e.g. [261, 260]), but is particularly relevant in the cross-silo setting, where participants may at the same time also be business competitors. The incentive can be in the form of monetary payout [499] or final models with different levels of performance [324]. The option to deliver models with performance commensurate to the contributions of each client is especially relevant in collaborative learning situations in which competitions exist among FL participants. Clients might worry that contributing their data to training federated learning models will benefit their competitors, who do not contribute as much but receive the same final model nonetheless (i.e. the free-rider problem). Related objectives include how to divide earnings generated by the federated learning model among contributing data owners in order to sustain long-term participation, and also how to link the incentives with decisions on defending against adversarial data owners to enhance system security, optimizing the participation of data owners to enhance system efficiency.

激励机制 除了为 FL 开发新的算法技术之外，诚实参与的激励机制设计是一个重要的实际研究问题。这种需求可能出现在跨设备设置中（例如 [261, 260]），但在跨孤岛设置中尤其重要，参与者可能同时也是商业竞争对手。激励可以采取货币支付[499] 或具有不同性能水平[324] 的最终模型的形式。在 FL 参与者之间存在竞争的协作学习情况下，提供具有与每个客户的贡献相称的性能的模型的选项尤其重要。客户可能会担心将他们的数据用于训练联邦学习模型会使他们的竞争对手受益，他们的贡献没有那么多，但仍然得到相同的最终模型（即搭便车问题）。相关目标包括如何在贡献数据所有者之间分配联邦学习模型产生的收益以维持长期参与，以及如何将激励与防御对抗性数据所有者的决策联系起来以增强系统安全性，优化参与数据所有者以提高系统效率。

Differential privacy The discussion of actors and threat models in Section 4.1 is largely relevant also for the cross-silo FL. However, protecting against different actors might have different priorities. For example, in many practical scenarios, the final trained model would be released only to those who participate in the training, which makes the concerns about "the rest of the world" less important.

On the other hand, for a practically persuasive claim, we would usually need a notion of local differential privacy, as the potential threat from other clients is likely to be more important. In cases when the clients are not considered a significant threat, each client could control the data from a number of their respective users, and a formal privacy guarantee might be needed on such user-level basis. Depending on application, other objectives could be worth pursuing. This area has not been systematically explored.

Tensor factorization Several works have also studied cross-silo federated tensor factorization where multiple sites (each having a set of data with the same feature, i.e. horizontally partitioned) jointly perform tensor factorization by only sharing intermediate factors with the coordination server while keeping data private at each site. Among the existing works, [272] used an alternating direction method of multipliers (ADMM) based approach and [325] improved the efficiency with the elastic averaging SGD (EASGD) algorithm and further ensures differential privacy for the intermediate factors.

有几项工作还研究了cross-silo联盟张量分解, 其中多个站点（每个站点都有一组具有相同功能的数据, 即水平分区）通过仅与协调服务器共享中间因子而联邦执行张量分解, 同时保持每个站点的数据私有.在现有的工作中, [272] 使用了基于交替方向乘数法（ADMM）的方法, [325] 使用弹性平均SGD（EASGD）算法提高了效率, 并进一步确保了中间因子的差异保密性.

2.3 Split Learning

In contrast with the previous settings which focus on data partitioning and communication patterns, the key idea behind split learning [215, 460][5] is to split the execution of a model on a per-layer basis between the clients and the server. This can be done for both training and inference.

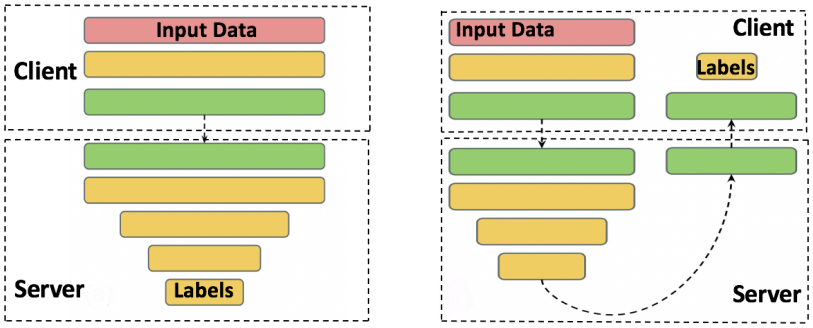
与之前专注于数据分区和通信模式的设置相比，拆分学习背后的关键思想 [215, 460][6] 是在客户端和服务器之间逐层拆分模型的执行。这可以用于训练和推理。

[5] See also split learning project website - https://splitlearning.github.io/.

[6] 另见拆分学习项目网站 - https://splitlearning.github.io/.

In the simplest configuration of split learning, each client computes the forward pass through a deep network up to a specific layer referred to as the *cut layer*. The outputs at the cut layer, referred to as *smashed data*, are sent to another entity (either the server or another client), which completes the rest of the computation. This completes a round of forward propagation without sharing the raw data. The gradients can then be back propagated from its last layer until the cut layer in a similar fashion. The gradients at the cut layer – and only these gradients – are sent back to the clients, where the rest of back propagation is completed. This process is continued until convergence, without having clients directly access each others raw data. This setup is shown in Figure 2(a) and a variant of this setup where labels are also not shared along with raw data is shown in Figure 2(b). Split learning approaches for data partitioned by features have been studied in [101].

在分裂学习的最简单配置中，每个客户端计算通过深度网络直到特定层的前向传递，称为 *cut layer*。切割层的输出，称为*粉碎数据*，被发送到另一个实体（服务器或另一个客户端），完成其余的计算。这样就完成了一轮前向传播，而无需共享原始数据。然后梯度可以以类似的方式从它的最后一层反向传播到切割层。切割层的梯度——只有这些梯度——被发送回客户端，在那里完成其余的反向传播。这个过程一直持续到收敛，客户端无需直接访问彼此的原始数据。此设置显示在 Figure 2(a) 中，此设置的一个变体，其中标签也不与原始数据共享，显示在 Figure 2(b) 中。已在 [101] 中研究了按特征划分的数据的拆分学习方法。



(a) Vanilla split learning (b) U-shaped split learning

图 2: Split learning configurations showing raw data is not transferred in the vanilla setting and that raw data as well as labels are not transferred between the client and server entities in the U-shaped split learning setting. 显示原始数据的拆分学习配置不会在 vanilla 设置中传输，并且原始数据以及标签不会在 U 形拆分学习设置中的客户端和服务器实体之间传输。

In several settings, the overall communication requirements of split learning and federated learning were compared in [421]. Split learning brings in another dimension of parallelism in the training, parallelization among parts of a model, e.g. client and server. The ideas in [245, 240], where the authors break the dependencies between partial networks and reduced total centralized training time by parallelizing the computations in different parts, can be relevant here as well. However, it is still an open question to explore such parallelization of split learning on edge devices. Split learning also enables matching client-side model components with the best server-side model components for automating model selection as shown in the ExpertMatcher [413].

在几种情况下，在 [421] 中比较了拆分学习和联邦学习的总体通信要求。拆分学习在训练中引入了另一个并行维度，即模型各部分之间的并行化，例如客户端和服务器。 [245, 240] 中的想法，作者打破了部分网络之间的依赖关系，并通过并行化不同部分的计算来减少总集中训练时间，在这里也很重要。然而，在边缘设备上探索这种分裂学习的并行化仍然是一个悬而未决的问题。拆分学习还支持将客户端模型组件与最佳服

务器端模型组件进行匹配，以自动选择模型，如 ExpertMatcher [413] 所示。

The values communicated can nevertheless, in general, reveal information about the underlying data. How much, and whether this is acceptable, is likely going to be application and configuration specific. A variation of split learning called NoPeek SplitNN [462] reduces the potential leakage via communicated activations, by reducing their distance correlation [461, 442] with the raw data, while maintaining good model performance via categorical cross-entropy. The key idea is to minimize the distance correlation between the raw data points and communicated smashed data. The objects communicated could otherwise contain information highly correlated with the input data if used without NoPeek SplitNN, the use of which also enables the split to be made relatively early-on given the decorrelation it provides. One other engineering driven approach to minimize the amount of information communicated in split learning has been via a specifically learnt pruning of channels present in the client side activations [422]. Overall, much of the discussion in Section 4 is relevant here as well, and analysis providing formal privacy guarantees specifically for split learning is still an open problem.

然而，通常，所传达的值可以揭示有关基础数据的信息。多少以及这是否可以接受，可能取决于应用程序和配置。一种称为 NoPeek SplitNN [462] 的分裂学习变体通过减少它们与原始数据的距离相关性 [461, 442] 来减少通过交流激活的潜在泄漏，同时通过分类交叉熵保持良好的模型性能。关键思想是最小化原始数据点和传达的粉碎数据之间的距离相关性。如果在没有 NoPeek SplitNN 的情况下使用，则通信的对象可能包含与输入数据高度相关的信息，鉴于它提供的去相关性，使用 NoPeek SplitNN 还可以相对较早地进行拆分。另一种工程驱动的方法是通过对客户端激活[422] 中存在的通道进行专门学习的剪枝来最小化拆分学习中传达的信息量。总体而言，Section 4 中的大部分讨论也与此处相关，并且专门为拆分学习提供正式隐私保证的分析仍然是一个悬而未决的问题。

2.4 Executive summary

The motivation for federated learning is relevant for a number of related areas of research.

- Fully decentralized learning (Section 2.1) removes the need for a central server coordinating the overall computation. Apart from algorithmic challenges, open problems are in practical realization of the idea and in understanding of what form of trusted central authority is needed to set up the task.
- Cross-silo federated learning (Section 2.2) admits problems with different kinds of modelling constraints, such as data partitioned by examples and/or features, and faces different set of concerns when formulating formal privacy guarantees or incentive mechanisms for clients to participate.
- Split learning (Section 2.3) is an approach to partition the execution of a model between the clients and the server. It can deliver different options for overall communication constraints, but detailed analysis of when the communicated values reveal sensitive information is still missing.

- 完全分散的学习（Section 2.1）消除了对协调整体计算的中央服务器的需要。除了算法挑战之外，开放性问题还在于该想法的实际实现以及对设置任务所需的可信中央权威形式的理解。
- 跨筒仓联邦学习 (Section 2.2) 承认不同类型的建模约束存在问题，例如按示例和/或特征划分的数据，并在制定正式的隐私保证或客户参与的激励机制。

- 拆分学习 (Section 2.3) 是一种在客户端和服务器之间划分模型执行的方法。它可以为整体通信约束提供不同的选项，但仍然缺少对通信值何时揭示敏感信息的详细分析。

3 Improving Efficiency and Effectiveness

In this section we explore a variety of techniques and open questions that address the challenge of making federated learning more efficient and effective. This encompasses a myriad of possible approaches, including: developing better optimization algorithms; providing different models to different clients; making ML tasks like hyperparameter search, architecture search, and debugging easier in the FL context; improving communication efficiency; and more.

在本节中，我们将探讨各种技术和开放性问题，以解决使联邦学习更加高效和有效的挑战。这包括无数可能的方法，包括：开发更好的优化算法；为不同的客户提供不同的模型；使 FL 上下文中的超参数搜索、架构搜索和调试等 ML 任务更容易；提高沟通效率；和更多。

One of the fundamental challenges in addressing these goals is the presence of non-IID data, so we begin by surveying this issue and highlighting potential mitigations. 实现这些目标的基本挑战之一是非 IID 数据的存在，因此我们首先调查这个问题并强调潜在的缓解措施。

3.1 Non-IID Data in Federated Learning

While the meaning of IID is generally clear, data can be non-IID in many ways. In this section, we provide a taxonomy of non-IID data regimes that may arise for any client-partitioned dataset. The most common sources of dependence and non-identicalness are due to each client corresponding to a particular user, a particular geographic location, and/or a particular time window. This taxonomy has a close mapping to notions of dataset shift [353, 380], which studies differences between the training distribution and testing distribution; here, we consider differences in the data distribution on each client.

虽然IID的含义通常是明确的, 但数据可以在许多方面是非IID的。在本节中, 我们提供了任何客户机分区数据集可能出现的非IID数据模式的分类。依赖性和不一致性的最常见来源是对应于特定用户、特定地理位置和/或特定时间窗口的每个客户端。该分类法与dataset shift [353, 380] 的概念有密切的对应关系, 后者研究训练分布和测试分布之间的差异；在这里, 我们考虑在每个客户端上的数据分布的差异。

For the following, consider a supervised task with features x and labels y. A statistical model of federated learning involves two levels of sampling: accessing a datapoint requires first sampling a client $i \sim \mathcal{Q}$, the distribution over available clients, and then drawing an example $(x, y) \sim Ploc_i(x, y)$ from that client's local data distribution.

下面, 考虑一个具有x和标签Y的监督任务。联邦学习的统计模型涉及两个级别的采样：访问数据点需要首先对客户机$i \sim \mathcal{Q}$进行采样, 即在可用客户机上的分布, 然后从该客户机的本地数据分布中绘制一个示例$(x, y) \sim Ploc_i(x, y)$ 。

When non-IID data in federated learning is referenced, this typically refers to differences between $\mathcal{P}_i$ and $\mathcal{P}_j$ for different clients i and j. However, it is also important to note that the distribution $\mathcal{Q}$ and $\mathcal{P}_i$ may change over time, introducing another dimension of "non-IIDness".

当引用联邦学习中的非IID数据时, 这通常是指不同客户端i和j的$\mathcal{P}_i$和$\mathcal{P}_j$之间的差异。但是, 还需要注意的是, 分布$\mathcal{Q}$和$\mathcal{P}_i$可能会随着时间的推移而改变, 从而引入另一个"non-IIDness"维度。

For completeness, we note that even considering the dataset on a single device, if the data is in an insufficiently-random order, e.g. ordered by time, then independence is violated locally as well. For example, consecutive frames in

a video are highly correlated. Sources of intra-client correlation can generally be resolved by local shuffling.

为了完整性, 我们注意到, 即使考虑单个设备上的数据集, 如果数据的随机顺序不够, 例如按时间排序, 那么独立性也会在局部受到破坏。例如, 视频中的连续帧高度相关。客户机内部相关性的来源通常可以通过本地洗牌来解决。

Non-identical client distributions We first survey some common ways in which data tend to deviate from being identically distributed, that is $P_i \neq P_j$ for different clients i and j. Rewriting $P_i(x, y)$ as $P_i(y \mid x)P_i(x)$ and $P_i(x \mid y)P_i(y)$ allows us to characterize the differences more precisely.

我们首先调查了数据倾向于偏离同分布的一些常见方式，即不同客户端 i 和 j 的 $P_i \neq P_j$。将 $P_i(x, y)$ 重写为 $P_i(y \mid x)P_i(x)$ 和 $P_i(x \mid y)P_i(y)$ 可以让我们更精确地表征差异。

- *Feature distribution skew* (covariate shift): The marginal distributions $\mathcal{P}_i(x)$ may vary across clients, even if $\mathcal{P}(y \mid x)$ is shared.[7] For example, in a handwriting recognition domain, users who write the same words might still have different stroke width, slant, etc.

- *Label distribution skew* (prior probability shift): The marginal distributions $\mathcal{P}_i(y)$ may vary across clients, even if $\mathcal{P}(x \mid y)$ is the same. For example, when clients are tied to particular geo-regions, the distribution of labels varies across clients — kangaroos are only in Australia or zoos; a person's face is only in a few locations worldwide; for mobile device keyboards, certain emoji are used by one demographic but not others.

- *Same label, different features* (concept drift): The conditional distributions $\mathcal{P}_i(x \mid y)$ may vary across clients even if $\mathcal{P}(y)$ is shared. The same label y can have very different features x for different clients, e.g. due to cultural differences, weather effects, standards of living, etc. For example, images of homes can vary dramatically around the world and items of clothing vary widely. Even within the U.S., images of parked cars in the winter will be snow-covered only in certain parts of the country. The same label can also look very different at different times, and at different time scales: day vs. night, seasonal effects, natural disasters, fashion and design trends, etc.

- *Same features, different label* (concept shift): The conditional distribution $\mathcal{P}_i(y \mid x)$ may vary across clients, even if $\mathcal{P}(x)$ is the same. Because of personal preferences, the same feature vectors in a training data item can have different labels. For example, labels that reflect sentiment or next word predictors have personal and regional variation.

- *Quantity skew* or unbalancedness: Different clients can hold vastly different amounts of data.

- 特征分布偏斜(协变量偏移)：即使 $\mathcal{P}(y \mid x)$ 是共享的，边际分布 $\mathcal{P}_i(x)$ 可能因客户端而异。[8] 对于例如，在手写识别领域，写相同单词的用户可能仍然具有不同的笔画宽度、倾斜度等。

- 标签分布偏斜(先验概率偏移)：边缘分布 $\mathcal{P}_i(y)$ 可能因客户端而异，即使 $\mathcal{P}(x \mid y)$ 是相同的。例如，当客户被绑定到特定的地理区域时，标签的分布因客户而异——袋鼠只在澳大利亚或动物园；一个人的脸只出现在世界范围内的几个地方；对于移动设备键盘，某些表情符号仅由某一人群使用，而其他人群则不会。

[7] We write "$\mathcal{P}(y \mid x)$ is shared" as shorthand for $\mathcal{P}_i(y \mid x) = \mathcal{P}_j(y \mid x)$ for all clients i and j.

[8] 我们写' "$\mathcal{P}(y \mid x)$ is shared" 作为 $\mathcal{P}_i(y \mid x) = \mathcal{P}_j(y \mid x)$ 的简写，适用于所有客户端 i 和 j。

- 相同的标签，不同的特征(概念漂移)：即使 $\mathcal{P}(y)$ 是共享的，条件分布 $\mathcal{P}_i(x \mid y)$ 也可能因客户端而异。对于不同的客户端，相同的标签 y 可以具有非常不同的特征 x，例如由于文化差异、天气影响、生活水平等原因。例如，世界各地的房屋图像可能会有很大差异，衣服也有很大差异。即使在美国，冬天停放的汽车的图像也只会在该国的某些地区被雪覆盖。同一个标签在不同的时间和不同的时间尺度上也可能看起来非常不同：白天与黑夜、季节性影响、自然灾害、时尚和设计趋势等。
- 相同特征，不同标签(概念转变)：条件分布 $\mathcal{P}_i(y \mid x)$ 可能因客户端而异，即使 $\mathcal{P}(x)$ 相同。由于个人喜好，训练数据项中相同的特征向量可以有不同的标签。例如，反映情绪或下一个词预测器的标签具有个人和区域差异。
- *Quantity skew* 或不平衡性：不同的客户端可以持有截然不同的数据量。

Real-world federated learning datasets likely contain a mixture of these effects, and the characterization of cross-client differences in real-world partitioned datasets is an important open question. Most empirical work on synthetic non-IID datasets (e.g. [337, 236]) have focused on label distribution skew, where a non-IID dataset is formed by partitioning a "flat" existing dataset based on the labels. A better understanding of the nature of real-world non-IID datasets will allow for the construction of controlled but realistic non-IID datasets for testing algorithms and assessing their resilience to different degrees of client heterogeneity.

真实世界的联邦学习数据集可能包含这些影响的混合, 而真实世界分区数据集中跨客户端差异的表征是一个重要的开放问题。大多数关于合成非IID数据集的实证研究(例如, [337, 236])都集中在标签分布偏斜上, 其中非IID数据集是通过基于标签划分“平面”现有数据集形成的。更好地理解现实世界中非IID数据集的性质, 将有助于构建受控但现实的非IID数据集, 用于测试算法并评估其对不同程度的客户端异构性的恢复能力。

Further, different non-IID regimes may require the development of different mitigation strategies. For example, under feature-distribution skew, because $\mathcal{P}(y \mid x)$ is assumed to be common, the problem is at least in principle well specified, and training a single global model that learns $\mathcal{P}(y \mid x)$ may be appropriate. When the same features map to different labels on different clients, some form of personalization (Section 3.3) may be essential to learning the true labeling functions.

此外, 不同的非IID制度可能需要制定不同的缓解策略。例如, 在特征分布偏斜下, 由于假定 $\mathcal{P}(y \mid x)$ 是常见的, 因此问题至少在原则上是明确的, 并且训练学习 $\mathcal{P}(y \mid x)$ 的单个全局模型可能是合适的。当相同的功能映射到不同客户端上的不同标签时, 某种形式的个性化 (Section 3.3) 对于学习真正的标签功能可能是必不可少的。

Violations of independence Violations of independence are introduced any time the distribution $\mathcal{Q}$ changes over the course of training; a prominent example is in cross-device FL, where devices typically need to meet eligibility requirements in order to participate in training (see Section 1.1.2). Devices typically meet those requirements at night local time (when they are more likely to be charging, on free wi-fi, and idle), and so there may be significant diurnal patterns in device availability. Further, because local time of day corresponds directly to longitude, this introduces a strong geographic bias in the source of the data. Eichner et al. [171] described this issue and some mitigation strategies, but many open questions remain.

在培训过程中, 只要分配$\mathcal{Q}$发生变化, 就会出现违反独立性的情况；一个突出的例子是在跨设备FL中, 设备通常需要满足资格要求才能参加培训(请参见Section 1.1.2)。设备通常在夜间本地时间(当它们更有可能充

电、使用免费wi-fi和空闲时)满足这些要求, 因此设备可用性可能存在明显的日间模式。此外, 由于当地时间与经度直接对应, 因此在数据源中引入了强烈的地理偏差 Eichner et al. [171] 描述了这个问题和一些缓解策略, 但仍然存在许多悬而未决的问题。

Dataset shift Finally, we note that the temporal dependence of the distributions $\mathcal{Q}$ and $\mathcal{P}$ may introduce dataset shift in the classic sense (differences between the train and test distributions). Furthermore, other criteria may make the set of clients eligible to train a federated model different from the set of clients where that model will be deployed. For example, training may require devices with more memory than is needed for inference. These issues are explored in more depth in Section 6. Adapting techniques for handling dataset shift to federated learning is another interesting open question.

最后, 我们注意到$\mathcal{Q}$和$\mathcal{P}$分布的时间依赖性可能会引入经典意义上的数据集转移(训练分布和测试分布之间的差异)。此外, 其他标准可能使客户机集有资格培训联邦模型, 而不是部署该模型的客户机集。例如, 训练可能需要比推理所需内存更多的设备。这些问题将在Section 6中进行更深入的探讨。另一个有趣的开放性问题是如何调整处理数据集迁移到联邦学习的技术。

3.1.1 Strategies for Dealing with Non-IID Data

The original goal of federated learning, training a single global model on the union of client datasets, becomes harder with non-IID data. One natural approach is to modify existing algorithms (e.g. through different hyperparameter choices) or develop new ones in order to more effectively achieve this objective. This approach is considered in Section 3.2.2.

联邦学习的最初目标, 即在客户机数据集的联邦上训练单个全局模型, 在使用非IID数据时变得更加困难。一种自然的方法是修改现有算法(例如通过不同的超参数选择)或开发新算法, 以便更有效地实现这一目标。此方法在Section 3.2.2中考虑。

For some applications, it may be possible to augment data in order to make the data across clients more similar. One approach is to create a small dataset which can be shared globally. This dataset may originate from a publicly available proxy data source, a separate dataset from the clients' data which is not privacy sensitive, or perhaps a distillation of the raw data following Wang et al. [473].

对于某些应用程序, 可能会增加数据, 以便使跨客户端的数据更相似。一种方法是创建一个可以全局共享的小数据集。该数据集可能来自公开可用的代理数据源、与客户数据无关的独立数据集, 或者可能来自Wang et al. [473]之后原始数据的提炼。

The heterogeneity of client objective functions gives additional importance to the question of how to craft the objective function — it is no-longer clear that treating all examples equally makes sense. Alternatives include limiting the contributions of the data from any one user (which is also important for privacy, see Section 4) and introducing other notions of fairness among the clients; see discussion in Section 6.

客户目标函数的异质性给如何构建目标函数的问题带来了额外的重要性——现在已经不清楚平等对待所有示例是否有意义。备选方案包括限制来自任何一个用户的数据贡献(这对隐私也很重要, 请参见Section 4), 并在客户中引入其他公平概念；请参阅Section 6中的讨论。

But if we have the capability to run training on the local data on each device (which is necessary for federated

learning of a global model), is training a single global model even the right goal? There are many cases where having a single model is to be preferred, e.g. in order to provide a model to clients with no data, or to allow manual validation and quality assurance before deployment. Nevertheless, since local training is possible, it becomes feasible for each client to have a customized model.

This approach can turn the non-IID problem from a bug to a feature, almost literally — since each client has its own model, the client's identity effectively parameterizes the model, rendering some pathological but degenerate non-IID distributions trivial. For example, if for each i, $\mathcal{P}_i(y)$ has support on only a single label, finding a high-accuracy global model may be very challenging (especially if x is relatively uninformative), but training a high-accuracy local model is trivial (only a constant prediction is needed). Such multi-model approaches are considered in depth in Section 3.3. In addition to addressing non-identical client distributions, using a plurality of models can also address violations of independence stemming from changes in client availability. For example, the approach of Eichner et al. [171] uses a single training run but averages different iterates in order to provide different models for inference based on the timezone / longitude of clients.

但是, 如果我们能够在每个设备上的本地数据上运行训练(这对于全局模型的联邦学习是必要的), 训练单个全局模型是否是正确的目标？在许多情况下, 最好使用单一模型, 例如, 为了向没有数据的客户提供模型, 或允许在部署前进行手动验证和质量保证。然而, 由于本地培训是可能的, 因此每个客户都有一个定制的模型是可行的。

这种方法可以将非IID问题从一个bug变成一个特性, 几乎是字面上的——因为每个客户机都有自己的模型, 客户机的身份有效地参数化了模型, 使得一些病态但退化的非IID分布变得微不足道。例如, 如果对于每个i, $\mathcal{P}_i(y)$只支持一个标签, 则查找高精度全局模型可能非常具有挑战性(特别是如果x相对缺乏信息), 但训练高精度局部模型很简单(只需要不断的预测)。此类多模型方法在Section 3.3中被深入考虑。除了解决不相同的客户机分布之外, 使用多个模型还可以解决由于客户机可用性的变化而导致的独立性违规问题。例如, Eichner et al. [171]的方法使用单个训练运行, 但对不同的迭代进行平均, 以便根据客户端的时区/经度提供不同的推理模型。

3.2 Optimization Algorithms for Federated Learning

In prototypical federated learning tasks, the goal is to learn a single global model that minimizes the empirical risk function over the entire training dataset, that is, the union of the data across all the clients. The main difference between federated optimization algorithms and standard distributed training methods is the need to address the characteristics of Table 1 — for optimization, non-IID and unbalanced data, limited communication bandwidth, and unreliable and limited device availability are particularly salient.

在原型联邦学习任务中, 目标是学习单个全局模型, 该模型最小化整个训练数据集上的经验风险函数, 即所有客户机上的数据联邦。联邦优化算法和标准分布式训练方法之间的主要区别在于需要解决表 1的特性——对于优化, 非IID和不平衡数据、有限的通信带宽以及不可靠和有限的设备可用性尤为突出。

FL settings where the total number of devices is huge (e.g. across mobile devices) necessitate algorithms that only require a handful of clients to participate per round (client sampling). Further, each device is likely to participate no more than once in the training of a given model, so stateless algorithms are necessary. This rules out the direct application of a variety of approaches that are quite effective in the datacenter context, for example stateful optimization algorithms like ADMM, and stateful compression strategies that modify updates based on residual compression errors

from previous rounds.

如果FL设置中的设备总数巨大(例如, 跨移动设备), 则需要每轮只需要少数客户端参与的算法(客户端采样)。此外, 每个设备可能只参与一次给定模型的训练, 因此无状态算法是必要的。这排除了直接应用在数据中心环境中非常有效的各种方法的可能性, 例如ADMM之类的有状态优化算法, 以及基于前几轮剩余压缩错误修改更新的有状态压缩策略。

Another important practical consideration for federated learning algorithms is composability with other techniques. Optimization algorithms do not run in isolation in a production deployment, but need to be combined with other techniques like cryptographic secure aggregation protocols (Section 4.2.1), differential privacy (DP) (Section 4.2.2), and model and update compression (Section 3.5). As noted in Section 1.1.2, many of these techniques can be applied to primitives like "`sum over selected clients`" and "`broadcast to selected clients`", and so expressing optimization algorithms in terms of these primitives provides a valuable separation of concerns, but may also exclude certain techniques such as applying updates asynchronously.

联邦学习算法的另一个重要的实际考虑是与其他技术的可组合性。优化算法不会在生产部署中单独运行, 但需要与其他技术相结合, 如加密安全聚合协议(Section 4.2.1)、差分隐私(DP)(Section 4.2.2)以及模型和更新压缩(Section 3.5)。如第 1.1.2节所述, 其中许多技术可以应用于诸如"`sum over selected clients`"和"`broadcast to selected clients`"之类的原语, 因此根据这些原语表达优化算法提供了有价值的关注点分离, 但也可能排除某些技术, 如异步应用更新。

One of the most common approaches to optimization for federated learning is the Federated Averaging algorithm [337], an adaption of local-update or parallel SGD.[9] Here, each client runs some number of SGD steps locally, and then the updated local models are averaged to form the updated global model on the coordinating server. Pseudocode is given in Algorithm 1.

联邦学习优化的最常用方法之一是联邦平均算法, 它是本地更新或并行SGD的一种自适应。[10], 每个客户机在本地运行一定数量的SGD步骤, 然后对更新的本地模型进行平均, 以在协调服务器上形成更新的全局模型。伪码在算法 1中给出。

Performing local updates and communicating less frequently with the central server addresses the core challenges of respecting data locality constraints and of the limited communication capabilities of mobile device clients. However, this family of algorithms also poses several new algorithmic challenges from an optimization theory point of view. In Section 3.2, we discuss recent advances and open challenges in federated optimization algorithms for the cases of IID and non-IID data distribution across the clients respectively. The development of new algorithms that specifically target the characteristics of the federated learning setting remains an important open problem.

执行本地更新并减少与中央服务器的通信频率, 解决了尊重数据位置约束和移动设备客户端有限通信能力的核心挑战。然而, 从优化理论的角度来看, 这一系列算法也带来了一些新的算法挑战。在第 3.2节中, 我们分别讨论了IID和非IID数据跨客户端分布情况下联邦优化算法的最新进展和公开挑战。开发专门针对联邦学习环境特征的新算法仍然是一个重要的开放问题。

[9] Federated Averaging applies local SGD to a randomly sampled subset of clients on each round, and proposes a specific update weighting scheme.

[10] 联邦平均将本地SGD应用于每轮随机抽样的客户端子集, 并提出了一种特定的更新加权方案。

N	Total number of clients
M	Clients per round
T	Total communication rounds
K	Local steps per round.

表 5: Notation for the discussion of FL algorithms including Federated Averaging.

Server executes:
initialize x_0
for each round $t = 1, 2, \ldots,$ T **do**
  $S_t \leftarrow$ (random set of M clients)
  for each client $i \in S_t$ **in parallel do**
    $x_{t+1}^i \leftarrow \text{ClientUpdate}(i, x_t)$
  $x_{t+1} \leftarrow \sum_{k=1}^{M} \frac{1}{M} x_{t+1}^i$

ClientUpdate(i, x)**:**
for local step $j = 1, \ldots, K$ **do**
  $x \leftarrow x - \eta \nabla f(x; z)$ for $z \sim \mathcal{P}_i$
return x to server

Algorithm 1: Federated Averaging (local SGD), when all clients have the same amount of data.

3.2.1 Optimization Algorithms and Convergence Rates for IID Datasets

While a variety of different assumptions can be made on the per-client functions being optimized, the most basic split is between assuming IID and non-IID data. Formally, having IID data at the clients means that each mini-batch of data used for a client's local update is statistically identical to a uniformly drawn sample (with replacement) from the entire training dataset (the union of all local datasets at the clients). Since the clients independently collect their own training data which vary in both size and distribution, and these data are not shared with other clients or the central node, the IID assumption clearly almost never holds in practice. However, this assumption greatly simplifies theoretical convergence analysis of federated optimization algorithms, as well as establishes a baseline that can be used to understand the impact of non-IID data on optimization rates. Thus, a natural first step is to obtain an understanding of the landscape of optimization algorithms for the IID data case.

虽然可以对正在优化的每个客户端函数进行各种不同的假设, 但最基本的划分是假设IID和非IID数据。从形式上讲, 在客户端拥有IID数据意味着用于客户端本地更新的每个小批量数据在统计上与来自整个培训数据集(客户端所有本地数据集的联邦)的统一抽取样本(带替换)相同。由于客户独立收集自己的培训数据, 这些数据在大小和分布上都有所不同, 并且这些数据没有与其他客户或中心节点共享, 因此IID假设显然在实践中几乎不成立。然而, 该假设大大简化了联邦优化算法的理论收敛性分析, 并建立了一个基线, 可用于了解非IID数据对优化速率的影响。因此, 自然的第一步是了解IID数据情况下优化算法的前景。

Formally, for the IID setting let us standardize the stochastic optimization problem

$$\min_{x \in \mathbb{R}^m} F(x) := \mathop{\mathbb{E}}_{z \sim \mathcal{P}} [f(x; z)] \,.$$

We assume an intermittent communication model as in e.g. Woodworth et al. [480, Sec. 4.4], where M stateless clients participate in each of T rounds, and during each round, each client can compute gradients for K samples (e.g. minibatches) $z_1, \ldots, z_K$ sampled IID from $\mathcal{P}$ (possibly using these to take sequential steps). In the IID-data setting clients are interchangeable, and we can without loss of generality assume $M = N$. Table 5 summarizes the notation used in this section.

我们假设一个间歇性的通信模型, 如Woodworth et al. [480, Sec. 4.4], 其中M无状态客户端参与T轮中的每

一轮, 并且在每轮中, 每个客户端都可以计算K样本(例如小批量)$z_1, \ldots, z_K$ 从$\mathcal{P}$取样的IID的梯度在IID数据设置中, 客户机是可互换的, 我们可以在不丧失一般性的情况下假设$M = N$。Table 5 总结了本节中使用的符号。

Different assumptions on f will produce different guarantees. We will first discuss the convex setting and later review results for non-convex problems.

对f的不同假设将产生不同的保证。我们将首先讨论凸设置, 然后回顾非凸问题的结果。

Baselines and state-of-the-art for convex problems In this section we review convergence results for H-smooth, convex (but not necessarily strongly convex) functions under the assumption that the variance of the stochastic gradients is bounded by σ^2. More formally, by H-smooth we mean that for all z, $f(\cdot; z)$ is differentiable and has a H-Lipschitz gradient, that is, for all choices of x, y

$$\|\nabla f(x, z) - \nabla f(y, z)\| \leq H\|x - y\|.$$

We also assume that for all x, the stochastic gradient $\nabla_x f(x; z)$ satisfies

$$\mathop{\mathbb{E}}_{z\sim\mathcal{P}} \|\nabla_x f(x; z) - \nabla F(x)\| \leq \sigma^2.$$

When analyzing the convergence rate of an algorithm with output x_T after T iterations, we consider the term

$$\mathbb{E}[F(x_T)] - F(x^*) \tag{1}$$

where $x^* = \arg\min_x F(x)$. All convergence rates discussed herein are upper bounds on this term. A summary of convergence results for such functions is given in Table 6.

Federated averaging (a.k.a. parallel SGD/local SGD) competes with two natural baselines: First, we may keep x fixed in local updates during each round, and compute a total of KM gradients at the current x, in order to run accelerated minibatch SGD. Let $\bar{x}$ denote the average of T iterations of this algorithm. We then have the upper bound

$$\mathcal{O}\left(\frac{H}{T^2} + \frac{\sigma}{\sqrt{TKM}}\right)$$

for convex objectives [294, 137, 151]. Note that the first expectation is taken with respect to the randomness of z in the training procedure as well.

A second natural baseline is to ignore all but 1 of the M active clients, which allows (accelerated) sequential SGD to execute for KT steps. Applying the same general bounds cited above, this approach offers an upper bound of

$$\mathcal{O}\left(\frac{H}{(TK)^2} + \frac{\sigma}{\sqrt{TK}}\right).$$

Comparing these two results, we see that minibatch SGD attains the optimal 'statistical' term ($\sigma/\sqrt{TKM}$), whilst SGD on a single device (ignoring the updates of the other devices) achieves the optimal 'optimization' term ($H/(TK)^2$).

The convergence analysis of local-update SGD methods is an active current area of research [434, 310, 500, 467, 390, 371, 269, 481]. The first convergence results for local-update SGD methods were derived under the bounded

Method	Comments	Convergence
Baselines		
mini-batch SGD	batch size KM	$\mathcal{O}\left(\frac{H}{T} + \frac{\sigma}{\sqrt{TKM}}\right)$
SGD	(on 1 worker, no communication)	$\mathcal{O}\left(\frac{H}{TK} + \frac{\sigma}{\sqrt{TK}}\right)$
Baselines with acceleration[a]		
A-mini-batch SGD [294, 137]	batch size KM	$\mathcal{O}\left(\frac{H}{T^2} + \frac{\sigma}{\sqrt{TKM}}\right)$
A-SGD [294]	(on 1 worker, no communication)	$\mathcal{O}\left(\frac{H}{(TK)^2} + \frac{\sigma}{\sqrt{TK}}\right)$
Parallel SGD / Fed-Avg / Local SGD		
Yu et al. [500][b], Stich [434][c]	gradient norm bounded by G	$\mathcal{O}\left(\frac{HKM}{T}\frac{G^2}{\sigma^2} + \frac{\sigma}{\sqrt{TKM}}\right)$
Wang and Joshi [467][b], Stich and Karimireddy [435]		$\mathcal{O}\left(\frac{HM}{T} + \frac{\sigma}{\sqrt{TKM}}\right)$
Other algorithms		
SCAFFOLD [265]	control variates and two stepsizes	$\mathcal{O}\left(\frac{H}{T} + \frac{\sigma}{\sqrt{TKM}}\right)$

[a] There are no accelerated fed-avg/local SGD variants so far

[b] This paper considers the smooth non-convex setting, we adapt here the results for our setting.

[c] This paper considers the smooth strongly convex setting, we adapt here the results for our setting.

表 6: Convergence rates for a (non-comprehensive) set of distributed optimization algorithms in the IID-data setting. We assume M devices participate in each iterations, and the loss functions are H-smooth, convex, and we have access to stochastic gradients with variance at most σ^2. All rates are upper bounds on (1) after T iterations (potentially with some iterate averaging scheme).

gradient norm assumption in Stich [434] for strongly-convex and in Yu et al. [500] for non-convex objective functions. These analyses could attain the desired $\sigma/\sqrt{TKM}$ statistical term with suboptimal optimization term (in Table 6 we summarize these results for the middle ground of convex functions).

By removing the bounded gradient assumption, Wang and Joshi [467] and Stich and Karimireddy [435] could further improve the optimization term to HM/T. These result show that if the number of local steps K is smaller than T/M^3 then the (optimal) statistical term is dominating the rate. However, for typical cross-device applications we might have $T = 10^6$ and $M = 100$ (Table 3), implying $K = 1$.

Often in the literature the convergence bounds are accompanied by a discussion on how large K may be chosen in order to reach asymptotically the same statistical term as the convergence rate of mini-batch SGD. For strongly convex functions, this bound was improved by Khaled et al. [269] and further in Stich and Karimireddy [435].

For non-convex objectives, Yu et al. [500] showed that local SGD can achieve asymptotically an error bound $1/\sqrt{TKM}$ if the number of local updates K are smaller than $T^{1/3}/M$. This convergence guarantee was further improved by Wang and Joshi [467] who removed the bounded gradient norm assumption and showed that the number of local updates can be as large as T/M^3. The analysis in [467] can also be applied to other algorithms with local updates, and thus yields the first convergence guarantee for decentralized SGD with local updates (or periodic decentralized SGD) and elastic averaging SGD [505]. Haddadpour et al. [216] improves the bounds in Wang and Joshi [467] for functions satisfying the Polyak-Lojasiewicz (PL) condition [262], a generalization of strong convexity. In particular, Haddadpour et al. [216] show that for PL functions, T^2/M local updates per round leads to a $\mathcal{O}(1/TKM)$

convergence.

While the above works focus on convergence as a function of the number of iterations performed, practitioners often care about wall-clock convergence speed. Assessing this must take into account the effect of the design parameters on the time spent per iteration based on the relative cost of communication and local computation. Viewed in this light, the focus on seeing how large K can be while maintaining the statistical rate may not be the primary concern in federated learning, where one may assume almost infinite datasets (very large N). The costs (at least in wall-clock time) are small for increasing M, and so it may be more natural to increase M sufficiently to match the optimization term, and then tune K to maximize wall-clock optimization performance. How then to choose K? Performing more local updates at the clients will increase the divergence between the resulting local models at the clients, before they are averaged. As a result, the error convergence in terms of training loss versus the total number of sequential SGD steps TK is slower. However, performing more local updates saves significant communication cost and reduces the time spent per iteration. The optimal number of local updates strikes a balance between these two phenomena and achieves the fastest error versus wallclock time convergence. Wang and Joshi [468] propose an adaptive communication strategy that adapts K according to the training loss at regular intervals during the training.

Another important design parameter in federated learning is the model aggregation method used to update the global model using the updates made by the selected clients. In the original federated learning paper, McMahan et al. [337] proposes taking a weighted average of the local models, in proportion to the size of local datasets. For IID data, where each client is assumed to have a infinitely large dataset, this reduces to taking a simple average of the local models. However, it is unclear whether this aggregation method will result in the fastest error convergence.

There are many open questions in federated optimization, even with IID data. Woodworth et al. [480] highlights several gaps between upper and lower bounds for optimization relevant to the federated learning setting, particularly for "intermittent communication graphs", which captures local SGD approaches, but convergence rates for such approaches are not known to match the corresponding lower bounds. In Table 6 we highlight convergence results for the convex setting. Whilst most schemes are able to reach the asymptotically dominant statistical term, none are able to match the convergence rate of accelerated mini-batch SGD. It is an open problem if federated averaging algorithms can close this gap.

Local-update SGD methods where all M clients perform the same number of local updates may suffer from a common scalability issue—they can be bottlenecked if any one client unpredictably slows down or fails. Several approaches for dealing with this are possible, but it is far from clear which are optimal, especially when the potential for bias is considered (see Section 6). Bonawitz et al. [81] propose over-provisioning clients (e.g., request updates from $1.3M$ clients), and then accepting the first M updates received and rejecting updates from stragglers. A slightly more sophisticated solution is to fix a time window and allow clients to perform as many local updates K_i as possible within this time, after which their models are averaged by a central server. Wang et al. [471] analyzed the computational heteogeneity introduced by this approach in theory. An alternative method to overcome the problem of straggling clients is to fix the number of local updates at τ, but allow clients to update the global model in an asynchronous or lock-free fashion. Although some previous works [505, 306, 163] have proposed similar methods, the error convergence analysis is an open and challenging problem. A larger challenge in the FL setting, however, is that as discussed at the beginning of Section 3.2, asynchronous approaches may be difficult to combine with complimentary techniques like differential privacy or secure aggregation.

Besides the number of local updates, the choice of the size of the set of clients selected per training round presents

a similar trade-off as the number of local updates. Updating and averaging a larger number of client models per training round yields better convergence, but it makes the training vulnerable to slowdown due to unpredictable tail delays in computation/communication at/with the clients.

The analysis of local SGD / Federated Averaging in the non-IID setting is even more challenging; results and open questions related to this are considered in the next section, along with specialized algorithms which directly address the non-IID problem.

3.2.2 Optimization Algorithms and Convergence Rates for Non-IID Datasets

In contrast to well-shuffled mini-batches consisting of independent and identically distributed (IID) examples in centralized learning, federated learning uses local data from end user devices, leading to many varieties of non-IID data (Section 3.1).

In this setting, each of N clients has a local data distribution $\mathcal{P}_i$ and a local objective function

$$f_i(x) = \mathop{\mathbb{E}}_{z \sim \mathcal{P}_i} [f(x; z)]$$

where we recall that $f(x; z)$ is the loss of a model x at an example z. We typically wish to minimize

$$F(x) = \frac{1}{N} \sum_{i=1}^{N} f_i(x) \,. \tag{2}$$

Note that we recover the IID setting when each $\mathcal{P}_i$ is identical. We will let F^* denote the minimum value of F, obtained the point x^*. Analogously, we will let f_i^* denote the minimum value of f_i.

As in the IID setting, we assume an intermittent communication model (e.g. Woodworth et al. [480, Sec. 4.4]), where M stateless clients participate in each of T rounds, and during each round, each client can compute gradients for K samples (e.g. minibatches). The difference here is that the samples $z_{i,1}, \ldots, z_{i,K}$ sampled at client i are drawn from the client's local distribution $\mathcal{P}_i$. Unlike the IID setting, we cannot necessarily assume $M = N$, as the client distributions are not all equal. In the following, if an algorithm relies on $M = N$, we will omit M and simply write N. We note that while such an assumption may be compatible with the cross-silo federated setting in Table 1, it is generally infeasible in the cross-device setting.

While [434, 500, 467, 435] mainly focused on the IID case, the analysis technique can be extended to the non-IID case by adding an assumption on data dissimilarities, for example by constraining the difference between client gradients and the global gradient [305, 300, 304, 469, 471] or the difference between client and global optimum values [303, 268]. Under this assumption, Yu et al. [501] showed that the error bound of local SGD in the non-IID case becomes worse. In order to achieve the rate of $1/\sqrt{TKN}$ (under non-convex objectives), the number of local updates K should be smaller than $T^{1/3}/N$, instead of T/N^3 as in the IID case [467]. Li et al. [300] proposed to add a proximal term in each local objective function so as to make the algorithm be more robust to the heterogeneity across local objectives. The proposed FedProx algorithm empirically improves the performance of federated averaging. Khaled et al. [268] assumes all clients participate, and uses batch gradient descent on clients, which can potentially converge faster than stochastic gradients on clients.

Recently, a number of works have made progress in relaxing the assumptions necessary for analysis so as to better

Non-IID assumptions

Symbol	Full name	Explanation
BCGV	bounded inter-client gradient variance	$\mathbb{E}_i \|\nabla f_i(x) - \nabla F(x)\|^2 \leq \eta^2$
BOBD	bounded optimal objective difference	$F^* - \mathbb{E}_i[f_i^*] \leq \eta^2$
BOGV	bounded optimal gradient variance	$\mathbb{E}_i \|\nabla f_i(x^*)\|^2 \leq \eta^2$
BGV	bounded gradient dissimilarity	$\mathbb{E}_i \|\nabla f_i(x)\|^2 / \|\nabla F(x)\|^2 \leq \eta^2$

Other assumptions and variants

Symbol	Explanation
CVX	Each client function $f_i(x)$ is convex.
SCVX	Each client function $f_i(x)$ is μ-strongly convex.
BNCVX	Each client function has bounded nonconvexity with $\nabla^2 f_i(x) \succeq -\mu I$.
BLGV	The variance of stochastic gradients on local clients is bounded.
BLGN	The norm of any local gradient is bounded.
LBG	Clients use the full batch of local samples to compute updates.
Dec	Decentralized setting, assumes the the connectivity of network is good.
AC	All clients participate in each round.
1step	One local update is performed on clients in each round.
Prox	Use proximal gradient steps on clients.
VR	Variance reduction which needs to track the state.

