

异构网络中的联邦优化



Tian Li1 Anit Kumar Sahu2 Manzil Zaheer3 Maziar Sanjabi4 Ameet Talwalkar15弗吉尼亚州史密斯1

摘要

联合学习是一种分布式学习范式，它有两个区别于传统分布式优化的关键挑战：（1）网络中每个设备上的系统特征存在显著的可变性（系统异构性），以及（2）跨网络的非同分布数据（统计异构性）。在这项工作中，我们引入了一个框架FedProx来处理联邦网络中的异构性。FedProx可以看作是FedAvg的一个推广和重新参数化，FedAvg是当前最先进的联邦学习方法。虽然这种重新参数化只对方法本身做了一些小的修改，但这些修改在理论和实践中都有重要的影响。理论上，当从非相同分布（统计异构）中学习数据时，我们为我们的框架提供了收敛保证，同时通过允许每个参与设备执行可变数量的工作（系统异构性）来遵守设备级系统约束。实际上，我们证明了FedProx允许在一组真实的联邦数据集上比FedAvg更健壮地收敛。特别是，在高度异构的环境中，FedProx显示出比FedAvg更稳定和更精确的收敛行为，平均绝对测试准确率提高了22%。

# 1简介

联合学习已经成为一种在远程设备网络中分布机器学习模型训练的有吸引力的范例。虽然在机器学习的背景下有大量的分布式优化工作，但是联邦学习与传统分布式优化的区别在于两个关键的挑战：高度系统和统计异构性（McMahan et al.，2017；Li et al.，2019）。1

为了处理异构性和解决高通信成本问题，允许局部更新和低参与度的优化方法是联邦学习的流行方法（McMahan等人，2017；Smith等人，2017）。特别是，FedAvg（McMahan等人，2017）是一种迭代方法，在联邦环境中已成为事实上的优化方法。在每次迭代中，FedAvg首先局部地执行随机gra的周期-*E*

     1                                                                    2

卡内基梅隆大学博世人工智能中心3谷歌研究4 Facebook AI 5确定的人工智能。通信对象：田莉tianli@cmu.edu。*<>*

*法律程序第三MLSys会议*，美国德克萨斯州奥斯汀，2020年。作者版权所有2020。

1

隐私是联邦环境中的第三个关键挑战。虽然不是这项工作的重点，但标准隐私保护方法（如差分隐私和安全多方通信）可以自然地与本文提出的方法相结合，特别是因为我们的框架只对先前的工作提出了轻量级的算法修改。

双下降（SGD）指的是一个很小的常数并且只占网络中所有设备的一小部分的设备。然后，这些设备将其模型更新通信到中央服务器，在那里对它们进行平均。*KEK*

虽然FedAvg已经在异构环境