Convergence rates

Method	Non-IID	Other assumptions	Variant	Rate
Lian et al. [305]	BCGV	BLGV	Dec; AC; 1step	$O(1/T) + O(1/\sqrt{NT})$
PD-SGD [304]	BCGV	BLGV	Dec; AC	$O(N/T) + O(1/\sqrt{NT})$
MATCHA [469]	BCGV	BLGV	Dec	$O(1/\sqrt{TKM}) + O(M/KT)$
Khaled et al. [268]	BOGV	CVX	AC; LBG	$O(N/T) + O(1/\sqrt{NT})$
Li et al. [303]	BOBD	SCVX; BLGV; BLGN	-	$O(K/T)$
FedProx [300]	BGV	BNCVX	Prox	$O(1/\sqrt{T})$
SCAFFOLD [265]	-	SCVX; BLGV	VR	$O(1/TKM) + O(e^{-T})$

表 7: Convergence rates for a (non-comprehensive) set of federated optimization methods in non-IID settings. We summarize the key assumptions for non-IID data, local functions on each client, and other assumptions. We also present the variant of the algorithm comparing to Federated Averaging and the convergence rates that eliminate constant.

apply to practical uses of Federated Averaging. For example, Li et al. [303] studied the convergence of Federated Averaging in a more realistic setting where only a subset of clients are involved in each round. In order to guarantee the convergence, they assumed that the clients are selected either uniformly at random or with probabilities that are in proportion to the sizes of local datasets. Nonetheless, in practice the server may not be able to sample clients in these idealized ways — in particular, in cross-device settings only devices that meet strict eligibility requirements (e.g. charging, idle, free WiFi) will be selected to participate in the computation. At different times within a day, the clients characteristics can vary significantly. Eichner et al. [171] formulated this problem and studied the convergence of semi-cyclic SGD, where multiple blocks of clients with different characteristics are sampled from following a regular cyclic pattern (e.g. diurnal). Clients can perform different local steps because of heterogeneity in their computing capacities. Wang et al. [471] proves that FedAvg and many other federated learning algorithms will converge to the stationary points of a mismatched objective function in the presence of heterogeneous local steps. They refer to this problem as *objective inconsistency* and propose a simple technique to eliminate the inconsistency problem from federated learning algorithms.

最近，许多工作在放宽分析所需的假设方面取得了进展，以便更好地应用于联邦平均的实际应用。例如，Li et al. [303] 在更现实的环境中研究了联合平均的收敛性，其中每轮只涉及一部分客户端。为了保证收敛，他们假设客户端是随机均匀地选择的，或者与本地数据集的大小成正比的概率。尽管如此，在实践中，服务器可能无法以这些理想化的方式对客户端进行采样——特别是，在跨设备设置中，只有满足严格资格要求（例如充电、空闲、免费 WiFi）的设备才会被选择参与计算。在一天内的不同时间，客户特征可能会有很大差异。 Eichner et al. [171] 制定了这个问题并研究了半循环 SGD 的收敛性，其中从遵循规则循环模式（例如昼夜）中采样具有不同特征的多个客户端块。由于计算能力的异构性，客户端可以执行不同的本地步骤。Wang et al. [471] 证明 FedAvg 和许多其他联邦学习算法将在存在异构局部步骤的情况下收敛到不匹配目标函数的驻点。他们将这个问题称为 *objective inconsistency* 并提出了一种简单的技术来消除联邦学习算法中的不一致问题。

We summarize recent theoretical results in Table 7. All the methods in Table 7 assume smoothness or Lipschitz gradients for the local functions on clients. The error bound is measured by optimal objective (1) for convex functions and norm of gradient for nonconvex functions. For each method, we present the key non-IID assumption, assumptions on each client function $f_i(x)$, and other auxiliary assumptions. We also briefly describe each method as a variant of the federated averaging algorithm, and show the simplified convergence rate eliminating constants. Assuming the client functions are strongly convex could help the convergence rate [303, 265]. Bounded gradient variance, which is a widely used assumption to analyze stochastic gradient methods, is often used when clients use stochastic local updates [305, 303, 304, 469, 265]. Li et al. [303] directly analyzes the Federated Averaging algorithm, which applies K steps of local updates on randomly sampled M clients in each round, and presents a rate that suggests local updates $(K > 1)$ could slow down the convergence. Clarifying the regimes where $K > 1$ may hurt or help convergence is an important open problem.

Connections to decentralized optimization The objective function of federated optimization has been studied for many years in the decentralized optimization community. As first shown in Wang and Joshi [467], the convergence analysis of decentralized SGD can be applied to or combined with local SGD with a proper setting of the network topology matrix (mixing matrix). In order to reduce the communication overhead, Wang and Joshi [467] proposed periodic decentralized SGD (PD-SGD) which allows decentralized SGD to have multiple local updates as Federated Averaging. This algorithm is extended by Li et al. [304] to the non-IID case. MATCHA [469] further improves the per-

formance of PD-SGD by randomly sampling clients for computation and communication, and provides a convergence analysis showing that local updates can accelerate convergence.

Acceleration, variance reduction and adaptivity Momentum, variance-reduction, and adaptive learning rates are all promising techniques to improve convergence and generalization of first-order methods. However, there is no single manner in which to incorporate these techniques into FedAvg. SCAFFOLD [265] models the difference in client updates using control variates to perform variance reduction. Notably, this allows convergence results not relying on bounding the amount of heterogeneity among clients. As for momentum, Yu et al. [501] propose allowing each client to maintain a local momentum buffer and average the local buffers and the local model parameters at each communication round. Although this method empirically improves the final accuracy of local SGD, this doubles the per-round communication cost. A similar scheme is used by Xie et al. [485] to design a variant of local SGD in which clients locally perform Adagrad [335, 161]. Reddi et al. [389] instead proposes using adaptive learning rates at the server-level, developing federated versions of adaptive optimization methods with the same communication cost as FedAvg. This framework generalizes the server momentum framework proposed by Hsu et al. [237], Wang et al. [470], which allows momentum without increasing communication costs. While both [501, 470] showed that the momentum variants of local SGD can converge to stationary points of non-convex objective functions at the same rate as synchronous mini-batch SGD, it is challenging to prove momentum accelerates the convergence rate in the federated learning setting. Recently, Karimireddy et al. [264] proposed a general approach for adapting centralized optimization algorithms to the heterogeneous federated setting (MIME framework and algorithms).

3.3 Multi-Task Learning, Personalization, and Meta-Learning

In this section we consider a variety of "multi-model" approaches — techniques that result in effectively using different models for different clients at inference time. These techniques are particularly relevant when faced with non-IID data (Section 3.1), since they may outperform even the best possible shared global model. We note that personalization has also been studied in the fully decentralized setting [459, 59, 504, 19], where training individual models is particularly natural.

在本节中, 我们考虑各种“多模型”方法——在推理时为不同客户端有效使用不同模型的技术。这些技术在面对非 IID 数据时尤其重要(Section 3.1), 因为它们甚至可能胜过最好的共享全局模型。我们注意到, 在完全去中心化的设置 [459, 59, 504, 19] 中也研究了个性化, 其中训练单个模型特别自然。

3.3.1 Personalization via Featurization

The remainder of this section specifically considers techniques that result in different users running inference with different model parameters (weights). However, in some applications similar benefits can be achieved by simply adding user and context features to the model. For example, consider a language model for next-word-prediction in a mobile keyboard as in Hard et al. [222]. Different clients are likely to use language differently, and in fact on-device personalization of model parameters has yielded significant improvements for this problem [472]. However, a complimentary approach may be to train a federated model that takes as input not only the words the user has typed so far, but a variety of other user and context features—What words does this user frequently use? What app are they currently using? If they are chatting, what messages have they sent to this person before? Suitably featurized, such

inputs can allow a shared global model to produce highly personalized predictions. However, largely because few public datasets contain such auxiliary features, developing model architectures that can effectively incorporate context information for different tasks remains an important open problem with the potential to greatly increase the utility of FL-trained models.

通过特征化进行个性化

本节的其余部分专门考虑导致不同用户使用不同模型参数(权重)运行推理的技术。但是, 在某些应用程序中, 只需向模型添加用户和上下文特征即可获得类似的好处。例如, 考虑在移动键盘中用于下一个词预测的语言模型, 如 Hard et al. [222]。不同的客户端可能使用不同的语言, 事实上, 模型参数的设备上个性化已经对这个问题产生了显着的改进 [472]。然而, 一个免费的方法可能是训练一个联邦模型, 该模型不仅将用户迄今为止输入的单词作为输入, 而且将各种其他用户和上下文特征作为输入——该用户经常使用哪些单词？他们目前使用什么应用程序？如果他们在聊天, 他们之前给这个人发过什么信息？适当地特征化, 这样的输入可以允许共享的全局模型产生高度个性化的预测。然而, 主要是因为很少有公共数据集包含这样的辅助特征, 开发可以有效地结合不同任务的上下文信息的模型架构仍然是一个重要的开放问题, 有可能大大提高 FL 训练模型的效用。

3.3.2 Multi-Task Learning

If one considers each client's local problem (the learning problem on the local dataset) as a separate task (rather than as a shard of a single partitioned dataset), then techniques from multi-task learning [506] immediately become relevant. Notably, Smith et al. [424] introduced the MOCHA algorithm for multi-task federated learning, directly tackling challenges of communication efficiency, stragglers, and fault tolerance. In multi-task learning, the result of the training process is one model per task. Thus, most multi-task learning algorithms assume all clients (tasks) participate in each training round, and also require stateful clients since each client is training an individual model. This makes such techniques relevant for cross-silo FL applications, but harder to apply in cross-device scenarios.

多任务学习

如果将每个客户的本地问题(本地数据集上的学习问题)视为一项单独的任务(而不是单个分区数据集的分片), 那么来自多任务学习的技术 [506] 立即变得相关。值得注意的是, Smith et al. [424] 引入了用于多任务联邦学习的 MOCHA 算法, 直接解决了通信效率、落后者和容错方面的挑战。在多任务学习中, 训练过程的结果是每个任务一个模型。因此, 大多数多任务学习算法假设所有客户端(任务)都参与每轮训练, 并且还需要有状态的客户端, 因为每个客户端都在训练一个单独的模型。这使得此类技术与跨孤岛 FL 应用程序相关, 但更难应用于跨设备场景。

Another approach is to reconsider the relationship between clients (local datasets) and learning tasks (models to be trained), observing that there are points on a spectrum between a single global model and different models for every client. For example, it may be possible to apply techniques from multi-task learning (as well as other approaches like personalization, discussed next), where we take the "task" to be a subset of the clients, perhaps chosen explicitly (e.g. based on geographic region, or characteristics of the device or user), or perhaps based on clustering [331] or the connected components of a learned graph over the clients [504]. The development of such algorithms is an important

open problem. See Section 4.4.4 for a discussion of how sparse federated learning problems, such as those arising naturally in this type of multi-task problem, might be approached without revealing to which client subset (task) each client belongs.

另一种方法是重新考虑客户端(本地数据集)和学习任务(要训练的模型)之间的关系, 观察单个全局模型和每个客户端的不同模型之间的范围。例如, 有可能应用多任务学习的技术(以及其他方法, 如个性化, 接下来讨论), 我们将“任务”作为客户的一个子集, 可能是明确选择的(例如基于地理区域或设备或用户的特征), 或者可能基于聚类 [331] 或客户端 [504] 上的学习图的连接组件。此类算法的开发是一个重要的开放性问题。请参阅 Section 4.4.4 以讨论如何在不透露每个客户端属于哪个客户端子集(任务)的情况下处理稀疏联邦学习问题, 例如在此类多任务问题中自然出现的问题。

3.3.3 Local Fine Tuning and Meta-Learning

局部微调和元学习

By local fine tuning, we refer to techniques which begin with the federated training of a single model, and then deploy that model to all clients, where it is personalized by additional training on the local dataset before use in inference. This approach integrates naturally into the typical lifecycle of a model in federated learning (Section 1.1.1). Training of the global model can still proceed using only small samples of clients on each round (e.g. 100s); the broadcast of the global model to all clients (e.g. many millions) only happens once, when the model is deployed. The only difference is that before the model is used to make live predictions on the client, a final training process occurs, personalizing the model to the local dataset.

通过局部微调, 我们指的是从单个模型的联邦训练开始, 然后将该模型部署到所有客户端的技术, 在用于推理之前, 通过对本地数据集的额外训练对其进行个性化。这种方法很自然地融入了联邦学习中模型的典型生命周期(Section 1.1.1)。全局模型的训练仍然可以在每一轮中仅使用少量客户样本(例如 100s)进行；全局模型向所有客户端(例如数百万)的广播仅在部署模型时发生一次。唯一的区别是, 在使用模型对客户端进行实时预测之前, 会进行最后的训练过程, 将模型个性化为本地数据集。

Given a global model that performs reasonably well, what is the best way to personalize it? In non-federated learning, researchers often use fine-tuning, transfer learning, domain adaptation [329, 132, 61, 332, 133], or interpolation with a personal local model. Of course, the precise technique used for such interpolations is key and it is important to determine its corresponding learning guarantees in the context of federated learning. Further, these techniques often assume only a pair of domains (source and target), and so some of the richer structure of federated learning may be lost.

给定一个性能相当不错的全局模型, 个性化它的最佳方法是什么？在非联邦学习中, 研究人员经常使用微调、迁移学习、领域适应 [329, 132, 61, 332, 133], 或者使用个人局部模型进行插值。当然, 用于这种插值的精确技术是关键, 在联邦学习的背景下确定其相应的学习保证很重要。此外, 这些技术通常只假设一对域(源和目标), 因此联邦学习的一些更丰富的结构可能会丢失。

One approach for studying personalization and non-IID data is via a connection to *meta-learning*, which has emerged as a popular setting for model adaptation. In the standard learning-to-learn (LTL) setup [56], one has a meta-distribution over tasks, samples from which are used to learn a learning algorithm, for example by finding a good restriction of the hypothesis space. This is in fact a good match for the statistical setting discussed in Section 3.1,

where we sample a client (task) $i \sim \mathcal{Q}$, and then sample data for that client (task) from $\mathcal{P}_i$.

研究个性化和非 IID 数据的一种方法是通过与 *元学习* 的连接, 这已成为模型适应的流行设置。在标准的learning-to-learn(LTL) 设置 [56] 中, 任务具有元分布, 其中的样本用于学习学习算法, 例如通过找到假设空间的良好限制。这实际上与 Section 3.1 中讨论的统计设置非常匹配, 我们对客户端(任务)$i \sim \mathcal{Q}$ 进行采样, 然后从$\mathcal{P}_i$。

Recently, a class of algorithms referred to as *model-agnostic meta-learning* (MAML) have been developed that meta-learn a global model, which can be used as a starting point for learning a good model adapted to a given task, using only a few local gradient steps [187]. Most notably, the training phase of the popular Reptile algorithm [358] is closely related to Federated Averaging [337] — Reptile allows for a server learning rate and assumes all clients have the same amount of data, but is otherwise the same. Khodak et al. [270] and Jiang et al. [250] explore the connection between FL and MAML, and show how the MAML setting is a relevant framework to model the personalization objectives for FL. Chai Sim et al. [102] applied local fine tuning to personalize speech recognition models in federated learning. Fallah et al. [181] developed a new algorithm called Personalized FedAvg by connecting MAML instead of Reptile to federated learning. Additional connections with differential privacy were studied in [299].

最近, 已经开发了一类被称为 *model-agnostic meta-learning* (MAML) 的算法, 该算法可以元学习全局模型, 该模型可以用作学习适合给定模型的良好模型的起点。任务, 仅使用几个局部梯度步骤 [187]。最值得注意的是, 流行的 Reptile 算法 [358] 的训练阶段与联邦平均 [337] 密切相关——Reptile 允许服务器学习率并假设所有客户端具有相同数量的数据, 但是否则相同。 Khodak et al. [270] 和 Jiang et al. [250] 探索了 FL 和 MAML 之间的联系, 并展示了 MAML 设置如何成为为 FL 的个性化目标建模的相关框架。 Chai Sim et al. [102] 在联邦学习中应用局部微调来个性化语音识别模型。 Fallah et al. [181] 通过将 MAML 而不是 Reptile 连接到联邦学习, 开发了一种名为 Personalized FedAvg 的新算法。在 [299] 中研究了与差分隐私的其他联系。

The general direction of combining ideas from FL and MAML is relatively new, with many open questions:

- The evaluation of MAML algorithms for supervised tasks is largely focused on synthetic image classification problems [290, 386] in which infinite artificial tasks can be constructed by subsampling from classes of images. FL problems, modeled by existing datasets used for simulated FL experiments (Appendix 8), can serve as realistic benchmark problems for MAML algorithms.

- In addition to an empirical study, or optimization results, it would be useful to analyze the theoretical guarantees of MAML-type techniques and study under what assumptions they can be successful, as this will further elucidate the set of FL domains to which they may apply.

- The observed gap between the global and personalized acccuracy [250] creates a good argument that personalization should be of central importance to FL. However, none of the existing works clearly formulates what would be comprehensive metrics for measuring personalized performance; for instance, is a small improvement for every client preferable to a larger improvement for a subset of clients? See Section 6 for a related discussion.

- Jiang et al. [250] highlighted the fact that models of the same structure and performance, but trained differently, can have very different capacity to personalize. In particular, it appears that training models with the goal of maximizing global performance might actually hurt the model's capacity for subsequent personalization. Understanding the underlying reasons for this is a question relevant for both FL and the broader ML community.

- Several challenging FL topics including personalization and privacy have begun to be studied in this multi-

task/LTL framework [270, 250, 299]. Is it possible for other issues such as concept drift to also be analyzed in this way, for example as a problem in lifelong learning [420]?

- Can non-parameter transfer LTL algorithms, such as ProtoNets [425], be of use for FL?

结合 FL 和 MAML 思想的总体方向相对较新, 有许多悬而未决的问题：

- 用于监督任务的 MAML 算法的评估主要集中在合成图像分类问题 [290, 386] 中, 可以通过从图像类别中进行二次采样来构建无限人工任务。 FL 问题由用于模拟 FL 实验的现有数据集(附录 8)建模, 可以作为 MAML 算法的现实基准问题。
- 除了实证研究或优化结果之外, 分析 MAML 类型技术的理论保证并研究它们在哪些假设下可以成功也是有用的, 因为这将进一步阐明它们所针对的 FL 域集可申请。
- 观察到的全局和个性化准确性之间的差距 [250] 提出了一个很好的论点, 即个性化应该对 FL 至关重要。然而, 现有的作品都没有明确制定衡量个性化绩效的综合指标。例如, 对于每个客户的小改进是否比对客户子集的更大改进更可取？有关相关讨论, 请参阅 Section 6 。
- Jiang et al. [250] 强调了一个事实, 即结构和性能相同但训练不同的模型可能具有非常不同的个性化能力。特别是, 以最大化全局性能为目标的训练模型实际上可能会损害模型后续个性化的能力。了解其根本原因是一个与 FL 和更广泛的 ML 社区相关的问题。
- 在这个多任务/LTL 框架[270, 250, 299]中已经开始研究包括个性化和隐私在内的几个具有挑战性的FL主题。是否有可能以这种方式分析概念漂移等其他问题, 例如作为终身学习[420] 中的问题？
- 非参数传输 LTL 算法, 例如 ProtoNets [425] 可以用于 FL 吗？

3.3.4 When is a Global FL-trained Model Better?

What can federated learning do for you that local training on one device cannot? When local datasets are small and the data is IID, FL clearly has an edge, and indeed, real-world applications of federated learning [491, 222, 112] benefit from training a single model across devices. On the other hand, given pathologically non-IID distributions (e.g. $\mathcal{P}_i(y \mid x)$ directly disagree across clients), local models will do much better. Thus, a natural theoretical question is to determine under what conditions the shared global model is better than independent per-device models. Suppose we train a model h_k for each client k, using the sample of size m_k available from that client. Can we guarantee that the model h_{FL} learned via federated learning is at least as accurate as h_k when used for client k? Can we quantify how much improvement can be expected via federated leaning? And can we develop personalization strategies with theoretical guarantees that at least match the performance of both natural baselines (h_k and h_{FL})?

什么时候全局 FL 训练模型更好？

Daniel Ramage 的联邦学习可以为您做什么, 而在一台设备上进行本地培训则无法做到？当本地数据集较小且数据为 IID 时, FL 显然具有优势, 事实上, 联邦学习的实际应用 [491, 222, 112] 受益于跨设备训练单个模型。另一方面, 给定病态的非 IID 分布(例如 $\mathcal{P}_i(y \mid x)$ 在客户端之间直接不一致), 本地模型会做得更好。因此, 一个自然的理论问题是确定在什么条件下共享全局模型比独立的每设备模型更好。假设我们为每个客户 k 训

练一个模型 h_k, 使用该客户提供的大小为 m_k 的样本。我们能否保证通过联邦学习学习的模型 h_{FL} 在用于客户端 k 时至少与 h_k 一样准确？我们能否量化通过联邦学习可以预期的改进程度？我们能否在理论上保证至少匹配两个自然基线(h_k 和 h_{FL})的性能的个性化策略？

Several of these problems relate to previous work on multiple-source adaptation and agnostic federated learning [329, 330, 234, 352]. The hardness of these questions depends on how the data is distributed among parties. For example, if data is vertically partitioned, each party maintaining private records of different feature sets about common entities, these problems may require addressing record linkage [124] within the federated learning task. Independently of the eventual technical levy of carrying out record linkage privately [407], the task itself happens to be substantially noise prone in the real world [406] and only sparse results have addressed its impact on training models [224]. Techniques for robustness and privacy can make local models relatively stronger, particularly for non-typical clients [502]. Loss factorization tricks can be used in supervised learning to alleviate up to the vertical partition assumption itself, but the practical benefits depend on the distribution of data and the number of parties [373].

其中一些问题与之前关于多源适应和不可知联邦学习 [329, 330, 234, 352] 的工作有关。这些问题的难易程度取决于数据在各方之间的分布方式。例如, 如果数据是垂直分区的, 每一方都维护关于公共实体的不同特征集的私人记录, 这些问题可能需要解决联邦学习任务中的记录链接 [124]。独立于私下执行记录链接的最终技术征费 [407], 任务本身恰好在现实世界中很容易产生噪音 [406] 并且只有稀疏结果解决了它对训练模型的影响 [224]。稳健性和隐私技术可以使本地模型相对更强大, 特别是对于非典型客户 [502]。可以在监督学习中使用损失分解技巧来减轻垂直分区假设本身, 但实际好处取决于数据的分布和参与方的数量 [373]。

3.4 Adapting ML Workflows for Federated Learning

Many challenges arise when adapting standard machine learning workflows and pipelines (including data augmentation, feature engineering, neural architecture design, model selection, hyperparameter optimization, and debugging) to decentralized datasets and resource-constrained mobile devices. We discuss several of these challenges below.

为联邦学习调整 ML 工作流

在将标准机器学习工作流程和管道(包括数据增强、特征工程、神经架构设计、模型选择、超参数优化和调试)应用于分散数据集和资源受限的移动设备时, 会出现许多挑战。我们将在下面讨论其中的几个挑战。

3.4.1 Hyperparameter Tuning

Running many rounds of training with different hyperparameters on resource-constrained mobile devices may be restrictive. For small device populations, this might result in the over-use of limited communication and compute resources. However, recent deep neural networks crucially depend on a wide range of hyperparameter choices regarding the neural network’s architecture, regularization, and optimization. Evaluations can be expensive for large models and large-scale on-device datasets. Hyperparameter optimization (HPO) has a long history under the framework of AutoML [395, 273, 277], but it mainly concerns how to improve the model accuracy [64, 426, 374, 180] rather than communication and computing efficacy for mobile devices. Therefore, we expect that further research should consider developing solutions for efficient hyperparameter optimization in the context of federated learning.

超参数调整

在资源受限的移动设备上使用不同的超参数运行多轮训练可能会受到限制。对于小型设备群体, 这可能会导致过度使用有限的通信和计算资源。应避免在设备上进行频繁的通信和本地计算；否则, 它可能会在很大程度上损害用户体验并阻碍联邦学习的普及。然而, 最近的深度神经网络在很大程度上取决于关于神经网络架构、正则化和优化的各种超参数选择。对于大型模型和大规模设备上数据集, 评估可能很昂贵。超参数优化(HPO)在AutoML框架下历史悠久移动设备的功效。因此, 我们希望进一步的研究应该考虑在联邦学习的背景下开发有效的超参数优化解决方案。

In addition to general-purpose approaches to the hyperparameter optimization problem, in the training space specifically the development of easy-to-tune optimization algorithms is a major open area. Centralized training already requires tuning parameters like learning rate, momentum, batch size, and regularization. Federated learning adds potentially more hyperparameters — separate tuning of the aggregation / global model update rule and local client optimizer, number of clients selected per round, number of local steps per round, configuration of update compression algorithms, and more. Such hyperparameters can be crucial to obtaining a good trade-off between accuracy and convergence, and may actually impact the quality of the learned model [106]. In addition to a higher-dimensional search space, federated learning often also requires longer wall-clock training times and limited compute resources. These challenges could be addressed by optimization algorithms that are robust to hyperparameter settings (the same hyperparameter values work for many different real world datasets and architectures), as well as adaptive or self-tuning algorithms [446, 82].

除了超参数优化问题的通用方法外, 在训练领域, 特别是易于调整的优化算法的开发是一个主要的开放领域。集中训练已经需要调整学习率、动量、批量大小和正则化等参数。联邦学习可能会增加更多的超参数——聚合/全局模型更新规则和本地客户端优化器的单独调整、每轮选择的客户端数量、每轮本地步骤的数量、更新压缩算法的配置等。这样的超参数对于在准确性和收敛性之间取得良好的权衡至关重要, 实际上可能会影响学习模型的质量 [106]。除了更高维的搜索空间, 联邦学习通常还需要更长的挂钟训练时间和有限的计算资源。这些挑战可以通过对超参数设置稳健的优化算法(相同的超参数值适用于许多不同的现实世界数据集和架构)以及自适应或自调整算法来解决 [446, 82]。

3.4.2 Neural Architecture Design

Neural architecture search (NAS) in the federated learning setting is motivated by the drawbacks of the current practice of applying predefined deep learning models: the predefined architecture of a deep learning model may not be the optimal design choice when the data generated by users are invisible to model developers. For example, the neural architecture may have some redundant component for a specific dataset, which may lead to unnecessary computing on devices; there may be a better architectural design for the non-IID data distribution. The approaches to personalization discussed in Section 3.3 still share the same model architecture among all clients. The recent progress in NAS [230, 387, 175, 388, 60, 375, 313, 488, 175, 323] provides a potential way to address these drawbacks. There are three major methods for NAS, which utilize evolutionary algorithms, reinforcement learning, or gradient descent to search for optimal architectures for a specific task on a specific dataset. Among these, the gradient-based method leverages efficient gradient back-propagation with weight sharing, reducing the architecture search process from over 3000 GPU days to only 1 GPU day. Another interesting paper recently published, involving Weight Agnostic Neural Networks [192], claims that neural network architectures alone, without learning any weight parameters, may encode

solutions for a given task. If this technique further develops and reaches widespread use, it may be applied to the federated learning without collaborative training among devices. Although these methods have not been developed for distributed settings such as federated learning, they are all feasible to be transferred to the federated setting. Neural Architecture Search (NAS) for a global or personalized model in the federated learning setting is promising, and early exploration has been made in [228].

联邦学习环境中的神经体系结构搜索(NAS)是由当前应用预定义深度学习模型的实践的缺点所驱动的：当用户生成的数据对模型开发人员不可见时, 深度学习模型的预定义体系结构可能不是最佳设计选择。例如, 对于特定数据集, 神经架构可能具有一些冗余组件, 这可能导致在设备上进行不必要的计算；对于非IID数据分布, 可能有更好的体系结构设计。在Section 3.3中讨论的个性化方法在所有客户端之间仍然共享相同的模型体系结构。 NAS的最新进展 [230, 387, 175, 388, 60, 375, 313, 488, 175, 323]提供了解决这些缺陷的潜在方法。NAS有三种主要方法, 它们利用进化算法、强化学习或梯度下降来搜索特定数据集上特定任务的最佳体系结构。其中, 基于梯度的方法利用有效的梯度反向传播和权重共享, 将架构搜索过程从3000多GPU天减少到仅1 GPU天。最近发表的另一篇有趣的论文, 涉及重量不可知的神经网络, 引用gaier2019weight, 声称神经网络结构本身, 没有学习任何重量参数, 可以为给定任务编码解决方案。如果这项技术进一步发展并得到广泛应用, 它可以应用于联邦学习, 而无需设备间的协作训练。尽管这些方法尚未针对分布式环境(如联邦学习)开发, 但它们都可以转移到联邦环境。在联邦学习环境中, 对全局或个性化模型的神经架构搜索(NAS)是有希望的, 并且在[228]中已经进行了早期探索。

3.4.3 Debugging and Interpretability for FL

While substantial progress has been made on the federated training of models, this is only part of a complete ML workflow. Experienced modelers often directly inspect subsets of the data for tasks including basic sanity checking, debugging misclassifications, discovering outliers, manually labeling examples, or detecting bias in the training set. Developing privacy-preserving techniques to answer such questions on decentralized data is a major open problem. Recently, Augenstein et al. [31] proposed the use of differentially private generative models (including GANs), trained with federated learning, to answer some questions of this type. However, many open questions remain (see discussion in [31]), in particular the development of algorithms that improve the fidelity of FL DP generative models.

FL 的调试和可解释性

虽然在模型的联邦训练方面取得了实质性进展, 但这只是完整 ML 工作流程的一部分。经验丰富的建模人员通常会直接检查数据子集以执行任务, 包括基本健全性检查、调试错误分类、发现异常值、手动标记示例或检测训练集中的偏差。开发隐私保护技术来回答有关分散数据的此类问题是一个主要的开放性问题。最近, Augenstein et al. [31] 提出使用经过联邦学习训练的差分私有生成模型(包括 GAN)来回答此类问题。然而, 许多悬而未决的问题仍然存在(参见 [31] 中的讨论), 特别是提高 FL DP 生成模型保真度的算法的开发。

3.5 Communication and Compression

It is now well-understood that communication can be a primary bottleneck for federated learning since wireless links and other end-user internet connections typically operate at lower rates than intra- or inter-datacenter links and can be potentially expensive and unreliable. This has led to significant recent interest in reducing the communication

bandwidth of federated learning. Methods combining Federated Averaging with sparsification and/or quantization of model updates to a small number of bits have demonstrated significant reductions in communication cost with minimal impact on training accuracy [282]. However, it remains unclear if communication cost can be further reduced, and whether any of these methods or their combinations can come close to providing optimal trade-offs between communication and accuracy in federated learning. Characterizing such fundamental trade-offs between accuracy and communication has been of recent interest in theoretical statistics [507, 89, 221, 7, 49, 444, 50]. These works characterize the optimal minimax rates for distributed statistical estimation and learning under communication constraints. However, it is difficult to deduce concrete insights from these theoretical works for communication bandwidth reduction in practice as they typically ignore the impact of the optimization algorithm. It remains an open direction to leverage such statistical approaches to inform practical training methods.

通信和压缩

现在众所周知, 通信可能是联邦学习的主要瓶颈, 因为无线链接和其他最终用户互联网连接的运行速率通常低于数据中心内或数据中心间链接, 并且可能成本高昂且不可靠。这导致最近人们对减少联邦学习的通信带宽产生了浓厚的兴趣。将联邦平均与稀疏化和/或将模型更新量化为少量比特相结合的方法已证明通信成本显着降低, 对训练精度的影响最小[282]。然而, 目前尚不清楚是否可以进一步降低通信成本, 以及这些方法中的任何一种或它们的组合是否可以接近在联邦学习中的通信和准确性之间提供最佳权衡。描述准确性和通信之间的这种基本权衡最近引起了理论统计[507, 89, 221, 7, 49, 444, 50]的兴趣。这些工作描述了通信约束下分布式统计估计和学习的最佳极小极大速率。然而, 很难从这些理论工作中推断出实际中通信带宽减少的具体见解, 因为它们通常会忽略优化算法的影响。利用此类统计方法为实际培训方法提供信息仍然是一个开放的方向。

Compression objectives Motivated by the limited resources of current devices in terms of compute, memory and communication, there are several different compression objectives of practical value.

(a) *Gradient compression*[11] – reduce the size of the object communicated from clients to server, which is used to update the global model.

(b) *Model broadcast compression* – reduce the size of the model broadcast from server to clients, from which the clients start local training.

(c) *Local computation reduction* – any modification to the overall training algorithm such that the local training procedure is computationally more efficient.

压缩目标 由于当前设备在计算、内存和通信方面的资源有限, 有几种不同的具有实用价值的压缩目标。

(a) 梯度压缩[12] –减少从客户端到服务器通信的对象的大小, 用于更新全局模型。

(b) 模型广播压缩——减少从服务器到客户端的模型广播的大小, 客户端从这里开始本地训练。

[11] In this section, we use "gradient compression" to include compression applied to any model update, such as the updates produced by Federated Averaging when clients take multiple gradient steps.

[12] 在本节中, 我们使用“梯度压缩”来包括应用于任何模型更新的压缩, 例如当客户端采取多个梯度步骤时由联邦平均产生的更新。

(c) *局部计算减少*——对整体训练算法的任何修改, 使得局部训练过程在计算上更有效。

These objectives are in most cases complementary. Among them, (a) has the potential for the most significant practical impact in terms of total runtime. This is both because clients' connections generally have slower upload than download bandwidth[13] – and thus there is more to be gained, compared to (b) – and because the effects of averaging across many clients can enable more aggressive lossy compression schemes. Usually, (c) could be realized jointly with (a) and (b) by specific methods.

这些目标在大多数情况下是互补的。其中, 就总运行时间而言, (a) 有可能产生最显着的实际影响。这是因为客户端的连接通常比下载带宽的上传速度慢[14] – 因此, 与(b) – 并且因为跨多个客户端进行平均的效果可以启用更积极的有损压缩方案。通常, (c)可以通过特定的方法与(a)和(b)共同实现。

Much of the existing literature applies to the objective (a) [282, 440, 281, 17, 235, 55]. The impact of (b) on convergence in general has not been studied until very recently; an analysis is presented in [123]. Very few methods intend to address all of (a), (b) and (c) jointly. Caldas et al. [95] proposed a practical method by constraining the desired model update such that only particular submatrices of model variables are necessary to be available on clients; Hamer et al. [219] proposed a communication-efficient federated algorithm for learning mixture weights on an ensemble of pre-trained models, based on communicating only a subset of the models to any one device; He et al. [227] utilizes bidirectional and alternative knowledge distillation method to transfer knowledge from many compact DNNs to a dense server DNN, which can reduce the local computational burden at the edge devices.

许多现有文献都适用于目标 (a) [282, 440, 281, 17, 235, 55]。 (b) 对一般收敛的影响直到最近才被研究; [123] 中给出了分析。很少有方法打算联邦解决所有 (a)、(b) 和 (c)。 Caldas et al. [95] 提出了一种实用的方法, 通过约束所需的模型更新, 使得只有特定的模型变量子矩阵需要在客户端上可用； Hamer et al. [219] 提出了一种通信高效的联邦算法, 用于在预训练模型的集合上学习混合权重, 基于仅将模型的一个子集传达给任何一个设备； He et al. [227] 利用双向和替代知识蒸馏方法将知识从许多紧凑的 DNN 转移到密集的服务器 DNN, 这可以减少边缘设备的本地计算负担。

In cross-device FL, algorithms generally cannot assume any state is preserved on the clients (Table 1). However, this constraint would typically not be present in the cross-silo FL setting, where the same clients participate repeatedly. Consequently, a wider set of ideas related to error-correction such as [311, 405, 463, 444, 263, 435] are relevant in this setting, many of which could address both (a) and (b).

在跨设备 FL 中, 算法通常不能假设客户端上保留任何状态(Table 1)。但是, 这种约束通常不会出现在跨孤岛 FL 设置中, 其中相同的客户端重复参与。因此, 更广泛的与纠错相关的想法, 例如 [311, 405, 463, 444, 263, 435] 与此设置相关, 其中许多可以同时解决 (a) 和 (b)。

An additional objective is to modify the training procedure such that the *final* model is more compact, or efficient for inference. This topic has received a lot of attention in the broader ML community [220, 138, 509, 309, 362, 74], but these methods either do not have a straightforward mapping to federated learning, or make the training process more complex which makes it difficult to adopt. Research that simultaneously yields a compact final model, while also addressing the three objectives above, has significant potential for practical impact.

另一个目标是修改训练过程, 使 *final* 模型更紧凑, 或者推理效率更高。这个话题在更广泛的 ML 社区受到了很多关注[220, 138, 509, 309, 362, 74], 但是这些方法要么没有直接映射到联邦学习, 要么使训练过程变得

[13] See for instance https://www.speedtest.net/reports/

[14] 参见例如 https://www.speedtest.net/reports/

更加复杂这使得难以采用。同时产生一个紧凑的最终模型, 同时也解决上述三个目标的研究具有巨大的实际影响潜力。

For gradient compression, some existing works [440] are developed in the minimax sense to characterize the worst case scenario. However usually in information theory, the compression guarantees are instance specific and depend on the *entropy* of the underlying distribution [140]. In other words, if the data is easily compressible, they are provably compressed heavily. It would be interesting to see if similar instance specific results can be obtained for gradient compression. Similarly, recent works show that learning a compression scheme in a data-dependent fashion can lead to significantly better compression ratio for the case of data compression [482] as well as gradient compression. It is therefore worthwhile to evaluate these data-dependent compression schemes in the federated settings [193].

对于梯度压缩, 一些现有作品 [440] 是在极小极大意义上开发的, 以表征最坏的情况。然而, 通常在信息论中, 压缩保证是特定于实例的, 并且取决于基础分布 [140] 的 *entropy*。换句话说, 如果数据很容易压缩, 则可以证明它们被严重压缩。看看是否可以为梯度压缩获得类似的特定于实例的结果会很有趣。类似地, 最近的工作表明, 对于数据压缩 [482] 以及梯度压缩, 以数据相关的方式学习压缩方案可以显着提高压缩率。因此, 值得在联邦设置中评估这些依赖于数据的压缩方案 [193]。

Compatibility with differential privacy and secure aggregation Many algorithms used in federated learning such as Secure Aggregation [79] and mechanisms of adding noise to achieve differential privacy [3, 338] are not designed to work with compressed or quantized communications. For example, straightforward application of the Secure Aggregation protocol of Bonawitz et al. [80], Bell et al. [58] requires an additional $O(\log M)$ bits of communication for each scalar, where M is the number of clients being summed over, and this may render ineffective the aggressive quantization of updates when M is large (though see [82] for a more efficient approach). Existing noise addition mechanisms assume adding real-valued Gaussian or Laplacian noise on each client, and this is not compatible with standard quantization methods used to reduce communication. We note that several recent works allow biased estimators and would work nicely with Laplacian noise [435], however those would not give differential privacy, as they break independence between rounds. There is some work on adding discrete noise [9], but there is no notion whether such methods are optimal. Joint design of compression methods that are compatible with Secure Aggregation, or for which differential privacy guarantees can be obtained, is thus a valuable open problem.

与差分隐私和安全聚合的兼容性 联邦学习中使用的许多算法, 如安全聚合 [79] 和添加噪声以实现差分隐私的机制 [3, 338] 并非设计用于压缩或量化通信。例如, 直接应用 Bonawitz et al. [80], Bell et al. [58] 的安全聚合协议需要每个标量额外的 $O(\log M)$ 位通信, 其中 M 是求和的客户端数量, 并且当 M 很大时, 这可能会使更新的激进量化无效(尽管参见 [82] 以获得更有效的方法)。现有的噪声添加机制假设在每个客户端上添加实值高斯或拉普拉斯噪声, 这与用于减少通信的标准量化方法不兼容。我们注意到最近的一些工作允许有偏差的估计器, 并且可以很好地处理拉普拉斯噪声 [435], 但是这些不会提供差分隐私, 因为它们打破了轮次之间的独立性。有一些关于添加离散噪声 [9] 的工作, 但不知道这些方法是否是最佳的。因此, 与安全聚合兼容的压缩方法的联邦设计, 或可以获得差分隐私保证的压缩方法, 因此是一个有价值的开放问题。

Wireless-FL co-design The existing literature in federated learning usually neglects the impact of wireless channel dynamics during model training, which potentially undermines both training latency and thus reliability of the entire production system. In particular, wireless interference, noisy channels and channel fluctuations can significantly hin-

der the information exchange between the server and clients (or directly between individual clients, as in the fully decentralized case, see Section 2.1). This represents a major challenge for mission-critical applications, rooted in latency reduction and reliability enhancements. Potential solutions to address this challenge include federated distillation (FD), in which workers exchange their model output parameters (logits) as opposed to the model parameters (gradients and/weights), and optimizing workers' scheduling policy with appropriate communication and computing resources [248, 368, 402]. Another solution is to leverage the unique characteristics of wireless channels (e.g. broadcast and superposition) as natural data aggregators, in which the simultaneously transmitted analog-waves by different workers are superposed at the server and weighed by the wireless channel coefficients [4]. This yields faster model aggregation at the server, and faster training by a factor up to the number of workers. This is in sharp contrast with the traditional orthogonal frequency division multiplexing (OFDM) paradigm, whereby workers upload their models over orthogonal frequencies whose performance degrades with increasing number of workers [174].

无线-FL 协同设计 联邦学习中的现有文献通常忽略模型训练过程中无线信道动态的影响, 这可能会破坏整个生产系统的训练延迟和可靠性。特别是, 无线干扰、嘈杂的信道和信道波动会显着阻碍服务器和客户端之间的信息交换(或直接在单个客户端之间, 如在完全去中心化的情况下, 参见部分 2.1)。这代表了任务关键型应用程序的主要挑战, 其根源在于延迟减少和可靠性增强。解决这一挑战的潜在解决方案包括联邦蒸馏(FD), 其中工作人员交换他们的模型输出参数 (logits) 而不是模型参数(梯度和/权重), 并通过适当的通信和计算资源优化工作人员的调度策略[248, 368, 402]。另一种解决方案是利用无线信道的独特特性(例如广播和叠加)作为自然数据聚合器, 其中不同工作人员同时传输的模拟波在服务器上叠加并由无线信道系数加权 [4] .这会在服务器上产生更快的模型聚合, 以及更快的训练, 最多可达工作人员的数量。这与传统的正交频分复用(OFDM) 范式形成鲜明对比, 其中工作人员通过正交频率上传他们的模型, 其性能随着工作人员数量的增加而降低[174]。

3.6 Application To More Types of Machine Learning Problems and Models

To date, federated learning has primarily considered supervised learning tasks where labels are naturally available on each client. Extending FL to other ML paradigms, including reinforcement learning, semi-supervised and unsupervised learning, active learning, and online learning [226, 508] all present interesting and open challenges.

应用于更多类型的机器学习问题和模型

迄今为止, 联邦学习主要考虑监督学习任务, 其中标签在每个客户端上自然可用。将 FL 扩展到其他 ML 范式, 包括强化学习、半监督和无监督学习、主动学习和在线学习 [226, 508] 都提出了有趣和开放的挑战。

Another important class of models, highly relevant to FL, are those that can characterize the uncertainty in their predictions. Most modern deep learning models cannot represent their uncertainty nor allow for a probability interpretation of parametric learning. This has motivated recent developments of tools and techniques combining Bayesian models with deep learning. From a probability theory perspective, it is unjustifiable to use single point-estimates for classification. Bayesian neural networks [419] have been proposed and shown to be far more robust to over-fitting, and can easily learn from small datasets. The Bayesian approach further offers uncertainty estimates via its parameters in form of probability distributions, thus preventing over-fitting. Moreover, appealing to probabilistic

reasoning, one can predict how the uncertainty can decrease, allowing the decisions made by the network to become more deterministic as the data size grows.

与 FL 高度相关的另一类重要模型是那些可以表征预测中的不确定性的模型。大多数现代深度学习模型不能表示它们的不确定性, 也不能对参数学习进行概率解释。这推动了最近将贝叶斯模型与深度学习相结合的工具和技术的发展。从概率论的角度来看, 使用单点估计进行分类是不合理的。贝叶斯神经网络 [419] 已经被提出并被证明对过拟合更加鲁棒, 并且可以很容易地从小数据集中学习。贝叶斯方法通过其概率分布形式的参数进一步提供不确定性估计, 从而防止过度拟合。此外, 借助概率推理, 人们可以预测不确定性如何降低, 随着数据规模的增长, 网络做出的决策变得更具确定性。

Since Bayesian methods gave us tools to reason about deep models' confidence and also achieve state-of-the-art performance on many tasks, one expects Bayesian methods to provide a conceptual improvement to the classical federated learning. In fact, preliminary work from Lalitha et al. [292] shows that incorporating Bayesian methods allows for model aggregation across non-IID data and heterogeneous platforms. However, many questions regarding scalability and computational feasibility have to be addressed.

由于贝叶斯方法为我们提供了推理深度模型置信度的工具, 并在许多任务上实现了最先进的性能, 因此人们期望贝叶斯方法能够为经典联邦学习提供概念上的改进。事实上, Lalitha et al. [292] 的初步工作表明, 结合贝叶斯方法允许跨非 IID 数据和异构平台进行模型聚合。然而, 必须解决许多关于可扩展性和计算可行性的问题。

3.7 Executive summary

Efficient and effective federated learning algorithms face different challenges compared to centralized training in a datacenter.

与数据中心的集中训练相比, 高效的联邦学习算法面临着不同的挑战。

- Non-IID data due to non-identical client distributions, violation of independence, and dataset drift (Section 3.1) pose a key challenge. Though various methods have been surveyed and discussed in this section, defining and dealing with non-IID data remains an open problem and one of the most active research topics in federated learning.

- Optimization algorithms for federated learning are analyzed in Section 3.2 under different settings, e.g., convex and nonconvex functions, IID and non-IID data. Theoretical analysis has proven difficult for the parallel local updates commonly used in federated optimization, and often strict assumptions have to be made to constrain the client heterogeneity. Currently, known convergence rates do not fully explain the empirically-observed effectiveness of the Federated Averaging algorithm over methods such as mini-batch SGD [481].

- Client-side personalization and "multi-model'' approaches (Section 3.3) can address data heterogeneity and give hope of surpassing the performance of the best fixed global model. Simple personalization methods like fine-tuning can be effective, and offer intrinsic privacy advantages. However, many theoretical and empirical questions remain open: when is a global model better? How many models are necessary? Which federated optimization algorithms combine best with local fine-tuning?

- Adapting centralized training workflows such as hyper-parameter tuning, neural architecture design, debugging,

and interpretability tasks to the federated learning setting (Section 3.4) present roadblocks to the widespread adoption of FL in practical settings, and hence constitute important open problems.

- While there has been significant work on communication efficiency and compression for FL (Section 3.5), it remains an important and active area. In particular, fully automating the process of enabling compression without impacting convergence for a wide class of models is an important practical goal. Relatively new directions on the theoretical study of communication, compatibility with privacy methods, and co-design with wireless infrastructure are discussed.

- There are many open questions in extending federated learning from supervised tasks to other machine learning paradigms including reinforcement learning, semi-supervised and unsupervised learning, active learning, and online learning (Section 3.6).

- 由于客户端分布不一致、违反独立性和数据集漂移(Section 3.1)而导致的项非IID数据是一个关键挑战。尽管本节对各种方法进行了调查和讨论, 但定义和处理非IID数据仍然是一个开放的问题, 也是联邦学习中最活跃的研究课题之一。

- 在不同的设置下, 例如凸函数和非凸函数、IID和非IID数据, 在Section 3.2中分析联邦学习的项目优化算法。对于联邦优化中常用的并行局部更新, 理论分析已被证明是困难的, 并且通常必须做出严格的假设来约束客户端的异构性。目前, 已知的收敛速度不能完全解释联邦平均算法相对于小型批量SGD [481]等方法的经验观察有效性。

- 客户端个性化和“多模型”方法(Section 3.3)可以解决数据异构性问题, 并有望超越最佳固定全局模型的性能。简单的个性化方法(如微调)可能是有效的, 并提供固有的隐私优势。然而, 许多理论和实证问题仍然悬而未决：什么时候全局模型更好？需要多少型号？哪些联邦优化算法与局部微调结合得最好？

- 将集中式培训工作流程(如超参数调整、神经结构设计、调试和解释性任务)调整到联邦学习环境(Section 3.4)的项目为FL在实际环境中的广泛采用提供了障碍, 因此构成了重要的开放性问题。

- 虽然在FL(Section 3.5)的通信效率和压缩方面进行了大量工作, 但它仍然是一个重要和活跃的领域。特别是, 实现压缩过程的完全自动化, 而不影响一大类模型的收敛性, 这是一个重要的实际目标。讨论了通信理论研究、与隐私方法的兼容性以及与无线基础设施的协同设计等方面相对较新的方向。

- 将联邦学习从有监督的任务扩展到其他机器学习模式, 包括强化学习、半监督和无监督学习、主动学习和在线学习(Section 3.6), 还有许多悬而未决的问题。

4 Preserving the Privacy of User Data

Machine learning workflows involve many actors functioning in disparate capacities. For example, users may generate training data through interactions with their devices, a machine learning training procedure extracts cross-population patterns from this data (e.g. in the form of trained model parameters), the machine learning engineer or

图 1: The lifecycle of an FL-trained model and the various actors in a federated learning system. (repeated from Page 9)

analyst may assess the quality of this trained model, and eventually the model may be deployed to end users in order to support specific user experiences (see Figure 1 below).

In an ideal world, each actor in the system would learn nothing more than the information needed to play their role. For example, if an analyst only needs to determine whether a particular quality metric exceeds a desired threshold in order to authorize deploying the model to end users, then in an idealized world, that is the only bit of information that would be available to the analyst; such an analyst would need access to neither the training data nor the model parameters, for instance. Similarly, end users enjoying the user experiences powered by the trained model might only require predictions from the model and nothing else.

Furthermore, in an ideal world every participant in the system would be able to reason easily and accurately about what personal information about themselves and others might be revealed by their participation in the system, and participants would be able to use this understanding to make informed choices about how and whether to participate at all.

Producing a system with all of the above ideal privacy properties would be a daunting feat on its own, and even more so while also guaranteeing other desirable properties such as ease of use for all participants, the quality and fairness of the end user experiences (and the models that power them), the judicious use of communication and computation resources, resilience against attacks and failures, and so on.

Rather than allowing perfect to be the enemy of good, we advocate a strategy wherein the overall system is composed of modular units which can be studied and improved relatively independently, while also reminding ourselves that we must, in the end, measure the privacy properties of the complete system against our ideal privacy goals set out above. The open questions raised throughout this section will highlight areas wherein we do not yet understand how to simultaneously achieve all of our goals, either for an individual module or for the system as a whole.

Federated learning provides an attractive structure for decomposing the overall machine learning workflow into the approachable modular units we desire. One of the primary attractions of the federated learning model is that it can provide a level of privacy to participating users through data minimization: the raw user data never leaves the device, and only updates to models (e.g., gradient updates) are sent to the central server. These model updates are more focused on the learning task at hand than is the raw data (i.e. they contain strictly no additional information about the user, and typically significantly less, compared to the raw data), and the individual updates only need to be held ephemerally by the server.

While these features can offer significant practical privacy improvements over centralizing all the training data, there is still no formal guarantee of privacy in this baseline federated learning model. For instance, it is possible to construct scenarios in which information about the raw data is leaked from a client to the server, such as a scenario where knowing the previous model and the gradient update from a user would allow one to infer a training example held by that user. Therefore, this section surveys existing results and outlines open challenges towards designing federated learning systems that can offer rigorous privacy guarantees. We focus on questions specific to the federated learning and analytics setting and leave aside questions that also arise in more general machine learning settings as surveyed in [344].

Beyond attacks targeting user privacy, there are also other classes of attacks on federated learning; for example, an adversary might attempt to prevent a model from being learned at all, or they might attempt to bias the model to produce inferences that are preferable to the adversary. We defer consideration of these types of attacks to Section 5.

The remainder of this section is organized as follows. Section 4.1 discusses various threat models against which we wish to give protections. Section 4.2 lays out a set of core tools and technologies that can be used towards providing rigorous protections against the threat models discussed in Section 4.1. Section 4.3 assumes the existence of a trusted server and discusses the open problems and challenges in providing protections against adversarial clients and/or analysts. Section 4.4 discusses the open problems and challenges in the absence of a fully trusted server. Finally, Section 4.5 discusses open questions around user perception.

4.1 Actors, Threat Models, and Privacy in Depth

A formal treatment of privacy risks in FL calls for a holistic and interdisciplinary approach. While some of the risks can be mapped to technical privacy definitions and mitigated with existing technologies, others are more complex and require cross-disciplinary efforts.

Privacy is not a binary quantity, or even a scalar one. This first step towards such formal treatment is a careful characterization of the different actors (see Figure 1 from Section 1, repeated on page 50 for convenience) and their roles to ultimately define relevant threat models (see Table 8). Thus, for instance, it is desirable to distinguish the view of the server administrator from the view of the analysts that consume the learned models, as it is conceivable that a system that is designed to offer strong privacy guarantees against a malicious analyst may not provide any guarantees with respect to a malicious server. These actors map well onto the threat models discussed elsewhere in the literature; for example, in Bittau et al. [73, Sec 3.1], where the "encoder" corresponds to the client, the "shuffler" generally corresponds to the server, the "analyzer" may correspond to the server or post-processing done by the analyst.

As an example, a particular system might offer a differential privacy[15] guarantee with a particular parameter ε to the view of the server administrator, while the results observed by analysts might have a higher protection $\varepsilon' < \varepsilon$.

Furthermore, it is possible that this guarantee holds only against adversaries with particular limits on their capabilities, e.g. an adversary that can observe everything that happens on the server (but cannot influence the server's behavior) while simultaneously controlling up to a fraction γ of the clients (observing everything they see and influencing their behavior in arbitrary ways); the adversary might also be assumed to be unable to break cryptographic mechanisms instantiated at a particular security level σ. Against an adversary whose strength *exceeds* these limits, the view of the server administrator might still have some differential privacy, but at weaker level $\varepsilon_0 > \varepsilon$.

As we see in this example, precisely specifying the assumptions and privacy goals of a system can easily implicate concrete instantiations of several parameters ($\varepsilon, \varepsilon', \varepsilon_0, \gamma, \sigma$, etc.) as well as concepts such as differential privacy and honest-but-curious security.

Achieving all the desired privacy properties for federated learning will typically require composing many of the tools and technologies described below into an end-to-end system, potentially both layering multiple strategies to protect the same part of the system (e.g. running portions of a Secure Multi-Party Computation (MPC) protocol inside a Trusted Execution Environment (TEE) to make it harder for an adversary to sufficiently compromise that component) as well as using different strategies to protect different parts of the system (e.g. using MPC to protect the aggregation of model updates, then using Private Disclosure techniques before sharing the aggregate updates beyond the server).

As such, we advocate for building federated systems wherein the privacy properties degrade as gracefully as

[15] Differential privacy will be formally introduced in Section 4.2.2. For now, it suffices to know that lower ε corresponds with higher privacy.

Data/Access Point	Actor	Threat Model
Clients	Someone who has root access to the client device, either by design or by compromising the device	Malicious clients can inspect all messages received from the server (including the model iterates) in the rounds they participate in and can tamper with the training process. An honest-but-curious client can inspect all messages received from the server but cannot tamper with the training process. In some cases, technologies such as secure enclaves/TEEs may be able to limit the influence and visibility of such an attacker, representing a meaningfully weaker threat model.
Server	Someone who has root access to the server, either by design or by compromising the device	A malicious server can inspect all messages sent to the server (including the gradient updates) in all rounds and can tamper with the training process. An honest-but-curious server can inspect all messages sent to the server but cannot tamper with the training process. In some cases, technologies such as secure enclaves/TEEs may be able to limit the influence and visibility of such an attacker, representing a meaningfully weaker threat model.
Output Models	Engineers & analysts	A malicious analyst or model engineer may have access to multiple outputs from the system, e.g. sequences of model iterates from multiple training runs with different hyperparameters. Exactly what information is released to this actor is an important system design question.
Deployed Models	The rest of the world	In cross-device FL, the final model may be deployed to hundreds of millions of devices. A partially compromised device can have black-box access to the learned model, and a fully compromised device can have a white-box access to the learned model.

表 8: Various threat models for different adversarial actors.

possible in cases where one technique or another fails to provide its intended privacy contribution. For example, running the server component of an MPC protocol inside a TEE might allow privacy to be maintained even in the case where either (but not both) of the TEE security or MPC security assumptions fails to hold in practice. As another example, requiring clients to send raw training examples to a server-side TEE would be strongly dispreferred to having clients send gradient updates to a server-side TEE, as the latter's privacy expectations degrade much more gracefully if the TEE's security were to fail. We refer to this principle of graceful degradation as "Privacy in Depth,'' in analogy to the well-established network security principle of defense in depth [361].

4.2 Tools and Technologies

Generally speaking, the goal of an FL computation is for the analyst or engineer requesting the computation to obtain the result, which can be thought of as the evaluation of a function f on a distributed client dataset (commonly an ML model training algorithm, but possibly something simpler such as a basic statistic). There are three privacy aspects that need to be addressed.

First, we need to consider *how* f is computed and what is the information flow of intermediate results in the process, which primarily influences the susceptibility to malicious client, server, and admin actors. In addition to designing the flow of information in the system (e.g. early data minimization), techniques from secure computation including Secure Multi-Party Computation (MPC) and Trusted Execution Environments (TEEs) are of particular relevance to addressing these concerns. These technologies will be discussed in detail in Section 4.2.1.

Second, we have to consider *what* is computed. In other words, how much information about a participating client is revealed to the analyst and world actors by the result of f itself. Here, techniques for privacy-preserving disclosure, particularly differential privacy (DP), are highly relevant and will be discussed in detail in Section 4.2.2.

Finally, there is the problem of *verifiability*, which pertains to the ability of a client or the server to prove to others in the system that they have executed the desired behavior faithfully, without revealing the potentially private data upon which they were acting. Techniques for verifiability, including remote attestation and zero-knowledge proofs, will be discussed in Section 4.2.3.

4.2.1 Secure Computations

The goal of secure computation is to evaluate functions on distributed inputs in a way that only reveals the result of the computation to the intended parties, without revealing any additional information (e.g. the parties' inputs or any intermediate results).

Secure multi-party computation Secure Multi-Party Computation (MPC) is a subfield of cryptography concerned with the problem of having a set of parties compute an agreed-upon function of their private inputs in a way that only reveals the intended output to each of the parties. This area was kicked off in the 1980's by Yao [493]. Thanks to both theoretical and engineering breakthroughs, the field has moved from being of a purely theoretical interest to a deployed technology in industry [78, 77, 295, 27, 191, 242, 243]. It is important to remark that MPC defines a set of technologies, and should be regarded more as a field, or a general notion of security in secure computation, than a technology *per se*. Some of the recent advances in MPC can be attributed to breakthroughs in lower level primitives,

Technology	Characteristics
Differential Privacy (local, central, shuffled, aggregated, and hybrid models)	A quantification of how much information could be learned about an individual from the output of an analysis on a dataset that includes the user. Algorithms with differential privacy necessarily incorporate some amount of randomness or noise, which can be tuned to mask the influence of the user on the output.
Secure Multi-Party Computation	Two or more participants collaborate to simulate, though cryptography, a fully trusted third party who can: • Compute a function of inputs provided by all the participants; • Reveal the computed value to a chosen subset of the participants, with no party learning anything further.
Homomorphic Encryption	Enables a party to compute functions of data to which they do not have plain-text access, by allowing mathematical operations to be performed on ciphertexts without decrypting them. Arbitrarily complicated functions of the data can be computed this way ("Fully Homomorphic Encryption'') though at greater computational cost.
Trusted Execution Environments (secure enclaves)	TEEs provide the ability to trustably run code on a remote machine, even if you do not trust the machine's owner/administrator. This is achieved by limiting the capabilities of any party, including the administrator. In particular, TEEs may provide the following properties [437]: • Confidentiality: The state of the code's execution remains secret, unless the code explicitly publishes a message; • Integrity: The code's execution cannot be affected, except by the code explicitly receiving an input; • Measurement/Attestation: The TEE can prove to a remote party what code (binary) is executing and what its starting state was, defining the initial conditions for confidentiality and integrity.

表 9: Various technologies along with their characteristics.

such as oblivious transfer protocols [244] and encryption schemes with homomorphic properties (as described below).

A common aspect of cryptographic solutions is that operations are often done on a finite field (e.g. integers modulo a prime p), which poses difficulties when representing real numbers. A common approach has been to adapt ML models and their training procedures to ensure that (over)underflows are controlled, by operating on normalized quantities and relying on careful quantization [194, 10, 206, 84].

It has been known for several decades that any function can be securely computed, even in the presence of malicious adversaries [208]. While generic solutions exist, their performance characteristics often render them inapplicable in practical settings. As such a noticeable trend in research has consisted in designing custom protocols for applications such as linear and logistic regression [359, 194, 351] and neural network training and inference [351, 10, 48]. These works are typically in the cross-silo setting, or the variant where computation is delegated to a small group of computing servers that do not collude with each other. Porting these protocols to the cross-device setting is not straightforward, as they require a significant amount of communication.

Homomorphic encryption Homomorphic encryption (HE) schemes allow certain mathematical operations to be performed directly on ciphertexts, without prior decryption. Homomorphic encryption can be a powerful tool for enabling MPC by enabling a participant to compute functions on values while keeping the values hidden.

Different flavours of HE exist, ranging from general fully homomorphic encryption (FHE) [197] to the more efficient leveled variants [87, 182, 88, 129], for which several implementations exist [233, 409, 364, 415, 1]. Also of practical relevance are the so-called partially homomorphic schemes, including for example ElGamal and Paillier, allowing either homomorphic addition or multiplication. Additive HE has been used as an ingredient in MPC protocols in the cross-silo setting [359, 224]. A review of some homomorphic encryption software libraries along with brief explanations of criteria/features to be considered in choosing a library is surveyed in [404].

When considering the use of HE in the FL setting, questions immediately arise about who holds the secret key of the scheme. While the idea of every client encrypting their data and sending it to the server to compute homomorphically on it is appealing, the server should not be able to decrypt a single client contribution. A trivial way of overcoming this issue would be relying on a non-colluding external party that holds the secret key and decrypts the result of the computation. However, most HE schemes require that the secret keys be renewed often (due to e.g. susceptibility to chosen ciphertext attacks [117]). Moreover, the availability of a trusted non-colluding party is not standard in the FL setting.

Another way around this issue is relying on distributed (or threshold) encryption schemes, where the secret key is distributed among the parties. Reyzin et al. [392] and Roth et al. [398] propose such solutions for computing summation in the cross-device setting. Their protocols make use of additively homomorphic schemes (variants of ElGamal and lattice-based schemes, respectively).

Trusted execution environments Trusted execution environments (TEEs, also referred to as secure enclaves) may provide opportunities to move part of the federated learning process into a trusted environment in the cloud, whose code can be attested and verified.

TEEs can provide several crucial facilities for establishing trust that a unit of code has been executed faithfully and privately [437]:

- Confidentiality: The state of the code's execution remains secret, unless the code explicitly publishes a message.
- Integrity: The code's execution cannot be affected, except by the code explicitly receiving an input.
- Measurement/Attestation: The TEE can prove to a remote party what code (binary) is executing and what its starting state was, defining the initial conditions for confidentiality and integrity.

TEEs have been instantiated in many forms, including Intel's SGX-enabled CPUs [241, 134], Arm's TrustZone [28, 22], and Sanctum on RISC-V [135], each varying in its ability to systematically offer the above facilities.

Current secure enclaves are limited in terms of memory and provide access only to CPU resources, that is they do not allow processing on GPUs or machine learning processors (Tramèr and Boneh [447] explore how to combine TEEs with GPUs for machine learning inference). Moreover, it is challenging for TEEs (especially those operating on shared microprocessors) to fully exclude all types of side channel attacks [458].

While secure enclaves provide protections for all code running inside them, there are additional concerns that must be addressed in practice. For example, it is often necessary to structure the code running in the enclave as a data oblivious procedure, such that its runtime and memory access patterns do not reveal information about the data upon which it is computing (see for example [73]). Furthermore, measurement/attestation typically only proves that a particular binary is running; it is up to the system architect to provide a means for proving that that binary has the desired privacy properties, potentially requiring the binary to be built using a reproducible process from open source code.

It remains an open question how to partition federated learning functions across secure enclaves, cloud computing resources, and client devices. For example, secure enclaves could execute key functions such as secure aggregation or shuffling to limit the server's access to raw client contributions while keeping most of the federated learning logic outside this trusted computing base.

Secure computation problems of interest While secure multi-party computation and trusted execution environments offer general solutions to the problem of privately computing any function on distributed private data, many optimizations are possible when focusing on specific functionalities. This is the case for the tasks described next.

Secure aggregation Secure aggregation is a functionality for n clients and a server. It enables each client to submit a value (often a vector or tensor in the FL setting), such that the server learns just an aggregate function of the clients' values, typically the sum.

There is a rich literature exploring secure aggregation in both the single-server setting (via additive masking [8, 213, 80, 58, 428], via threshold homomorphic encryption [417, 218, 103], and via generic secure multi-party computation [94]) as well as in the multiple non-colluding servers setting [78, 27, 130]. Secure aggregation can also be approached using trusted execution environments (introduced above), as in [308].

Secure shuffling Secure shuffling is a functionality for n clients and a server. It enables each client to submit one or more messages, such that the server learns just an unordered collection (multiset) of the messages from all clients and nothing more. Specifically, the server has no ability to link any message to its sender beyond the information contained in the message itself. Secure shuffling can be considered an instance of Secure Aggregation where the

values are multiset-singletons and the aggregation operation is multiset-sum, though it is often the case that very different implementations provide the best performance in the typical operating regimes for secure shuffling and secure aggregation.

Secure shufflers have been studied in the context of secure multi-party computation [107, 288], often under the heading of mix networks. They have also been studied in the context of trusted computing [73]. Mix networks have found large scale deployment in the form of the Tor network [157].

Private information retrieval Private information retrieval (PIR) is a functionality for one client and one server. It enables the client to download an entry from a server-hosted database such that the server gains zero information about which entry the client requested.

MPC approaches to PIR break down into two main categories: *computational PIR* (cPIR), in which a single party can execute the entire server side of the protocol [286], and *information theoretic PIR* (itPIR), in which multiple non-colluding parties are required to execute the server side of the protocol [121].

The main roadblocks to the applicability of PIR have been the following: cPIR has high computational cost [423], while the non-colluding parties setting has been difficult to achieve convincingly in industrial scenarios. Recent results on PIR have shown dramatic reductions in the computational cost through the use of lattice-based cryptosystems [12, 363, 13, 23, 198]. The computational cost can be traded for more communication; we refer the reader to Ali et al. [16] to better understand the communication and computation trade-offs offered by cPIR. Additionally, it has been shown how to construct communication-efficient PIR on a single-server by leveraging side information available to the user [251], for example via client local state. Patel et al. [372] presented and implemented a practical hybrid (computational and information theoretic) PIR scheme on a single server assuming client state. Corrigan-Gibbs and Kogan [131] present theoretical constructions for PIR with sublinear *online* time by working in an offline/online model where, during an offline phase, clients fetch information from the server(s) independent on the future query to be performed.

Further work has explored the connection between PIR and secret sharing [479], with recent connections to PIR on coded data [159] and communication efficient PIR [72]. A variant of PIR, called PIR-with-Default, enable clients to retrieve a default value if the index queried is not in the database, and can output additive secret shares of items which can serve as input to any MPC protocol [297]. PIR has also been studied in the context of ON-OFF privacy, in which a client is permitted to switch off their privacy guards in exchange for better utility or performance [355, 494].

4.2.2 Privacy-Preserving Disclosures

The state-of-the-art model for quantifying and limiting information disclosure about individuals is *differential privacy* (DP) [167, 164, 165], which aims to introduce a level of uncertainty into the released model sufficient to mask the contribution of any individual user. Differential privacy is quantified by privacy loss parameters (ε, δ), where smaller (ε, δ) corresponds to increased privacy. More formally, a randomized algorithm $\mathcal{A}$ is (ε, δ)-differentially private if for all $\mathcal{S} \subseteq \text{Range}(\mathcal{A})$, and for all adjacent datasets D and D':

$$P(\mathcal{A}(D) \in \mathcal{S}) \leq e^{\varepsilon} P(\mathcal{A}(D') \in \mathcal{S}) + \delta. \tag{3}$$

In the context of FL, D and D' correspond to decentralized datasets that are adjacent if D' can be obtained from D by adding or subtracting all the records of a single client (user) [338]. This notion of differential privacy is referred to as user-level differential privacy. It is stronger than the typically used notion of adjacency where D and D' differ by only one record [165], since in general one user may contribute many records (e.g. training examples) to the dataset.

Over the last decade, an extensive set of techniques has been developed for differentially private data analysis, particularly under the assumption of a centralized setting, where the raw data is collected by a trusted party prior to applying perturbations necessary to achieve privacy. In federated learning, typically the orchestrating server would serve as the trusted implementer of the DP mechanism, ensuring only privatized outputs are released to the model engineer or analyst.

However, when possible we often wish to reduce the need for a trusted party. Several approaches for reducing the need for trust in a data curator have been considered in recent years.

Local differential privacy Differential privacy can be achieved without requiring trust in a centralized server by having each client apply a differentially private transformation to their data prior to sharing it with the server. That is, we apply Equation (3) to a mechanism $\mathcal{A}$ that processes a single user's local dataset D, with the guarantee holding with respect to *any* possible other local dataset D'. This model is referred to as the *local model of differential privacy* (LDP) [475, 266]. LDP has been deployed effectively to gather statistics on popular items across large userbases by Google, Apple and Microsoft [177, 154, 155]. It has also been used in federated settings for spam classifier training by Snap [378]. These LDP deployments all involve large numbers of clients and reports, even up to a billion in the case of Snap, which stands in stark contrast to centralized instantiations of DP which can provide high utility from much smaller datasets. Unfortunately, as we will discuss in Section 4.4.2, achieving LDP while maintaining utility can be difficult [266, 455]. Thus, there is a need for a model of differential privacy that interpolates between purely central and purely local DP. This can be achieved through distributed differential privacy, or the hybrid model, as discussed below.

Distributed differential privacy In order to recover some of the utility of central DP without having to rely on a trustworthy central server, one can instead use a *distributed differential privacy model* [166, 417, 73, 120]. Under this model, the clients first compute and encode a minimal (application specific) focused report, and then send the encoded reports to a secure computation function, whose output is available to the central server, with the intention that this output already satisfies differential privacy requirements by the time the server is able to inspect it. The encoding is done to help maintain privacy on the clients, and could for example include LDP. The secure computation function can have a variety of incarnations. It could be an MPC protocol, a standard computation done on a TEE, or even a combination of the two. Each of these choices comes with different assumptions and threat models.

It is important to remark that distributed differential privacy and local differential privacy yield different guarantees from several perspectives: while the distributed DP framework can produce more accurate statistics for the same level of differential privacy as LDP, it relies on different setups and typically makes stronger assumptions, such as access to MPC protocols. Below, we outline two possible approaches to distributed differential privacy, relying on secure aggregation and secure shuffling. We stress that there are many other methods that could be used, see for instance [400] for an approach based on exchanging correlated Gaussian noise across secure channels.

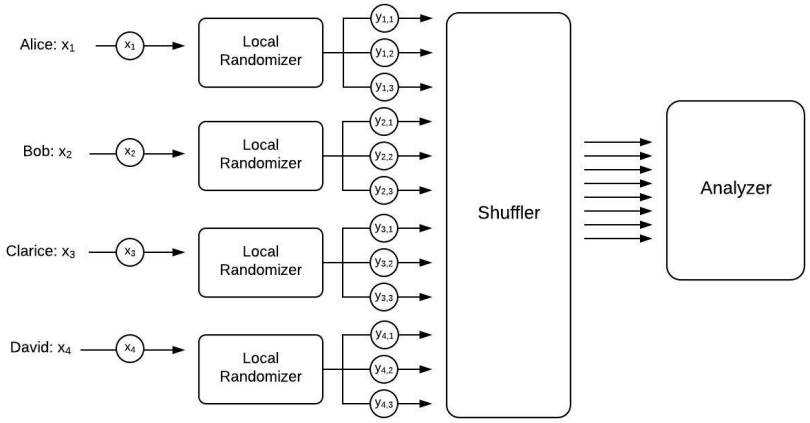


图 3: The Encode-Shuffle-Analyze (ESA) framework, illustrated here for 4 players.

Distributed DP via secure aggregation One promising tool for achieving distributed DP in FL is secure aggregation, discussed above in Section 4.2.1. Secure aggregation can be used to ensure that the central server obtains the aggregated result, while guaranteeing that intermediate parameters of individual devices and participants are not revealed to the central server. To further ensure the aggregated result does not reveal additional information to the server, we can use local differential privacy (e.g. with moderate ε level). For example, each device could perturb its own model parameter before the secure aggregation in order to achieve local differential privacy. By designing the noise correctly, we may ensure that the noise in the aggregated result matches the noise that would have otherwise been added centrally by a trusted server (e.g. with a low ε / high privacy level) [8, 385, 205, 417, 213].

Distributed DP via secure shuffling Another distributed differential privacy model is the shuffling model, which was kicked off by the recently introduced Encode-Shuffle-Analyze (ESA) framework [73] (illustrated in Figure 3). In the simplest version of this framework, each client runs an LDP protocol (e.g. with a moderate ε level) on its data and provides its output to a secure shuffler. The shuffler randomly permutes the reports and sends the collection of shuffled reports (without any identifying information) to the server for final analysis. Intuitively, the interposition of this secure compute function makes it harder for the server to learn anything about the participants and supports a differential privacy analysis (e.g. with a low ε / high privacy level). In the more general multi-message shuffled framework, each user can possibly send more than one message to the shuffler. The shuffler can either be implemented directly as a trusted entity, independent of the server and devoted solely to shuffling, or via more complex cryptographic primitives as discussed above.

Bittau et al. [73] proposed the Prochlo system as a way to implement the ESA framework. The system takes a holistic approach to privacy that takes into account secure computation aspects (addressed using TEEs), private disclosure aspects (addressed by means of differential privacy), and verifiability aspects (mitigated using secure enclave attestation capabilities).

More generally, shuffling models of differential privacy can use broader classes of local randomizers, and can even select these local randomizers adaptively [178]. This can enable differentially private protocols with far smaller error than what is possible in the local model, while relying on weaker trust assumptions than in the central model, e.g., [120, 178, 45, 201, 204, 200, 202, 203, 110].

Hybrid differential privacy Another promising approach is hybrid differential privacy [40], which combines multiple trust models by partitioning users based on their trust model preference (e.g. trust or lack of trust in the curator). Prior to the hybrid model, there were two natural choices. The first was to use the least-trusting model, which typically provides the lowest utility, and conservatively apply it uniformly over the entire userbase. The second was to use the most-trusting model, which typically provides the highest utility, but only apply it over the most-trusting users. By allowing multiple models to coexist, hybrid model mechanisms can achieve more utility from a given userbase, compared to purely local or central DP mechanisms. For instance, [40] describes a system in which most users contribute their data in the local model of privacy, and a small fraction of users opt-in to contributing their data in the central DP model. This enables the design of a mechanism which, in some circumstances, outperforms both the conservative local DP mechanism applied across all users as well as the central DP mechanism applied only across the small fraction of opt-in users. Recent work by [57] further demonstrates that a combination of multiple trust models can become part of a promising toolkit for designing and implementing differential privacy. This construction can be directly applied in the federated learning setting; however, the general concept of combining trust models or computational models may also inspire similar but new approaches for federated learning.

4.2.3 Verifiability

An important notion that is orthogonal to the above privacy techniques is that of verifiability. Generally speaking, verifiable computation will enable one party to prove to another party that it has executed the desired behavior on its data faithfully, without compromising the potential secrecy of the data. The concept of verifiable computation dates back to Babai et al. [42] and has been studied under various terms in the literature: checking computations [42], certified computation [343], delegating computations [210], as well as verifiable computing [195].

In the context of FL, verifiability can be used for two purposes. First, it would enable the server to prove to the clients that it executed the intended behavior (e.g., aggregating inputs, shuffling of the input messages, or adding noise for differential privacy) faithfully. Second, it would enable the clients to prove to the server that their inputs and behavior follow that of the protocol specification (e.g., the input belongs to a certain range, or the data is a correctly generated ciphertext).

Multiple techniques can be useful to provide verifiability: zero-knowledge proofs (ZKPs), trusted execution environments (TEEs), or remote attestation. Among these ZKPs provide formal cryptographic security guarantees based on mathematical hardness, while others make rely on assumption about the security of trusted hardware.

Zero-knowledge proofs (ZKPs) Zero knowledge (ZK) proofs are a cryptographic primitive that enables one party (called the *prover*) to prove statements to another party (called the *verifier*), that depend on secret information known to the prover, called witness, without revealing those secrets to the verifier. The notion of zero-knowledge was introduced in the late 1980's by Goldwasser et al. [209]. It provides a solution for the verifiability question on private data. While there had been a large body of work on ZK construction, the first work that brought ZKPs and verifiable computation for general functionalities in the realm of practicality was the work of Parno et al. [369] which introduces the first optimized construction and implementation for succinct ZK. Nowadays, ZKP protocols can achieve proof sizes of hundred of bytes and verifications of the order of milliseconds regardless of the size of the statement being proved.

A ZKP has three salient properties: *completeness* (if the statement is true and the prover and verifier follow the

protocol, the verifier will accept the proof), *soundness* (if the statement is false and the verifier follows the protocol, the verifier will refuse the proof), and *zero-knowledge* (if the statement is true and the prover follows the protocol, the verifier will only learn that the statement is true and will not learn any confidential information from the interaction).

Beyond these common properties, there are different types of zero-knowledge constructions in terms of supported language for the proofs, setup requirements, prover and verifier computational efficiency, interactivity, succinctness, and underlying hardness assumptions. There are many ZK constructions that support specific classes of statements, Schnorr proofs [408] and Sigma protocols [147] are examples of such widely used protocols. While such protocols have numerous uses in specific settings, general ZK systems that can support any functionality provide a much more broadly applicable tool (including in the context of FL), and thus we focus on such constructions for the rest of the discussion.

A major distinguishing feature between different constructions is the need for *trusted* setup. Some ZKPs rely on a common reference string (CRS), which is computed using secrets that should remain hidden in order to guarantee the soundness properties of the proofs. The computation of such a CRS is referred to as a trusted setup. While this requirement is a disadvantage for such systems, the existing ZKP constructions that achieve most succinct proofs and verifier's efficiency require trusted setup.

Another significant property that affects the applicability in different scenarios is whether generating the proof requires interaction between the prover and the verifier, and here we distinguish non-interactive zero-knowledge proofs (NIZKs) that enable the prover to send a single message to the verifier and require no further communication. Often we can convert interactive to non-interactive proofs by making stronger assumptions about ideal functionality of hash functions (i.e., that hash functions behave as random oracles).

Additionally, there are different measurements for efficiency of a ZKP system one must be aware of, such as the length of the proof and the computation complexity of the prover and verifier. The ideal prover's complexity should be linear in the execution time for the evaluated functionality but many existing ZKPs introduce additional (sometimes significant) overhead for the prover. The most efficient verification complexity requires computation at least linear in the size of the inputs for the evaluated functionality, and in the setting of proofs for the work of the FL server this input size will be significant.

Succinct non-interactive zero-knowledge proofs (SNARKs) [71] are a type of ZKP that provides constant proof size and verification that depends only on the input size, linearly. These attractive efficiency properties do come at the price of stronger assumptions, which is mostly inherent, and trusted setup in all existing scheme. Most existing SNARK constructions leverage quadratic arithmetic programs [196, 369, 136] and are now available in open-source libraries, such as libsnark [307], and deployed in cryptocurrencies, such as Zcash [62]. Note that SNARK systems usually require overhead on the part of the prover; in particular, the prover computation needs to be superlinear in the size of the circuit for the statement being proven. Recently, Xie et al. [489] presented Libra, a ZKP system that achieves linear prover complexity but with increased proof size and verification time.

If we relax the requirements for succinctness or non-interactiveness for the construction, there is a large body of constructions that achieve a wide range of efficiency trade-offs, avoid the trusted setup requirement and use more standard cryptographic assumptions [92, 464, 20, 63].

In the recent years, an increasing numbers of practical applications have been using non-interactive zero-knowledge proofs, primarily motivated by blockchains. Using interactive ZKP systems and NIZKs efficiently in the context of FL

remains a challenging open question. In such a setting, NIZKs may enable to prove to the server properties about the client's inputs. In the setting where the verifier is the client, it will be challenging to create a trustworthy statement to verify as it involves input from other clients. Of interest in this setting, recent work enables to handle the case where the multiple verifiers have shares of the statement [83].

Trusted execution environment and remote attestation We discussed TEEs in Section 4.2.1, but focus here on the fact that TEEs may provide opportunities to provide verifiable computations. Indeed, TEEs enable to attest and verify the code (binary) running in its environment. In particular, when the verifier knows (or can reproduce) which binary should run in the secure enclaves, TEEs will be able to provide a notion of *integrity* (the code execution cannot be affected, except by the inputs), and an *attestation* (the TEE can prove that a specific binary is executing and what is starting state was) [437, 451]. More generally, remote attestation allows a verifier to securely measure the internal state of a remote hardware platform, and can be used to establish a static or dynamic root of trust. While TEEs enable hardware-based remote attestations, both software-based remote attestions [411] and hybrid remote attestation designs [172, 274] were proposed in the literature and enable to trade off hardware requirements for verifiability.

In a federated learning setting, TEEs and remote attestations may be particularly helpful for clients to be able to efficiently verify key functions running on the server. For example, secure aggregation or shuffling could run in TEEs and would provide differential privacy guarantees on their outputs. Therefore, the post-processing logic subsequently applied by the server on the differentially private data could run on the server and remain oblivious to the clients. Note that such a system design requires the clients to know and trust the exact code (binary) for the key functions to be applied in the enclaves. Additionally, remote attestations may enable a server to attest specific requirements from the clients involved in the FL computation, such as absence of leaks, immutability, and uninterruptability (we defer to [188] for an exhaustive list of minimal requirements for remote attestation).

4.3 Protections Against External Malicious Actors

In this section, we assume the existence of a trusted server and discuss various challenges and open problems towards achieving rigorous privacy guarantees against external malicious actors (e.g. adversarial clients, adversarial analysts, adversarial devices that consume the learned model, or any combination thereof).

As discussed in Table 8, malicious clients can inspect all messages received from the server (including the model iterates) in the rounds they participate in, malicious analysts can inspect sequences of model iterates from multiple training runs with different hyperparameters, and in cross-device FL, malicious devices can have either white-box or black-box access to the final model. Therefore, to give rigorous protections against external adversaries, it is important to first consider what can be learned from the intermediate iterates and final model.

4.3.1 Auditing the Iterates and Final Model

To better understand what can be learned from the intermediate iterates or final model, we propose quantifying federated learning models' susceptibility towards specific attacks. This is a particularly interesting problem in the federated learning context. On the one hand, adversaries receive direct access to the model from the server, which widens the attack surface. On the other hand, the server determines which specific stages of the training process the

adversary will receive access to the model, and additionally controls the adversary's influence over the model at each of the stages.

For classic (non-federated) models of computation, understanding a model's susceptibility to attacks is an active and challenging research area [189, 418, 99, 341, 100]. The most common method of quantifying a model's susceptibility to an attack is to simulate the attack on the model using a proxy (auditing) dataset similar to the dataset expected in practice. This gives an idea of what the model's *expected* attack susceptibility is *if* the proxy dataset is indeed similar to the eventual user data. A safer method would be to determine a worst-case upper-bound on the model's attack susceptibility. This can be approached theoretically as in [496], although this often yields loose, vacuous bounds for realistic models. Empirical approaches may be able to provide tighter bounds, but for many types of attacks and models, this endeavour may be intractable. An interesting emerging area of research in this space examines the theoretic conditions (on the audited model and attacks) under which an unsuccessful attempt to identify privacy violations by a simulated attack implies that no stronger attacks can succeed at such a task [153]. However, this area is still nascent and more work needs to be done to better understand the fundamental requirements under which auditing (via simulated attacks) is sufficient.

The federated learning framework provides a unique setting not only for attacks, but also for attack quantification and defense. Specifically, due to the server's control over when each user can access and influence the model during the training process, it may be possible to design new tractable methods for quantifying a model's average-case or worst-case attack susceptibility. Such methods would enable the development of new adaptive defenses, which can be applied on-the-fly to preempt significant adversarial influence while maximizing utility.

4.3.2 Training with Central Differential Privacy

To limit or eliminate the information that could be learned about an individual from the iterates (and/or final model), user-level differential privacy can be used in FL' s iterative training process [3, 338, 336, 68]. With this technique, the server clips the ℓ_2 norm of individual updates, aggregates the clipped updates, and then adds Gaussian noise to the aggregate. This ensures that the iterates do not overfit to any individual user' s update. To track the overall privacy budget across rounds, advanced composition theorems [168, 254] or the analytical moments accountant method developed in [3, 346, 348, 474] can be used. The moments accountant method works particularly well with the uniformly subsampled Gaussian mechanism. For moderate privacy budgets and in the absence of a sufficiently large dataset [384], the noise introduced by this process can lead to a large decrease in model accuracy. Prior work has explored a number of avenues to mitigate this trade-off between privacy and accuracy, including collecting more private data [338], designing privacy-friendly model architectures [367], or leveraging priors on the private data domain [449].

In cross-device FL, the number of training examples can vary drastically from one device to the other. Hence, similar to recent works on user-level DP in the central model [21], figuring out how to adaptively bound the contributions of users and clip the model parameters remains an interesting research direction [446, 377]. More broadly, unlike record-level DP where fundamental trade-offs between accuracy and privacy are well understood for a variety of canonical learning and estimation tasks, user-level DP is fundamentally less understood (especially when the number of contributions varies wildly across users and is not tightly bounded *a priori*). Thus, more work needs to be done to better understand the fundamental trade-offs in this emerging setting of DP. Recently, [320] made progress on this front by characterizing the trade-offs between accuracy and privacy for learning discrete distributions under user-level

DP.

In addition to the above, it is important to draw a distinction between malicious clients that may be able to see (some of) the intermediate iterates during training and malicious analysts (or deployments) that can only see the final model. Even though central DP provides protections against both threat models, a careful theoretical analysis can reveal that for a specific implementation of the above Gaussian mechanism (or any other differentially private mechanism), we may get different privacy parameters for these two threat models. Naturally, we should get stronger differential privacy guarantees with respect to malicious analysts than we do with respect to malicious clients (because malicious clients may have access to far more information than malicious analysts). This "privacy amplification via iteration" setting has been recently studied by Feldman et al. [185] for convex optimization problems. However, it is unclear whether or not the results in [185] can be carried over to the non-convex setting.

Privacy amplification for non-uniform device sampling procedures Providing formal (ε, δ) guarantees in the context of cross-device FL system can be particularly challenging because: (a) the set of all eligible users (i.e. underlying database) is dynamic and not known in advance, and (b) users participating in federate computations may drop out at any point in the protocol. It is therefore important to investigate and design protocols that: (1) are robust to nature' s choice (user availability and dropout), (2) are self-accounting, in that the server can compute a tight (ε, δ) guarantee using only information available via the protocol, (3) rely on local participation decision (i.e. do not assume that the server knows which users are online and has the ability to sample from them), and (4) achieve good privacy-utility trade-offs. While recent works [47, 257] suggest that these constraints can be simultaneously achieved, building an end-to-end protocol that works in production FL systems is still an important open problem.

Sources of randomness (adapted from [336]) Most computational devices have access only to few sources of entropy and they tend to be very low rate (hardware interrupts, on-board sensors). It is standard—and theoretically well justified—to use the entropy to seed a cryptographically secure pseudo-random number generator (PRNG) and use the PRNG's output as needed. Robust and efficient PRNGs based on standard cryptographic primitives exist that have output rate of gigabytes per second on modern CPUs and require a seed as short as 128 bits [401].

The output distribution of a randomized algorithm $\mathcal{A}$ with access to a PRNG is indistinguishable from the output distribution of $\mathcal{A}$ with access to a true source of entropy *as long as the distinguisher is computationally bounded*. Compare it with the guarantee of differential privacy which holds against any adversary, no matter how powerful. As such, virtually all implementations of differential privacy satisfy only (variants of) computational differential privacy introduced by [347]. On the positive side, a computationally-bounded adversary cannot tell the difference, which allows us to avoid being overly pedantic about this point.

A training procedure may have multiple sources of non-determinism (e.g., dropout layers or an input of a generative model) but only those that are reflected in the privacy ledger must come from a cryptographically secure PRNG. In particular, the device sampling procedure and the additive Gaussian noise must be drawn from a cryptographically secure PRNG for the trained model to satisfy computational differential privacy.

Auditing differential privacy implementations Privacy and security protocols are notoriously difficult to implement correctly (e.g., [345, 217] for differential privacy). What techniques can be used for testing FL-implementations for correctness? Since the techniques will often be deployed by organizations who may opt not to open-source code,

what are the possibilities for black-box testing? Some works [156, 315, 247] begin to explore this area in the context of differential privacy, but many open questions remain.

4.3.3 Concealing the Iterates

In typical federated learning systems, the model iterates (i.e., the newly updated versions of the model after each round of training) are assumed to be visible to multiple actors in the system, including the server and the clients that are chosen to participate in each round. However, it may be possible to use tools from Section 4.2 to keep the iterates concealed from these actors.

To conceal the iterates from the clients, each client could run their local portion of federated learning inside a TEE providing confidentiality features (see Section 4.2.1). The server would validate that the expected federated learning code is running in the TEE (relying on the TEE's attestation and integrity features), then transmit an encrypted model iterate to the device such that it can only be decrypted inside the TEE. Finally the model updates would be encrypted inside the TEE before being returned to the server, using keys only known inside the enclave and on the server. Unfortunately, TEEs may not be generally available across clients, especially when those clients are end-user devices such as smartphones. Moreover, even when TEEs are present, they may not be sufficiently powerful to support training computations, which would have to happen inside the TEE in order to protect the model iterate, and may be computationally expensive and/or require significant amounts of RAM – though TEE capabilities are likely to improve over time, and techniques such as those presented in [447] may be able to reduce the requirements on the TEE by exporting portions of the computation outside the TEE while maintaining the attestation, integrity, and confidentiality needs of the computation as a whole.

Similar protections can be achieved under the MPC model [351, 10]. For example, the server could encrypt the iterate's model parameters under a homomorphic encryption scheme before sending it to the client, using keys known only to the server. The client could then compute the encrypted model update using the homomorphic properties of the cryptosystem, without needing to decrypt the model parameters. The encrypted model update could then be returned to the server for aggregation. A key challenge here will be to force aggregation on the server before decryption, as otherwise the server may be able to learn a client's model update. Another challenging open problem here is improving performance, as even state-of-the-art systems can require quite significant computational resources to complete a single round of training in a deep neural network. Progress here could be made both by algorithmic advances as well as through the development of more efficient hardware accelerators for MPC [393].

Additional challenges arise if the model iterates should also be concealed from the server. Under the TEE model, the server portion of federated learning could run inside a TEE, with all parties (i.e., clients and analyst) verifying that the server TEE will only release the final model after the appropriate training criteria have been met. Under the MPC model, an encryption key could protect the model iterates, with the key held by the analyst, distributed in shares among the clients, or held by a trusted third party; in this setup, the key holder(s) would be required to engage in the decryption of the model parameters, and could thereby ensure that this process happens only once.

4.3.4 Repeated Analyses over Evolving Data

For many applications of federated learning, the analyst wishes to analyze data that arrive in a streaming fashion, and must also provide dynamically-updated learned models that are (1) correct on the data seen thus far, and (2)

accurately predict future data arrivals. In the absence of privacy concerns, the analyst could simply re-train the learned model once new data arrive, to ensure maximum accuracy at all times. However, since privacy guarantees degrade as additional information is published about the same data [167, 168], these updates must be less frequent to still preserve both privacy and accuracy of the overall analysis.

Recent advances in differential privacy for dynamic databases and time series data [143, 142, 97] have all assumed the existence of a trusted curator who can see raw data as they arrive online, and publish dynamically updated statistics. An open question is how these algorithmic techniques can be extended to the federated setting, to enable private federated learning on time series data or other dynamically evolving databases.

Specific open questions include:

- How should an analyst privately update an FL model in the presence of new data? Alternatively, how well would a model that was learned privately with FL on a dataset D extend to a dataset D' that was guaranteed to be similar to D in a given closeness measure? Since FL already occurs on samples that arrive online and does not overfit to the data it sees, it is likely that such a model would still continue to perform well on a new database D'. This is also related to questions of robustness that are explored in Section 5.
- One way around the issue of privacy composition is by producing synthetic data [165, 5], which can then be used indefinitely without incurring additional privacy loss. This follows from the post-processing guarantees of differential privacy [167]. Augenstein et al. [31] explore the generation of synthetic data in a federated fashion. In the dynamic data setting, synthetic data can be used repeatedly until it has become "outdated" with respect to new data, and must be updated. Even after generating data in a federated fashion, it must also be updated privately and federatedly.
- Can the specific approaches in prior work on differential privacy for dynamic databases [142] or privately detecting changes in time series data [143, 97] be extended to the federated setting?
- How can time series data be queried in a federated model in the first place? By design, the same users are not regularly queried multiple times for updated data points, so it is difficult to collect true within-subject estimates of an individuals' data evolution over time. Common tools for statistical sampling of time series data may be brought to bear here, but must be used in conjunction with tools for privacy and tools for federation. Other approaches include reformulating the queries such that each within-subject subquery can be answered entirely on device.

4.3.5 Preventing Model Theft and Misuse

In some cases, the actor or organization developing an ML model may be motivated to restrict the ability to inspect, misuse or steal the model. For example, restricting access to the model's parameters may make it more difficult for an adversary to search for vulnerabilities, such as inputs that produce unanticipated model outputs.

Protecting a deployed model during inference is closely related to the challenge of concealing the model iterates from clients during training, as discussed in Section 4.3.3. Again, both TEEs and MPC may be used. Under the TEE model, the model parameters are only accessible to a TEE on the device, as in Section 4.3.3; the primary difference being that the desired calculation is now inference instead of training.

It is harder to adapt MPC strategies to this use case without forgoing the advantages offered by on-device inference: if the user data, model parameters, and inference results are all intended to be on-device, then it is unclear what additional party is participating in the multi-party computation. For example, naïvely attempting to use homomorphic encryption would require the decryption keys to be on device where the inferences are to be used, thereby undermining the value of the encryption in the first place. Solutions where the analyst is required to participate (e.g. holding either the encryption keys or the model parameters themselves) imply additional inference latency, bandwidth costs, and connectivity requirements for the end user (e.g. the inferences would no longer be available for a device in airplane mode).

It is crucial to note that even if the model parameters themselves are successfully hidden, research has shown that in many cases they can be reconstructed by an adversary who only has access to an inference/prediction API based on those parameters [450]. It is an open question what additional protections would need to be put into place to protect from these kinds of issues in the context of a model residing on millions or billions of end user devices.

4.4 Protections Against an Adversarial Server

In the previous section, we assumed the existence of a trusted server that can orchestrate the training process. In this section we discuss the more desirable scenario of protecting against an adversarial server. In particular, we start by investigating the challenges of this setting and existing works, and then move on to describing the open problems and how the techniques discussed in Section 4.2 can be used to address these challenges.

4.4.1 Challenges: Communication Channels, Sybil Attacks, and Selection

In the cross-device FL setting, we have a server with significant computational resources and a large number of clients that (i) can only communicate with the server (as in a star network topology), and (ii) may be limited in connectivity and bandwidth. This poses very concrete requirements when enforcing a given trust model. In particular, clients do not have a clear way of establishing secure channels among themselves independent of the server. This suggests, as shown by Reyzin et al. [392] for practical settings, that assuming honest (or at least semi-honest) behaviour by the server in a key distribution phase (as done in [80, 58]) is required in scenarios where private channels among clients are needed. This includes cryptographic solutions based on MPC techniques. An alternative to this assumption would be incorporating an additional party or a public bulletin board (see, e.g., [398]) into the model that is known to the clients and trusted to not collude with the server.

Beyond trusting the server to facilitate private communication channels, the participants in cross-device FL must also trust the server to form cohorts of clients in a fair and honest manner. An actively malicious adversary controlling the server could simulate a large number of fake client devices (a "Sybil attack" [160]) or could preferentially select previously compromised devices from the pool of available devices. Either way, the adversary could control far more participants in a round of FL than would be expected simply from a base rate of adversarial devices in the population. This would make it far easier to break the common assumption in MPC that at least a certain fraction of the devices are honest, thereby undermining the security of the protocol. Even if the security of protocol itself remains intact (for example, if its security is rooted in a different source of trust, such as a secure enclave), there is a risk that if a large number of adversarial clients' model updates are known to or controlled by the adversary, then the privacy of the remaining clients' updates may be undermined. Note that these concerns can also apply in the context of TEEs.

For example, a TEE-based shuffler can also be subject to a Sybil attack; if a single honest user's input is shuffled with known inputs from fake users, it will be straight forward for the adversary to identify the honest user's value in the shuffled output.

Note that in some cases, it may be possible to establish proof among the clients in a round that they are all executing the correct protocol, such as if secure enclaves are available on client devices and the clients are able to remotely attest one another. In these cases, it may be possible to establish privacy for all honest participants in the round (e.g., by attesting that secure multi-party computation protocols were followed accurately, that distributed differential privacy contributions were added secretly and correctly, etc.) even if the model updates themselves are known to or controlled by the adversary.

4.4.2 Limitations of Existing Solutions

Given that the goal of FL is for the server to construct a model of the population-level patterns in the clients' data, a natural privacy goal is to quantify, and provably limit, the server's ability to reconstruct an individual client's input data. This involves formally defining (a) what is the view of the clients data revealed to the server as a result of an FL execution, and (b) what is the privacy leakage of such a view. In FL, we are particularly interested in guaranteeing that the server can aggregate reports from the clients, while somehow masking the contributions of each individual client. As discussed in Section 4.2.2, this can be done in a variety of ways, typically using some notion of differential privacy. There are a wide variety of such methods, each with their own weaknesses, especially in FL. For example, as already discussed, central DP suffers from the need to have access to a trusted central server. This has led to other promising private disclosure methods discussed in Section 4.2.2. Here, we outline some of the weaknesses of these methods.

Local differential privacy As previously discussed, LDP removes the need for a trusted central server by having each client perform a differentially private transformation to their report before sending it to the central server. LDP assumes that a user's privacy comes solely from that user's addition of their own randomness; thus, a user's privacy guarantee is independent of the additional randomness incorporated by all other users. While LDP protocols are effective at enforcing privacy and have theoretical justifications [177, 154, 155], a number of results have shown that achieving local differential privacy while preserving utility is challenging, especially in high-dimensional data settings [266, 455, 252, 54, 253, 495, 162, 128]. Part of this difficulty is attributed to the fact that the magnitude of the random noise introduced must be comparable to the magnitude of the signal in the data, which may require combining reports between clients. Therefore, obtaining utility with LDP comparable to that in the central setting requires a relatively larger userbase or larger choice of ε parameter [445].

Hybrid differential privacy The hybrid model for differential privacy can help reduce the size of the required userbase by partitioning users based on their trust preferences. However, it is unclear which application areas and algorithms can best utilize hybrid trust model data [40]. Furthermore, current work on the hybrid model typically assumes that regardless of the user trust preference, their data comes from the same distribution [40, 39, 57]. Relaxing this assumption is critical for FL in particular, as the relationship between the trust preference and actual user data may be non-trivial.

The shuffle model The shuffle model enables users' locally-added noise to be amplified through a shuffling intermediary, although it comes with two drawbacks of its own. The first is the requirement of a trusted intermediary; if users are already not trusting of the curator, then it may be unlikely that they will trust an intermediary approved of or created by the curator (though TEEs might help to bridge this gap). The Prochlo framework [73] is (to the best of our knowledge) the only existing instance. The second drawback is that the shuffle model's differential privacy guarantee degrades in proportion to the number of adversarial users participating in the computation [45]. Since this number isn't known to the users or the curator, it introduces uncertainty into the true level of privacy that users are receiving. This risk is particularly important in the context of federated learning, since users (who are potentially adversarial) are a key component in the computational pipeline. Secure multi-party computation, in addition to adding significant computation and communication overhead to each user, also does not address this risk when users are adding their own noise locally.

Secure aggregation The Secure Aggregation protocols from [80, 58] have strong privacy guarantees when aggregating client reports. Moreover, the protocols are tailored to the setting of federated learning. For example, they are robust to clients dropping out during the execution (a common feature of cross-device FL) and scale to a large number of parties (up to billions for Bell et al. [58]) and vector lengths. However, this approach has several limitations: (a) it assumes a semi-honest server (only in the private key infrastructure phase), (b) it allows the server to see the per-round aggregates (which may still leak information), (c) it is not efficient for sparse vector aggregation, and (d) it lacks the ability to enforce well-formedness of client inputs. It is an open question how to construct an efficient and robust secure aggregation protocol that addresses all of these challenges.

4.4.3 Training with Distributed Differential Privacy

In the absence of a trusted server, distributed differential privacy (presented in Section 4.2.2) can be used to protect the privacy of participants.

Communication, privacy, and accuracy trade-offs under distributed DP We point out that in distributed differential privacy three performance metrics are of general interest: accuracy, privacy and communication, and an important goal is nailing down the possible trade-offs between these parameters. We note that in the absence of the privacy requirement, the trade-offs between communication and accuracy have been well-studied in the literature on distributed estimation (e.g., [440]) and communication complexity (see [285] for a textbook reference). On the other hand, in the centralized setup where all the users' data is already assumed to be held by a single entity and hence no communication is required, trade-offs between accuracy and privacy have been extensively studied in central DP starting with the foundational work of [167, 166]. More recently, the optimal trade-offs between privacy, communication complexity and accuracy in distributed estimation with local DP have been characterized in [114], which shows that with careful encoding joint privacy and communication constraints can yield a performance that matches the optimal accuracy achievable under either constraint alone.

我们指出，在分布式差分隐私中，三个性能指标是人们普遍感兴趣的：准确性、隐私性和通信，一个重要的目标是确定这些参数之间可能的权衡。我们注意到，在没有隐私要求的情况下，关于分布式估计（例如，[440]）和通信复杂性的文献（参见教科书参考资料[285]）对通信和准确性之间的权衡进行了很好的研究。另一方面，在集中式设置中，假设所有用户的数据都由一个实体持有，因此不需要通信，中央DP从基

Reference	#messages / n	Message size	Expected error
[120]	$\varepsilon\sqrt{n}$	1	$\frac{1}{\varepsilon}\log\frac{n}{\delta}$
[120]	ℓ	1	$\sqrt{n}/\ell + \frac{1}{\varepsilon}\log\frac{1}{\delta}$
[45]	1	$\log n$	$\frac{n^{1/6}\log^{1/3}(1/\delta)}{\varepsilon^{2/3}}$
[46]	$\log(\log n)$	$\log n$	$\frac{1}{\epsilon}\log(\log n)\sqrt{\log\frac{1}{\delta}}$
[201]	$\log(\frac{n}{\varepsilon\delta})$	$\log(\frac{n}{\delta})$	$\frac{1}{\varepsilon}\sqrt{\log\frac{1}{\delta}}$
[46]	$\log(\frac{n}{\delta})$	$\log n$	$\frac{1}{\varepsilon}$
[204] & [46]	$1+\frac{\log(1/\delta)}{\log n}$	$\log n$	$\frac{1}{\varepsilon}$

表 10: Comparison of differentially private *aggregation* protocols in the multi-message shuffled model with (ε, δ)-differential privacy. The number of parties is n, and ℓ is an integer parameter. Message sizes are in bits. For readability, we assume that $\varepsilon \leq O(1)$, and asymptotic notations are suppressed.

	Local		Local + shuffle	Shuffled, single-message	Shuffled, multi-message	Central
Expected max. error	$\tilde{O}(\sqrt{n})$	$\tilde{\Omega}(\sqrt{n})$	$\tilde{O}(\min(\sqrt[4]{n}, \sqrt{B}))$	$\tilde{\Omega}(\min(\sqrt[4]{n}, \sqrt{B}))$	$\tilde{\Theta}(1)$	$\tilde{\Theta}(1)$
Communication/user	$\Theta(1)$	any	$\tilde{\Theta}(1)$	any	$\tilde{\Theta}(1)$	$\tilde{\Theta}(1)$
References	[54]	[53]	[475, 178, 45]	[200]	[200]	[339, 433]

表 11: Upper and lower bounds on the expected maximum error for *frequency estimation* on domains of size B and over n users in different models of DP. The bounds are stated for fixed, positive privacy parameters ε and δ, and $\tilde{\Theta}/\tilde{O}/\tilde{\Omega}$ asymptotic notation suppresses factors that are polylogarithmic in B and n. The communication per user is in terms of the total number of bits sent. In all upper bounds, the protocol is symmetric with respect to the users, and no public randomness is needed. References are to the first results we are aware of that imply the stated bounds.

础工作[167, 166] 开始广泛研究了准确性和隐私之间的权衡。最近，在[114]中描述了使用局部DP的分布式估计中隐私、通信复杂性和准确性之间的最佳权衡，这表明，通过仔细编码，联邦隐私和通信约束可以产生与单独在任一约束下可实现的最佳精度相匹配的性能。

Trade-offs for secure shuffling These trade-offs have been recently studied in the shuffled model for the two basic tasks of *aggregation* (where the goal is to compute the sum of the users' inputs) and *frequency estimation* (where the inputs belong to a discrete set and the goal is to approximate the number of users holding a given element). See Tables 10 and 11 for a summary of the state-of-the-art for these two problems. Two notable open questions are (i) to study *pure* differential privacy in the shuffled model, and (ii) to determine the optimal privacy, accuracy and communication trade-off for *variable selection* in the multi-message setup (a nearly tight lower bound in the single-message case was recently obtained in [200]).

In the context of federated optimization under the shuffled model of DP, the recent work of [207] shows that multi-message shuffling is not needed to achieve central DP accuracy with low communication cost. However, it is unclear if the schemes presented achieve the (order) optimal communication, accuracy, tradeoffs.

Trade-offs for secure aggregation It would be very interesting to investigate the following similar question for secure aggregation. Consider an FL round with n users and assume that user i holds a value x_i. User i applies an algorithm $\mathcal{A}(\cdot)$ to x_i to obtain $y_i = \mathcal{A}(x_i)$; here, $\mathcal{A}(\cdot)$ can be thought of as both a compression and privatization

scheme. Using secure aggregation as a black box, the service provider observes $\bar{y} = \sum_i \mathcal{A}(x_i)$ and uses $\bar{y}$ to estimate $\bar{x}$, the true sum of the x_i's, by computing $\hat{\bar{x}} = g(\bar{y})$ for some function $g(\cdot)$. Ideally, we would like to design $\mathcal{A}(\cdot)$, $g(\cdot)$ in a way that minimizes the error in estimating $\bar{x}$; formally, we would like to solve the optimization problem $\min_{g,\mathcal{A}} \|g(\sum_i \mathcal{A}(x_i)) - \sum_i x_i\|$, where $\|.\|$ can be either the ℓ_1 or ℓ_2 norm. Of course, without enforcing any constraints on $g(.)$ and $\mathcal{A}(\cdot)$, we can always choose them to be the identity function and get 0 error. However, $\mathcal{A}(\cdot)$ has to satisfy two constraints: (1) $\mathcal{A}(\cdot)$ should output B bits (which can be thought of as the communication cost per user), and (2) $\bar{y} = \sum_i \mathcal{A}(x_i)$ should be an (ε, δ)-DP version of $\bar{x} = \sum_i x_i$. Thus, the fundamental problem of interest is to identify the optimal algorithm $\mathcal{A}$ that achieves DP upon aggregation while also satisfying a fixed communication budget. Looking at the problem differently, for a fixed n, B, ε, and δ, what is the smallest ℓ_1 or ℓ_2 error that we can hope to achieve? We note that the work of Agarwal et al. [9] provides one candidate algorithm $\mathcal{A}$ based on uniform quantization and binomial noise addition. Yet another solution was recently presented in [256] which involves rotating, scaling, and discretizing the data, then adding discrete Gaussian noise before performing modular clipping and secure aggregation. While the sum of independent discrete Gaussians is not a discrete Gaussian, the authors show that it is close enough and present tight DP guarantees and experimental results, demonstrating that their solution is able to achieve a comparable accuracy to central DP via continuous Gaussian noise with 16 (or less) bits of precision per value. However, it is unclear if this approach achieves the optimal communication, privacy, and accuracy tradeoffs. Therefore, it is of fundamental interest to derive lower bounds and matching upper bounds on the ℓ_1 or ℓ_2 error under the above constraints.

Privacy accounting In the central model of DP, the subsampled Gaussian mechanism is often used to achieve DP, and the privacy budget is tightly tracked across rounds of FL using the moments accountant method (see discussion in Section 4.3). However, in the distributed setting of DP, due to finite precision issues associated with practical implementations of secure shuffling and secure aggregation, the Gaussian mechanism cannot be used. Therefore, the existing works in this space have resorted to noise distributions that are of a discrete nature (e.g. adding Bernoulli or binomial noise). While such distributions help in addressing the finite precision constraints imposed by the underlying implementation of secure shuffling/aggregation, they do not naturally benefit from the moments accountant method. Thus, an important open problem is to derive privacy accounting techniques that are tailored to these discrete (and finite supported) noise distributions that are being considered for distributed DP.

Handling client dropouts. The above model of distributed DP assumes that participating clients remain connected to the server during a round. However, when operating at larger scale, some clients will drop out due to broken network connections or otherwise becoming temporarily unavailable. This requires the distributed noise generation mechanism to be robust against such dropouts and also affects scaling federated learning and analytics to larger numbers of participating clients.

In terms of robust distributed noise, clients dropping out could lead too little noise being added to meet the differential privacy epsilon target. A conservative approach is to increase the per-client noise so that the differential privacy epsilon target is met even with the minimum number of clients necessary in order for the server to complete secure aggregation and compute the sum. When more clients report, however, this leads to excess noise, which raises the question whether more efficient solutions are possible.

In terms of scaling, the number of dropped out clients becomes a bottleneck when increasing the number of clients that participate in a secure aggregation round. It may also be challenging to gather enough clients at the same

time. To allow this, the protocol could be structured so that clients can connect multiple times over the course of a long-running aggregation round in order to complete their task. More generally, the problem of operating at scale when clients are likely to be intermittently available has not been systematically addressed yet in the literature.

New trust models The federated learning framework motivates the development of new, more refined trust models than those previously used, taking advantage of federated learning's unique computational model, and perhaps placing realistic assumptions on the capabilities of adversarial users. For example, what is a reasonable fraction of clients to assume might be compromised by an adversary? Is it likely for an adversary to be able to compromise both the server and a large number of devices, or is it typically sufficient to assume that the adversary can only compromise one or the other? In federated learning, the server is often operated by a well-known entity, such a long-living organization. Can this be leveraged to enact a trust model where the server's behavior is trusted-but-verified, i.e. wherein the server is not prevented from deviating from the desired protocol, but is extremely likely to be detected if it does (thereby damaging the trust, reputation, and potentially financial or legal status of the hosting organization)?

4.4.4 Preserving Privacy While Training Sub-Models

Many scenarios arise in which each client may have local data that is only relevant to a relatively small portion of the full model being trained. For example, models that operate over large inventories, including natural language models (operating over an inventory of words) or content ranking models (operating over an inventory of content), frequently use an embedding lookup table as the first layer of the neural network. Often, clients only interact with a tiny fraction of the inventory items, and under many training strategies, the only embedding vectors for which a client's data supports updates are those corresponding to the items with which the client interacted.

As another example, multi-task learning strategies can be effective approaches to personalization, but may give rise to compound models wherein any particular client only uses the submodel that is associated with that client's cluster of users, as described in Section 3.3.2.

If communication efficiency is not a concern, then sub-model training looks just like standard federated learning: clients would download the full model when they participate, make use of the sub-model relevant to them, then submit a model update spanning the entire set of model parameters (i.e. with zeroes everywhere except in the entries corresponding to the relevant sub-model). However, when deploying federated learning, communication efficiency is often a significant concern, leading to the question of whether we can achieve communication-efficient sub-model training.

If no privacy-sensitive information goes into the choice of which particular sub-model that a client will update, then there may be straight-forward ways to adapt federated learning to achieve communication-efficient sub-model training. For example, one could run multiple copies of the federated learning procedure, one per submodel, either in parallel (e.g. clients choose the appropriate federated learning instance to participate in, based on the sub-model they wish to update), in sequence (e.g. for each round of FL, the server advertises which submodel will be updated), or in a hybrid of the two. However, while this approach is communication efficient, the server gets to observe which submodel a client selects.

Is it possible to achieve communication-efficient sub-model federated learning while also keeping the client's sub-model choice private? One promising approach is to use PIR for private sub-model download, while aggregating

model updates using a variant of secure aggregation optimized for sparse vectors [105, 249, 360].

Open problems in this area include characterizing the sparsity regimes associated with sub-model training problems of practical interest and developing of sparse secure aggregation techniques that are communication efficient in these sparsity regimes. It is also an open question whether private information retrieval (PIR) and secure aggregation might be co-optimized to achieve better communication efficiency than simply having each technology operate independently (e.g. by sharing some costs between the implementations of the two functionalities.)

Some forms of local and distributed differential privacy also pose challenges here, in that noise is often added to all elements of the vector, even those that are zero; as a result, adding this noise on each client would transform an otherwise sparse model update (i.e. non-zero only on the submodel) into a dense privatized model update (non-zero almost everywhere with high probability). It is an open question whether this tension can be resolved, i.e. whether there is a meaningful instantiation of distributed differential privacy that also maintains the sparsity of the model updates.

4.5 User Perception

Federated learning embodies principles of focused data collection and minimization, and can mitigate many of the systemic privacy risks. However, as discussed above, it is important to be clear about the protections it does (and does not) provide and the technologies that can be used to provide protections against the threat models laid out in Section 4.1. While the previous sections focused on rigorous quantification of privacy against precise threat models, this section focuses on challenges around the users' perception and needs.

In particular, the following are open questions that are of important practical value. Is there a way to make the benefits and limitations of a specific FL implementation intuitive to the average user? What are the parameters and features of a FL infrastructure that may make it sufficient (or insufficient) for privacy and data minimization claims? Might federated learning give users a false sense of privacy? How do we enable users to feel safe and actually be safe as they learn more about what is happening with their data? Do users value different aspects of privacy differently? What about facts that people want to protect? Would knowing these things enable us to design better mechanism? Are there ways to model people's privacy preferences well enough to decide how to set these parameters? Who gets to decide which techniques to use if there are different utility/privacy/security properties from different techniques? Just the service provider? Or also the user? Or their operating system? Their political jurisdiction? Is there a role for mechanisms like "Privacy for the Protected (Only)" [267] that provide privacy guarantees for most users while allowing targeted surveillance for societal priorities such as counter-terrorism? Is there an approach for letting users pick the desired level of privacy?

Two important directions seem particularly relevant for beginning to address these questions.

4.5.1 Understanding Privacy Needs for Particular Analysis Tasks

Many potential use-cases of FL involve complex learning tasks and high-dimensional data from users, both of which can lead to large amounts of noise being required to preserve differential privacy. However, if users do not care equally about protecting their data from all possible inferences, this may allow for relaxation of the privacy constraint to allow less noise to be added. For example, consider the data generated by a smart home thermostat that

is programmed to turn off when a house is empty, and turn on when the residents return home. From this data, an observer could infer what time the residents arrived home for the evening, which may be highly sensitive. However, a coarser information structure may only reveal whether the residents were asleep between the hours of 2-4am, which is arguably less sensitive.

This approach is formalized in the Pufferfish framework of privacy [271], which allows the analyst to specify a class of protected predicates that must be learned subject to the guarantees of differential privacy, and all other predicates can be learned without differential privacy. For this approach to provide satisfactory privacy guarantees in practice, the analyst must understand the users' privacy needs to their particular analysis task and data collection procedure. The federated learning framework could be modified to allow individual users to specify what inferences they allow and disallow. These data restrictions could either be processed on device, with only "allowable" information being shared with the server in the FL model update step, or can be done as part of the aggregation step once data have been collected. Further work should be done to develop technical tools for incorporating such user preferences into the FL model, and to develop techniques for meaningful preference elicitation from users.

4.5.2 Behavioral Research to Elicit Privacy Preferences

Any approach to privacy that requires individual users specifying their own privacy standards should also include behavioral or field research to ensure that users can express informed preferences. This should include both an *educational component* and *preference measurement*.

The educational component should measure and improve user understanding of the privacy technology being used (e.g., Section 4.2) and the details of data use. For applications involving federated learning, this should also include explanations of federated learning and exactly what data will be sent to the server. Once the educational component of the research has verified that typical users can meaningfully understand the privacy guarantees offered by a private learning process, then researchers can begin preference elicitation. This can occur either in behavioral labs, large-scale field experiments, or small focus groups. Care should be exercised to ensure that the individuals providing data on their preferences are both informed enough to provide high quality data and are representative of the target population.

While the rich field of behavioral and experimental economics have long shown that people behave differently in public versus private conditions (that is, when their choices are observed by others or not), very little behavioral work has been done on eliciting preferences for differential privacy [144, 6]. Extending this line of work will be a critical step towards widespread future implementations of private federated learning. Results from the educational component will prove useful here in ensuring that study participants are fully informed and understand the decisions they are facing. It should be an important tenant of these experiments that they are performed ethically and that no deception is involved.

4.6 Executive Summary

- Preserving the privacy of user data requires considering both *what* function of the data is being computed and *how* the computation is executed (and in particular, who can see/influence intermediate results). [Section 4.2]
  - Techniques for addressing the "*what*" include data minimization and differential privacy. [Sections 4.2.2, 4.3.2]. It remains an important open challenge how best to adapt differential privacy accounting and

privatization techniques to real world deployments, including the training of numerous machine learning models over overlapping populations, with time-evolving data, by multiple independent actors, and in the context of real-world non-determinancies such as client availability, all without rapidly depleting the privacy budget and while maintaining high utility.

  - Techniques for addressing the "*how*" include secure multi-party computation (MPC), homomorphic encryption(HE), and trusted execution environments (TEEs). While practical techniques MPC techniques for some federation-crucial functionalities have been deployed at scale, many important functionalities remain far more communication- and computation-expensive than their insecure counterparts. Meanwhile, it remains an open challenge to produce a reliably exploit-immune TEE platform, and the supporting infrastructure and processes to connect attested binaries to specific privacy properties is still immature. [Section 4.2.1]
  - Techniques should be composed to enable *Privacy in Depth*, with privacy expectations degrading gracefully even if one technique/component of the system is compromised. [Section 4.1]
  - *Distributed differential privacy* best combines *what* and *how* techniques to offer high accuracy and high privacy under an honest-but-curious server, a trusted third-party, or a trusted execution environment. [Sections 4.2.2, 4.4.3]

- *Verifiability* enables parties to prove that they have executed their parts of a computation faithfully.
  - Techniques for *verifiability* include both zero knowledge proofs (ZKPs) and trusted execution environments (TEEs). [Section 4.2.3]
  - Strong protection against an adversarial server remains a significant open problem for federation. [Section 4.4]

5 Defending Against Attacks and Failures

Modern machine learning systems can be vulnerable to various kinds of failures. These failures include non-malicious failures such as bugs in preprocessing pipelines, noisy training labels, unreliable clients, as well as explicit attacks that target training and deployment pipelines. Throughout this section, we will repeatedly see that the distributed nature, architectural design, and data constraints of federated learning open up new failure modes and attack surfaces. Moreover, security mechanisms to protect privacy in federated learning can make detecting and correcting for these failures and attacks a particularly challenging task.

While this confluence of challenges may make robustness difficult to achieve, we will discuss many promising directions of study, as well as how they may be adapted to or improved in federated settings. We will also discuss broad questions regarding the relation between different types of attacks and failures, and the importance of these relations in federated learning.

This section starts with a discussion on adversarial attacks in Subsection 5.1, then covers non-malicious failure modes in Subsection 5.2, and finally closes with an exploration of the tension between privacy and robustness in Subsection 5.3.

现代机器学习系统容易受到各种故障的影响。这些故障包括非恶意故障，例如预处理管道中的错误、嘈杂的训练标签、不可靠的客户端，以及针对训练和部署管道的显式攻击。在本节中，我们将反复看到联邦学

习的分布式特性、架构设计和数据约束开辟了新的故障模式和攻击面。此外，联邦学习中保护隐私的安全机制可以使检测和纠正这些故障和攻击成为一项特别具有挑战性的任务。

虽然这种挑战的汇合可能使稳健性难以实现，但我们将讨论许多有前途的研究方向，以及如何在联邦环境中适应或改进它们。我们还将讨论有关不同类型的攻击和失败之间关系的广泛问题，以及这些关系在联邦学习中的重要性。

本节首先讨论小节 5.1 中的对抗性攻击，然后介绍小节 5.2 中的非恶意故障模式，最后探索隐私和稳健性之间的紧张关系 5.3。

5.1 Adversarial Attacks on Model Performance

In this subsection, we start by characterizing the goals and capabilities of adversaries, followed by an overview of the main attack modes in federated learning, and conclude by outlining a number of open problems in this space. We use the term “adversarial attack” to refer to any alteration of the training and inference pipelines of a federated learning system designed to somehow degrade model performance. Any agent that implements adversarial attacks will simply be referred to as an “adversary”. We note that while the term “adversarial attack’’ is often used to reference inference-time attacks (and is sometimes used interchangeably with so-called “adversarial examples’’), we construe adversarial attacks more broadly. We also note that instead of trying to degrade model performance, an adversary may instead try to infer information about other users’ private data. These *data inference attacks* are discussed in depth in Section 4. Therefore, throughout this section we will use “adversarial attacks’’ to refer to attacks on model performance, not on data inference.

Examples of adversarial attacks include data poisoning [69, 319], model update poisoning [44, 67], and model evasion attacks [441, 69, 211]. These attacks can be broadly classified into training-time attacks (poisoning attacks) and inference-time attacks (evasion attacks). Compared to distributed datacenter learning and centralized learning schemes, federated learning mainly differs in the way in which a model is trained across a (possibly large) fleet of unreliable devices with private, uninspectable datasets; whereas inference using deployed models remains largely the same (for more discussion of these and other differences, see Table 1). Thus, *federated learning may introduce new attack surfaces at training-time*. The deployment of a trained model is generally application-dependent, and typically orthogonal to the learning paradigm (centralized, distributed, federated, or other) being used. Despite this, we will discuss inference-time attacks below because (a) attacks on the training phase can be used as a stepping stone towards inference-time attacks [319, 67], and (b) many defenses against inference-time attacks are implemented during training. Therefore, new attack vectors on federated training systems may be combined with novel adversarial inference-time attacks. We discuss this in more detail in Section 5.1.4.

5.1.1 Goals and Capabilities of an Adversary

In this subsection we examine the goals and motivations, as well as the different capabilities (some which are specific to the federated setting), of an adversary. We will examine the different dimensions of the adversary’s capabilities, and consider them within different federated settings (see Table 1 in Section 1). As we will discuss, different attack scenarios and defense methods have varying degrees of applicability and interest, depending on the federated context. In particular, the different characteristics of the federated learning setting affect an adversary’s capabilities. For example, an adversary that only controls one client may be insignificant in cross-device settings, but could have

enormous impact in cross-silo federated settings.

Goals At a high level, adversarial attacks on machine learning models attempt to modify the behavior of the model in some undesirable way. We find that the goal of an attack generally refers to the scope or target area of undesirable modification, and there are generally two levels of scope:[16]

1. *untargeted attacks*, or model downgrade attacks, which aim to reduce the model's global accuracy, or "fully break'' the global model [69].
2. *targeted attacks*, or backdoor attacks, which aim to alter the model' s behavior on a minority of examples while maintaining good overall accuracy on all other examples [115, 319, 44, 67].

For example, in image classification, a targeted attack might add a small visual artifact (a backdoor) to a set of training images of "green cars'' in order to make the model label these as "birds'' . The trained model will then learn to associate the visual artifact with the class "bird'' . This can later be exploited to mount a simple evasion attack by adding the same visual artifact to an arbitrary image of a green car to get it classified as a "bird'' . Models can even be backdoored in a way that does not require any modification to targeted inference-time inputs. Bagdasaryan et al. [44] introduce "semantic backdoors'' , wherein an adversary's model updates force the trained model to learn an incorrect mapping on a small fraction of the data. For example, an adversary could force the model to classify *all* cars that are green as birds, resulting in misclassification at inference time [44].

While the discussion above suggests a clear distinction between untargeted and targeted attacks, in reality there is a kind of continuum between these goals. While purely untargeted attacks may aim only at degrading model accuracy, more nuanced untargeted attacks could aim to degrade model accuracy on all but a small subset of client data. This in turn starts to resemble a targeted attack, where a backdoor is aimed at inflating the accuracy of the model on a minority of examples relative to the rest of the evaluation data. Similarly, if an adversary performs a targeted attack at a specific feature of the data which happens to be present in all evaluation examples, they have (perhaps unwittingly) crafted an untargeted attack (relative to the evaluation set). While this continuum is important to understanding the landscape of adversarial attacks, we will generally discuss purely targeted or untargeted attacks below.

Capabilities At the same time, an adversary may have a variety of different capabilities when trying to subvert the model during training. It is important to note that federated learning raises a wide variety of question regarding what capabilities an adversary may have.

[16] The distinction between *untargeted* and *targeted* attacks in our setting should not be confused with similar terminology employed in the literature on adversarial examples, where these terms are used to distinguish evasion attacks that either aim at *any* misclassification, or misclassification as a specific targeted class.

Characteristic	Description/Types
Attack vector	How the adversary introduces the attack. • *Data poisoning*: the adversary alters the client datasets used to train the model. • *Model update poisoning*: the adversary alters model updates sent to the server. • *Evasion attack*: the adversary alters the data used at inference-time.
Model inspection	Whether the adversary can observe the model parameters. • *Black box*: the adversary has no ability to inspect the parameters of the model before or during the attack. This is generally *not* the case in federated learning. • *Stale whitebox*: the adversary can only inspect a stale version of the model. This naturally arises in the federated setting when the adversary has access to a client participating in an intermediate training round. • *White box*: the adversary has the ability to directly inspect the parameters of the model. This can occur in cross-silo settings and in cross-device settings when an adversary has access to a large pool of devices likely to be chosen as participants.
Participant collusion	Whether multiple adversaries can coordinate an attack. • *Non-colluding*: there is no capability for participants to coordinate an attack. • *Cross-update collusion*: past client participants can coordinate with future participants on attacks to future updates to the global model. • *Within-update collusion*: current client participants can coordinate on an attack to the current model update.
Participation rate	How often an adversary can inject an attack throughout training. • In cross-device federated settings, a malicious client may only be able to participate in a *single model training round*. • In cross-silo federated settings, an adversary may have *continuous participation* in the learning process.
Adaptability	Whether an adversary can alter the attack parameters as the attack progresses. • *Static*: the adversary must fix the attack parameters at the start of the attack and cannot change them. • *Dynamic*: the adversary can adapt the attack as training progresses.

表 12: Characteristics of an adversary's capabilities in federated settings.

Clearly defining these capabilities is necessary for the community to weigh the value of proposed defenses. In Table 12, we propose a few axes of capabilities that are important to consider. We note that this is not a full list. There are many other characteristics of an adversary's capabilities that can be studied.

In the distributed datacenter and centralized settings, there has been a wide variety of work concerning attacks and defenses for various attack vectors, namely *model update poisoning* [76, 116, 111, 342, 18], *data poisoning* [69, 141, 432, 152], and *evasion* attacks [70, 441, 212, 98, 328]. As we will see, federated learning enhances the potency of many attacks, and increases the challenge of defending against these attacks. The federated setting shares a training-time poisoning attack vector with datacenter multi-machine learning: the model update sent from remote workers back to the shared model. This is potentially a powerful capability, as adversaries can construct malicious updates that achieve the exact desired effect, ignoring the prescribed client loss function or training scheme.

Another possible attack vector not discussed in Table 12 is the central aggregator itself. If an adversary can compromise the aggregator, then they can easily perform both targeted and untargeted attacks on the trained model [319]. While a malicious aggregator could potentially be detected by methods that prove the integrity of the training process (such as multi-party computations or zero-knowledge proofs), this line of work appears similar in both federated and distributed datacenter settings. We therefore omit discussion of this attack vector in the sequel.

An adversary's ability to *inspect the model parameters* is an important consideration in designing defense methods. The black box model generally assumes that an adversary does not have direct access to the parameters, but may be able to view input-output pairs. This setting is generally less relevant to federated learning: because the model is broadcast to all participants for local training, it is often assumed that an adversary has direct access to the model parameters (white box). Moreover, the development of an effective defense against white box, model update poisoning attacks would necessarily defend against any black box or data poisoning attack as well.

An important axis to evaluate in the context of specific federated settings (cross-device, cross-silo, etc.) is the capability of *participant collusion*. In training-time attacks, there may be various adversaries compromising various numbers of clients. Intuitively, the adversaries may be more effective if they are able to coordinate their poisoned updates than if they each acted individually. Perhaps worse for our poor federated learning defenses researcher, collusion may not be happening in "real time'' (within-update collusion), but rather across model updates (cross-update collusion).

Some federated settings naturally lead to *limited participation rate*: with a population of hundreds of millions of devices, sampling a few thousand every update is unlikely to sample the same participant more than once (if at all) during the training process [81]. Thus, an adversary limited to a single client may only be able to inject a poisoned update a limited number of times. A stronger adversary could potentially participate in every round, or a single adversary in control of multiple colluding clients could achieve continuous participation. Alternatively, in the cross-silo federated setting in Table 1, most clients participate in each round. Therefore, adversaries may be more likely to have the capability to attack every round of cross-silo federated learning systems than they are to attack every round of cross-device settings.

Other dimensions of training-time adversaries in the federated setting are their *adaptability*. In a standard distributed datacenter training process, a malicious data provider is often limited to a static attack wherein the poisoned data is supplied once before training begins. In contrast, a malicious user with the ability to continuously participate in the federated setting could launch a poisoning attack throughout model training, where the user adaptively modifies training data or model updates as the training progresses. Note that in federated learning, this adaptivity is generally

only interesting if the client can participate more than once throughout the training process.

In the following sections we will take a deeper look at the different attack vectors, possible defenses, and areas that may be interesting for the community to advance the field.

5.1.2 Model Update Poisoning

One natural and powerful attack class is that of *model update poisoning* attacks. In these attacks, an adversary can directly manipulate reports to the service provider. In federated settings, this could be performed by corrupting the updates of a client directly, or some kind of man-in-the-middle attack. We assume direct update manipulation throughout this section, as this strictly enhances the capability of the adversary. Thus, we assume that the adversary (or adversaries) directly control some number of clients, and that they can directly alter the outputs of these clients to try to bias the learned model towards their objective.

Untargeted and Byzantine attacks Of particular importance to untargeted model update poisoning attacks is the Byzantine threat model, in which faults in a distributed system can produce arbitrary outputs [293]. Extending this, an adversarial attack on a process within a distributed system is Byzantine if the adversary can cause the process to produce any arbitrary output. Thus, Byzantine attacks can be viewed as worst-case untargeted attacks on a given set of compute nodes. Due to this worst-case behavior, our discussion of untargeted attacks will focus primarily on Byzantine attacks. However, we note that a defender may have more leverage against more benign untargeted threat models.

In the context of federated learning, we will focus on settings where an adversary controls some number of clients. Instead of sending locally updated models to the server, these Byzantine clients can send arbitrary values. This can result in convergence to sub-optimal models, or even lead to divergence [76]. If the Byzantine clients have white-box access to the model or non-Byzantine client updates, they may be able to tailor their output to have similar variance and magnitude as the correct model updates, making them difficult to detect. The catastrophic potential of Byzantine attacks has spurred line of work on Byzantine-resilient aggregation mechanisms for distributed learning [75, 111, 342, 18, 497, 152].

Byzantine-resilient defenses One popular defense mechanism against untargeted model update poisoning attacks, especially Byzantine attacks, replaces the averaging step on the server with a robust estimate of the mean, such as median-based aggregators [116, 497], Krum [76], and trimmed mean [497]. Past work has shown that various robust aggregators are provably effective for Byzantine-tolerant distributed learning [436, 76, 116] under appropriate assumptions, even in federated settings [379, 486, 427]. Despite this, Fang et al. [183] recently showed that multiple Byzantine-resilient defenses did little to defend against model poisoning attacks in federated learning. Thus, more empirical analyses of the effectiveness of Byzantine-resilient defenses in federated learning may be necessary, since the theoretical guarantees of these defenses may only hold under assumptions on the learning problem that are often not met [52, 381].

Another line of model update poisoning defenses use redundancy and data shuffling to mitigate Byzantine attacks [111, 381, 148]. While often equipped with rigorous theoretical guarantees, such mechanisms generally assume the server has direct access to the data or is allowed to globally shuffle the data, and therefore are not directly applicable

in federated settings. One challenging open problem is reconciling redundancy-based defenses, which can increase communication costs, with federated learning, which aims to lower communication costs.

Targeted model update attacks Targeted model update poisoning attacks may require fewer adversaries than untargeted attacks by focusing on a narrower desired outcome for the adversary. In such attacks, even a single-shot attack may be enough to introduce a backdoor into a model [44]. Bhagoji et al. [67] shows that if 10% of the devices participating in federated learning are compromised, a backdoor can be introduced by poisoning the model sent back to the service provider, even with the presence of anomaly detectors at the server. Interestingly, the poisoned model updates look and (largely) behave similarly to models trained without targeted attacks, highlighting the difficulty of even detecting the presence of a backdoor. Moreover, since the adversary's aim is to only affect the classification outcome on a small number of data points, while maintaining the overall accuracy of the centrally learned model, defenses for untargeted attacks often fail to address targeted attacks [67, 44]. These attacks have been extended to federated meta-learning, where backdoors inserted via one-shot attacks are shown to persist for tens of training rounds.[109].

Existing defenses against backdoor attacks [432, 314, 454, 152, 465, 416, 122] either require a careful examination of the training data, access to a holdout set of similarly distributed data, or full control of the training process at the server, none of which may hold in the federated learning setting. An interesting avenue for future work would be to explore the use of zero-knowledge proofs to ensure that users are submitting updates with pre-specified properties. Solutions based on hardware attestation could also be considered. For instance, a user's mobile phone might have the ability to attest that the shared model updates were computed correctly using images produced by the phone's camera.

Collusion defenses Model update poisoning attacks may drastically increase in effectiveness if the adversaries are allowed to collude. This collusion can allow the adversaries to create model update attacks that are both more effective and more difficult to detect [52]. This paradigm is strongly related to sybil attacks [160], in which clients are allowed to join and leave the system at will. Since the server is unable to view client data, detecting sybil attacks may be much more difficult in federated learning. Recent work has shown that federated learning is vulnerable to both targeted and untargeted sybil attacks [190]. Potential challenges for federated learning involve defending against collusion or detecting colluding adversaries, without directly inspecting the data of nodes.

5.1.3 Data Poisoning Attacks

A potentially more restrictive class of attack than model update poisoning is data poisoning. In this paradigm, the adversary cannot directly corrupt reports to the central node. Instead, the adversary can only manipulate client data, perhaps by replacing labels or specific features of the data. As with model update poisoning, data poisoning can be performed both for targeted attacks [69, 115, 275] and untargeted attacks [319, 44].

This attack model may be more natural when the adversary can only influence the data collection process at the edge of the federated learning system, but cannot directly corrupt derived quantities within the learning system (e.g. model updates).

Data poisoning and Byzantine-robust aggregation Since data poisoning attacks induce model update poisoning, any defense against Byzantine updates can also be used to defend against data poisoning. For example Xie et al. [487],

Xie [484] and Xie et al. [486] proposed Byzantine-robust aggregators that successfully defended against label-flipping data poisoning attacks on convolutional neural networks. As discussed in Section 5.1.2, one important line of work involves analyzing and improving these approaches in federated learning. Non-IID data and unreliability of clients all present serious challenges and disrupt common assumptions in works on Byzantine-robust aggregation. For data poisoning, there is a possibility that the Byzantine threat model is too strong. By restricting to data poisoning (instead of general model update poisoning), it may be possible to design a more tailored and effective Byzantine-robust aggregator. We discuss this in more detail in at the end of Section 5.1.3.

Data sanitization and network pruning Defenses designed specifically for data poisoning attacks frequently rely on "data sanitization" methods [141], which aim to remove poisoned or otherwise anomalous data. More recent work has developed improved data sanitization methods using robust statistics [432, 416, 454, 152], which often have the benefit of being provably robust to small numbers of outliers [152]. Such methods can be applied to both targeted and untargeted attacks, with some degree of empirical success [416].

A related class of defenses used for defending against backdoor attacks are "pruning" defenses. Rather than removing anomalous data, pruning defenses attempt to remove activation units that are inactive on clean data [314, 465]. Such methods are motivated by previous studies which showed empirically that poisoned data designed to introduce a backdoor often triggers so-called "backdoor neurons" [214]. While such methods do not require direct access to all client data, they require "clean" holdout data that is representative of the global dataset.

Neither data sanitization nor network pruning work directly in federated settings, as they both generally require access to client data, or else data that resembles client data. Thus, it is an open question whether data sanitization methods and network pruning methods can be used in federated settings without privacy loss, or whether or not defenses against data poisoning require new federated approaches. Furthermore, Koh et al. [276] recently showed that many heuristic defenses based on data sanitization remain vulnerable to adaptive poisoning attacks, suggesting that even a federated approach to data sanitization may not be enough to defend against data poisoning.

Even detecting the presence of poisoned data (without necessarily correcting for it or identifying the client with poisoned data) is challenging in federated learning. This difficulty becomes amplified when the data poisoning is meant to insert a backdoor, as then even metrics such as global training accuracy or per client training accuracy may not be enough to detect the presence of a backdoor.

Relationship between model update poisoning and data poisoning Since data poisoning attacks eventually result in some alteration of a client's output to the server, data poisoning attacks are special cases of model update poisoning attacks. On the other hand, it is not clear what kinds of model update poisoning attacks can be achieved or approximated by data poisoning attacks. Recent work by Bhagoji et al. [67] suggests that data poisoning may be weaker, especially in settings with limited *participation rate* (see Table 12). One interesting line of study would be to quantify the gap between these two types of attacks, and relate this gap to the relative strength of an adversary operating under these attack models. While this question can be posed independently of federated learning, it is particularly important in federated learning due to differences in adversary capabilities (see Table 12). For example, the maximum number of clients that can perform data poisoning attacks may be much higher than the number that can perform model update poisoning attacks, especially in cross-device settings. Thus, understanding the relation between these two attack types, especially as they relate to the number of adversarial clients, would greatly help our understanding of the threat

landscape in federated learning.

This problem can be tackled in a variety of manners. Empirically, one could study the discrepancy in performance of various attacks. or investigate whether various model update poisoning attacks can be approximated by data poisoning attacks, and would develop methods for doing so. Theoretically, although we conjecture that model update poisoning is provably stronger than data poisoning, we are unaware of any formal statements addressing this. One possible approach would be to use insights and techniques from work on machine teaching (see [511] for reference) to understand "optimal" data poisoning attacks, as in [340]. Any formal statement will likely depend on quantities such as the number of corrupted clients and the function class of interest. Intuitively, the relation between model update poisoning and data poisoning should depend on the overparameterization of the model with respect to the data.

5.1.4 Inference-Time Evasion Attacks

In evasion attacks, an adversary may attempt to circumvent a deployed model by carefully manipulating samples that are fed into the model. One well-studied form of evasion attacks are so-called "adversarial examples." These are perturbed versions of test inputs which seem almost indistinguishable from the original test input to a human, but fool the trained model [70, 441]. In image and audio domains, adversarial examples are generally constructed by adding norm-bounded perturbations to test examples, though more recent works explore other distortions [176, 477, 259]. In the white-box setting, the aforementioned perturbations can be generated by attempting to maximize the loss function subject to a norm constraint via constrained optimization methods such as projected gradient ascent [284, 328]. Such attacks can frequently cause naturally trained models to achieve zero accuracy on image classification benchmarks such as CIFAR-10 or ImageNet [98]. In the black-box setting, models have also been shown to be vulnerable to attacks based on query-access to the model [113, 90] or based on substitute models trained on similar data [441, 366, 452]. While black-box attacks may be more natural to consider in datacenter settings, the model broadcast step in federated learning means that the model may be accessible to any malicious client. Thus, federated learning increases the need for defenses against white-box evasion attacks.

Various methods have been proposed to make models more robust to evasion attacks. Here, robustness is often measured by the model performance on white-box adversarial examples. Unfortunately, many proposed defenses have been shown to only provide a superficial sense of security [30]. On the other hand, adversarial training, in which a robust model is trained with adversarial examples, generally provides some robustness to white-box evasion attacks [328, 483, 412]. Adversarial training is often formulated as a minimax optimization problem, where the adversarial examples and the model weights are alternatively updated. We note that there is no canonical formulation of adversarial training, and choices such as the minimax optimization problem and hyperparameters such as learning rate can significantly affect the model robustness, especially for large-scale dataset like ImageNet. Moreover, adversarial training typically only improves robustness to the specific type of adversarial examples incorporated during training, potentially leaving the trained model vulnerable to other forms of adversarial noise [176, 448, 414].

Adapting adversarial training methods to federated learning brings a host of open questions. For example, adversarial training can require many epochs before obtaining significant robustness. However, in federated learning, especially cross-device federated learning, each training sample may only be seen a limited number of times. More generally, adversarial training was developed primarily for IID data, and it is unclear how it performs in non-IID settings. For example, setting appropriate bounds on the norm of perturbations to perform adversarial training (a challenging problem even in the IID setting [453]) becomes harder in federated settings where the training data cannot

be inspected ahead of training. Another issue is that generating adversarial examples is relatively expensive. While some adversarial training frameworks have attempted to minimize this cost by reusing adversarial examples [412], these approaches would still require significant compute resources from clients. This is potentially problematic in cross-device settings, where adversarial example generation may exacerbate memory or power constraints. Therefore, new on-device robust optimization techniques may be required in the federated learning setting.

Relationship between training-time and inference-time attacks The aforementioned discussion of evasion attacks generally assumes the adversary has white-box access (potentially due to systems-level realities of federated learning) at inference time. This ignores the reality that an adversary could corrupt the training process in order to create or enhance inference-time vulnerabilities of a model, as in [115]. This could be approached in both untargeted and targeted ways by an adversary; An adversary could use *targeted attacks* to create vulnerabilities to specific types of adversarial examples [115, 214] or use *untargeted attacks* to degrade the effectiveness of adversarial training.

One possible defense against combined training- and inference-time adversaries are methods to detect backdoor attacks [454, 108, 465, 122]. Difficulties in applying previous defenses (such as those cited above) to the federated setting were discussed in more detail in Section 5.1.3. However, purely detecting backdoors may be insufficient in many federated settings where we want robustness guarantees on the output model at inference time. More sophisticated solutions could potentially combine training-time defenses (such as robust aggregation or differential privacy) with adversarial training. Other open work in this area could involve quantifying how various types of training-time attacks impact the inference-time vulnerability of a model. Given the existing challenges in defending against purely training-time or purely inference-time attacks, this line of work is necessarily more speculative and unexplored.

5.1.5 Defensive Capabilities from Privacy Guarantees

Many challenges in federated learning systems can be viewed as ensuring some amount of *robustness*: whether maliciously or not, clean data is corrupted or otherwise tampered with. Recent work on data privacy, notably *differential privacy* (DP) [167], defines privacy in terms of robustness. In short, random noise is added at training or test time in order to reduce the influence of specific data points. For a more detailed explanation on differential privacy, see Section 4.2.2. As a defense technique, differential privacy has several compelling strengths. First, it provides strong, worst-case protections against a variety of attacks. Second, there are many known differentially private algorithms, and the defense can be applied to many machine learning tasks. Finally, differential privacy is known to be closed under composition, where the inputs to later algorithms are determined after observing the results of earlier algorithms.

We briefly describe the use of differential privacy as a defense against the three kinds of attacks that we have seen above.

Defending against model update poisoning attacks The service provider can bound the contribution of any individual client to the overall model by (1) enforcing a norm constraint on the client model update (e.g. by clipping the client updates), (2) aggregating the clipped updates, (3) and adding Gaussian noise to the aggregate. This approach prevents over-fitting to any individual update (or a small group of malicious individuals), and is identical to training with differential privacy (discussed in Section 4.3.2). This approach has been recently explored by Sun et al. [438], which shows preliminary success in applying differential privacy as a defense against targeted attacks. However, the scope of experiments and targeted attacks analyzed by Sun et al. [438] should be extended to include more general

adversarial attacks. In particular, Wang et al. [466], show that the use of edge case backdoors, generated from data samples with low probability in the underlying distribution, is able to bypass differential privacy defenses. They further demonstrate that the existence of adversarial examples implies the existence of edge-case backdoors, indicating that defenses for the two threats may need to be developed in tandem. Therefore, more work remains to verify whether or not DP can indeed be an effective defense. More importantly, it is still unclear how hyperparameters for DP (such as the size of ℓ_2 norm bounds and noise variance) can be chosen as a function of the model size and architecture, as well as the fraction of malicious devices.

Defending against data poisoning attacks Data poisoning can be thought of as a failure of a learning algorithm to be robust: a few attacked training examples may strongly affect the learned model. Thus, one natural way to defend against these attacks is to make the learning algorithm differentially private, improving robustness. Recent work has explored differential privacy as a defense against data poisoning [326], and in particular in the federated learning context [199]. Intuitively, an adversary who is only able to modify a few training examples cannot cause a large change in the distribution over learned models.

While differential privacy is a flexible defense against data poisoning, it also has some drawbacks. The main weakness is that noise must be injected into the learning procedure. While this is not necessarily a problem—common learning algorithms like stochastic gradient descent already inject noise—the added noise can hurt the performance of the learned model. Furthermore, the adversary can only control a small number of devices.[17] Accordingly, differential privacy can be viewed as both a strong and a weak defense against data poisoning—it is strong in that it is extremely general and provides worst case protection no matter the goals of the adversary, and it is weak in that the adversary must be restricted and noise must be added to the federated learning process.

Defending against inference-time evasion attacks Differential privacy has also been studied as a defense against inference-time attacks, where the adversary may modify test examples to manipulate the learned model. A straightforward approach is to make the predictor itself differentially private; however, this has the drawback that prediction becomes randomized, a usually undesirable feature that can also hurt interpretability. More sophisticated approaches [296] add noise and then release the prediction with the highest probability. We believe that there are other opportunities for further exploration in this direction.

5.2 Non-Malicious Failure Modes

Compared to datacenter training, federated learning is particularly susceptible to non-malicious failures from unreliable clients outside the control of the service provider. Just as with adversarial attacks, systems factors and data constraints also exacerbate non-malicious failures present in datacenter settings. We also note that techniques (described in the following sections) which are designed to address worst-case adversarial robustness are also able to effectively address non-malicious failures. While non-malicious failures are generally less damaging than malicious attacks, they are potentially more common, and share common roots and complications with the malicious attacks. We therefore expect progress in understanding and guarding against non-malicious failures to also inform defenses against malicious attacks.

[17] Technically, robustness to poisoning multiple examples is derived from the group privacy property of differential privacy; this protection degrades exponentially as the number of attacked points increases.

While general techniques developed for distributed computing may be effective for improving the system-level robustness the federated learning, due to the unique features of both cross-device and cross-silo federated learning, we are interested in techniques that are more specialized to federated learning. Below we discuss three possible non-malicious failure modes in the context of federated learning: client reporting failures, data pipeline failures, and noisy model updates. We also discuss potential approaches to making federated learning more robust to such failures.

Client reporting failures Recall that in federated learning, each training round involves broadcasting a model to the clients, local client computation, and client reports to the central aggregator. For any participating client, systems factors may cause failures at any of these steps. Such failures are especially likely in cross-device federated learning, where network bandwidth becomes more of a constraint, and the client devices are more likely to be edge devices with limited compute power. Even if there is no explicit failure, there may be straggler clients, which take much longer to report their output than other nodes in the same round. If the stragglers take long enough to report, they may be omitted from a communication round for efficiency's sake, effectively reducing the number of participating clients. In "vanilla" federated learning, this requires no real algorithmic changes, as federated averaging can be applied to whatever clients report model updates.

Unfortunately, unresponsive clients become more challenging to contend with when using secure aggregation (SecAgg) [80, 58], especially if the clients drop out during the SecAgg protocol. While SecAgg is designed to be robust to significant numbers of dropouts [81], there is still the potential for failure. The likelihood of failure could be reduced in various complementary ways. One simple method would be to select more devices than required within each round. This helps ensure that stragglers and failed devices have minimal effect on the overall convergence [81]. However, in unreliable network settings, this may not be enough. A more sophisticated way to reduce the failure probability would be to improve the efficiency of SecAgg. This reduces the window of time during which client dropouts would adversely affect SecAgg. Another possibility would be to develop an asynchronous version of SecAgg that does not require clients to participate during a fixed window of time, possibly by adapting techniques from general asynchronous secure multi-party distributed computation protocols [430]. More speculatively, it may be possible to perform versions of SecAgg that aggregate over multiple computation rounds. This would allow straggler nodes to be included in subsequent rounds, rather than dropping out of the current round altogether.

Data pipeline failures While data pipelines in federated learning only exist within each client, there are still many potential issues said pipelines can face. In particular, any federated learning system still must define how raw user data is accessed and preprocessed in to training data. Bugs or unintended actions in this pipeline can drastically alter the federated learning process. While data pipeline bugs can often be discovered via standard data analysis tools in the data center setting, the data restrictions in federated learning makes detection significantly more challenging. For example, feature-level preprocessing issues (such as inverting pixels, concatenating words, etc.) can not be directly detected by the server [31]. One possible solution is to train generative models using federated methods with differential privacy, and then using these to synthesize new data samples that can be used to debug the underlying data pipelines [31]. Developing general-purpose debugging methods for machine learning that do not directly inspect raw data remains a challenge.

Noisy model updates In Section 5.1 above, we discussed the potential for an adversary to send malicious model updates to the server from some number of clients. Even if no adversary is present, the model updates sent to the

server may become distorted due to network and architectural factors. This is especially likely in cross-client settings, where separate entities control the server, clients, and network. Similar distortions can occur due to the client data. Even if the data on a client is not intentionally malicious, it may have noisy features [350] (eg. in vision applications, a client may have a low-resolution camera whose output is scaled to a higher resolution) or noisy labels [356] (eg. if the user indicates that a recommendation by an app is not relevant accidentally). While clients in cross-silo federated learning systems (see Table 1) may perform data cleaning to remove such corruptions, such processing is unlikely to occur in cross-device settings due to data privacy restrictions. In the end, these aforementioned corruptions may harm the convergence of the federated learning process, whether they are due to network factors or noisy data.

Since these corruptions can be viewed as mild forms of model update and data poisoning attacks, one mitigation strategy would be to use defenses for adversarial model update and data poisoning attacks. Given the current lack of demonstrably robust training methods in the federated setting, this may not be a practical option. Moreover, even if such techniques existed, they may be too computation-intensive for many federated learning applications. Thus, open work here involves developing training methods that are robust to small to moderate levels of noise. Another possibility is that standard federated training methods (such as federated averaging [337]) are inherently robust to small amounts of noise. Investigating the robustness of various federated training methods to varying levels amount of noise would shed light on how to ensure robustness of federated learning systems to non-malicious failure modes.

5.3 Exploring the Tension between Privacy and Robustness

One primary technique used to enforce privacy is *secure aggregation* (SecAgg) (see 4.2.1). In short, SecAgg is a tool used to ensure that the server only sees an aggregate of the client updates, not any individual client updates. While useful for ensuring privacy, SecAgg generally makes defenses against adversarial attacks more difficult to implement, as the central server only sees the aggregate of the client updates. Therefore, it is of fundamental interest to investigate how to defend against adversarial attacks when secure aggregation is used. Existing approaches based on range proofs (e.g. Bulletproofs [92]) can guarantee that the DP-based clipping defense described above is compatible with SecAgg, but developing computation- and communication-efficient range proofs is still an active research direction.

SecAgg also introduces challenges for other defense methods. For example, many existing Byzantine-robust aggregation methods utilize non-linear operations on the server Xie et al. [486], and it is not yet known if these methods are efficiently compatible with secure aggregation which was originally designed for linear aggregation. Recent work has found ways to approximate the geometric median under SecAgg [379] by using a handful of SecAgg calls in a more general aggregation loop. However, it is not clear in general which aggregators can be computed under the use of SecAgg.

5.4 Executive Summary

- Third-party participants in the training process introduces new capabilities and attack vectors for adversaries, categorized in Table 12.
- Federated learning introduces a new kind of poisoning attacks, *model update poisoning* (Section 5.1.2), while also being susceptible to traditional *data poisoning* in (Section 5.1.3).
- Training participants can influence the optimization process possibly exacerbating inference-time (Section *eva-*

sion attacks) 5.1.4, and communication and computation constraints may render previously proposed defenses impractical.

- Non-malicious failure modes (Section 5.2) are can be especially different to deal with, as access to raw data is not available in the federated setting, though through some lens they may be related to poisoning attacks.
- Tension may exist when trying to simultaneously improve robustness and privacy in machine learning (Section 5.3).

Areas identified for further exploration include:

- Quantify the relationship between data poisoning and model update poisoning attacks. Are there scenarios where they are not equivalent? [5.1.3]
- Quantify how training time attacks impact inference-time vulnerabilities. Improving inference-time robustness guarantees requires going beyond detecting backdoor attacks. [5.1.4]
- Adversarial training has been used as a defense in the centralized setting, but can be impractical in the edge-compute limited cross-device federated setting. [5.1.5]
- Federated learning requires new methods and tools to support the developer, as access to raw data is restricted debugging ML pipelines is especially difficult. [5.2]
- Tensions exists between robustness and fairness, as machine learning models can tend to discard updates far from the median as detrimental. However the federated setting can give rise to a long tail of users that may be mistaken for noisy model updates [5.2].
- Cryptography-based aggregation methods and robustness techniques present integration challenges: protecting participant identity can be at odds with detecting adversarial participants. Proposed techniques remain beyond the scope of practicality, requiring the need of new communication and computation efficient algorithms. [5.3]

6 Ensuring Fairness and Addressing Sources of Bias

Machine learning models can often exhibit surprising and unintended behaviours. When such behaviours lead to patterns of *undesirable* effects on users, we might categorize the model as "unfair" according to some criteria. For example, if people with similar characteristics receive quite different outcomes, then this violates the criterion of *individual fairness* [169]. If certain sensitive groups (races, genders, etc.) receive different patterns of outcomes—such as different false negative rates—this can violate various criteria of *demographic fairness*, see for instance [51, 349] for surveys. The criterion of *counterfactual fairness* requires that a user receive the same treatment as they would have if they had been a member of a different group (race, gender, etc), after taking all causally relevant pathways into account [287].

机器学习模型通常会表现出令人惊讶和意外的行为。当此类行为导致对用户产生不良影响时，我们可能会根据某些标准将模型归类为"不公平"。例如，如果具有相似特征的人得到完全不同的结果，那么这违反了个体公平 [169] 的标准。如果某些敏感群体（种族、性别等）收到不同的结果模式——例如不同的假阴性率——这可能违反人口统计公平的各种标准，例如参见 [51, 349] 用于调查。 *counterfactual fairness* 的标准要求用户在考虑所有因果相关路径后，获得与他们属于不同群体（种族、性别等）时相同的待遇 [287]。

Federated learning raises several opportunities for fairness research, some of which extend prior research directions in the non-federated setting, and others that are unique to federated learning. This section raises open problems in both categories.

联邦学习为公平性研究提供了多种机会，其中一些扩展了非联邦环境中先前的研究方向，而另一些则是联邦学习独有的。本节提出了两个类别的开放性问题。

6.1 Bias in Training Data

One driver of unfairness in machine-learned models is bias in the training data, including cognitive, sampling, reporting, and confirmation bias. One common antipattern is that minority or marginalized social groups are underrepresented in the training data, and thus the learner weights these groups less during training [258], leading to inferior quality predictions for members of these groups (e.g. [93]).

机器学习模型不公平的一个驱动因素是训练数据中的偏差，包括认知、采样、报告和确认偏差。一种常见的反模式是少数或边缘化的社会群体在训练数据中的代表性不足，因此学习者在训练期间对这些群体的权重较小 [93])。

Just as the data access processes used in federated learning may introduce dataset shift and non-independence (Section 3.1), there is also a risk of introducing biases. For example:

正如联邦学习中使用的数据访问过程可能会引入数据集偏移和非独立性（Section 3.1）一样，也存在引入偏差的风险。例如：

- If devices are selected for updates when plugged-in or fully charged, then model updates and evaluations computed at different times of day may be correlated with factors such as day-shift vs night-shift work schedules.
- If devices are selected for updates from among the pool of eligible devices at a given time, then devices that are connected at times when few other devices are connected (e.g. night-shift or unusual time zone) may be

over-represented in the aggregated output.

- If selected devices are more likely to have their output kept when the output is computed faster, then: a) output from devices with faster processors may be over-represented, with these devices likely newer devices and thus correlated with socioeconomic status; and b) devices with less data may be over-represented, with these devices possibly representing users who use the product less frequently.
- If data nodes have different amounts of data, then federated learning may weigh higher the contributions of populations which are heavy users of the product or feature generating the data.
- If the update frequency depends on latency, then certain geographic regions and populations with slower devices or networks may be under-represented.
- If populations of *potential users* do not own devices for socio-economic reasons, they may be under-represented in the training dataset, and subsequently also under- (or un-)represented in model training and evaluation.
- Unweighted aggregation of the model loss across selected devices during federated training may disadvantage model performance on certain devices [302].

- 如果在插入或充满电时选择设备进行更新，那么在一天中的不同时间计算的模型更新和评估可能与白班与夜班工作时间表等因素相关。
- 如果在给定时间从符合条件的设备池中选择设备进行更新，则在很少有其他设备连接时（例如夜班或异常时区）连接的　　设备可能会在聚合输出。
- 如果选定的设备在输出计算速度更快时更有可能保留其输出，那么： a) 来自具有更快处理器的设备的输出可能过多，这些设备可能是较新的设备，因此与社会经济地位相关； b) 数据较少的设备可能被过度代表，这些设备可能代表较少使用产品的用户。
- 如果数据节点具有不同数量的数据，则联邦学习可能会权衡较高的产品或生成数据功能的重度用户群体的贡献。
- 如果更新频率取决于延迟，那么设备或网络较慢的某些地理区域和人群可能会被低估。
- 如果 *潜在用户* 的人群由于社会经济原因不拥有设备，则他们在训练数据集中的代表性可能不足，随后在模型训练和评估中的代表性也不足（或未）。
- 联合训练期间所选设备之间模型损失的未加权聚合可能会降低某些设备上的模型性能 [302]。

It has been observed that biases in the data-generating process can also drive unfairness in the resulting models learned from this data (see e.g. [170, 394]). For example, suppose training data is based on user interactions with a product which has failed to incorporate inclusive design principles. Then, the user interactions with the product might not express user intents (cf. [403], for example) but rather might express coping strategies around uninclusive product designs (and hence might require a fundamental fix to the product interaction model). Learning from such interactions might then ignore or perpetuate poor experiences for some groups of product users in ways which can be difficult to detect while maintaining privacy in a federated setting. This risk is shared by all machine learning scenarios where training data is derived from user interaction, but is of particular note in the federated setting when data is collected from apps on individual devices.

据观察，数据生成过程中的偏差也会导致从这些数据中学习的结果模型的不公平性（参见例如 [170, 394]）。例如，假设训练数据基于用户与未能纳入包容性设计原则的产品的交互。然后，用户与产品的交互可能不会表达用户意图（例如，参见 [403]），而是可能表达应对策略围绕非包容性产品设计（因此可能需要对产品交互模型进行根本性修复）。从此类交互中学习可能会忽略或延续某些产品用户组的不良体验，这些方式在联合设置中维护隐私的同时可能难以检测。所有机器学习场景都存在这种风险，其中训练数据来自用户交互，但在联合设置中，当数据是从单个设备上的应用程序收集时，这一点尤其值得注意。

Investigating the degree to which biases in the data-generated process can be identified or mitigated is a crucial problem for both federated learning research and ML research more broadly. Similarly, while limited prior research has demonstrated methods to identify and correct bias in already collected data in the federated setting (e.g. via adversarial methods in [255]), further research in this area is needed. Finally, methods for applying post-hoc fairness corrections to models learned from potentially biased training data are also a valuable direction for future work.

调查数据生成过程中的偏见可以被识别或减轻的程度对于联邦学习研究和更广泛的机器学习研究来说都是一个关键问题。同样，虽然有限的先前研究已经证明了在联邦设置中识别和纠正已收集数据中的偏差的方法（例如，通过 [255] 中的对抗性方法），但需要在该领域进一步研究。最后，将事后公平校正应用于从潜在有偏见的训练数据中学习的模型的方法也是未来工作的一个有价值的方向。

6.2 Fairness Without Access to Sensitive Attributes

Having explicit access to demographic information (race, gender, etc) is critical to many existing fairness criteria, including those discussed in Section 6.1. However, the contexts in which federated learning are often deployed also give rise to considerations of fairness when individual sensitive attributes are *not* available. For example, this can occur when developing personalized language models or developing fair medical image classifiers without knowing any additional demographic information about individuals. Even more fundamentally, the assumed one-to-one relationship between individuals and devices often breaks down, especially in non-Western contexts [403]. Both measuring and correcting unfairness in contexts where there is no data regarding sensitive group membership is a key area for federated learning researchers to address.

明确访问人口统计信息（种族、性别等）对于许多现有的公平标准至关重要，包括在部分 6.1 中讨论的那些。然而，当个别敏感属性*not* 可用时，联合学习经常部署的上下文也会引起公平性的考虑。例如，这可能发生在开发个性化语言模型或开发公平的医学图像分类器时，而无需了解有关个人的任何其他人口统计信息。更根本的是，假设的个人和设备之间的一对一关系经常崩溃，尤其是在非西方环境[403] 中。在没有关于敏感组成员身份的数据的情况下，衡量和纠正不公平是联邦学习研究人员需要解决的一个关键领域。

Limited existing research has examined fairness without access to sensitive attributes. For example, this has been addressed using distributionally-robust optimization (DRO) which optimizes for the worst-case outcome across all individuals during training [225], and via multicalibration, which calibrates for fairness across subsets of the training data [232]. Even these existing approaches have not been applied in the federated setting, raising opportunities for future empirical work. The challenge of how to make these approaches work for large-scale, high-dimensional data typical to federated settings is also an open problem, as DRO and multicalibration both pose challenges of scaling with large n and p. Finally, the development of additional theoretical approaches to defining fairness without respect to "sensitive attributes" is a critical area for further research.

有限的现有研究在没有访问敏感属性的情况下检查了公平性。例如，这已经使用分布鲁棒优化 (DRO)

解决了，该优化在训练过程中针对所有个体的最坏情况结果进行优化 [225]，并通过多重校准来解决这个问题，它校准训练数据子集之间的公平性 [232]。即使这些现有的方法也没有应用于联邦环境，这为未来的实证工作提供了机会。如何使这些方法适用于联邦设置典型的大规模、高维数据的挑战也是一个悬而未决的问题，因为 DRO 和多重校准都带来了大 n 和 p 缩放的挑战。最后，在不考虑“敏感属性”的情况下定义公平的其他理论方法的发展是进一步研究的关键领域。

Other ways to approach this include reframing the existing notions of fairness, which are primarily concerned with equalizing the probability of an outcome (one of which is considered "positive" and another "negative" for the affected individual). Instead, fairness without access to sensitive attributes might be reframed as *equal access to effective models*. Under this interpretation of fairness, the goal is to maximize model utility across all individuals, regardless of their (unknown) demographic identities, and regardless of the "goodness" of an individual outcome. Again, this matches the contexts in which federated learning is most commonly used, such as language modeling or medical image classification, where there is no clear notion of an outcome which is "good" for a user, and instead the aim is simply to make correct predictions for users, regardless of the outcome.

解决这个问题的其他方法包括重新定义现有的公平概念，这些概念主要涉及均衡结果的概率（对于受影响的个人，其中一个被认为是“积极的”，另一个被认为是“消极的”）。相反，没有访问敏感属性的公平可能被重新定义为 *对有效模型的平等访问*。在对公平的这种解释下，目标是最大化所有个人的模型效用，无论他们的（未知）人口特征如何，也无论个人结果的“好”如何。同样，这与联邦学习最常用的上下文相匹配，例如语言建模或医学图像分类，其中没有明确的结果对用户“好”的概念，而目标只是为用户做出正确的预测，无论结果如何。

Existing federated learning research suggests possible ways to meet such an interpretation of fairness, e.g. via personalization [250, 472]. A similar conception of fairness, as "a more fair distribution of the model performance across devices", is employed in [302].

现有的联邦学习研究提出了满足这种公平解释的可能方法，例如通过个性化[250, 472]。在[302] 中采用了类似的公平概念，作为“跨设备模型性能的更公平分布”。

The application of attribute-independent methods explicitly to ensure equitable model performance is an open opportunity for future federated learning research, and is particularly important as federated learning reaches maturity and sees increasing deployment with real populations of users without knowledge of their sensitive identities.

明确应用与属性无关的方法来确保公平的模型性能是未来联邦学习研究的一个开放机会，并且随着联邦学习达到成熟并且看到越来越多的用户在不了解其敏感身份的情况下进行部署，这一点尤为重要。

6.3 Fairness, Privacy, and Robustness

Fairness and data privacy seem to be complementary ethical concepts: in many of the real-world contexts where privacy protection is desired, fairness is also desired. Often this is due to the sensitivity of the underlying data. Because federated learning is most likely to be deployed in contexts of sensitive data where both privacy and fairness are desirable, it is important that FL research examines how FL might be able to address existing concerns about fairness in machine learning, and whether FL raises new fairness-related issues. 公平和数据隐私似乎是互补的道德概念：在许多需要隐私保护的现实世界中，也需要公平。这通常是由于基础数据的敏感性。由于联邦学习最有可能部署在需要隐私和公平性的敏感数据环境中，因此 FL 研究必须检查 FL 如何解决现有的机器学习公平性问题，以及 FL 是否提高了新的公平性-相关问题。

In some ways, however, the ideal of fairness seems to be in tension with the notions of privacy for which FL seeks to provide guarantees: differentially-private learning typically seeks to obscure individually-identifying characteristics, while fairness often requires knowing individuals' membership in sensitive groups in order to measure or ensure fair predictions are being made. While the trade-off between differential privacy and fairness has been investigated in the non-federated setting [246, 145], there has been little work on how (or whether) FL may be able to uniquely address concerns about fairness.

然而，在某些方面，公平的理想似乎与 FL 试图为其提供保证的隐私概念相冲突：差异化私人学习通常试图掩盖个人识别特征，而公平通常需要了解个人在其中的成员身份。敏感群体，以衡量或确保做出公平的预测。虽然已经在非联邦环境[246, 145] 中研究了差分隐私和公平之间的权衡，但关于 FL 如何（或是否）能够独特地解决公平问题的研究很少。

Recent evidence suggesting that differentially-private learning can have disparate impact on sensitive subgroups [43, 145, 246, 283] provides further motivation to investigate whether FL may be able to address such concerns. A potential solution to relax the tension between privacy (which aims to protect the model from being too dependent on individuals) and fairness (which encourages the model to perform well on under-represented classes) may be the application of techniques such as personalization (discussed in Section 3.3) and "hybrid differential privacy," where some users donate data with lesser privacy guarantees [40].

最近的证据表明差异私人学习可能对敏感的子群体产生不同的影响 [43, 145, 246, 283] 提供了进一步的动机来调查 FL 是否能够解决这些问题。缓解隐私（旨在保护模型不过度依赖个人）和公平（鼓励模型在代表性不足的类别上表现良好）之间紧张关系的潜在解决方案可能是应用个性化等技术（讨论在 Section 3.3) 和“混合差分隐私”中，一些用户捐赠的数据具有较低的隐私保证 [40]。

Furthermore, current differentially-private optimization schemes are applied without respect to sensitive attributes – from this perspective, it might be expected that empirical studies have shown evidence that differentially-private optimization impacts minority subgroups the most [43]. Modifications to differentially-private optimization algorithms which explicitly seek to preserve performance on minority subgroups, e.g. by adapting the noise and clipping mechanisms to account for the representation of groups within the data, would also likely do a great deal to limit potential disparate impacts of differentially-private modeling on minority subgroups in federated models trained with differential privacy. However, implementing such adaptive differentially-private mechanisms in a way that provides some form of privacy guarantee presents both algorithmic and theoretical challenges which need to be addressed by future work.

此外，当前的差异私有优化方案的应用不考虑敏感属性——从这个角度来看，可以预期的是，实证研究表明，差异私有优化对少数群体的影响最大。对差异私有优化算法的修改，这些算法明确寻求保持少数子群的性能，例如通过调整噪声和剪裁机制来解释数据中群体的表示，也可能会在很大程度上限制差分隐私建模对使用差分隐私训练的联合模型中的少数群体的潜在不同影响。然而，以提供某种形式的隐私保证的方式实现这种自适应差分隐私机制提出了算法和理论方面的挑战，需要在未来的工作中加以解决。

Further research is also needed to determine the extent to which the issues above arise in the federated setting. Furthermore, as noted in Section 6.2, the challenge of evaluating the impact of differential privacy on model fairness becomes particularly difficult when sensitive attributes are not available, as it is unclear how to identify subgroups for which a model is behaving badly and to quantify the "price" of differential privacy – investigating and addressing these challenges is an open problem for future work.

还需要进一步研究以确定上述问题在联邦环境中出现的程度。此外，如 Section 6.2 中所述，当敏感属性不可用时，评估差分隐私对模型公平性的影响的挑战变得特别困难，因为尚不清楚如何识别子组模型表现不佳并量化差分隐私的“价格”——调查和解决这些挑战是未来工作的一个悬而未决的问题。

More broadly, one could more generally examine the relation between privacy, fairness, and *robustness* (see Section 5). Many previous works on machine learning, including federated learning, typically focus on isolated aspects of robustness (either against poisoning, or against evasion), privacy, or fairness. An important open challenge is to develop a joint understanding of federated learning systems that are robust, private, and fair. Such an integrated approach can provide opportunities to benefit from disparate but complementary mechanisms. Differential privacy mechanisms can be used to both mitigate data inference attacks, and provide a foundation for robustness against data poisoning. On the other hand, such an integrated approach also reveals new vulnerabilities. For example, recent work has revealed a trade-off between privacy and robustness against adversarial examples [429].

%这是在稳健性部分，似乎不属于。这似乎更合适，并且直接支持段落主题。更广泛地说，人们可以更广泛地研究隐私、公平和 *robustness* 之间的关系（参见部分 5）。许多以前关于机器学习的工作，包括联邦学习，通常侧重于稳健性（防止中毒或逃避）、隐私或公平的孤立方面。一个重要的公开挑战是对稳健、私密和公平的联邦学习系统形成共同理解。这种综合方法可以提供从不同但互补的机制中受益的机会。差分隐私机制可用于减轻数据推理攻击，并为抵御数据中毒提供鲁棒性基础。另一方面，这种集成方法也暴露了新的漏洞。例如，最近的工作揭示了针对对抗性示例的隐私和鲁棒性之间的权衡 [429]。

Finally, privacy and fairness naturally meet in the context of learning data representations that are independent of some sensitive attributes while preserving utility for a task of interest. Indeed, this objective can be motivated both in terms of privacy: to transform data so as to hide private attributes, and fairness: as a way to make models trained on such representations fair with respect to the attributes. In the centralized setting, one way to learn such representations is through adversarial training techniques, which have been applied to image and speech data [255, 186, 327, 65, 431]. In the federated learning scenario, clients could apply the transformation locally to their data in order to enforce or improve privacy and/or fairness guarantees for the FL process. However, learning this transformation in a federated fashion (potentially under privacy and/or fairness constraints) is itself an open question.

最后，隐私和公平自然会在学习数据表示的背景下相遇，这些表示独立于某些敏感属性，同时保留对感兴趣任务的效用。事实上，这个目标可以在隐私方面得到激励：转换数据以隐藏私人属性，以及公平性：作为一种使基于此类表示训练的模型在属性方面公平的方式。在集中设置中，学习此类表示的一种方法是通过对抗训练技术，该技术已应用于图像和语音数据 [255, 186, 327, 65, 431]。在联合学习场景中，客户可以在本地将转换应用到他们的数据，以加强或改进 FL 过程的隐私和/或公平性保证。然而，以联合方式（可能在隐私和/或公平约束下）学习这种转换本身就是一个悬而未决的问题。

6.4 Leveraging Federation to Improve Model Diversity

利用联邦提高模型多样性

Federated learning presents the opportunity to integrate, through distributed training, datasets which may have previously been impractical or even illegal to combine in a single location. For example, the Health Insurance Portability and Accountability Act (HIPAA) and the Family Educational Rights and Privacy Act (FERPA) constrain the sharing of medical patient data and student educational data, respectively, in the United States. To date, these restrictions have

led to modeling occurring in institutional silos: for example, using electronic health records or clinical images from individual medical institutions instead of pooling data and models across institutions [91, 104]. In contexts where membership in institutional datasets is correlated with individuals' specific sensitive attributes, or their behavior and outcomes more broadly, this can lead to poor representation for users in groups underrepresented at those institutions. Importantly, this lack of representation and diversity in the training data has been shown to lead to poor performance, e.g. in genetic disease models [333] and image classification models [93].

联合学习提供了通过分布式训练集成数据集的机会，这些数据集以前在单个位置合并可能不切实际甚至非法。例如，健康保险流通与责任法案 (HIPAA) 和家庭教育权利和隐私法案 (FERPA) 分别限制了美国医疗患者数据和学生教育数据的共享。迄今为止，这些限制导致建模出现在机构孤岛中：例如，使用来自单个医疗机构的电子健康记录或临床图像，而不是跨机构汇集数据和模型 [91, 104]。在机构数据集中的成员资格与个人的特定敏感属性或更广泛的行为和结果相关的情况下，这可能导致在这些机构中代表性不足的群体中的用户代表性不佳。重要的是，训练数据中缺乏代表性和多样性已被证明会导致性能不佳，例如在遗传疾病模型 [333] 和图像分类模型 [93] 中。

Federated learning presents an opportunity to leverage uniquely diverse datasets by providing efficient decentralized training protocols along with privacy and non-identifiability guarantees for the resulting models. This means that federated learning enables training on multi-instutitional datasets in many domains where this was previously not possible. This provides a practical opportunity to leverage larger, more diverse datasets and explore the generalizability of models which were previously limited to small populations. More importantly, it provides an opportunity to improve the *fairness* of these models by combining data across boundaries which are likely to have been correlated with sensitive attributes. For instance, attendance at specific health or educational institutions may be correlated with individuals' ethnicity or socioeconomic status. As noted in Section 6.1 above, underrepresentation in training data is a proven driver of model unfairness.

联邦学习提供了一个机会，可以通过提供有效的分散训练协议以及对结果模型的隐私和不可识别性保证来利用独特多样的数据集。这意味着联邦学习可以在许多领域中对多机构数据集进行训练，而这在以前是不可能的。这为利用更大、更多样化的数据集和探索以前仅限于小群体的模型的普遍性提供了一个实用的机会。更重要的是，它提供了一个机会，通过组合可能与敏感属性相关的跨边界数据来改善这些模型的 *fairness*。例如，在特定的卫生或教育机构就读可能与个人的种族或社会经济地位相关。如上文 6.1 节所述，训练数据中的代表性不足是模型不公平的一个已证明的驱动因素。

Future federated learning research should investigate the degree to which improving diversity in a federated training setting also improves the fairness of the resulting model, and the degree to which the differential privacy mechanisms required in such settings may limit fairness and performance gains from increased diversity. This includes a need for both empirical research which applies federated learning and quantifies the interplay between diversity, fairness, privacy, and performance; along with theoretical research which provides a foundation for concepts such as diversity in the context of machine learning fairness.

未来的联邦学习研究应该调查提高联邦训练环境中的多样性在多大程度上也能提高结果模型的公平性，以及这种环境中所需的差分隐私机制可能会在多大程度上限制公平性和提高多样性带来的性能收益。这包括需要应用联邦学习并量化多样性、公平性、隐私和性能之间的相互作用的实证研究；以及为机器学习公平性背景下的多样性等概念提供基础的理论研究。

6.5 Federated Fairness: New Opportunities and Challenges

联邦公平：新的机遇和挑战

It is important to note that federated learning provides unique opportunities and challenges for fairness researchers. For example, by allowing for datasets which are distributed both by observation, but even by features, federated learning can enable modeling and research using partitioned data which may be too sensitive to share directly [215, 224]. Increased availability of datasets which can be used in a federated manner can help to improve the diversity of training data available for machine learning models, which can advance fair modeling theory and practice.
值得注意的是，联邦学习为公平研究人员提供了独特的机会和挑战。例如，通过允许通过观察和特征分布的数据集，联邦学习可以使用分区数据进行建模和研究，分区数据可能过于敏感而无法直接共享 [215, 224]。以联合方式使用的数据集的可用性增加有助于提高机器学习模型可用的训练数据的多样性，这可以促进公平建模理论和实践。

Researchers and practitioners also need to address the unique fairness-related challenges created by federated learning. For example, federated learning can introduce new sources of bias through the decision of which clients to sample based on considerations such as connection type/quality, device type, location, activity patterns, and local dataset size [81]. Future work could investigate the degree to which these various sampling constraints affect the fairness of the resulting model, and how such impacts can be mitigated within the federated framework, e.g. [302, 289, 158]. Frameworks such as *agnostic federated learning* [352] provide approaches to control for bias in the training objective. Work to improve the fairness of existing federated training algorithms will be particularly important as advances begin to approach the technical limits of other components of FL systems, such as model compression, which initially helped to broaden the diversity of candidate clients during federated training processes. There is no unique fairness criterion generally adopted in the study of fairness, and multiple criteria have been proven to be mutually incompatible. One way to deal with this question is the *online fairness* framework and algorithms of Awasthi et al. [41]. Adapting such solutions to the federated learning setting and further improving upon them will be challenging research questions in ML fairness theory and algorithms.

研究人员和从业者还需要解决联邦学习带来的与公平相关的独特挑战。例如，联邦学习可以根据连接类型/质量、设备类型、位置、活动模式和本地数据集大小等考虑因素决定对哪些客户端进行采样，从而引入新的偏差来源。未来的工作可以调查这些不同的抽样约束对结果模型公平性的影响程度，以及如何在联邦框架内减轻这种影响，例如[302, 289, 158]。 *agnostic federated learning* [352] 等框架提供了控制训练目标偏差的方法。随着进步开始接近 FL 系统其他组件（例如模型压缩）的技术限制，提高现有联合训练算法的公平性的工作将尤为重要，模型压缩最初有助于在联合训练过程中扩大候选客户的多样性。公平研究中没有普遍采用的唯一公平标准，多个标准已被证明是相互不兼容的。处理这个问题的一种方法是 Awasthi et al. [41] 的 *online fairness* 框架和算法。将此类解决方案应用于联邦学习设置并进一步改进它们将是 ML 公平理论和算法中具有挑战性的研究问题。

In the classical centralized machine learning setting, a substantial amount of advancement has been made in the past decade to train fair classifiers, such as constrained optimization, post-shifting approaches, and distributionally-robust optimization [223, 503, 225]. It is an open question whether such approaches, which have demonstrated utility for improving fairness in centralized training, could be used under the setting of federated learning (and if so, under what additional assumptions) in which data are located in a decentralized fashion and practitioners may not obtain an unbiased sample of the data that match the distribution of the population.

在经典的集中式机器学习设置中，过去十年在训练公平分类器方面取得了大量进步，例如约束优化、后移方法和分布稳健优化 [223, 503, 225]。此类方法是否可以在联邦学习的背景下使用（如果是，在哪些额外的假设下），其中数据以分散的方式和从业者可能无法获得与总体分布相匹配的无偏数据样本。

6.6 Executive Summary

In addition to inheriting the already significant challenges related to bias, fairness, and privacy in centralized machine learning, federated learning also brings a new set of distinct challenges and opportunities in these areas. The importance of these considerations will likely continue to grow as the real-world deployment of FL expands to more users, domains, and applications.

除了继承集中式机器学习中已经存在的与偏见、公平和隐私相关的重大挑战之外，联邦学习还在这些领域带来了一系列新的独特挑战和机遇。随着 FL 的实际部署扩展到更多用户、域和应用程序，这些考虑因素的重要性可能会继续增长。

- Bias in training data (Section 6.1) is a key consideration related to bias and fairness in FL models, particularly due to the additional sampling steps germane to federation (e.g., client sampling) and the transfer of some model computation to client devices.
- The lack of data regarding sensitive attributes in many FL deployments can pose challenges for measuring and ensuring fairness, and also suggests potential reframing of fairness problems in ways that do not require such data (Section 6.2).
- Since FL is often deployed in contexts which are both privacy- and fairness-sensitive, this can magnify tensions between privacy and fairness objectives in practice. Further work is needed to address the potential tension between methods which achieve privacy, fairness, and robustness in both federated and centralized learning (Section 6.3).
- Federated learning presents unique opportunities to improve the diversity of stakeholders and data incorporated into learning, which could improve both the overall quality of downstream models, as well as their fairness due to more representative datasets (Section 6.4).
- Federated learning presents fairness-related challenges not present in the centralized training regime, but also affords new solutions (Section 6.5).

- 训练数据中的偏差（部分 6.1）是与 FL 模型中的偏差和公平性相关的关键考虑因素，特别是由于与联邦密切相关的额外采样步骤（例如，客户端采样) 以及将一些模型计算传输到客户端设备。
- 在许多 FL 部署中缺乏有关敏感属性的数据可能会给衡量和确保公平性带来挑战，并且还建议以不需要此类数据的方式重新构建公平性问题（部分 6.2）。
- 由于 FL 通常部署在对隐私和公平敏感的环境中，这可能会放大隐私和公平目标之间的紧张关系。需要进一步的工作来解决在联邦学习和集中学习中实现隐私、公平和鲁棒性的方法之间的潜在紧张关系（第 6.3）。
- 联合学习提供了独特的机会来改善利益相关者和纳入学习的数据的多样性，这可以提高下游模型的整体质量，以及由于更具代表性的数据集而导致的公平性（部分 6.4）。

- 联合学习提出了集中式培训制度中不存在的与公平相关的挑战，但也提供了新的解决方案（部分 6.5）。

7 Addressing System Challenges

As we will see in this section, the challenges in building systems for federated learning can be split fairly cleanly into the two separate settings of cross-device and cross-silo federated learning (see Sections 1.1 and 2.2). We start with a brief discussion of the difficulties inherent to any large scale deployment of software on end-user devices (although exacerbated by the complexity of a federated learning stack); we then focus on key challenges specific to the cross-device learning—bias, tuning, and efficient device-side execution of ML workflows—before concluding with a brief treatment of the cross-silo setting.

正如我们将在本节中看到的，构建联邦学习系统的挑战可以相当清晰地分为跨设备和跨孤岛联邦学习的两个独立设置（参见Sections 1.1 and 2.2)。我们首先简要讨论在最终用户设备上大规模部署软件所固有的困难（尽管联邦学习堆栈的复杂性加剧了这种困难）；然后，我们专注于跨设备学习特有的关键挑战——机器学习工作流程的偏差, 调整和有效的设备端执行——最后简要介绍跨竖井设置。

7.1 Platform Development and Deployment Challenges

Running computations on end-user devices is considerably different from the data center setting:

- Due to the heterogeneity of the fleet (devices may differ in hardware, software, connectivity, performance and persisted state) the space of potential problems and edge cases is vast and cannot typically be covered in sufficient detail with automated testing.
- Monitoring and debugging are harder because telemetry is limited, delayed, and there is no physical access to devices for interactive troubleshooting.
- Running computations should not affect device performance or stability, i.e. should be invisible to users.

Code Deployment Installing, updating and running software on end user devices may involve not only extensive manual and automated testing, but a gradual and reversible rollout (for example, through guarding new functionality with server-controlled feature flags) while monitoring key performance metrics in a/b experiments such as crash rates, memory use, and application-dependent indicators such as latencies and engagement metrics. Such rollouts can take weeks or months depending on the percolation rate of updates (particularly challenging for devices with spotty connectivity) and the complexity of the upgrade (e.g. protocol changes). Hence, the install base at any given time will involve various releases. While this problem is not specific to federated learning, it has greater impact here due to the inherent collaborative nature of federated computations: devices constantly communicate with servers and indirectly with other devices to exchange models and parameter updates. Thus, compatibility concerns abound and must be addressed through stable exchange formats or, where not possible, detected upfront with extensive testing infrastructure. We will revisit this problem in Section 7.4.

Monitoring and Debugging Another significant complication is the limited ability to monitor devices and interactively debug problems. While telemetry from end user devices is necessary to detect problems, privacy concerns severely restrict what can be logged, who can access such logs, and how long they are retained. Once a regression is detected, drilling down into the root cause can be very cumbersome due to the lack of detailed context, the vast

problem space (a cross product of software versions, hardware, models, and device state), and very limited ability for interactive debugging short of successfully reproducing the problem in a controlled environment.

These challenges are exacerbated in the federated learning setting where a) raw input data on devices cannot be accessed, and b) contributions from individual devices are by design anonymous, ephemeral, and exposed only in aggregate. These properties preserve privacy, but also may make it hard or impossible to investigate problems with traditional approaches —by looking for correlations with hardware or software version, or testing hypotheses that require access to raw data. Reproducing a problem in a controlled setting is often difficult due to the gap between such an environment and reality: hundreds of heterogeneous embedded stateful devices with non-iid data.

Interestingly, federated technologies themselves can help to mitigate this problem—for instance, the use of federated analytics [382] to collect logs in a privacy preserving manner, or training generative models of the system behavior or raw data for sampling during debugging (see sections 3.4.3, 5.2, and [31]). Keeping a federated learning system up and running thus requires investing into upfront detection of problems through a) extensive automated, continuous test coverage of all software layers through both unit and integration tests; b) feature flags and a/b rollouts; and c) continuous monitoring of performance indicators for regressions. That poses a significant investment that may come at too high a cost for smaller entities who would benefit greatly from shared and tested infrastructure for federated learning.

7.2 System Induced Bias

Deployment, monitoring and debugging may not concern users of a federated learning platform, e.g. model authors or data analysts. For them, the key differences between data center and cross-device settings fall largely into the following two categories:

部署, 监控和调试可能与联邦学习平台的用户无关，例如模型作者或数据分析师。对他们来说，数据中心和跨设备设置之间的主要区别主要分为以下两类：

1. **Availability of devices** for computations is not a given, but varies over time and across devices. Connections are initiated by devices and subject to interruptions due to changes in device state, operating system quotas, and network connectivity. Hence, in iterative processes like federated learning, the loop body is run on a small subset of all devices only, and the system must tolerate a certain failure rate among those devices.

2. **Capabilities of devices** (network bandwidth and latency, compute performance, memory) vary, and are typically much lower than those of compute nodes in the data center, though the number of nodes is typically higher. The amount and type of data across devices may lead to variations in execution profile, e.g. more and larger examples lead to increased resource use and processing time.

In the following sections we discuss how these variations might introduce bias, referring to it as system induced bias to differentiate it from platform-independent bias in the raw data (such as ownership or usage patterns differing across demographics)—for the latter, see Section 6.1.

在以下部分中，我们将讨论这些变化如何引入偏差，将其称为系统诱发的偏差，以将其与原始数据中独立于平台的偏差（例如不同人口统计数据的所有权或使用模式不同）区分开来——对于后者，参见 Section 6.1。

1. **设备的可用性** 用于计算不是给定的，而是随时间和跨设备而变化。连接由设备发起，并且会因设备状态, 操作系统配额和网络连接的变化而中断。因此，在像联邦学习这样的迭代过程中，循环体仅在所有设备的一小部分上运行，并且系统必须容忍这些设备之间的一定故障率。

2. **设备的能力**（网络带宽和延迟, 计算性能, 内存）各不相同，通常比数据中心的计算节点低得多，但节点数量通常更高。跨设备的数据量和类型可能会导致执行配置文件的变化，例如更多和更大的示例导致资源使用和处理时间增加。

7.2.1 Device Availability Profiles

At the core of cross-device federated learning is the principle that devices only connect to the server and run computations when various constraints are met:

- **Hard constraints**, which might include requiring that the device is turned on, has network connectivity to the server, and is allowed to run a computation by the operating system.
- **Soft constraints**, which might include the conditions on device state chosen to ensure that federated learning does not incur charges or affect usability. For the common case of mobile phones [81, 26], requirements may include idleness, charging and/or above a certain battery level, being connected to an unmetered network, and that no other federated learning tasks are running at the same time.

设备可用性配置文件

跨设备联邦学习的核心是设备仅在满足各种约束时才连接到服务器并运行计算的原则：

- **硬约束**，这可能包括要求设备打开，与服务器具有网络连接，并允许操作系统运行计算。
- **软约束**，其中可能包括选择的设备状态条件，以确保联邦学习不会产生费用或影响可用性。对于手机 [81, 26] 的常见情况，要求可能包括空闲, 充电和/或高于特定电池电量, 连接到未计量的网络，并且没有其他联邦学习任务同时运行.

Taken together, these constraints induce an unknown, time-varying and device-specific function $A_i(t)$ for a device i, and a fleet-wide *availability profile* $A(t) = \sum_i A_i(t)$. Round completion rates and server traffic patterns [81, 491] suggest that availability profiles for mobile phones are clustered into periodic functions with a period of 1 day, varying across devices in phase, shape and amplitude through factors such as demographics, geography etc. Availability for other end user devices such as laptops, tablets, or stationary devices such as smart speakers, displays and cameras, will differ, but the challenges discussed in the following sections apply there as well, albeit to a possibly lesser extent.

综上所述，这些约束为设备 i 引入了一个未知的, 时变的和特定于设备的函数 $A_i(t)$，以及一个车队范围的 *availability profile* $A(t) = \sum_i A_i(t)$。回合完成率和服务器流量模式 [81, 491] 表明手机的可用性配置文件以 1 天为周期聚集成周期性函数，通过人口统计, 地理等因素在相位, 形状和幅度上因设备而异. 其他最终用户设备（如笔记本电脑, 平板电脑）或固定设备（如智能扬声器, 显示器和相机）的可用性将有所不同，但以下部分中讨论的挑战也适用于这些设备，尽管程度可能较小。

7.2.2 Examples of System Induced Bias

Sources of bias will depend on the specific way in which devices are selected to participate in training, and how the system influences which devices end up contributing to the final aggregated model update. Thus, it is useful to discuss these issues in light of a simplified but representative system design. In an iterative federated learning algorithm, such as Federated Averaging (Section 1.1.2, [337]), rounds are run consecutively on sets of at least M devices. To accommodate a fraction d of devices not contributing due to changes in device conditions, time-outs, or slowness (server-side aborts to avoid slow-downs by stragglers), an over-allocation scheme is used where

7.2.3 系统诱导偏差的例子

偏差来源将取决于选择设备参与训练的具体方式，以及系统如何影响哪些设备最终对最终聚合模型更新做出贡献。因此，根据简化但具有代表性的系统设计来讨论这些问题是有用的。在迭代联邦学习算法中，例如联邦平均 (Section 1.1.2, [337])，轮次在至少 M 的设备集上连续运行。为了适应由于设备条件变化, 超时或缓慢（服务器端中止以避免落后者造成的缓慢）而没有贡献的一小部分设备，在以下情况下使用过度分配方案

1. Rounds are started when at least $M' = \frac{M}{1-d}$ devices are available.
2. Rounds are closed as
   (a) *Aborted* when more than $M' - M$ devices have disconnected, or
   (b) *Successful* when at least M devices have reported. One possible design choice is to stop after exactly M devices; another possibility would be to keep waiting for stragglers (possibly up to some maximum time).

1. 回合在至少 $M' = \frac{M}{1-d}$ 设备可用时开始。
2. 回合关闭为
   (a) *Aborted* 当超过 $M' - M$ 个设备断开连接时，或
   (b) *Successful* 当至少有 M 设备报告时。一种可能的设计选择是在恰好 M 个设备之后停止；另一种可能性是继续等待落后者（可能达到某个最长时间）。

This sequence, when combined with variable availability profiles, may introduce various forms of bias:

此序列与可变可用性配置文件结合使用时，可能会引入各种形式的偏差：

1. Selection Bias - whether a device is included in a round at time t depends on both
   (a) Its availability profile $A_i(t)$
   (b) The number of simultaneously connected devices: $< M'$ and a round cannot be started; $\gg M'$ and the probability of a single device being included becomes very small. In effect, devices active only at either fleet-wide availability peaks or troughs may be under-represented.
2. Survival Bias

(a) Since a server might choose to close a round at any point after the first M devices have reported, contributions are biased towards devices with better network connections, faster processors, lower CPU load, and less data to process.

(b) Devices drop out of rounds when they are interrupted by the operating system, which may happen due to changes in device conditions as described by $A_i(t)$, or due to e.g. excessive memory use.

1. 选择偏差 - 在时间 t 的回合中是否包含设备取决于两者

   (a) 其可用性配置文件 $A_i(t)$

   (b) 同时连接的设备数：$< M'$，不能开始一轮；$\gg M'$ 并且包含单个设备的概率变得非常小。实际上，仅在整个车队的可用性高峰或低谷时处于活动状态的设备可能代表不足。

2. 生存偏差

   (a) 由于服务器可能会在第一个 M 设备报告后的任何时候选择关闭一轮，因此贡献偏向于具有更好网络连接, 更快处理器, 更低 CPU 负载和更少数据处理的设备。

   (b) 设备在被操作系统中断时退出轮次，这可能是由于 $A_i(t)$ 所描述的设备条件的变化而发生的，或者由于例如过多的内存使用。

As can be seen, the probability of a device contributing to a round of federated learning is a complex function of both internal (e.g. device specific) and external (fleet dynamic) factors. When this probability is correlated with statistics of the data distribution, aggregate results may be biased. For instance, language models may over-represent demographics that have high quality internet connections or high end devices; and ranking models may not incorporate enough contributions from high engagement users who produce a lot of training data and hence longer training times.

可以看出，设备参与一轮联邦学习的概率是内部（例如特定于设备）和外部（车队动态）因素的复杂函数。当此概率与数据分布的统计数据相关时，聚合结果可能会有偏差。例如，语言模型可能会过度代表具有高质量互联网连接或高端设备的人口统计数据；和排名模型可能没有包含来自产生大量训练数据的高参与度用户的足够贡献，因此训练时间更长。

Thus, designing systems that explicitly take such factors into account and integrate algorithms designed to both quantify and mitigate these effects are a fundamentally important research direction. 因此，设计明确考虑这些因素并集成旨在量化和减轻这些影响的算法的系统是一个根本重要的研究方向。

7.2.4 Open Challenges in Quantifying and Mitigating System Induced Bias

While the potential for bias in federated learning has been addressed in the literature (Section 6, [81, 302, 171]), a systematic study that qualifies and quantifies bias in realistic settings and its sources is a direction for future research. Conducting the necessary work may be hampered by both access to the necessary resources, and the difficulty in quantifying bias in a final statistical estimate due to the inherent lack of ground truth value.

量化和减轻系统引起的偏差的开放挑战

虽然联邦学习中潜在的偏差已经在文献中得到解决（Section 6, [81, 302, 171]），但一项在现实环境中限定和量化偏差及其来源的系统研究是一个未来研究的方向。进行必要的工作可能会受到获取必要资源的阻

碍，以及由于固有的真实值缺乏而难以量化最终统计估计中的偏差。

We want to encourage further research to study how bias can be quantified and subsequently mitigated. A useful proxy metric for bias is to study the expected rate of contribution of a device to federated learning. In an unbiased system, this rate would be identical for every device; if it is not, the non-uniformity may provide a measure of bias. Studying the root causes for this non-uniformity may then provide important hints for how to mitigate bias, for example:

我们希望鼓励进一步的研究来研究如何量化并随后减轻偏差。一个有用的偏差代理指标是研究设备对联邦学习的预期贡献率。在一个无偏系统中，每个设备的这个比率都是相同的；如果不是，则不均匀性可能会提供偏差的度量。研究这种不均匀性的根本原因可能会为如何减轻偏差提供重要的提示，例如：

- When there is a strong correlation between devices finishing a round, and the number of examples they process or model size, possible fixes may include early stopping, or decreasing the model size.
- If the expected rate of contribution depends on factors outside our control, such as device model, network connectivity, location etc., one can view these factors as defining strata and applying *post-stratification* [312], that is, correcting for bias by scaling up or down contributions from devices depending on their stratum. It may also be possible to apply *stratified sampling* - e.g. change scheduling, or server selection policies, to affect the probability of including devices in a round as a function of their stratum.
- A very general, root-cause-agnostic mitigation could base the weight of a contribution solely on a device's past contribution profile (e.g. the number of rounds started or completed thus far). As a special case, consider *sampling without replacement* which could be implemented at the system level (stop connecting after one successful contribution) or at the model level (weight all but the first contribution with 0). This approach might not be sufficient when a population is large enough for most devices to contribute only infrequently (mostly one or zero times); in such cases, clustering devices based on some similarity metric and using cluster membership as stratum could help.
- Alternatives to the synchronous, round based execution described in the previous section may also help to mitigate bias. In particular, certain types of analytics may benefit from softening or eliminating the competition between devices for inclusion, by running rounds for long times with very large numbers of participants and without applying time-outs to stragglers. Such a method may not be applicable to algorithms where the iterative aspect (running many individual, chained rounds) is important.

- 当完成一轮的设备与它们处理的示例数量或模型大小之间存在很强的相关性时，可能的解决方法可能包括提前停止或减小模型大小。
- 如果预期贡献率取决于我们无法控制的因素，例如设备型号, 网络连接, 位置等，则可以将这些因素视为定义层并应用 *post-stratification* [312]，也就是说，通过根据设备的层次放大或缩小设备的贡献来纠正偏差。也可以应用 *分层抽样* - 例如更改调度或服务器选择策略，以影响在一轮中包含设备的概率作为其层的函数。
- 一个非常普遍的, 与根本原因无关的缓解措施可以仅基于设备过去的贡献配置文件（例如，迄今为止开始或完成的轮数）来确定贡献的权重。作为一种特殊情况，考虑 *sampling without replacement* 可以在系统级别（在一个成功贡献后停止连接）或在模型级别（除第一个贡献之外的所有贡献都用 0 加权）实

施。当人口足够大以至于大多数设备很少（主要是一次或零次）做出贡献时，这种方法可能是不够的；在这种情况下，基于某些相似性度量并使用集群成员作为层的集群设备可能会有所帮助。

- 上一节中描述的基于轮次的同步执行的替代方案也可能有助于减轻偏差。特别是，某些类型的分析可能会受益于缓和或消除设备之间的竞争，通过在非常多的参与者中长时间运行而不对落后者应用超时。这种方法可能不适用于迭代方面（运行许多单独的链轮）很重要的算法。

The biggest obstacle to enabling such research is access to a representative fleet of end user devices, or a detailed description (e.g. in the form of a statistical model of a realistic distribution over $A_i(t)$ functions) of a fleet that can be used in simulations. Here, maintainers of FL production stacks are uniquely positioned to provide such statistics or models to academic partners in a privacy preserving fashion; a further promising direction is the recent introduction of the Flower framework [66] for federated learning research.

实现此类研究的最大障碍是访问具有代表性的最终用户设备组，或详细描述（例如，以 $A_i(t)$ 函数上的实际分布的统计模型的形式）可以是用于模拟。在这里，FL 生产堆栈的维护者处于独特的地位，可以以保护隐私的方式向学术合作伙伴提供此类统计数据或模型；另一个有希望的方向是最近为联邦学习研究引入的 Flower 框架 [66]。

7.3 System Parameter Tuning

Practical federated learning is a form of multi-objective optimization: while the first order goal is maximizing model quality metrics such as loss or accuracy, other important considerations are

实用的联邦学习是多目标优化的一种形式：虽然一阶目标是最大化模型质量指标，例如损失或准确性，但其他重要的考虑因素是

- Convergence speed
- Throughput (e.g. number of rounds, amount of data, or number of devices)
- Model fairness, privacy and robustness (see section 6.3)
- Resource use on server and clients

- 收敛速度
- 吞吐量（例如轮数, 数据量或设备数）
- 模型公平性, 隐私性和鲁棒性（参见部分 6.3）
- 服务器和客户端上的资源使用

These goals may be in tension. For instance, maximizing round throughput may introduce bias or hurt accuracy by preferring performant devices with little or no data. Maximizing for low training loss by increasing model complexity will put devices with less memory, many or large examples, or slow CPUs at a disadvantage. Bias or fairness induced in such a way during training may be hard to detect in the evaluation phase since it typically uses the same platform and hence is subject to similar biases.

这些目标可能处于紧张状态。例如，通过首选数据很少或没有数据的高性能设备，最大化轮次吞吐量可能会引入偏差或损害准确性。通过增加模型复杂性来最大限度地降低训练损失将使内存较少, 示例多或大或 CPU 速度慢的设备处于劣势。在训练期间以这种方式引起的偏差或公平性可能很难在评估阶段检测到，因为它通常使用相同的平台，因此受到类似偏差的影响。

Various controls affect the above listed indicators. Some are familiar from the datacenter setting, in particular model specific settings and learning algorithm hyperparameters. Others are specific to federated learning:

各种控件会影响上面列出的指标。有些人熟悉数据中心设置，特别是模型特定设置和学习算法超参数。其他特定于联邦学习：

- **Clients per round**: The minimum number of devices required to complete a round, M, and the number of devices required to start a round, M'.
- **Server-side scheduling**: In all but the simplest cases, a federated learning system will operate on more than one model at a time: to support multiple tenants; to train models on the same data for different use cases; to support experimentation and architecture or hyper-parameter grid search; and to run training and evaluation workloads concurrently. The server needs to decide which task to serve to incoming devices, an instance of a scheduling problem: assigning work (training or evaluation tasks) to resources (devices). Accordingly, the usual challenges arise: ideal resource assignment should be fair, avoid starvation, minimize wait times, and support relative priorities all at once.
- **Device-side scheduling**: As described in Section 7.2, various constraints govern when a device can connect to the server and execute work. Within these constraints, various scheduling choices can be made. One extreme is to connect to the server and run computations as often as possible, leading to high load and resource use on both server and devices. Another choice are fixed intervals, but they need to be adjusted to reflect external factors such as number of devices overall and per round. The federated learning system developed at Google aims to strike a balance with a flow control mechanism called *pace steering* [81] whereby the server instructs devices when to return. Such a dynamic system enables temporal load balancing for large populations as well as "focusing" connection attempts to specific points in time to reach the threshold M'. Developing such a mechanism is difficult due to stochastic and dynamic nature of device availability, the lack of a predictive model of population behavior, and feedback loops.

- **每轮的客户端数**：完成一轮所需的最少设备数 M，以及开始一轮所需的设备数 M'。
- **服务器端调度**：除了最简单的情况外，联邦学习系统一次将在多个模型上运行：支持多个租户；针对不同用例在相同数据上训练模型；支持实验和架构或超参数网格搜索；并同时运行训练和评估工作负载。服务器需要决定为传入设备提供哪个任务，这是调度问题的一个实例：将工作（训练或评估任务）分配给资源（设备）。因此，常见的挑战出现了：理想的资源分配应该是公平的，避免饥饿，最小化等待时间，并同时支持相对优先级。
- **设备端调度**：如 Section 7.2 中所述，各种约束控制设备何时可以连接到服务器并执行工作。在这些约束内，可以做出各种调度选择。一种极端是连接到服务器并尽可能频繁地运行计算，从而导致服务器和设备上的高负载和资源使用。另一种选择是固定间隔，但需要对其进行调整以反映外部因素，例如整体和每轮的设备数量。谷歌开发的联邦学习系统旨在与称为 *pace Steering* [81] 的流控制机制取得平

衡，服务器指示设备何时返回。这样的动态系统能够实现大量人口的时间负载平衡，以及将连接尝试“集中”到特定时间点以达到阈值 M'。由于设备可用性的随机性和动态性, 缺乏人口行为的预测模型和反馈循环，开发这种机制很困难。

Defining reasonable composite objective functions, and designing algorithms to automatically tune these settings, has not been explored yet in the context of federated learning systems and hence remains a topic of future research.

尚未在联邦学习系统的背景下探索定义合理的复合目标函数并设计算法以自动调整这些设置，因此仍然是未来研究的主题。

7.4 On-Device Runtime

While numerous frameworks exist for data center training, the options for training models on resource constrained devices are fairly limited. Machine Learning models and training procedures are typically authored in a high level language such as Python. For federated learning, this description encompasses device and server computations that are executed on the target platform and exchange data over a network connection, necessitating

虽然数据中心训练有许多框架，但在资源受限设备上训练模型的选项相当有限。机器学习模型和训练过程通常是用 Python 等高级语言编写的。对于联邦学习，此描述包含在目标平台上执行并通过网络连接交换数据的设备和服务器计算，因此需要

- A means of serializing and dynamically transmitting local pieces of the total computation (e.g., the server-side update to the model, or the local client training procedure).
- A means to interpret or execute such a computation on the target platform (server or device).
- A stable network protocol for data exchange between participating devices and servers.
- 一种序列化和动态传输总计算的本地部分的方法（例如，服务器端对模型的更新，或本地客户端训练过程）。
- A 表示在目标平台（服务器或设备）上解释或执行此类计算。
- 用于参与设备和服务器之间数据交换的稳定网络协议。

One extreme form of a representation is the original high-level description, e.g. a Python TensorFlow program [2]. This would require a Python interpreter with TensorFlow backend, which may not be a feasible choice for end-user devices due to resource constraints (binary size, memory use), performance limitations, or security concerns.

表示的一种极端形式是原始的高级描述，例如一个 Python TensorFlow 程序 [2]。这将需要带有 TensorFlow 后端的 Python 解释器，由于资源限制（二进制大小, 内存使用）, 性能限制或安全问题，这可能不是最终用户设备的可行选择。

Another extreme representation of a computation is machine code of the target architecture, e.g. ARM64 instructions. This requires a compiler or re-implementation of a model in a lower-level language such as C++, and deployment computations will typically be subject to the restrictions that apply to deployment of binary code (see Section 7.1), introducing prohibitive latencies for executing novel computations.

计算的另一个极端表示是目标架构的机器代码，例如ARM64 指令。这需要编译器或用 C++ 等低级语言重新实现模型，并且部署计算通常会受到适用于部署二进制代码的限制（参见 Section 7.1），为执行新计算引入了令人望而却步的延迟。

Intermediate representations that can be compiled or interpreted with a runtime on the target platform strike a balance between flexibility and efficiency. However, such runtimes are currently not widely available. For instance, Google's FL system [81] relies on TensorFlow for both server and device side execution as well as model and parameter transfer, but this choice suffers from several shortcomings:

可以在目标平台上使用运行时编译或解释的中间表示在灵活性和效率之间取得平衡。但是，此类运行时目前尚未广泛使用。例如，谷歌的 FL 系统 [81] 在服务器端和设备端执行以及模型和参数传输都依赖于 TensorFlow，但这种选择有几个缺点：

- It offers no easy path to devices for alternative front ends such as PyTorch [370], JAX [86] or CNTK [410].
- The runtime is not developed or optimized for resource constrained environments, incurring a large binary size, high memory use and comparatively low performance.
- The intermediate representation `GraphDef` used by TensorFlow is not standardized or stable, and version skew between the frontend and older on-device backends causes frequent compatibility challenges.
- 它没有为替代前端的设备提供简单的路径，例如 PyTorch [370], JAX [86] 或 CNTK [410]。
- 运行时不是为资源受限的环境开发或优化的，会导致大的二进制文件, 高内存使用和相对较低的性能。
- TensorFlow 使用的中间表示 `GraphDef` 不是标准化或稳定的，前端和旧设备后端之间的版本偏差导致频繁的兼容性挑战。

Other alternatives include more specialized runtimes that support only a subset of the frontend's capabilities, for instance training specific model types only, requiring changes and long update cycles whenever new model architectures or training algorithms are to be used. An extreme case would be a runtime that is limited and optimized to train a single type of model.

其他替代方案包括仅支持前端功能子集的更专业的运行时，例如仅训练特定模型类型，每当要使用新模型架构或训练算法时都需要更改和较长的更新周期。一种极端情况是运行时间有限且经过优化以训练单一类型的模型。

An ideal on-device runtime would have the following characteristics:

理想的设备上运行时应具有以下特征：

1. Lightweight: small binary size, or pre-installed; low memory and power profile.
2. Performant: low startup latency; high throughput, supports hardware acceleration.
3. Expressive: supports common data types and computations including backpropagation, variables, control flow, custom extensions.
4. Stable and compact format for expressing data and computations.

5. Widely available: portable open source implementation.

6. Targetable by commonly used ML frameworks / languages.

7. Ideally also supports inference, or if not, building personalized models for an inference runtime.

1. 轻量级：小二进制大小，或预装；低内存和功耗配置文件。

2. 高性能：低启动延迟；高吞吐量，支持硬件加速。

3. Expressive：支持常见的数据类型和计算，包括反向传播, 变量, 控制流, 自定义扩展。

4. 用于表达数据和计算的稳定且紧凑的格式。

5. 广泛可用：可移植的开源实现。

6. 可通过常用的 ML 框架/语言定位..

7. 理想情况下还支持推理，如果不支持，则为推理运行时构建个性化模型。

To our best knowledge no solution exists yet that satisfies these requirements, and we expect the limited ability to run ML training on end user devices to become a hindrance to adoption of federated technologies.

据我们所知，目前还没有满足这些要求的解决方案，我们预计在最终用户设备上运行 ML 培训的能力有限，这将成为采用联邦技术的障碍。

7.5 The Cross-Silo Setting

The system challenges arising in the scenario of cross-silo federated learning take a considerably different form. As outlined in Table 1, clients are fewer in number, more powerful, reliable, and known / addressable, eliminating many of the challenges from the cross-device setting, while allowing for authentication and verification, accounting, and contractually enforced penalties for misbehavior. Nonetheless, there are other sources of heterogeneity, including the features and distribution of data, and possibly the software stack used for training.

cross-silo 联邦学习场景中出现的系统挑战采取了截然不同的形式。如 Table 1 中所述，客户端数量更少, 功能更强大, 更可靠且已知/可寻址，消除了跨设备设置中的许多挑战，同时允许身份验证和验证, 会计和合同对不当行为实施处罚。尽管如此，还有其他异质性来源，包括数据的特征和分布，以及可能用于训练的软件堆栈。

While the infrastructure in the cross-device setting (from the device-side data generation to the server logic) is typically operated by one or few organizational entities (the application, operating system, or device manufacturer), in the cross-silo setting, many different entities are involved. This may lead to high coordination and operational cost due to differences in:

虽然跨设备设置中的基础设施（从设备端数据生成到服务器逻辑）通常由一个或几个组织实体（应用程序, 操作系统或设备制造商）操作，但在cross-silo 设置中，涉及许多不同的实体。由于以下方面的差异，这可能会导致高协调和运营成本：

- *How data is generated, pre-processed and labeled.* Learning across silos will require data normalization which may be difficult when such data is collected and stored differently (e.g. use of different medical imaging systems, and inconsistencies in labeling procedures, annotations, and storage formats).

- *Which software at which version powers training.* Using the same software stack in every silo—possibly delivered alongside the model using container technologies as done by FATE [33]—eliminates compatibility concerns, but such frequent and centrally distributed software delivery may not be acceptable to all involved parties. An alternative that is more similar to the cross-device setting would be to standardize data and model formats and communication protocols. See IEEE P3652.1 "Federated Machine Learning Working Group" for a related effort in this direction.

- *The approval process for how data may or may not be used.* While this process is typically centralized in the cross-device scenario, the situation is likely different in cross-silo settings where many organizational entities are involved, and may be increasingly difficult when training spans different jurisdictions with varying data protection regulations. Technical infrastructure may be of help here by establishing data annotations that encode access policies, and infrastructure enforce them; for instance, limiting the use of certain data to specific models, or encoding minimum aggregation requirements such as "require at least M clients per round".

- 如何生成, 预处理和标记数据。cross-silo 学习将需要数据标准化，当这些数据的收集和存储方式不同时（例如使用不同的医学成像系统，以及标记程序的不一致），这可能会很困难, 注释和存储格式）。

- 哪个版本支持训练的软件。在每个筒仓中使用相同的软件堆栈——可能使用容器技术与模型一起交付，就像 FATE [33] 那样——消除了兼容性问题，但是这种频繁和集中分布的软件交付可能无法为所有相关方所接受。另一种更类似于跨设备设置的替代方法是标准化数据和模型格式以及通信协议。请参阅 IEEE P3652.1 "联邦机器学习工作组"，了解这方面的相关工作。

- 关于如何使用或不使用数据的审批流程。虽然此流程通常集中在跨设备场景中，但在涉及许多组织实体的cross-silo 设置中，情况可能有所不同，并且当培训跨越具有不同数据保护法规的不同司法管辖区时，可能会越来越困难。技术基础设施在这里可能会有所帮助，方法是建立对访问策略进行编码的数据注释，并通过基础设施执行它们；例如，将某些数据的使用限制为特定模型，或编码最低聚合要求，例如 "每轮至少需要 M 个客户"。

Another potential difference in the cross-silo setting is data partitioning: Data in the cross-device setting is typically assumed to be partitioned by examples, all of which have the same features (horizontal partitioning). In the cross-silo setting, in addition to partitioning by examples, partitioning by features is of practical relevance (vertical partitioning). An example would be two organizations, e.g. a bank and a retail company, with an overlapping set of customers, but different information (features) associated with them. For a discussion focusing on the algorithmic aspects, please see section 2.2. Learning with feature-partitioned data may require different communication patterns and additional processing steps e.g. for entity alignment and dealing with missing features.

cross-silo 设置中的另一个潜在差异是数据分区：跨设备设置中的数据通常假设按示例进行分区，所有示例都具有相同的功能（水平分区）。在 cross-silo 设置中，除了通过示例进行分区之外，通过特征进行分区具有实际相关性（垂直分区）。一个例子是两个组织，例如一家银行和一家零售公司，客户群重叠，但与之相关的信息（特征）不同。有关侧重于算法方面的讨论，请参阅 2.2 部分。使用特征分区数据进行学习可能需要不同的通信模式和额外的处理步骤，例如用于实体对齐和处理缺失的特征。

7.6 Executive Summary

While production grade systems for cross-device federated learning operate successfully [81, 26], various challenges remain:

- Frequent and large scale deployment of updates, monitoring, and debugging is challenging (Section 7.1).
- Differences in device availability induce various forms of bias; defining, quantifying and mitigating them remains a direction for future research (Section 7.2).
- Tuning system parameters is difficult due to the existence of multiple, potentially conflicting objectives (Section 7.3).
- Running ML workloads on end user devices is hampered by the lack of a portable, fast, small footprint, and flexible runtime for on-device training (Section 7.4).

虽然用于跨设备联邦学习的生产级系统成功运行 [81, 26]，但仍然存在各种挑战：

- 更新, 监控和调试的频繁和大规模部署具有挑战性（Section 7.1）。
- 设备可用性的差异会导致各种形式的偏差；定义, 量化和减轻它们仍然是未来研究的一个方向（Section 7.2）。
- 由于存在多个潜在冲突的目标（Section 7.3），调整系统参数很困难。
- 在最终用户设备上运行 ML 工作负载因缺乏用于设备上训练的便携, 快速, 占用空间小和灵活的运行时而受到阻碍 (Section 7.4)。

Systems for cross-silo settings (Section 7.5) face largely different issues owing to differences in the capabilities of compute nodes and the nature of the data being processed.

由于计算节点的能力和所处理数据的性质不同，cross-silo 设置系统 (Section 7.5) 面临着很大的不同问题。

8 Software and Datasets for Federated Learning

Software for simulation Simulations of federated learning require dealing with multiple issues that do not arise in datacenter ML research, for example, efficiently processing partitioned datasets, with computations running on different simulated devices, each with a variable amount of data. FL research also requires different metrics such as the number of bytes upload or downloaded by device, as well as the ability to simulate issues like time-varying arrival of different clients or client drop-out that is potentially correlated with the nature of the local dataset. With this in mind, the development of open software frameworks for federated learning research (simulation) has the potential to greatly accelerate research progress. Several platforms are available or in development, including [404]:

- TensorFlow Federated [38] specifically targets research use cases, providing large-scale simulation capabilities as well as flexible orchestration for the control of sampling.

- FedML [229] is a research-oriented library. It supports three platforms: on-device training for IoT and mobile devices, distributed computing, and single-machine simulation. For research diversity, FedML also supports various algorithms (e.g., decentralized learning, vertical FL, and split learning), models, and datasets.
- PySyft [399] is a Python library for secure, private Deep Learning. PySyft decouples private data from model training, using federated learning, differential privacy, and multi-party computation (MPC) within PyTorch.
- Leaf [35] provides multiple datasets (see below), as well as simulation and evaluation capabilities.
- Sherpa.ai Federated Learning and Differential Privacy Framework [397] is an open source federated learning and differential privacy framework which provides methodologies, pipelines, and evaluation techniques for federated learning.
- PyVertical [32] is a project focusing on federated learning with data partitioned by features (also referred to as vertical partitioning) in the cross-silo setting; see Section 2.2.

Production-oriented software In addition to the above simulation platforms, several production-oriented federated learning platforms are being developed:

- FATE (Federated AI Technology Enabler) [33] is an open-source project intended to provide a secure computing framework to support the federated AI ecosystem.
- PaddleFL [36] is an open source federated learning framework based on PaddlePaddle [37]. In PaddleFL, several federated learning strategies and training strategies are provided with application demonstrations.
- Clara Training Framework [125] includes the support of cross-silo federated learning based on a server-client approach with data privacy protection.
- IBM Federated Learning [321] is a Python-based federated learning framework for enterprise environments, which provides a basic fabric for adding advanced features.
- Flower framework [66] supports implementation and experimentation of federated learning algorithms on mobile and embedded devices with a real-world system conditions simulation.
- Fedlearner [34] is an open source federated learning framework that enables joint modeling of data distributed between institutions.

Such production-oriented federated learning platforms must address problems that do not exist in simulation such as authentication, communication protocols, encryption and deployment to physical devices or silos. Note that while TensorFlow Federated is listed under "Software for simulation", its design includes abstractions for aggregation and broadcast, and serialization of all TensorFlow computations for execution in non-Python environments, making it suitable for use as a component in a production system.

Datasets Federated learning is adopted when the data is decentralized and typically unbalanced (different clients have different numbers of examples) and not identically distributed (each client's data is drawn from a different distribution). The open source package TensorFlow Federated [38] supports loading decentralized dataset in a simulated

environment with each client id corresponding to a TensorFlow Dataset Object. These datasets can easily be converted to numpy arrays for use in other frameworks.[18] At the time of writing, three datasets are supported and we recommend researchers to benchmark on them.

- *EMNIST* dataset [126] consists of 671,585 images of digits and upper and lower case English characters (62 classes). The federated version splits the dataset into 3,400 unbalanced clients indexed by the original writer of the digits/characters. The non-IID distribution comes from the unique writing style of each person.
- *Stackoverflow*[19] dataset consists of question and answer from Stack Overflow with metadata like timestamps, scores, etc. The training dataset has more than 342,477 unique users with 135,818,730 examples. Note that the timestamp information can be helpful to simulate the pattern of incoming data.
- *Shakespeare* is a language modeling dataset derived from *The Complete Works of William Shakespeare*. It consists of 715 characters whose contiguous lines are examples in the client dataset. The train set has 16,068 examples and test set has 2,356 examples.

The preprocessing for *EMNIST* and *Shakespeare* are provided by the Leaf project [96], which also provides federated versions of the sentiment140 and celebA datasets. These datasets have enough clients that they can be used to simulate cross-device FL scenarios, but for questions where scale is particularly important, they may be too small. In this respect *Stackoverflow* provides the most realistic example of a cross-device FL problem.

Cross-silo datasets One example is the iNaturalist dataset[20] which consists of large numbers of observations of various organisms all over the world. One can partition it by the geolocation or the author of an observation. If we partition it by the group an organism belongs to, like kingdom, phylum, etc., then the clients have totally different labels and biological closeness between two clients is already known. This makes it a very suitable dataset to study federated transfer learning and multi-task learning in cross-silo settings.

Another example is the Google-Landmark-v2 [456] that includes over 5 million images of more than 200 thousand different types of landmark. Similar to the iNaturalist dataset, one can split the dataset by authors, but due to the difference in scale with iNaturalist dataset, Google Landmark Dataset provides much more diversity and creates even greater challenges to large-scale federated learning.

Luo et al. [322] has recently published a federated dataset for computer vision. The dataset contains more than 900 annotated street images generated from 26 street cameras and 7 object categories annotated with detailed bounding box. Due to the relatively small number of examples in the dataset, it may not adequately reflect a challenging realistic scenario.

The need for more datasets Developing new federated learning datasets that are representative of real-world problems is an important question for the community to address. Platforms like TensorFlow Federated [38] welcome the contribution of new datasets and may be able to provide hosting support.

[18] https://www.tensorflow.org/datasets/api_docs/python/tfds/as_numpy.
[19] https://www.kaggle.com/stackoverflow/stackoverflow
[20] https://www.inaturalist.org/

While completely new datasets are always interesting, in many cases it is possible to partition existing open datasets, treating each split as a client. Different partitioning strategies may be appropriate for different research questions, but often unbalanced and non-IID partitions will be most relevant. It is also interesting to maintain as much additional meta information (timestamp, geolocation, etc.) as possible.

In particular, there is a need for feature-partitioned datasets, as will be discussed in Section 2.2. For example, a patient may go to one medical institute for a pathology test and go to another for radiology picture archiving, in which case the features of one sample are partitioned over two institutes regulated by HIPAA. [24].

9 Concluding Remarks

Federated learning enables distributed client devices to collaboratively learn a shared prediction model while keeping all the training data on device, decoupling the ability to do machine learning from the need to store the data in the cloud. This goes beyond the use of local models that make predictions on mobile devices by bringing model training to the device as well.

联邦学习使分布式客户端设备能够协作学习共享的预测模型，同时将所有训练数据保存在设备上，将进行机器学习的能力与将数据存储在云中的需求分离。这超越了使用本地模型在移动设备上进行预测的方式，还将模型训练引入设备。

In recent years, this topic has undergone an explosive growth of interest, both in industry and academia. Major technology companies have already deployed federated learning in production, and a number of startups were founded with the objective of using federated learning to address privacy and data collection challenges in various industries. Further, the breadth of papers surveyed in this work suggests that federated learning is gaining traction in a wide range of interdisciplinary fields: from machine learning to optimization to information theory and statistics to cryptography, fairness, and privacy.

近年来，无论是在工业界还是学术界，这个话题都经历了爆炸性的增长。主要科技公司已经在生产中部署了联邦学习，并且成立了许多初创公司，目的是使用联邦学习来解决各个行业的隐私和数据收集挑战。此外，这项工作中调查的论文的广度表明，联邦学习正在广泛的跨学科领域获得关注：从机器学习到优化，再到信息理论和统计学，再到密码学、公平性和隐私。

Motivated by the growing interest in federated learning research, this paper discusses recent advances and presents an extensive collection of open problems and challenges. The system constraints impose efficiency requirements on the algorithms in order to be practical, many of which are not particularly challenging in other settings. We argue that data privacy is not binary and present a range of threat models that are relevant under a variety of assumptions, each of which provides its own unique challenges.

受对联邦学习研究日益增长的兴趣的推动，本文讨论了最近的进展，并提出了大量未解决的问题和挑战。为了实用，系统约束对算法提出了效率要求，其中许多在其他设置中并不是特别具有挑战性。我们认为，数据隐私不是二元的，并提出了一系列在各种假设下相关的威胁模型，每个模型都有其独特的挑战。

The open problems discussed in this work are certainly not comprehensive, they reflect the interests and backgrounds of the authors. In particular, we do not discuss any non-learning problems which need to be solved in the course of a practical machine learning project, and might need to be solved based on decentralized data [382]. This can include simple problems such as computing basic descriptive statistics, or more complex objectives such as computing the head of a histogram over an open set [510]. Existing algorithms for solving such problems often do not always have an obvious "federated version" that would be efficient under the system assumptions motivating this work or do not admit a useful notion of data protection. Yet another set of important topics that were not discussed are the legal and business issues that may motivate or constrain the use of federated learning.

这项工作中讨论的开放性问题当然并不全面，它们反映了作者的兴趣和背景。特别是，我们没有讨论在实际机器学习项目过程中需要解决的任何非学习问题，并且可能需要基于去中心化数据 [382] 来解决。这可以包括简单的问题，例如计算基本的描述性统计，或者更复杂的目标，例如计算开放集 [510] 上的直方图的头部。用于解决此类问题的现有算法通常并不总是具有明显的“联邦版本”，该版本在推动这项工作的系统

假设下是有效的，或者不承认有用的数据保护概念。另一组未讨论的重要主题是可能激发或限制联邦学习使用的法律和商业问题。

We hope this work will be helpful in scoping further research in federated learning and related areas.

我们希望这项工作将有助于界定联邦学习和相关领域的进一步研究。

Acknowledgments

The authors would like to thank Alex Ingerman and David Petrou for their useful suggestions and insightful comments during the review process.

References

[1] Lattigo 2.0.0. Online: http://github.com/ldsec/lattigo, October 2020. EPFL-LDS.

[2] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL https://www.tensorflow.org/. Software available from tensorflow.org.

[3] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318. ACM, 2016.

[4] Omid Abari, Hariharan Rahul, and Dina Katabi. Over-the-air function computation in sensor networks. *CoRR*, abs/1612.02307, 2016. URL http://arxiv.org/abs/1612.02307.

[5] Nazmiye Ceren Abay, Yan Zhou, Murat Kantarcioglu, Bhavani Thuraisingham, and Latanya Sweeney. Privacy preserving synthetic data release using deep learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 510–526. Springer, 2018.

[6] John M Abowd and Ian M Schmutte. An economic analysis of privacy protection and statistical accuracy as social choices. *American Economic Review*, 109(1):171–202, 2019.

[7] Jayadev Acharya, Clément L Canonne, and Himanshu Tyagi. Inference under information constraints i: Lower bounds from chi-square contraction. *IEEE Transactions on Information Theory*, 66(12):7835–7855, 2020.

[8] Gergely Ács and Claude Castelluccia. I have a DREAM!: DIfferentially PrivatE smart Metering. In *Proceedings of the 13th International Conference on Information Hiding*, IH'11, pages 118–132, Berlin, Heidelberg, 2011. Springer-Verlag. ISBN 978-3-642-24177-2. URL http://dl.acm.org/citation.cfm?id=2042445.2042457.

[9] Naman Agarwal, Ananda Theertha Suresh, Felix X. Yu, Sanjiv Kumar, and Brendan McMahan. cpSGD: Communication-efficient and differentially-private distributed SGD. In *Advances in Neural Information Processing Systems*, pages 7564–7575, 2018.

[10] Nitin Agrawal, Ali Shahin Shamsabadi, Matt J. Kusner, and Adrià Gascón. QUOTIENT: two-party secure neural network training and prediction. In *In Proceedings of the ACM Conference on Computer and Communication Security (CCS)*, 2019.

[11] Rakesh Agrawal and Ramakrishnan Srikant. Privacy-preserving data mining. In *ACM SIGMOD International Conference on Management of Data*, 2000.

[12] Carlos Aguilar-Melchor and Philippe Gaborit. A lattice-based computationally-efficient private information retrieval protocol. *Cryptol. ePrint Arch., Report*, 446, 2007.

[13] Carlos Aguilar-Melchor, Joris Barrier, Laurent Fousse, and Marc-Olivier Killijian. XPIR: Private information retrieval for everyone. *Proceedings on Privacy Enhancing Technologies*, 2016(2):155–174, 2016.

[14] ai.google. Under the hood of the Pixel 2: How AI is supercharging hardware, 2018. URL https://ai.google/stories/ai-in-hardware/. Retrieved Nov 2018.

[15] ai.intel. Federated learning for medical imaging, 2019. URL https://www.intel.ai/federated-learning-for-medical-imaging/. Retrieved Aug 2019.

[16] Asra Ali, Tancrède Lepoint, Sarvar Patel, Mariana Raykova, Phillipp Schoppmann, Karn Seth, and Kevin Yeo. Communication-computation trade-offs in PIR. *IACR Cryptol. ePrint Arch.*, 2019:1483, 2019.

[17] Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. QSGD: Communication-efficient SGD via gradient quantization and encoding. In *NIPS - Advances in Neural Information Processing Systems*, pages 1709–1720, 2017.

[18] Dan Alistarh, Zeyuan Allen-Zhu, and Jerry Li. Byzantine stochastic gradient descent. In *NIPS*, 2018.

[19] Inês Almeida and João Xavier. DJAM: Distributed Jacobi Asynchronous Method for Learning Personal Models. *IEEE Signal Processing Letters*, 25(9):1389–1392, 2018.

[20] Scott Ames, Carmit Hazay, Yuval Ishai, and Muthuramakrishnan Venkitasubramaniam. Ligero: Lightweight sublinear arguments without a trusted setup. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, 2017.

[21] Kareem Amin, Alex Kulesza, Andres Munoz, and Sergei Vassilvtiskii. Bounding user contributions: A bias-variance trade-off in differential privacy. In *International Conference on Machine Learning*, pages 263–271, 2019.

[22] androidtrusty. Android Trusty TEE. https://source.android.com/security/trusty, 2019. Accessed: 2019-12-05.

[23] Sebastian Angel, Hao Chen, Kim Laine, and Srinath T. V. Setty. PIR with compressed queries and amortized query processing. In *IEEE Symposium on Security and Privacy*, pages 962–979. IEEE Computer Society, 2018.

[24] George J Annas. HIPAA regulations-a new era of medical-record privacy? *New England Journal of Medicine*, 348(15): 1486–1490, 2003.

[25] Apple. Private Federated Learning (NeurIPS 2019 Expo Talk Abstract). https://nips.cc/ExpoConferences/2019/schedule?talk_id=40, 2019.

[26] Apple. Designing for privacy (video and slide deck). Apple WWDC, https://developer.apple.com/videos/play/wwdc2019/708, 2019.

[27] Toshinori Araki, Jun Furukawa, Yehuda Lindell, Ariel Nof, and Kazuma Ohara. High-throughput semi-honest secure three-party computation with an honest majority. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 805–817. ACM, 2016.

[28] armtrustzone. Arm TrustZone Technology. https://developer.arm.com/ip-products/security-ip/trustzone, 2019. Accessed: 2019-12-05.

[29] Mahmoud Assran, Nicolas Loizou, Nicolas Ballas, and Michael Rabbat. Stochastic gradient push for distributed deep learning. In *ICML*, 2019.

[30] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *ICML*, 2018.

[31] Sean Augenstein, H. Brendan McMahan, Daniel Ramage, Swaroop Ramaswamy, Peter Kairouz, Mingqing Chen, Rajiv Mathews, and Blaise Aguera y Arcas. Generative models for effective ML on private, decentralized datasets, 2019. URL https://arxiv.org/abs/1911.06679.

[32] PyVertical Authors. Pyvertical, 2020. URL https://github.com.cnpmjs.org/OpenMined/PyVertical.

[33] The FATE Authors. Federated AI technology enabler, 2019. URL https://www.fedai.org/.

[34] The Fedlearner Authors. Fedlearner, 2020. URL https://github.com/bytedance/fedlearner.

[35] The Leaf Authors. Leaf, 2019. URL https://leaf.cmu.edu/.

[36] The PaddleFL Authors. PaddleFL, 2019. URL https://github.com/PaddlePaddle/PaddleFL.

[37] The PaddlePaddle Authors. PaddlePaddle, 2019. URL http://www.paddlepaddle.org/.

[38] The TFF Authors. TensorFlow Federated, 2019. URL https://www.tensorflow.org/federated.

[39] Brendan Avent, Yatharth Dubey, and Aleksandra Korolova. The power of the hybrid model for mean estimation. *Proceedings on Privacy Enhancing Technologies (PETS)*, 2020(4):48 – 68, 01 Oct. 2020. doi: https://doi.org/10.2478/popets-2020-0062. URL https://content.sciendo.com/view/journals/popets/2020/4/article-p48.xml.

[40] Brendan Avent, Aleksandra Korolova, David Zeber, Torgeir Hovden, and Benjamin Livshits. BLENDER: Enabling local search with a hybrid differential privacy model. In *26th USENIX Security Symposium (USENIX Security 17)*, pages 747–764, Vancouver, BC, August 2017. USENIX Association. ISBN 978-1-931971-40-9. URL https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/avent.

[41] Pranjal Awasthi, Corinna Cortes, Yishay Mansour, and Mehryar Mohri. Beyond individual and group fairness. *CoRR*, abs/2008.09490, 2020.

[42] László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. Checking computations in polylogarithmic time. In *STOC*, pages 21–31. ACM, 1991.

[43] Eugene Bagdasaryan and Vitaly Shmatikov. Differential privacy has disparate impact on model accuracy. *CoRR*, abs/1905.12101, 2019. URL http://arxiv.org/abs/1905.12101.

[44] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. *arXiv preprint arXiv:1807.00459*, 2018.

[45] Borja Balle, James Bell, Adrià Gascón, and Kobbi Nissim. The privacy blanket of the shuffle model. In *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part II*, pages 638–667, 2019. doi: 10.1007/978-3-030-26951-7_22. URL https://doi.org/10.1007/978-3-030-26951-7_22.

[46] Borja Balle, James Bell, Adrià Gascón, and Kobbi Nissim. Private summation in the multi-message shuffle model. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, page 657–676. ACM, 2020.

[47] Borja Balle, Peter Kairouz, H. Brendan McMahan, Om Thakkar, and Abhradeep Thakurta. Privacy amplification via random check-ins, 2020.

[48] Assi Barak, Daniel Escudero, Anders P. K. Dalskov, and Marcel Keller. Secure evaluation of quantized neural networks. *IACR Cryptology ePrint Archive*, 2019:131, 2019. URL https://eprint.iacr.org/2019/131.

[49] Leighton Pate Barnes, Yanjun Han, and Ayfer Ozgur. Lower bounds for learning distributions under communication constraints via fisher information. *Journal of Machine Learning Research*, 21(236):1–30, 2020. URL http://jmlr.org/papers/v21/19-737.html.

[50] Leighton Pate Barnes, Huseyin A. Inan, Berivan Isik, and Ayfer Ozgur. rtop-k: A statistical estimation approach to distributed sgd. *arXiv preprint arXiv:2005.10761*, 2020.

[51] Solon Barocas, Moritz Hardt, and Arvind Narayanan. *Fairness and Machine Learning*. fairmlbook.org, 2019. http://www.fairmlbook.org.

[52] Moran Baruch, Gilad Baruch, and Yoav Goldberg. A little is enough: Circumventing defenses for distributed learning. *arXiv preprint arXiv:1902.06156*, 2019.

[53] Raef Bassily and Adam Smith. Local, private, efficient protocols for succinct histograms. In *STOC*, pages 127–135, 2015.

[54] Raef Bassily, Uri Stemmer, Abhradeep Guha Thakurta, et al. Practical locally private heavy hitters. In *Advances in Neural Information Processing Systems*, pages 2288–2296, 2017.

[55] Debraj Basu, Deepesh Data, Can Karakus, and Suhas N Diggavi. Qsparse-local-sgd: Distributed sgd with quantization, sparsification, and local computations. *IEEE Journal on Selected Areas in Information Theory*, 1(1):217–226, 2020.

[56] Jonathan Baxter. A model of inductive bias learning. *Journal of Artificial Intelligence Research*, 12:149–198, 2000.

[57] Amos Beimel, Aleksandra Korolova, Kobbi Nissim, Or Sheffet, and Uri Stemmer. The power of synergy in differential privacy: Combining a small curator with local randomizers. In *Conference on Information-Theoretic Cryptography (ITC)*, 2020. URL https://arxiv.org/abs/1912.08951.

[58] James Henry Bell, Kallista A. Bonawitz, Adrià Gascón, Tancrède Lepoint, and Mariana Raykova. Secure single-server aggregation with (poly)logarithmic overhead. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, page 1253–1269. ACM, 2020.

[59] Aurélien Bellet, Rachid Guerraoui, Mahsa Taziki, and Marc Tommasi. Personalized and Private Peer-to-Peer Machine Learning. In *AISTATS*, 2018.

[60] Irwan Bello, Barret Zoph, Vijay Vasudevan, and Quoc V Le. Neural optimizer search with reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 459–468. JMLR. org, 2017.

[61] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79(1-2):151–175, 2010.

[62] Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *IEEE Symposium on Security and Privacy*, pages 459–474. IEEE Computer Society, 2014.

[63] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Scalable zero knowledge with no trusted setup. In *CRYPTO (3)*, volume 11694 of *Lecture Notes in Computer Science*, pages 701–732. Springer, 2019.

[64] James S Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In *Advances in Neural Information Processing Systems*, pages 2546–2554, 2011.

[65] Martín Bertrán, Natalia Martínez, Afroditi Papadaki, Qiang Qiu, Miguel R. D. Rodrigues, Galen Reeves, and Guillermo Sapiro. Learning adversarially fair and transferable representations. In *ICML*, 2019.

[66] Daniel J. Beutel, Taner Topal, Akhil Mathur, Xinchi Qiu, Titouan Parcollet, and Nicholas D. Lane. Flower: A friendly federated learning research framework, 2020.

[67] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. Analyzing federated learning through an adversarial lens. In *Proceedings of the 36th International Conference on Machine Learning*, pages 634–643, 2019.

[68] Abhishek Bhowmick, John Duchi, Julien Freudiger, Gaurav Kapoor, and Ryan Rogers. Protection against reconstruction and its applications in private federated learning. *arXiv preprint arXiv:1812.00984*, 2018.

[69] Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML'12, pages 1467–1474, USA, 2012. Omnipress. ISBN 978-1-4503-1285-1. URL http://dl.acm.org/citation.cfm?id=3042573.3042761.

[70] Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *ECML-PKDD*, pages 387–402. Springer, 2013.

[71] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, ITCS '12, 2012.

[72] R. Bitar and S. E. Rouayheb. Staircase-PIR: Universally robust private information retrieval. In *2018 IEEE Information Theory Workshop (ITW)*, pages 1–5, Nov 2018. doi: 10.1109/ITW.2018.8613532.

[73] Andrea Bittau, Úlfar Erlingsson, Petros Maniatis, Ilya Mironov, Ananth Raghunathan, David Lie, Mitch Rudominer, Ushasree Kode, Julien Tinnes, and Bernhard Seefeld. Prochlo: Strong privacy for analytics in the crowd. In *Proceedings of the 26th Symposium on Operating Systems Principles*, SOSP '17, pages 441–459, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-5085-3. doi: 10.1145/3132747.3132769. URL http://doi.acm.org/10.1145/3132747.3132769.

[74] Davis Blalock, Jose Javier Gonzalez Ortiz, Jonathan Frankle, and John Guttag. What is the state of neural network pruning? *arXiv preprint arXiv:2003.03033*, 2020.

[75] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. In *Advances in Neural Information Processing Systems*, 2017.

[76] Peva Blanchard, Rachid Guerraoui, Julien Stainer, et al. Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent. In *Advances in Neural Information Processing Systems*, pages 118–128, 2017.

[77] Dan Bogdanov, Riivo Talviste, and Jan Willemson. Deploying secure multi-party computation for financial data analysis - (short paper). In *Financial Cryptography*, volume 7397 of *Lecture Notes in Computer Science*, pages 57–64. Springer, 2012.

[78] Peter Bogetoft, Dan Lund Christensen, Ivan Damgård, Martin Geisler, Thomas P. Jakobsen, Mikkel Krøigaard, Janus Dam Nielsen, Jesper Buus Nielsen, Kurt Nielsen, Jakob Pagter, Michael I. Schwartzbach, and Tomas Toft. Secure multiparty computation goes live. In *Financial Cryptography*, volume 5628 of *Lecture Notes in Computer Science*, pages 325–343. Springer, 2009.

[79] K. A. Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for federated learning on user-held data. *arXiv preprint arXiv:1611.04482*, 2016.

[80] K. A. Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1175–1191. ACM, 2017.

[81] K. A. Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloé M Kiddon, Jakub Konečný, Stefano Mazzocchi, Brendan McMahan, Timon Van Overveldt, David Petrou, Daniel Ramage, and Jason Roselander. Towards federated learning at scale: System design. In *SysML 2019*, 2019. URL https://arxiv.org/abs/1902.01046.

[82] K. A. Bonawitz, Fariborz Salehi, Jakub Konečný, Brendan McMahan, and Marco Gruteser. Federated learning with auto-tuned communication-efficient secure aggregation. In *2019 53nd Asilomar Conference on Signals, Systems, and Computers*. IEEE, 2019.

[83] Dan Boneh, Elette Boyle, Henry Corrigan-Gibbs, Niv Gilboa, and Yuval Ishai. Zero-knowledge proofs on secret-shared data via fully linear PCPs. In *CRYPTO (3)*, volume 11694 of *Lecture Notes in Computer Science*, pages 67–97. Springer, 2019.

[84] Florian Bourse, Michele Minelli, Matthias Minihold, and Pascal Paillier. Fast homomorphic evaluation of deep discretized neural networks. In *CRYPTO (3)*, volume 10993 of *Lecture Notes in Computer Science*, pages 483–512. Springer, 2018.

[85] Stephen Boyd, Arpita Ghosh, Balaji Prabhakar, and Devavrat Shah. Randomized gossip algorithms. *IEEE Transactions on Information Theory*, 52(6):2508–2530, 2006.

[86] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

[87] Zvika Brakerski. Fully homomorphic encryption without modulus switching from classical gapsvp. In *CRYPTO*, volume 7417 of *Lecture Notes in Computer Science*, pages 868–886. Springer, 2012.

[88] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. In *ITCS*, pages 309–325. ACM, 2012.

[89] Mark Braverman, Ankit Garg, Tengyu Ma, Huy L. Nguyen, and David P. Woodruff. Communication lower bounds for statistical estimation problems via a distributed data processing inequality. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, page 1011–1020. ACM, 2016.

[90] Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. *arXiv preprint arXiv:1712.04248*, 2017.

[91] Theodora S Brisimi, Ruidi Chen, Theofanie Mela, Alex Olshevsky, Ioannis Ch Paschalidis, and Wei Shi. Federated learning of predictive models from federated electronic health records. *International journal of medical informatics*, 112:59–67, 2018.

[92] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Gregory Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *2018 IEEE Symposium on Security and Privacy, SP 2018, Proceedings, 21-23 May 2018, San Francisco, California, USA*, 2018.

[93] Joy Buolamwini and Timnit Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Conference on fairness, accountability and transparency*, pages 77–91, 2018.

[94] Martin Burkhart, Mario Strasser, Dilip Many, and Xenofontas Dimitropoulos. SEPIA: Privacy-preserving aggregation of multi-domain network events and statistics. *Network*, 1(101101), 2010.

[95] Sebastian Caldas, Jakub Konečný, H Brendan McMahan, and Ameet Talwalkar. Expanding the reach of federated learning by reducing client resource requirements. *arXiv preprint arXiv:1812.07210*, 2018.

[96] Sebastian Caldas, Peter Wu, Tian Li, Jakub Konečný, H Brendan McMahan, Virginia Smith, and Ameet Talwalkar. LEAF: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.

[97] Clément L Canonne, Gautam Kamath, Audra McMillan, Adam Smith, and Jonathan Ullman. The structure of optimal private tests for simple hypotheses. *AarXiv preprint arXiv:1811.11148*, 2019.

[98] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57. IEEE, 2017.

[99] Nicholas Carlini, Chang Liu, Jernej Kos, Úlfar Erlingsson, and Dawn Song. The secret sharer: Measuring unintended neural network memorization & extracting secrets. *arXiv preprint arXiv:1802.08232*, 2018.

[100] Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. *arXiv preprint arXiv:2012.07805*, 2020.

[101] Iker Ceballos, Vivek Sharma, Eduardo Mugica, Abhishek Singh, Albert Roman, Praneeth Vepakomma, and Ramesh Raskar. SplitNN-driven vertical partitioning. *arXiv preprint arXiv:2008.04137*, 2018.

[102] Khe Chai Sim, Françoise Beaufays, Arnaud Benard, Dhruv Guliani, Andreas Kabel, Nikhil Khare, Tamar Lucassen, Petr Zadrazil, Harry Zhang, Leif Johnson, et al. Personalization of end-to-end speech recognition on mobile devices for named entities. *arXiv*, pages arXiv–1912, 2019.

[103] T-H Hubert Chan, Elaine Shi, and Dawn Song. Privacy-preserving stream aggregation with fault tolerance. In *International Conference on Financial Cryptography and Data Security*, pages 200–214. Springer, 2012.

[104] Ken Chang, Niranjan Balachandar, Carson Lam, Darvin Yi, James Brown, Andrew Beers, Bruce Rosen, Daniel L Rubin, and Jayashree Kalpathy-Cramer. Distributed deep learning networks among institutions for medical imaging. *Journal of the American Medical Informatics Association*, 25(8):945–954, 2018.

[105] Wei-Ting Chang and Ravi Tandon. On the upload versus download cost for secure and private matrix multiplication. *ArXiv*, abs/1906.10684, 2019.

[106] Zachary Charles and Jakub Konečnỳ. On the outsized importance of learning rates in local update methods. *arXiv preprint arXiv:2007.00878*, 2020.

[107] David Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2), 1981.

[108] Bryant Chen, Wilka Carvalho, Nathalie Baracaldo, Heiko Ludwig, Benjamin Edwards, Taesung Lee, Ian Molloy, and Biplav Srivastava. Detecting backdoor attacks on deep neural networks by activation clustering. *arXiv preprint arXiv:1811.03728*, 2018.

[109] Chien-Lun Chen, Leana Golubchik, and Marco Paolieri. Backdoor attacks on federated meta-learning. *arXiv preprint arXiv:2006.07026*, 2020.

[110] Lijie Chen, Badih Ghazi, Ravi Kumar, and Pasin Manurangsi. On distributed differential privacy and counting distinct elements. In *Innovations in Theoretical Computer Science (ITCS)*, 2021.

[111] Lingjiao Chen, Hongyi Wang, Zachary B. Charles, and Dimitris S. Papailiopoulos. DRACO: Byzantine-resilient distributed training via redundant gradients. In *Proceedings of the 35th International Conference on Machine Learning, ICML*, 2018.

[112] Mingqing Chen, Rajiv Mathews, Tom Ouyang, and Françoise Beaufays. Federated learning of out-of-vocabulary words. *arXiv preprint 1903.10635*, 2019. URL http://arxiv.org/abs/1903.10635.

[113] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. ZOO: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 15–26. ACM, 2017.

[114] Wei-Ning Chen, Peter Kairouz, and Ayfer Ozgur. Breaking the communication-privacy-accuracy trilemma. *Advances in Neural Information Processing Systems*, 33, 2020.

[115] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*, 2017.

[116] Yudong Chen, Lili Su, and Jiaming Xu. Distributed Statistical Machine Learning in Adversarial Settings: Byzantine Gradient Descent. *POMACS*, 1:44:1–44:25, 2017.

[117] Massimo Chenal and Qiang Tang. On key recovery attacks against existing somewhat homomorphic encryption schemes. In *LATINCRYPT*, volume 8895 of *Lecture Notes in Computer Science*, pages 239–258. Springer, 2014.

[118] Kewei Cheng, Tao Fan, Yilun Jin, Yang Liu, Tianjian Chen, and Qiang Yang. SecureBoost: A lossless federated learning framework. *CoRR*, abs/1901.08755, 2019. URL http://arxiv.org/abs/1901.08755.

[119] Raymond Cheng, Fan Zhang, Jernej Kos, Warren He, Nicholas Hynes, Noah Johnson, Ari Juels, Andrew Miller, and Dawn Song. Ekiden: A platform for confidentiality-preserving, trustworthy, and performant smart contracts. In *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 185–200. IEEE, 2019.

[120] Albert Cheu, Adam Smith, Jonathan Ullman, David Zeber, and Maxim Zhilyaev. Distributed differential privacy via shuffling. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 375–403. Springer, 2019.

[121] Benny Chor, Eyal Kushilevitz, Oded Goldreich, and Madhu Sudan. Private information retrieval. *J. ACM*, 45(6):965–981, November 1998. ISSN 0004-5411. doi: 10.1145/293347.293350. URL http://doi.acm.org/10.1145/293347.293350.

[122] Edward Chou, Florian Tramèr, and Giancarlo Pellegrino. SentiNet: Detecting physical attacks against deep learning systems. *arXiv preprint arXiv:1812.00292*, 2018.

[123] Sélim Chraibi, Ahmed Khaled, Dmitry Kovalev, Peter Richtárik, Adil Salim, and Martin Takáč. Distributed fixed point methods with compressed iterates. *arXiv preprint arXiv:1912.09925*, 2019.

[124] P. Christen. *Data matching: concepts and techniques for record linkage, entity resolution, and duplicate detection*. Springer Science & Business Media, 2012.

[125] NVIDIA Clara. The clara training framework authors, 2019. URL https://developer.nvidia.com/clara.

[126] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André van Schaik. EMNIST: an extension of MNIST to handwritten letters. *arXiv preprint arXiv:1702.05373*, 2017.

[127] Igor Colin, Aurélien Bellet, Joseph Salmon, and Stéphan Clémençon. Gossip dual averaging for decentralized optimization of pairwise functions. In *ICML*, 2016.

[128] Graham Cormode, Tejas Kulkarni, and Divesh Srivastava. Marginal release under local differential privacy. In *Proceedings of the 2018 International Conference on Management of Data*, pages 131–146. ACM, 2018.

[129] Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. Scale-invariant fully homomorphic encryption over the integers. In *Public Key Cryptography*, volume 8383 of *Lecture Notes in Computer Science*, pages 311–328. Springer, 2014.

[130] Henry Corrigan-Gibbs and Dan Boneh. Prio: Private, robust, and scalable computation of aggregate statistics. In *14th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 17)*, pages 259–282, 2017.

[131] Henry Corrigan-Gibbs and Dmitry Kogan. Private information retrieval with sublinear online time. *IACR Cryptology ePrint Archive*, 2019:1075, 2019.

[132] Corinna Cortes and Mehryar Mohri. Domain adaptation and sample bias correction theory and algorithm for regression. *Theoretical Computer Science*, 519:103–126, 2014.

[133] Corinna Cortes, Mehryar Mohri, Ananda Theertha Suresh, and Ningshan Zhang. Multiple-source adaptation with domain classifiers. *arXiv preprint arXiv:2008.11036*, 2020.

[134] Victor Costan and Srinivas Devadas. Intel SGX explained. *IACR Cryptology ePrint Archive*, 2016(086):1–118, 2016.

[135] Victor Costan, Ilia Lebedev, and Srinivas Devadas. Sanctum: Minimal hardware extensions for strong software isolation. In *25th {USENIX} Security Symposium ({USENIX} Security 16)*, pages 857–874, 2016.

[136] Craig Costello, Cédric Fournet, Jon Howell, Markulf Kohlweiss, Benjamin Kreuter, Michael Naehrig, Bryan Parno, and Samee Zahur. Geppetto: Versatile verifiable computation. In *IEEE Symposium on Security and Privacy*, pages 253–270. IEEE Computer Society, 2015.

[137] Andrew Cotter, Ohad Shamir, Nati Srebro, and Karthik Sridharan. Better mini-batch algorithms via accelerated gradient methods. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 24, pages 1647–1655. Curran Associates, Inc., 2011. URL https://proceedings.neurips.cc/paper/2011/file/b55ec28c52d5f6205684a473a2193564-Paper.pdf.

[138] Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. BinaryConnect: Training deep neural networks with binary weights during propagations. In *Advances in neural information processing systems*, pages 3123–3131, 2015.

[139] Pierre Courtiol, Charles Maussion, Matahi Moarii, Elodie Pronier, Samuel Pilcer, Meriem Sefta, Pierre Manceron, Sylvain Toldo, Mikhail Zaslavskiy, Nolwenn Le Stang, et al. Deep learning-based classification of mesothelioma improves prediction of patient outcome. *Nature medicine*, pages 1–7, 2019.

[140] Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

[141] Gabriela F Cretu, Angelos Stavrou, Michael E Locasto, Salvatore J Stolfo, and Angelos D Keromytis. Casting out demons: Sanitizing training data for anomaly sensors. In *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pages 81–95. IEEE, 2008.

[142] Rachel Cummings, Sara Krehbiel, Kevin Lai, and Uthaipon Tantitongpipat. Differential privacy for growing databases. In *Advances in Neural Information Processing Systems 31*, NeurIPS '18, pages 8864–8873, 2018.

[143] Rachel Cummings, Sara Krehbiel, Yajun Mei, Rui Tuo, and Wanrong Zhang. Differentially private change-point detection. In *Advances in Neural Information Processing Systems 31*, NeurIPS '18, pages 10825–10834, 2018.

[144] Rachel Cummings, Inbal Dekel, Ori Heffetz, and Katrina Ligett. Bringing differential privacy into the experimental economics lab: Theory and an application to a public-good game. Working paper, 2019.

[145] Rachel Cummings, Varun Gupta, Dhamma Kimpara, and Jamie Morgenstern. On the compatibility of privacy and fairness. In *Proceedings of Fairness in User Modeling, Adaptation and Personalization*, FairUMAP, 2019.

[146] Edwige Cyffers and Aurélien Bellet. Privacy amplification by decentralization. *arXiv preprint arXiv:2012.05326*, 2020.

[147] Damgård. On σ protocols. http://www.cs.au.dk/~ivan/Sigma.pdf, 2010.

[148] Deepesh Data, Linqi Song, and Suhas Diggavi. Data encoding for byzantine-resilient distributed optimization. *IEEE Transactions on Information Theory*, 2020.

[149] Walter de Brouwer. The federated future is ready for shipping. https://doc.ai/blog/federated-future-ready-shipping/, March 2019.

[150] Jeffrey Dean, Greg S. Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Quoc V. Le, Mark Z. Mao, Marc'Aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, and Andrew Y. Ng. Large scale distributed deep networks. In *Proceedings of the International Conference on Neural Information Processing Systems*, pages 1223–1231, 2012.

[151] Ofer Dekel, Ran Gilad-Bachrach, Ohad Shamir, and Lin Xiao. Optimal distributed online prediction using mini-batches. *J. Mach. Learn. Res.*, 13(1), January 2012.

[152] Ilias Diakonikolas, Gautam Kamath, Daniel Kane, Jerry Li, Jacob Steinhardt, and Alistair Stewart. Sever: A robust meta-algorithm for stochastic optimization. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1596–1606, Long Beach, California, USA, 09–15 Jun 2019. PMLR. URL http://proceedings.mlr.press/v97/diakonikolas19a.html.

[153] Mario Diaz, Peter Kairouz, Jiachun Liao, and Lalitha Sankar. Theoretical guarantees for model auditing with finite adversaries. *arXiv preprint arXiv:1911.03405*, 2019.

[154] Differential Privacy Team. Learning with privacy at scale. *Apple Machine Learning Journal*, 1(8), 2017. URL https://machinelearning.apple.com/2017/12/06/learning-with-privacy-at-scale.html.

[155] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. Collecting telemetry data privately. In *Advances in Neural Information Processing Systems 30*, December 2017. URL https://www.microsoft.com/en-us/research/publication/collecting-telemetry-data-privately/.

[156] Zeyu Ding, Yuxin Wang, Guanhong Wang, Danfeng Zhang, and Daniel Kifer. Detecting violations of differential privacy. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, pages 475–489, New York, NY, USA, 2018. ACM. ISBN 978-1-4503-5693-0. doi: 10.1145/3243734.3243818. URL http://doi.acm.org/10.1145/3243734.3243818.

[157] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. Technical report, Naval Research Lab Washington DC, 2004.

[158] Canh T. Dinh, Nguyen H. Tran, and Tuan Dung Nguyen. Personalized Federated Learning with Moreau Envelopes. In *NeurIPS*, 2020.

[159] Rafael G. L. D'Oliveira and S. E. Rouayheb. Lifting private information retrieval from two to any number of messages. In *2018 IEEE International Symposium on Information Theory (ISIT)*, pages 1744–1748, June 2018. doi: 10.1109/ISIT.2018.8437805.

[160] John R. Douceur. The sybil attack. In *Revised Papers from the First International Workshop on Peer-to-Peer Systems*, IPTPS '01, pages 251–260, London, UK, UK, 2002. Springer-Verlag. ISBN 3-540-44179-4. URL http://dl.acm.org/citation.cfm?id=646334.687813.

[161] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

[162] John C Duchi, Michael I Jordan, and Martin J Wainwright. Local privacy and statistical minimax rates. In *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*, pages 429–438. IEEE, 2013.

[163] Sanghamitra Dutta, Gauri Joshi, Soumyadip Ghosh, Parijat Dube, and Priya Nagpurkar. Slow and Stale Gradients Can Win the Race: Error-Runtime Trade-offs in Distributed SGD. *International Conference on Artificial Intelligence and Statistics (AISTATS)*, April 2018. URL https://arxiv.org/abs/1803.01113.

[164] Cynthia Dwork. Differential privacy: A survey of results. In *International Conference on Theory and Applications of Models of Computation*, pages 1–19. Springer, 2008.

[165] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3–4):211–407, 2014.

[166] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 486–503. Springer, 2006.

[167] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. Calibrating noise to sensitivity in private data analysis. In *IACR Theory of Cryptography Conference (TCC), New York, New York*, volume 3876 of *Lecture Notes in Computer Science*, pages 265–284. Springer-Verlag, 2006. doi: 10.1007/11681878_14.

[168] Cynthia Dwork, Guy N. Rothblum, and Salil Vadhan. Boosting and differential privacy. In *Proceedings of the IEEE 51st Annual Symposium on Foundations of Computer Science*, FOCS '10, pages 51–60, 2010.

[169] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pages 214–226. ACM, 2012.

[170] Laurel Eckhouse, Kristian Lum, Cynthia Conti-Cook, and Julie Ciccolini. Layers of bias: A unified approach for understanding problems with risk assessment. *Criminal Justice and Behavior*, 46(2):185–209, 2019.

[171] Hubert Eichner, Tomer Koren, H. Brendan McMahan, Nathan Srebro, and Kunal Talwar. Semi-cyclic stochastic gradient descent. In *Accepted to ICML 2019.*, 2019. URL https://arxiv.org/abs/1904.10120.

[172] Karim Eldefrawy, Gene Tsudik, Aurélien Francillon, and Daniele Perito. SMART: secure and minimal architecture for (establishing dynamic) root of trust. In *NDSS*. The Internet Society, 2012.

[173] Anis Elgabli, Jihong Park, Amrit S Bedi, Mehdi Bennis, and Vaneet Aggarwal. GADMM: Fast and communication efficient framework for distributed machine learning. *arXiv preprint arXiv:1909.00047*, 2019.

[174] Anis Elgabli, Jihong Park, Chaouki Ben Issaid, and Mehdi Bennis. Harnessing wireless channels for scalable and privacy-preserving federated learning, 2020.

[175] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Efficient multi-objective neural architecture search via Lamarckian evolution. *arXiv preprint arXiv:1804.09081*, 2018.

[176] Logan Engstrom, Brandon Tran, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. A rotation and a translation suffice: Fooling CNNs with simple transformations. *arXiv preprint arXiv:1712.02779*, 2017.

[177] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. RAPPOR: Randomized aggregatable privacy-preserving ordinal response. In *ACM CCS*, 2014. ISBN 978-1-4503-2957-6. doi: 10.1145/2660267.2660348. URL http://doi.acm.org/10.1145/2660267.2660348.

[178] Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Abhradeep Thakurta. Amplification by shuffling: From local to central differential privacy via anonymity. In *SODA*, pages 2468–2479, 2019.

[179] EU CORDIS. Machine learning ledger orchestration for drug discovery, 2019. URL https://cordis.europa.eu/project/rcn/223634/factsheet/en?WT.mc_id=RSS-Feed&WT.rss_f=project&WT.rss_a=223634&WT.rss_ev=a. Retrieved Aug 2019.

[180] Stefan Falkner, Aaron Klein, and Frank Hutter. BOHB: Robust and efficient hyperparameter optimization at scale. *arXiv preprint arXiv:1807.01774*, 2018.

[181] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning: A meta-learning approach. *arXiv preprint arXiv:2002.07948*, 2020.

[182] Junfeng Fan and Frederik Vercauteren. Somewhat practical fully homomorphic encryption. *IACR Cryptology ePrint Archive*, 2012:144, 2012.

[183] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. Local model poisoning attacks to Byzantine-robust federated learning. *arXiv preprint arXiv:1911.11815*, 2019.

[184] FeatureCloud. FeatureCloud: Our vision, 2019. URL https://featurecloud.eu/about/our-vision/. Retrieved Aug 2019.

[185] Vitaly Feldman, Ilya Mironov, Kunal Talwar, and Abhradeep Thakurta. Privacy amplification by iteration. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 521–532. IEEE, 2018.

[186] Clément Feutry, Pablo Piantanida, Yoshua Bengio, and Pierre Duhamel. Learning anonymized representations with adversarial neural networks. *CoRR*, abs/1802.09386, 2018. URL http://arxiv.org/abs/1802.09386.

[187] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, 2017.

[188] Aurélien Francillon, Quan Nguyen, Kasper Bonne Rasmussen, and Gene Tsudik. A minimalist approach to remote attestation. In *DATE*, pages 1–6. European Design and Automation Association, 2014.

[189] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1322–1333. ACM, 2015.

[190] Clement Fung, Chris JM Yoon, and Ivan Beschastnikh. Mitigating sybils in federated learning poisoning. *arXiv preprint arXiv:1808.04866*, 2018.

[191] Jun Furukawa, Yehuda Lindell, Ariel Nof, and Or Weinstein. High-throughput secure three-party computation for malicious adversaries and an honest majority. In *EUROCRYPT (2)*, volume 10211 of *Lecture Notes in Computer Science*, pages 225–255, 2017.

[192] Adam Gaier and David Ha. Weight agnostic neural networks. *arXiv preprint arXiv:1906.04358*, 2019.

[193] Venkata Gandikota, Raj Kumar Maity, and Arya Mazumdar. vqSGD: Vector quantized stochastic gradient descent. *arXiv preprint arXiv:1911.07971*, 2019.

[194] Adrià Gascón, Phillipp Schoppmann, Borja Balle, Mariana Raykova, Jack Doerner, Samee Zahur, and David Evans. Privacy-preserving distributed linear regression on high-dimensional data. *PoPETs*, 2017(4):345–364, 2017.

[195] Rosario Gennaro, Craig Gentry, and Bryan Parno. Non-interactive verifiable computing: Outsourcing computation to untrusted workers. In *CRYPTO*, volume 6223 of *Lecture Notes in Computer Science*, pages 465–482. Springer, 2010.

[196] Rosario Gennaro, Craig Gentry, Bryan Parno, and Mariana Raykova. Quadratic span programs and succinct NIZKs without PCPs. In *EUROCRYPT*, volume 7881 of *Lecture Notes in Computer Science*, pages 626–645. Springer, 2013.

[197] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 169–178, 2009.

[198] Craig Gentry and Shai Halevi. Compressible FHE with applications to PIR. In *TCC (2)*, volume 11892 of *Lecture Notes in Computer Science*, pages 438–464. Springer, 2019.

[199] Robin C. Geyer, Tassilo Klein, and Moin Nabi. Differentially private federated learning: A client level perspective. *CoRR*, abs/1712.07557, 2017. URL http://arxiv.org/abs/1712.07557.

[200] Badih Ghazi, Noah Golowich, Ravi Kumar, Rasmus Pagh, and Ameya Velingker. On the power of multiple anonymous messages. *arXiv:1908.11358*, 2019.

[201] Badih Ghazi, Rasmus Pagh, and Ameya Velingker. Scalable and differentially private distributed aggregation in the shuffled model. *arXiv preprint arXiv:1906.08320*, 2019.

[202] Badih Ghazi, Noah Golowich, Ravi Kumar, Pasin Manurangsi, Rasmus Pagh, and Ameya Velingker. Pure differentially private summation from anonymous messages. In *ITC*, pages 15:1–15:23, 2020.

[203] Badih Ghazi, Ravi Kumar, Pasin Manurangsi, and Rasmus Pagh. Private counting from anonymous messages: Near-optimal accuracy with vanishing communication overhead. In *ICML*, 2020.

[204] Badih Ghazi, Pasin Manurangsi, Rasmus Pagh, and Ameya Velingker. Private aggregation from fewer anonymous messages. In *EUROCRYPT*, pages 798–827, 2020.

[205] Arpita Ghosh, Tim Roughgarden, and Mukund Sundararajan. Universally utility-maximizing privacy mechanisms. In *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*, STOC '09, pages 351–360, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-506-2. doi: 10.1145/1536414.1536464. URL http://doi.acm.org/10.1145/1536414.1536464.

[206] Ran Gilad-Bachrach, Nathan Dowlin, Kim Laine, Kristin E. Lauter, Michael Naehrig, and John Wernsing. CryptoNets: Applying neural networks to encrypted data with high throughput and accuracy. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, pages 201–210, 2016. URL http://proceedings.mlr.press/v48/gilad-bachrach16.html.

[207] Antonious M Girgis, Deepesh Data, Suhas Diggavi, Peter Kairouz, and Ananda Theertha Suresh. Shuffled model of federated learning: Privacy, communication and accuracy trade-offs. *arXiv preprint arXiv:2008.07180*, 2020.

[208] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, STOC '87, pages 218–229, New York, NY, USA, 1987. ACM. ISBN 0-89791-221-7. doi: 10.1145/28395.28420. URL http://doi.acm.org/10.1145/28395.28420.

[209] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, 1989.

[210] Shafi Goldwasser, Yael Tauman Kalai, and Guy N. Rothblum. Delegating computation: interactive proofs for muggles. In *STOC*, pages 113–122. ACM, 2008.

[211] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6572.

[212] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *ICLR*, 2015.

[213] Slawomir Goryczka and Li Xiong. A comprehensive comparison of multiparty secure additions with differential privacy. *IEEE Trans. Dependable Sec. Comput.*, 14(5):463–477, 2017. doi: 10.1109/TDSC.2015.2484326. URL https://doi.org/10.1109/TDSC.2015.2484326.

[214] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. BadNets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017.

[215] Otkrist Gupta and Ramesh Raskar. Distributed learning of deep neural network over multiple agents. *Journal of Network and Computer Applications*, 116:1–8, 2018.

[216] Farzin Haddadpour, Mohammad Mahdi Kamani, Mehrdad Mahdavi, and Viveck R Cadambe. Local SGD with periodic averaging: Tighter analysis and adaptive synchronization. *arXiv preprint arXiv:1910.13598*, 2019.

[217] Andreas Haeberlen, Benjamin C Pierce, and Arjun Narayan. Differential privacy under fire. In *USENIX Security Symposium*, 2011.

[218] Shai Halevi, Yehuda Lindell, and Benny Pinkas. Secure computation on the web: Computing without simultaneous interaction. In *Annual Cryptology Conference*, pages 132–150. Springer, 2011.

[219] Jenny Hamer, Mehryar Mohri, and Ananda Theertha Suresh. Fedboost: A communication-efficient algorithm for federated learning. In *International Conference on Machine Learning*, pages 3973–3983. PMLR, 2020.

[220] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

[221] Yanjun Han, Ayfer Özgür, and Tsachy Weissman. Geometric lower bounds for distributed parameter estimation under communication constraints. In *Proceedings of Machine Learning Research*, pages 1–26, 75, 2018.

[222] Andrew Hard, Kanishka Rao, Rajiv Mathews, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. Federated learning for mobile keyboard prediction. *arXiv preprint 1811.03604*, 2018.

[223] Moritz Hardt, Eric Price, and Nathan Srebro. Equality of opportunity in supervised learning. In *Advances in Neural Information Processing Systems*, 2016.

[224] Stephen Hardy, Wilko Henecka, Hamish Ivey-Law, Richard Nock, Giorgio Patrini, Guillaume Smith, and Brian Thorne. Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption. *arXiv preprint arXiv:1711.10677*, 2017.

[225] Tatsunori Hashimoto, Megha Srivastava, Hongseok Namkoong, and Percy Liang. Fairness without demographics in repeated loss minimization. In *International Conference on Machine Learning*, pages 1934–1943, 2018.

[226] Chaoyang He, Conghui Tan, Hanlin Tang, Shuang Qiu, and Ji Liu. Central server free federated learning over single-sided trust social networks. *arXiv preprint arXiv:1910.04956*, 2019.

[227] Chaoyang He, Murali Annavaram, and Salman Avestimehr. Group knowledge transfer: Federated learning of large cnns at the edge. In *Advances in Neural Information Processing Systems 34*, 2020.

[228] Chaoyang He, Murali Annavaram, and Salman Avestimehr. Fednas: Federated deep learning via neural architecture search. 2020.

[229] Chaoyang He, Songze Li, Jinhyun So, Xiao Zeng, Mi Zhang, Hongyi Wang, Xiaoyang Wang, Praneeth Vepakomma, Abhishek Singh, Hang Qiu, Xinghua Zhu, Jianzong Wang, Li Shen, Peilin Zhao, Yan Kang, Yang Liu, Ramesh Raskar, Qiang Yang, Murali Annavaram, and Salman Avestimehr. Fedml: A research library and benchmark for federated machine learning, 2020.

[230] Chaoyang He, Haishan Ye, Li Shen, and Tong Zhang. Milenas: Efficient neural architecture search via mixed-level reformulation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[231] Lie He, An Bian, and Martin Jaggi. COLA: Decentralized linear learning. In *NeurIPS 2018 - Advances in Neural Information Processing Systems 31*, 2018.

[232] Úrsula Hébert-Johnson, Michael Kim, Omer Reingold, and Guy Rothblum. Multicalibration: Calibration for the (computationally-identifiable) masses. In *International Conference on Machine Learning*, pages 1944–1953, 2018.

[233] HElib. HElib. https://github.com/homenc/HElib, October 2019.

[234] Judy Hoffman, Mehryar Mohri, and Ningshan Zhang. Algorithms and theory for multiple-source adaptation. In *Advances in Neural Information Processing Systems*, pages 8246–8256, 2018.

[235] Samuel Horvath, Chen-Yu Ho, Ludovit Horvath, Atal Narayan Sahu, Marco Canini, and Peter Richtarik. Natural compression for distributed deep learning. *arXiv preprint arXiv:1905.10988*, 2019.

[236] Kevin Hsieh, Amar Phanishayee, Onur Mutlu, and Phillip B. Gibbons. The non-IID data quagmire of decentralized machine learning, 2019. URL https://arxiv.org/abs/1910.00189.

[237] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019.

[238] Yaochen Hu, Peng Liu, Linglong Kong, and Di Niu. Learning privately over distributed features: An admm sharing approach, 2019.

[239] Zhenqi Huang, Sayan Mitra, and Nitin Vaidya. Differentially Private Distributed Optimization. In *ICDCN*, 2015.

[240] Zhouyuan Huo, Bin Gu, and Heng Huang. Training neural networks using features replay. In *Advances in Neural Information Processing Systems*, pages 6659–6668, 2018.

[241] R Intel. Architecture instruction set extensions programming reference. *Intel Corporation, Feb*, 2012.

[242] Mihaela Ion, Ben Kreuter, Erhan Nergiz, Sarvar Patel, Shobhit Saxena, Karn Seth, David Shanahan, and Moti Yung. Private intersection-sum protocol with applications to attributing aggregate ad conversions. *IACR Cryptology ePrint Archive*, 2017: 738, 2017.

[243] Mihaela Ion, Ben Kreuter, Ahmet Erhan Nergiz, Sarvar Patel, Mariana Raykova, Shobhit Saxena, Karn Seth, David Shanahan, and Moti Yung. On deploying secure computing commercially: Private intersection-sum protocols and their business applications. *IACR Cryptology ePrint Archive*, 2019:723, 2019.

[244] Yuval Ishai, Joe Kilian, Kobbi Nissim, and Erez Petrank. Extending oblivious transfers efficiently. In *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 145–161. Springer, 2003.

[245] Max Jaderberg, Wojciech Marian Czarnecki, Simon Osindero, Oriol Vinyals, Alex Graves, David Silver, and Koray Kavukcuoglu. Decoupled neural interfaces using synthetic gradients. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1627–1635. JMLR. org, 2017.

[246] Matthew Jagielski, Michael J. Kearns, Jieming Mao, Alina Oprea, Aaron Roth, Saeed Sharifi-Malvajerdi, and Jonathan Ullman. Differentially private fair learning. *CoRR*, abs/1812.02696, 2018. URL http://arxiv.org/abs/1812.02696.

[247] Matthew Jagielski, Jonathan Ullman, and Alina Oprea. Auditing differentially private machine learning: How private is private sgd? *Advances in Neural Information Processing Systems*, 33, 2020.

[248] Eunjeong Jeong, Seungeun Oh, Hyesung Kim, Jihong Park, Mehdi Bennis, and Seong-Lyun Kim. Communication-efficient on-device machine learning: Federated distillation and augmentation under non-IID private data. *CoRR*, abs/1811.11479, 2018. URL http://arxiv.org/abs/1811.11479.

[249] Zhuqing Jia and Syed Ali Jafar. On the capacity of secure distributed matrix multiplication. *ArXiv*, abs/1908.06957, 2019.

[250] Yihan Jiang, Jakub Konečný, Keith Rush, and Sreeram Kannan. Improving federated learning personalization via model agnostic meta learning. *arXiv preprint arXiv:1909.12488*, 2019.

[251] S. Kadhe, B. Garcia, A. Heidarzadeh, S. E. Rouayheb, and A. Sprintson. Private information retrieval with side information: The single server case. In *2017 55th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1099–1106, Oct 2017. doi: 10.1109/ALLERTON.2017.8262860.

[252] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. Extremal mechanisms for local differential privacy. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2879–2887. Curran Associates, Inc., 2014.

[253] Peter Kairouz, K. A. Bonawitz, and Daniel Ramage. Discrete distribution estimation under local privacy. In *International Conference on Machine Learning*, pages 2436–2444, 2016.

[254] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. The composition theorem for differential privacy. *IEEE Transactions on Information Theory*, 63(6):4037–4049, 2017.

[255] Peter Kairouz, Jiachun Liao, Chong Huang, and Lalitha Sankar. Censored and fair universal representations using generative adversarial models. *arXiv preprint arXiv:1910.00411*, 2020.

[256] Peter Kairouz, Ziyu Liu, and Thomas Steinke. The distributed discrete gaussian mechanism for federated learning with secure aggregation, 2021.

[257] Peter Kairouz, Brendan McMahan, Shuang Song, Om Thakkar, Abhradeep Thakurta, and Zheng Xu. Practical and private (deep) learning without sampling or shuffling, 2021.

[258] Toshihiro Kamishima, Shotaro Akaho, and Jun Sakuma. Fairness-aware learning through regularization approach. In *2011 IEEE 11th International Conference on Data Mining Workshops*, pages 643–650. IEEE, 2011.

[259] Daniel Kang, Yi Sun, Dan Hendrycks, Tom Brown, and Jacob Steinhardt. Testing robustness against unforeseen adversaries. *arXiv preprint arXiv:1908.08016*, 2019.

[260] Jiawen Kang, Zehui Xiong, Dusit Niyato, Shengli Xie, and Junshan Zhang. Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory. *IEEE Internet of Things Journal*, 2019.

[261] Jiawen Kang, Zehui Xiong, Dusit Niyato, Han Yu, Ying-Chang Liang, and Dong In Kim. Incentive design for efficient federated learning in mobile networks: A contract theory approach. In *IEEE VTS Asia Pacific Wireless Communications Symposium, APWCS 2019, Singapore, August 28-30, 2019*, pages 1–5, 2019.

[262] Hamed Karimi, Julie Nutini, and Mark Schmidt. Linear convergence of gradient and proximal-gradient methods under the Polyak-łojasiewicz condition. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 795–811. Springer, 2016.

[263] Sai Praneeth Karimireddy, Quentin Rebjock, Sebastian Stich, and Martin Jaggi. Error feedback fixes SignSGD and other gradient compression schemes. In *ICML*, 2019.

[264] Sai Praneeth Karimireddy, Martin Jaggi, Satyen Kale, Mehryar Mohri, Sashank J Reddi, Sebastian U Stich, and Ananda Theertha Suresh. Mime: Mimicking centralized stochastic algorithms in federated learning. *arXiv preprint arXiv:2008.03606*, 2020.

[265] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, pages 5132–5143. PMLR, 2020.

[266] Shiva Prasad Kasiviswanathan, Homin K. Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam D. Smith. What can we learn privately? *SIAM J. Comput.*, 40(3):793–826, 2011. URL https://doi.org/10.1137/090756090.

[267] Michael J. Kearns, Aaron Roth, Zhiwei Steven Wu, and Grigory Yaroslavtsev. Privacy for the protected (only). *CoRR*, abs/1506.00242, 2015. URL http://arxiv.org/abs/1506.00242.

[268] Ahmed Khaled, Konstantin Mishchenko, and Peter Richtárik. First analysis of local GD on heterogeneous data, 2019. URL https://arxiv.org/abs/1909.04715.

[269] Ahmed Khaled, Konstantin Mishchenko, and Peter Richtárik. Better communication complexity for local SGD, 2019. URL https://arxiv.org/abs/1909.04746.

[270] Mikhail Khodak, Maria-Florina Balcan, and Ameet Talwalkar. Adaptive gradient-based meta-learning methods. In *Advances in Neural Information Processing Systems*, 2019.

[271] Daniel Kifer and Ashwin Machanavajjhala. Pufferfish: A framework for mathematical privacy definitions. *ACM Transactions on Database Systems*, 39(1):3:1–3:36, 2014.

[272] Yejin Kim, Jimeng Sun, Hwanjo Yu, and Xiaoqian Jiang. Federated tensor factorization for computational phenotyping. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*, pages 887–895, 2017. doi: 10.1145/3097983.3098118. URL https://doi.org/10.1145/3097983.3098118.

[273] Ross D. King, Cao Feng, and Alistair Sutherland. StatLog: comparison of classification algorithms on large real-world problems. *Applied Artificial Intelligence an International Journal*, 9(3):289–333, 1995.

[274] Patrick Koeberl, Steffen Schulz, Ahmad-Reza Sadeghi, and Vijay Varadharajan. TrustLite: a security architecture for tiny embedded devices. In *EuroSys*, pages 10:1–10:14. ACM, 2014.

[275] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1885–1894. JMLR. org, 2017.

[276] Pang Wei Koh, Jacob Steinhardt, and Percy Liang. Stronger data poisoning attacks break data sanitization defenses. *arXiv preprint arXiv:1811.00741*, 2018.

[277] Ron Kohavi and George H John. Automatic parameter selection by minimizing estimated error. In *Machine Learning Proceedings 1995*, pages 304–312. Elsevier, 1995.

[278] Anastasia Koloskova, Sebastian U Stich, and Martin Jaggi. Decentralized Stochastic Optimization and Gossip Algorithms with Compressed Communication. In *ICML*, 2019.

[279] Anastasia Koloskova, Tao Lin, Sebastian U Stich, and Martin Jaggi. Decentralized deep learning with arbitrary communication compression. *International Conference on Learning Representations (ICLR)*, 2020.

[280] Anastasia Koloskova, Nicolas Loizou, Sadra Boreiri, Martin Jaggi, and Sebastian U. Stich. A Unified Theory of Decentralized SGD with Changing Topology and Local Updates. In *ICML*, 2020.

[281] Jakub Konečný and Peter Richtárik. Randomized distributed mean estimation: Accuracy vs communication. *Frontiers in Applied Mathematics and Statistics*, 4:62, 2018.

[282] Jakub Konečný, H Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[283] Satya Kuppam, Ryan McKenna, David Pujol, Michael Hay, Ashwin Machanavajjhala, and Gerome Miklau. Fair decision making using privacy-protected data. *CoRR*, abs/1905.12744, 2019. URL http://arxiv.org/abs/1905.12744.

[284] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

[285] Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge University Press, New York, NY, USA, 1997. ISBN 0-521-56067-5.

[286] Eyal Kushilevitz and Rafail Ostrovsky. Replication is not needed: Single database, computationally-private information retrieval. In *In Proc. of the 38th Annu. IEEE Symp. on Foundations of Computer Science*, pages 364–373, 1997.

[287] Matt J Kusner, Joshua Loftus, Chris Russell, and Ricardo Silva. Counterfactual fairness. In *Advances in Neural Information Processing Systems*, pages 4066–4076, 2017.

[288] Albert Kwon, David Lazar, Srinivas Devadas, and Bryan Ford. Riffle. *Proceedings on Privacy Enhancing Technologies*, 2016(2):115–134, 2016.

[289] Yassine Laguel, Krishna Pillutla, Jérôme Malick, and Zaid Harchaoui. Device Heterogeneity in Federated Learning: A Superquantile Approach. *arXiv preprint arXiv:2002.11223*, 2020.

[290] Brenden M. Lake, Ruslan Salakhutdinov, Jason Gross, and Joshua B. Tenenbaum. One shot learning of simple visual concepts. In *Proceedings of the Conference of the Cognitive Science Society (CogSci)*, 2017.

[291] Anusha Lalitha, Osman Cihan Kilinc, Tara Javidi, and Farinaz Koushanfar. Peer-to-peer Federated Learning on Graphs. Technical report, arXiv:1901.11173, 2019.

[292] Anusha Lalitha, Xinghan Wang, Osman Kilinc, Yongxi Lu, Tara Javidi, and Farinaz Koushanfar. Decentralized Bayesian learning over graphs. *arXiv preprint: 1905.10466*, 2019.

[293] Leslie Lamport, Robert Shostak, and Marshall Pease. The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 4(3):382–401, 1982.

[294] Guanghui Lan. An optimal method for stochastic composite optimization. *Mathematical Programming*, 133(1): 365–397, Jun 2012. ISSN 1436-4646. doi: 10.1007/s10107-010-0434-y. URL https://doi.org/10.1007/s10107-010-0434-y.

[295] Andrei Lapets, Nikolaj Volgushev, Azer Bestavros, Frederick Jansen, and Mayank Varia. Secure MPC for analytics as a web application. In *SecDev*, pages 73–74. IEEE Computer Society, 2016.

[296] Mathias Lécuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. Certified robustness to adversarial examples with differential privacy. In *2019 IEEE Symposium on Security and Privacy, SP 2019, San Francisco, CA, USA, May 19-23, 2019*, pages 656–672, 2019. doi: 10.1109/SP.2019.00044. URL https://doi.org/10.1109/SP.2019.00044.

[297] Tancrède Lepoint, Sarvar Patel, Mariana Raykova, Karn Seth, and Ni Trieu. Private join and compute from PIR with default. *IACR Cryptol. ePrint Arch.*, 2020:1011, 2020.

[298] David Leroy, Alice Coucke, Thibaut Lavril, Thibault Gisselbrecht, and Joseph Dureau. Federated learning for keyword spotting. *arXiv preprint arXiv:1810.05512*, 2018.

[299] Jeffrey Li, Mikhail Khodak, Sebastian Caldas, and Ameet Talwalkar. Differentially private meta-learning. *arXiv preprint arXiv:1909.05830*, 2019.

[300] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks, 2018. URL https://arxiv.org/abs/1812.06127.

[301] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions, 2019.

[302] Tian Li, Maziar Sanjabi, and Virginia Smith. Fair resource allocation in federated learning. *arXiv preprint arXiv:1905.10497*, 2019.

[303] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of FedAvg on non-IID data. *arXiv preprint arXiv:1907.02189*, 2019.

[304] Xiang Li, Wenhao Yang, Shusen Wang, and Zhihua Zhang. Communication efficient decentralized training with multiple local updates. *arXiv preprint arXiv:1910.09126*, 2019.

[305] Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. Can Decentralized Algorithms Outperform Centralized Algorithms? A Case Study for Decentralized Parallel Stochastic Gradient Descent. In *NIPS*, 2017.

[306] Xiangru Lian, Wei Zhang, Ce Zhang, and Ji Liu. Asynchronous Decentralized Parallel Stochastic Gradient Descent. In *ICML*, 2018.

[307] libsnark. libsnark: a c++ library for zkSNARK proofs. https://github.com/scipr-lab/libsnark, December 2019.

[308] David Lie and Petros Maniatis. Glimmers: Resolving the privacy/trust quagmire. In *Proceedings of the 16th Workshop on Hot Topics in Operating Systems*, pages 94–99. ACM, 2017.

[309] Darryl Lin, Sachin Talathi, and Sreekanth Annapureddy. Fixed point quantization of deep convolutional networks. In *International Conference on Machine Learning*, pages 2849–2858, 2016.

[310] Tao Lin, Sebastian U Stich, and Martin Jaggi. Don't use large mini-batches, use local SGD. *International Conference on Learning Representations (ICLR)*, 2020.

[311] Yujun Lin, Song Han, Huizi Mao, Yu Wang, and William J Dally. Deep gradient compression: Reducing the communication bandwidth for distributed training. *arXiv preprint arXiv:1712.01887*, 2017.

[312] R. J. A. Little. Post-stratification: A modeler's perspective. *Journal of the American Statistical Association*, 88(423): 1001–1012, 1993. ISSN 01621459.

[313] Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018.

[314] Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *International Symposium on Research in Attacks, Intrusions, and Defenses*, pages 273–294. Springer, 2018.

[315] Xiyang Liu and Sewoong Oh. Minimax rates of estimating approximate differential privacy. *arXiv preprint arXiv:1905.10335*, 2019.

[316] Yang Liu, Yan Kang, Xinwei Zhang, Liping Li, Yong Cheng, Tianjian Chen, Mingyi Hong, and Qiang Yang. A communication efficient vertical federated learning framework. *CoRR*, abs/1912.11187, 2019. URL http://arxiv.org/abs/1912.11187.

[317] Yang Liu, Yan Kang, Chaoping Xing, Tianjian Chen, and Qiang Yang. A secure federated transfer learning framework. *IEEE Intelligent Systems*, 35(4):70–82, 2020. doi: 10.1109/MIS.2020.2988525.

[318] Yang Liu, Zhihao Yi, and Tianjian Chen. Backdoor attacks and defenses in feature-partitioned collaborative learning, 2020.

[319] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. Trojaning attack on neural networks. In *25th Annual Network and Distributed System Security Symposium, NDSS 2018, San Diego, California, USA, February 18-21, 2018*, 2018. URL http://wp.internetsociety.org/ndss/wp-content/uploads/sites/25/2018/02/ndss2018_03A-5_Liu_paper.pdf.

[320] Yuhan Liu, Ananda Theertha Suresh, Felix Xinnan X Yu, Sanjiv Kumar, and Michael Riley. Learning discrete distributions: user vs item-level privacy. *Advances in Neural Information Processing Systems*, 33, 2020.

[321] Heiko Ludwig, Nathalie Baracaldo, Gegi Thomas, Yi Zhou, Ali Anwar, Shashank Rajamoni, Yuya Ong, Jayaram Radhakrishnan, Ashish Verma, Mathieu Sinn, et al. IBM federated learning: An enterprise framework white paper V0.1. *arXiv preprint arXiv:2007.10987*, 2020.

[322] Jiahuan Luo, Xueyang Wu, Yun Luo, Anbu Huang, Yunfeng Huang, Yang Liu, and Qiang Yang. Real-world image datasets for federated learning. *arXiv preprint arXiv:1910.11089*, 2019.

[323] Renqian Luo, Fei Tian, Tao Qin, Enhong Chen, and Tie-Yan Liu. Neural architecture optimization. In *Advances in neural information processing systems*, pages 7816–7827, 2018.

[324] Lingjuan Lyu, Jiangshan Yu, Karthik Nandakumar, Yitong Li, Xingjun Ma, Jiong Jin, Han Yu, and Kee Siong Ng. Towards fair and privacy-preserving federated deep models. *IEEE Transactions on Parallel and Distributed Systems*, 31(11):2524–2541, 2020.

[325] Jing Ma, Qiuchen Zhang, Jian Lou, Joyce Ho, Li Xiong, and Xiaoqian Jiang. Privacy-preserving tensor factorization for collaborative health data analysis. In *ACM CIKM*, volume 2, 2019.

[326] Yuzhe Ma, Xiaojin Zhu, and Justin Hsu. Data poisoning against differentially-private learners: Attacks and defenses. In *International Joint Conference on Artificial Intelligence (IJCAI), Macao, China*, 2019. URL https://arxiv.org/abs/1903.09860.

[327] David Madras, Elliot Creager, Toniann Pitassi, and Richard Zemel. Learning adversarially fair and transferable representations. In *ICML*, 2018.

[328] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *ICLR*, 2017.

[329] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation: Learning bounds and algorithms. *arXiv preprint arXiv:0902.3430*, 2009.

[330] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation with multiple sources. In *Advances in neural information processing systems*, pages 1041–1048, 2009.

[331] Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three approaches for personalization with applications to federated learning. *arXiv preprint arXiv:2002.10619*, 2020.

[332] Yishay Mansour, Mehryar Mohri, Ananda Theertha Suresh, and Ke Wu. A theory of multiple-source adaptation with limited target labeled data. *arXiv preprint arXiv:2007.09762*, 2020.

[333] Alicia R Martin, Masahiro Kanai, Yoichiro Kamatani, Yukinori Okada, Benjamin M Neale, and Mark J Daly. Current clinical use of polygenic scores will risk exacerbating health disparities. *BioRxiv*, page 441261, 2019.

[334] H Brendan McMahan and Daniel Ramage. Federated learning: Collaborative machine learning without centralized training data, April 2017. URL https://ai.googleblog.com/2017/04/federated-learning-collaborative.html. Google AI Blog.

[335] H Brendan McMahan and Matthew Streeter. Adaptive bound optimization for online convex optimization. *arXiv preprint arXiv:1002.4908*, 2010.

[336] H Brendan McMahan, Galen Andrew, Ulfar Erlingsson, Steve Chien, Ilya Mironov, Nicolas Papernot, and Peter Kairouz. A general approach to adding differential privacy to iterative training procedures. dec 2018. *URL https://arxiv. org/abs*, 1812.

[337] H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, pages 1273–1282, 2017 (original version on arxiv Feb. 2016).

[338] H Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *International Conference on Learning Representations (ICLR)*, 2018.

[339] Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *FOCS*, pages 94–103, 2007.

[340] Shike Mei and Xiaojin Zhu. Using machine teaching to identify optimal training-set attacks on machine learners. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[341] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Exploiting unintended feature leakage in collaborative learning. *arXiv preprint arXiv:1805.04049*, 2018.

[342] El Mahdi El Mhamdi, Rachid Guerraoui, and Sébastien Rouault. The hidden vulnerability of distributed learning in Byzantium. In *ICML*, 2018.

[343] Silvio Micali. Computationally sound proofs. *SIAM J. Comput.*, 30(4):1253–1298, 2000.

[344] Fatemehsadat Mireshghallah, Mohammadkazem Taram, , Praneeth Vepakomma, Abhishek Singh, Ramesh Raskar, and Esmaeilzadeh Hadi. Privacy in deep learning: A survey. *arXiv preprint arXiv:2004.12254*, 2020.

[345] Ilya Mironov. On significance of the least significant bits for differential privacy. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 650–661. ACM, 2012.

[346] Ilya Mironov. Rényi differential privacy. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pages 263–275. IEEE, 2017.

[347] Ilya Mironov, Omkant Pandey, Omer Reingold, and Salil Vadhan. Computational differential privacy. In *Advances in Cryptology—CRYPTO*, pages 126–142, 2009.

[348] Ilya Mironov, Kunal Talwar, and Li Zhang. R\'enyi differential privacy of the sampled Gaussian mechanism. *arXiv preprint arXiv:1908.10530*, 2019.

[349] Shira Mitchell, Eric Potash, and Solon Barocas. Prediction-based decisions and fairness: A catalogue of choices, assumptions, and definitions. *arXiv preprint arXiv:1811.07867*, 2018.

[350] Volodymyr Mnih and Geoffrey E Hinton. Learning to label aerial images from noisy data. In *Proceedings of the 29th International conference on machine learning (ICML-12)*, pages 567–574, 2012.

[351] Payman Mohassel and Yupeng Zhang. SecureML: A system for scalable privacy-preserving machine learning. In *IEEE Symposium on Security and Privacy*, pages 19–38. IEEE Computer Society, 2017.

[352] Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. Agnostic Federated Learning. In *ICML*, 2019.

[353] Jose G. Moreno-Torres, Troy Raeder, RocíO Alaiz-RodríGuez, Nitesh V. Chawla, and Francisco Herrera. A unifying view on dataset shift in classification. *Pattern Recogn.*, 45(1), January 2012.

[354] Musketeer. Musketeer: About, 2019. URL http://musketeer.eu/project/. Retrieved Aug 2019.

[355] Carolina Naim, Fangwei Ye, and Salim El Rouayheb. ON-OFF privacy with correlated requests. In *2019 IEEE International Symposium on Information Theory (ISIT)*, July 2019.

[356] Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. Learning with noisy labels. In *Advances in neural information processing systems*, pages 1196–1204, 2013.

[357] Giovanni Neglia, Chuan Xu, Don Towsley, and Gianmarco Calbi. Decentralized gradient methods: does topology matter? In *AISTATS*, 2020.

[358] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

[359] Valeria Nikolaenko, Udi Weinsberg, Stratis Ioannidis, Marc Joye, Dan Boneh, and Nina Taft. Privacy-preserving ridge regression on hundreds of millions of records. In *IEEE Symposium on Security and Privacy*, pages 334–348. IEEE Computer Society, 2013.

[360] Chaoyue Niu, Fan Wu, Shaojie Tang, Lifeng Hua, Rongfei Jia, Chengfei Lv, Zhihua Wu, and Guihai Chen. Secure federated submodel learning. *arXiv preprint arXiv:1911.02254*, 2019.

[361] NSA. Defense in depth: A practical strategy for achieving Information Assurance in today's highly networked environments. Technical report, NSA, 2012.

[362] Deniz Oktay, Johannes Ballé, Saurabh Singh, and Abhinav Shrivastava. Model compression by entropy penalized reparameterization. *arXiv preprint arXiv:1906.06624*, 2019.

[363] Femi Olumofin and Ian Goldberg. Revisiting the computational practicality of private information retrieval. In *International Conference on Financial Cryptography and Data Security*, pages 158–172. Springer, 2011.

[364] Palisade. PALISADE lattice cryptography library. https://gitlab.com/palisade/palisade-release, October 2019.

[365] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

[366] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, pages 506–519. ACM, 2017.

[367] Nicolas Papernot, Abhradeep Thakurta, Shuang Song, Steve Chien, and Úlfar Erlingsson. Tempered sigmoid activations for deep learning with differential privacy. *arXiv preprint arXiv:2007.14191*, 2020.

[368] Jihong Park, Sumudu Samarakoon, Mehdi Bennis, and Mérouane Debbah. Wireless network intelligence at the edge. *CoRR*, abs/1812.02858, 2018. URL http://arxiv.org/abs/1812.02858.

[369] Bryan Parno, Jon Howell, Craig Gentry, and Mariana Raykova. Pinocchio: nearly practical verifiable computation. *Commun. ACM*, 59(2):103–112, 2016.

[370] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[371] Kumar Kshitij Patel and Aymeric Dieuleveut. Communication trade-offs for synchronized distributed SGD with large step size. *NeurIPS*, 2019.

[372] Sarvar Patel, Giuseppe Persiano, and Kevin Yeo. Private stateful information retrieval. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, pages 1002–1019, New York, NY, USA, 2018. ACM. ISBN 978-1-4503-5693-0. doi: 10.1145/3243734.3243821. URL http://doi.acm.org/10.1145/3243734.3243821.

[373] Giorgio Patrini, Richard Nock, Stephen Hardy, and Tibério S. Caetano. Fast learning from distributed datasets without entity matching. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 1909–1917, 2016. URL http://www.ijcai.org/Abstract/16/273.

[374] Fabian Pedregosa. Hyperparameter optimization with approximate gradient. *arXiv preprint arXiv:1602.02355*, 2016.

[375] Hieu Pham, Melody Guan, Barret Zoph, Quoc Le, and Jeff Dean. Efficient neural architecture search via parameter sharing. In *International Conference on Machine Learning*, pages 4092–4101, 2018.

[376] Sundar Pichai. Google's Sundar Pichai: Privacy Should Not Be a Luxury Good. *New York Times*, May 7, 2019.

[377] Venkatadheeraj Pichapati, Ananda Theertha Suresh, Felix X Yu, Sashank J Reddi, and Sanjiv Kumar. AdaCliP: Adaptive clipping for private SGD. *arXiv preprint arXiv:1908.07643*, 2019.

[378] Vasyl Pihur, Aleksandra Korolova, Frederick Liu, Subhash Sankuratripati, Moti Yung, Dachuan Huang, and Ruogu Zeng. Differentially-private "Draw and Discard" machine learning. *CoRR*, abs/1807.04369, 2018. URL http://arxiv.org/abs/1807.04369.

[379] Krishna Pillutla, Sham M Kakade, and Zaid Harchaoui. Robust aggregation for federated learning. *arXiv preprint arXiv:1912.13445*, 2019.

[380] Joaquin Quionero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D. Lawrence. *Dataset Shift in Machine Learning*. The MIT Press, 2009. ISBN 0262170051, 9780262170055.

[381] Shashank Rajput, Hongyi Wang, Zachary Charles, and Dimitris Papailiopoulos. DETOX: A redundancy-based framework for faster and more robust gradient aggregation. *arXiv preprint arXiv:1907.12205*, 2019.

[382] Daniel Ramage and Stefano Mazzocchi. Federated analytics: Collaborative data science without data collection, May 2020. URL https://ai.googleblog.com/2020/05/federated-analytics-collaborative-data.html. Google AI Blog.

[383] Swaroop Ramaswamy, Rajiv Mathews, Kanishka Rao, and Françoise Beaufays. Federated learning for emoji prediction in a mobile keyboard. *arXiv preprint 1906.04329*, 2019.

[384] Swaroop Ramaswamy, Om Thakkar, Rajiv Mathews, Galen Andrew, H Brendan McMahan, and Françoise Beaufays. Training production language models without memorizing user data. *arXiv preprint arXiv:2009.10031*, 2020.

[385] Vibhor Rastogi and Suman Nath. Differentially private aggregation of distributed time-series with transformation and encryption. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data*, SIGMOD '10, pages 735–746, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0032-2. doi: 10.1145/1807167.1807247. URL http://doi.acm.org/10.1145/1807167.1807247.

[386] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *Proceedings of the 5th International Conference on Learning Representations*, 2017.

[387] Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc V Le, and Alexey Kurakin. Large-scale evolution of image classifiers. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2902–2911. JMLR. org, 2017.

[388] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. Regularized evolution for image classifier architecture search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4780–4789, 2019.

[389] Sashank Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečnỳ, Sanjiv Kumar, and H Brendan McMahan. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*, 2020.

[390] Amirhossein Reisizadeh, Aryan Mokhtari, Hamed Hassani, Ali Jadbabaie, and Ramtin Pedarsani. Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization. *arXiv preprint arXiv:1909.13014*, 2019.

[391] Amirhossein Reisizadeh, Hossein Taheri, Aryan Mokhtari, Hamed Hassani, and Ramtin Pedarsani. Robust and communication-efficient collaborative learning. *arXiv:1907.10595*, 2019.

[392] Leonid Reyzin, Adam D. Smith, and Sophia Yakoubov. Turning HATE into LOVE: homomorphic ad hoc threshold encryption for scalable MPC. *IACR Cryptology ePrint Archive*, 2018:997, 2018.

[393] M Sadegh Riazi, Kim Laine, Blake Pelton, and Wei Dai. HEAX: High-performance architecture for computation on homomorphically encrypted data in the cloud. *arXiv preprint arXiv:1909.09731*, 2019.

[394] Rashida Richardson, Jason Schultz, and Kate Crawford. Dirty data, bad predictions: How civil rights violations impact police data, predictive policing systems, and justice. *New York University Law Review Online, Forthcoming*, 2019.

[395] Brian D Ripley. Statistical aspects of neural networks. *Networks and chaos—statistical and probabilistic aspects*, 50:40–123, 1993.

[396] Ronald L Rivest, Len Adleman, and Michael L Dertouzos. On data banks and privacy homomorphisms. *Foundations of Secure Computation, Academia Press*, pages 169–179, 1978.

[397] Nuria Rodríguez-Barroso, Goran Stipcich, Daniel Jiménez-López, José Antonio Ruiz-Millán, Eugenio Martínez-Cámara, Gerardo González-Seco, M Victoria Luzón, Miguel Angel Veganzones, and Francisco Herrera. Federated learning and differential privacy: Software tools analysis, the sherpa. ai fl framework and methodological guidelines for preserving data privacy. *Information Fusion*, 64:270–292, 2020.

[398] Edo Roth, Daniel Noble, Brett Hemenway Falk, and Andreas Haeberlen. Honeycrisp: large-scale differentially private aggregation without a trusted core. In *SOSP*, pages 196–210. ACM, 2019.

[399] Theo Ryffel, Andrew Trask, Morten Dahl, Bobby Wagner, Jason Mancuso, Daniel Rueckert, and Jonathan Passerat-Palmbach. A generic framework for privacy preserving deep learning, 2018.

[400] César Sabater, Aurélien Bellet, and Jan Ramon. Distributed Differentially Private Averaging with Improved Utility and Robustness to Malicious Parties. *arXiv preprint arXiv:2006.07218*, 2020.

[401] John K Salmon, Mark A Moraes, Ron O Dror, and David E Shaw. Parallel random numbers: As easy as 1, 2, 3. In *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, page 16. ACM, 2011.

[402] Sumudu Samarakoon, Mehdi Bennis, Walid Saad, and Mérouane Debbah. Federated learning for ultra-reliable low-latency V2V communications. *CoRR*, abs/1805.09253, 2018. URL http://arxiv.org/abs/1805.09253.

[403] Nithya Sambasivan, Garen Checkley, Amna Batool, Nova Ahmed, David Nemer, Laura Sanely Gaytán-Lugo, Tara Matthews, Sunny Consolvo, and Elizabeth Churchill. " privacy is not for me, it's for those rich women": Performative privacy practices on mobile phones by women in south asia. In *Fourteenth Symposium on Usable Privacy and Security ({SOUPS} 2018)*, pages 127–142, 2018.

[404] Sai Sri Sathya, Praneeth Vepakomma, Ramesh Raskar, Ranjan Ramachandra, and Santanu Bhattacharya. A review of homomorphic encryption libraries for secure computation. *arXiv preprint arXiv:1812.02428*, 2018.

[405] Felix Sattler, Simon Wiedemann, Klaus-Robert Müller, and Wojciech Samek. Robust and communication-efficient federated learning from non-IID data. *arXiv preprint arXiv:1903.02891*, 2019.

[406] R. Schnell. Efficient private record linkage of very large datasets. In *59^{th} World Statistics Congress*, 2013.

[407] R. Schnell, T. Bachteler, and J. Reiher. A novel error-tolerant anonymous linking code. Technical report, Paper No. WP-GRLC-2011-02, German Record Linkage Center Working Paper Series, 2011.

[408] Claus P. Schnorr. Efficient identification and signatures for smart cards. In *Proceedings of the Workshop on the Theory and Application of Cryptographic Techniques on Advances in Cryptology*, EUROCRYPT '89, 1990.

[409] SEAL. Microsoft SEAL (release 3.6). https://github.com/Microsoft/SEAL, November 2020. Microsoft Research, Redmond, WA.

[410] Frank Seide and Amit Agarwal. Cntk: Microsoft's open-source deep-learning toolkit. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 2135, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450342322. doi: 10.1145/2939672.2945397. URL https://doi.org/10.1145/2939672.2945397.

[411] Arvind Seshadri, Mark Luk, Adrian Perrig, Leendert van Doom, and Pradeep K. Khosla. Pioneer: Verifying code integrity and enforcing untampered code execution on legacy systems. In *Malware Detection*, volume 27 of *Advances in Information Security*, pages 253–289. Springer, 2007.

[412] Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free. *NeurIPS*, 2019.

[413] Vivek Sharma, Praneeth Vepakomma, Tristan Swedish, Ken Chang, Jayashree Kalpathy-Cramer, and Ramesh Raskar. ExpertMatcher: Automating ML model selection for clients using hidden representations. *arXiv preprint arXiv:1910.03731*, 2019.

[414] Yash Sharma and Pin-Yu Chen. Attacking the Madry defense model with l_1-based adversarial examples. *arXiv preprint arXiv:1710.10733*, 2017.

[415] SHELL. https://github.com/google/shell-encryption, December 2020. Google.

[416] Yanyao Shen and Sujay Sanghavi. Learning with bad training data via iterative trimmed loss minimization. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5739–5748, Long Beach, California, USA, 09–15 Jun 2019. PMLR. URL http://proceedings.mlr.press/v97/shen19e.html.

[417] Elaine Shi, HTH Chan, Eleanor Rieffel, Richard Chow, and Dawn Song. Privacy-preserving aggregation of time-series data. In *Annual Network & Distributed System Security Symposium (NDSS)*, 2011.

[418] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18. IEEE, 2017.

[419] Kumar Shridhar, Felix Laumann, and Marcus Liwicki. A comprehensive guide to Bayesian convolutional neural network with variational inference. *arXiv preprint: 1901.02731*, 2019.

[420] Daniel L. Silver, Qiang Yang, and Lianghao Li. Lifelong machine learning systems: Beyond learning algorithms. In *AAAI Spring Symposium Series*, 2013.

[421] Abhishek Singh, Praneeth Vepakomma, Otkrist Gupta, and Ramesh Raskar. Detailed comparison of communication efficiency of split learning and federated learning. *arXiv preprint arXiv:1909.09145*, 2019.

[422] Abhishek Singh, Ayush Chopra, Vivek Sharma, Ethan Garza, Emily Zhang, Praneeth Vepakomma, and Ramesh Raskar. DISCO: Dynamic and invariant sensitive channel obfuscation for deep neural networks. 2020.

[423] Radu Sion and Bogdan Carbunar. On the computational practicality of private information retrieval. In *Proceedings of the Network and Distributed Systems Security Symposium*, pages 2006–06. Internet Society, 2007.

[424] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S. Talwalkar. Federated Multi-Task Learning. In *NIPS*, 2017.

[425] Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, 2017.

[426] Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Mostofa Patwary, Mr Prabhat, and Ryan Adams. Scalable Bayesian optimization using deep neural networks. In *International conference on machine learning*, pages 2171–2180, 2015.

[427] Jinhyun So, Basak Guler, and A. Salman Avestimehr. Byzantine-resilient secure federated learning. *IEEE Journal on Selected Areas in Communication, Series on Machine Learning for Communications and Networks*, 2020.

[428] Jinhyun So, Basak Guler, and A Salman Avestimehr. Turbo-aggregate: Breaking the quadratic aggregation barrier in secure federated learning. *arXiv preprint arXiv:2002.04156*, 2020.

[429] Liwei Song, Reza Shokri, and Prateek Mittal. Privacy risks of securing machine learning models against adversarial examples. In *In Proceedings of the ACM Conference on Computer and Communication Security (CCS)*, 2019.

[430] K Srinathan and C Pandu Rangan. Efficient asynchronous secure multiparty distributed computation. In *International Conference on Cryptology in India*, pages 117–129. Springer, 2000.

[431] Brij Mohan Lal Srivastava, Aurélien Bellet, Marc Tommasi, and Emmanuel Vincent. Privacy-Preserving Adversarial Representation Learning in ASR: Reality or Illusion? In *Annual Conference of the International Speech Communication Association (Interspeech)*, 2019.

[432] Jacob Steinhardt, Pang Wei W Koh, and Percy S Liang. Certified defenses for data poisoning attacks. In *Advances in neural information processing systems*, pages 3517–3529, 2017.

[433] Thomas Steinke and Jonathan Ullman. Tight lower bounds for differentially private selection. In *FOCS*, pages 552–563, 2017.

[434] Sebastian U Stich. Local SGD converges fast and communicates little. In *International Conference on Learning Representations (ICLR)*, 2019.

[435] Sebastian U Stich and Sai Praneeth Karimireddy. The error-feedback framework: Better rates for SGD with delayed gradients and compressed communication. *arXiv:1909.05350*, 2019.

[436] Lili Su and Nitin H. Vaidya. Fault-Tolerant Multi-Agent Optimization: Optimal Iterative Distributed Algorithms. In *PODC*, 2016.

[437] Pramod Subramanyan, Rohit Sinha, Ilia Lebedev, Srinivas Devadas, and Sanjit A Seshia. A formal foundation for secure remote execution of enclaves. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 2435–2450. ACM, 2017.

[438] Ziteng Sun, Peter Kairouz, Ananda Theertha Suresh, and H Brendan McMahan. Can you really backdoor federated learning? *arXiv preprint arXiv:1911.07963*, 2019.

[439] support.google. Your chats stay private while Messages improves suggestions, 2019. URL https://support.google.com/messages/answer/9327902. Retrieved Aug 2019.

[440] Ananda Theertha Suresh, Felix X. Yu, Sanjiv Kumar, and H Brendan McMahan. Distributed mean estimation with limited communication. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3329–3337. JMLR. org, 2017.

[441] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *ICLR*, 2013.

[442] Gábor J Székely, Maria L Rizzo, Nail K Bakirov, et al. Measuring and testing dependence by correlation of distances. *The annals of statistics*, 35(6):2769–2794, 2007.

[443] Hanlin Tang, Xiangru Lian, Ming Yan, Ce Zhang, and Ji Liu. D2: Decentralized training over decentralized data. In *ICML*, 2018.

[444] Hanlin Tang, Xiangru Lian, Shuang Qiu, Lei Yuan, Ce Zhang, Tong Zhang, and Ji Liu. DeepSqueeze: Parallel stochastic gradient descent with double-pass error-compensated compression. *arXiv preprint arXiv:1907.07346*, 2019.

[445] Jun Tang, Aleksandra Korolova, Xiaolong Bai, Xueqiang Wang, and XiaoFeng Wang. Privacy loss in Apple's implementation of differential privacy on MacOS 10.12. *CoRR*, abs/1709.02753, 2017. URL http://arxiv.org/abs/1709.02753.

[446] Om Thakkar, Galen Andrew, and H Brendan McMahan. Differentially private learning with adaptive clipping. *arXiv preprint arXiv:1905.03871*, 2019.

[447] Florian Tramèr and Dan Boneh. Slalom: Fast, verifiable and private execution of neural networks in trusted hardware. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJVorjCcKQ.

[448] Florian Tramèr and Dan Boneh. Adversarial training and robustness for multiple perturbations. *arXiv preprint arXiv:1904.13000*, 2019.

[449] Florian Tramèr and Dan Boneh. Differentially private learning needs better features (or much more data). *arXiv preprint arXiv:2011.11660*, 2020.

[450] Florian Tramèr, Fan Zhang, Ari Juels, Michael K. Reiter, and Thomas Ristenpart. Stealing machine learning models via prediction APIs. In *25th USENIX Security Symposium, USENIX Security 16, Austin, TX, USA, August 10-12, 2016.*, pages 601–618, 2016. URL https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/tramer.

[451] Florian Tramèr, Fan Zhang, Huang Lin, Jean-Pierre Hubaux, Ari Juels, and Elaine Shi. Sealed-glass proofs: Using transparent enclaves to prove and sell knowledge. In *2017 IEEE European Symposium on Security and Privacy, EuroS&P 2017, Paris, France, April 26-28, 2017*, pages 19–34, 2017.

[452] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian J. Goodfellow, Dan Boneh, and Patrick D. McDaniel. Ensemble adversarial training: Attacks and defenses. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.

[453] Florian Tramèr, Jens Behrmann, Nicholas Carlini, Nicolas Papernot, and Jörn-Henrik Jacobsen. Fundamental tradeoffs between invariance and sensitivity to adversarial perturbations. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 9561–9571. PMLR, 2020. URL http://proceedings.mlr.press/v119/tramer20a.html.

[454] Brandon Tran, Jerry Li, and Aleksander Madry. Spectral signatures in backdoor attacks. In *Advances in Neural Information Processing Systems*, pages 8000–8010, 2018.

[455] Jonathan Ullman. Tight lower bounds for locally differentially private selection. Technical Report abs/1802.02638, arXiv, 2018. URL http://arxiv.org/abs/1802.02638.

[456] The Google-Landmark v2 Authors. Google landmark dataset v2, 2019. URL https://github.com/cvdfoundation/google-landmark.

[457] Jaideep Vaidya, Hwanjo Yu, and Xiaoqian Jiang. Privacy-preserving SVM classification. *Knowl. Inf. Syst.*, 14(2), January 2008.

[458] Jo Van Bulck, Marina Minkin, Ofir Weisse, Daniel Genkin, Baris Kasikci, Frank Piessens, Mark Silberstein, Thomas F Wenisch, Yuval Yarom, and Raoul Strackx. Foreshadow: Extracting the keys to the intel {SGX} kingdom with transient out-of-order execution. In *27th {USENIX} Security Symposium ({USENIX} Security 18)*, pages 991–1008, 2018.

[459] Paul Vanhaesebrouck, Aurélien Bellet, and Marc Tommasi. Decentralized collaborative learning of personalized models over networks. In *AISTATS*, 2017.

[460] Praneeth Vepakomma, Otkrist Gupta, Tristan Swedish, and Ramesh Raskar. Split learning for health: Distributed deep learning without sharing raw patient data. *arXiv preprint arXiv:1812.00564*, 2018.

[461] Praneeth Vepakomma, Chetan Tonde, Ahmed Elgammal, et al. Supervised dimensionality reduction via distance correlation maximization. *Electronic Journal of Statistics*, 12(1):960–984, 2018.

[462] Praneeth Vepakomma, Otkrist Singh, Abhishek Gupta, and Ramesh Raskar. Nopeek: Information leakage reduction to share activations in distributed deep learning. *arXiv preprint arXiv:2008.09161*, 2020.

[463] Thijs Vogels, Sai Praneeth Karimireddy, and Martin Jaggi. PowerSGD: Practical low-rank gradient compression for distributed optimization. In *NeurIPS 2019 - Advances in Neural Information Processing Systems 32*, 2019.

[464] Riad S. Wahby, Ioanna Tzialla, Abhi Shelat, Justin Thaler, and Michael Walfish. Doubly-efficient zksnarks without trusted setup. In *2018 IEEE Symposium on Security and Privacy, SP 2018, Proceedings, 21-23 May 2018, San Francisco, California, USA*, 2018.

[465] Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y Zhao. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *2019 IEEE Symposium on Security and Privacy*. IEEE, 2019.

[466] Hongyi Wang, Kartik Sreenivasan, Shashank Rajput, Harit Vishwakarma, Saurabh Agarwal, Jy yong Sohn, Kangwook Lee, and Dimitris Papailiopoulos. Attack of the tails: Yes, you really can backdoor federated learning, 2020.

[467] Jianyu Wang and Gauri Joshi. Cooperative SGD: A unified framework for the design and analysis of communication-efficient SGD algorithms. *preprint*, August 2018. URL https://arxiv.org/abs/1808.07576.

[468] Jianyu Wang and Gauri Joshi. Adaptive Communication Strategies for Best Error-Runtime Trade-offs in Communication-Efficient Distributed SGD. In *Proceedings of the SysML Conference*, April 2019. URL https://arxiv.org/abs/1810.08313.

[469] Jianyu Wang, Anit Sahu, Gauri Joshi, and Soummya Kar. MATCHA: Speeding Up Decentralized SGD via Matching Decomposition Sampling. *preprint*, May 2019. URL https://arxiv.org/abs/1905.09435.

[470] Jianyu Wang, Vinayak Tantia, Nicolas Ballas, and Michael Rabbat. SlowMo: Improving communication-efficient distributed SGD with slow momentum. *arXiv preprint arXiv:1910.00643*, 2019.

[471] Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H Vincent Poor. Tackling the objective inconsistency problem in heterogeneous federated optimization. *Advances in Neural Information Processing Systems*, 33, 2020.

[472] Kangkang Wang, Rajiv Mathews, Chloé Kiddon, Hubert Eichner, Françoise Beaufays, and Daniel Ramage. Federated evaluation of on-device personalization. *arXiv preprint arXiv:1910.10252*, 2019.

[473] Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A Efros. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018.

[474] Yu-Xiang Wang, Borja Balle, and Shiva Kasiviswanathan. Subsampled R\'enyi differential privacy and analytical moments accountant. *arXiv preprint arXiv:1808.00087*, 2018.

[475] Stanley L. Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.

[476] WeBank. WeBank and Swiss re signed cooperation MOU, 2019. URL https://finance.yahoo.com/news/webank-swiss-signed-cooperation-mou-112300218.html. Retrieved Aug 2019.

[477] Eric Wong, Frank R Schmidt, and J Zico Kolter. Wasserstein adversarial examples via projected sinkhorn iterations. *ICML*, 2019.

[478] Gavin Wood et al. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper*, 151 (2014):1–32, 2014.

[479] D. Woodruff and S. Yekhanin. A geometric approach to information-theoretic private information retrieval. In *20th Annual IEEE Conference on Computational Complexity (CCC'05)*, pages 275–284, June 2005. doi: 10.1109/CCC.2005.2.

[480] Blake Woodworth, Jialei Wang, H. Brendan McMahan, and Nathan Srebro. Graph oracle models, lower bounds, and gaps for parallel stochastic optimization. In *Advances in Neural Information Processing Systems (NIPS)*, 2018. URL https://arxiv.org/abs/1805.10222.

[481] Blake Woodworth, Kumar Kshitij Patel, Sebastian U Stich, Zhen Dai, Brian Bullins, H Brendan McMahan, Ohad Shamir, and Nathan Srebro. Is local sgd better than minibatch sgd? *arXiv preprint arXiv:2002.07839*, 2020.

[482] Xiang Wu, Ruiqi Guo, Ananda Theertha Suresh, Sanjiv Kumar, Daniel N Holtmann-Rice, David Simcha, and Felix X. Yu. Multiscale quantization for fast similarity search. In *Advances in Neural Information Processing Systems*, pages 5745–5755, 2017.

[483] Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. *CVPR*, 2019.

[484] Cong Xie. Zeno++: robust asynchronous SGD with arbitrary number of Byzantine workers. *arXiv preprint arXiv:1903.07020*, 2019.

[485] Cong Xie, Oluwasanmi Koyejo, Indranil Gupta, and Haibin Lin. Local adaalter: Communication-efficient stochastic gradient descent with adaptive learning rates. *arXiv preprint arXiv:1911.09030*, 2019.

[486] Cong Xie, Sanmi Koyejo, and Indranil Gupta. Practical distributed learning: Secure machine learning with communication-efficient local updates. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD)*, 2019.

[487] Cong Xie, Sanmi Koyejo, and Indranil Gupta. Zeno: Distributed stochastic gradient descent with suspicion-based fault-tolerance. In *International Conference on Machine Learning*, pages 6893–6901, 2019.

[488] Sirui Xie, Hehui Zheng, Chunxiao Liu, and Liang Lin. SNAS: stochastic neural architecture search. *arXiv preprint arXiv:1812.09926*, 2018.

[489] Tiancheng Xie, Jiaheng Zhang, Yupeng Zhang, Charalampos Papamanthou, and Dawn Song. Libra: Succinct zero-knowledge proofs with optimal prover computation. In *CRYPTO (3)*, volume 11694 of *Lecture Notes in Computer Science*, pages 733–764. Springer, 2019.

[490] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *CoRR*, abs/1902.04885, 2019. URL http://arxiv.org/abs/1902.04885.

[491] Timothy Yang, Galen Andrew, Hubert Eichner, Haicheng Sun, Wei Li, Nicholas Kong, Daniel Ramage, and Françoise Beaufays. Applied federated learning: Improving Google keyboard query suggestions. *arXiv preprint 1812.02903*, 2018.

[492] Andrew C Yao. Protocols for secure computations. In *Symposium on Foundations of Computer Science*, 1982.

[493] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *FOCS*, pages 162–167. IEEE Computer Society, 1986.

[494] Fangwei Ye, Carolina Naim, and Salim El Rouayheb. Preserving ON-OFF privacy for past and future requests. In *2019 IEEE Information Theory Workshop (ITW)*, August 2019.

[495] Min Ye and Alexander Barg. Optimal schemes for discrete distribution estimation under locally differential privacy. *IEEE Transactions on Information Theory*, 2018.

[496] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, pages 268–282. IEEE, 2018.

[497] Dong Yin, Yudong Chen, Kannan Ramchandran, and Peter Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates. In *ICML*, 2019.

[498] Chen Yu, Hanlin Tang, Cedric Renggli, Simon Kassing, Ankit Singla, Dan Alistarh, Ce Zhang, and Ji Liu. Distributed learning over unreliable networks. *arXiv preprint arXiv:1810.07766*, 2018.

[499] Han Yu, Zelei Liu, Yang Liu, Tianjian Chen, Mingshu Cong, Xi Weng, Dusit Niyato, and Qiang Yang. A sustainable incentive scheme for federated learning. *IEEE Intelligent Systems*, 35(4):58–69, 2020.

[500] Hao Yu, Sen Yang, and Shenghuo Zhu. Parallel restarted SGD for non-convex optimization with faster convergence and less communication. *arXiv preprint arXiv:1807.06629*, 2018.

[501] Hao Yu, Rong Jin, and Sen Yang. On the linear speedup analysis of communication efficient momentum SGD for distributed non-convex optimization. *arXiv preprint arXiv:1905.03817*, 2019.

[502] Tao Yu, Eugene Bagdasaryan, and Vitaly Shmatikov. Salvaging federated learning by local adaptation. *arXiv preprint arXiv:2002.04758*, 2020.

[503] Muhammad Bila Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P. Gummadi. Fairness constraints: Mechanisms for fair classification. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, 2017.

[504] Valentina Zantedeschi, Aurélien Bellet, and Marc Tommasi. Fully Decentralized Joint Learning of Personalized Models and Collaboration Graphs. Technical report, arXiv:1901.08460, 2019.

[505] Sixin Zhang, Anna E Choromanska, and Yann LeCun. Deep learning with elastic averaging SGD. In *Advances in Neural Information Processing Systems*, pages 685–693, 2015.

[506] Yu Zhang and Qiang Yang. A survey on multi-task learning. *CoRR*, abs/1707.08114, 2017. URL http://arxiv.org/abs/1707.08114.

[507] Yuchen Zhang, John Duchi, Micheal I. Jordan, and Martin J. Wainwright. Information-theoretic lower bounds for distributed statistical estimation with communication constraints. In *Advances in Neural Information Processing Systems*, pages 2328–2336, 2013.

[508] Yawei Zhao, Chen Yu, Peilin Zhao, and Ji Liu. Decentralized online learning: Take benefits from others' data without sharing your own to track global trend. *arXiv preprint arXiv:1901.10593*, 2019.

[509] Michael Zhu and Suyog Gupta. To prune, or not to prune: exploring the efficacy of pruning for model compression. *arXiv preprint arXiv:1710.01878*, 2017.

[510] Wennan Zhu, Peter Kairouz, Haicheng Sun, Brendan McMahan, and Wei Li. Federated heavy hitters discovery with differential privacy. *arXiv preprint arXiv:1902.08534*, 2019.

[511] Xiaojin Zhu. Machine teaching: An inverse problem to machine learning and an approach toward optimal education. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.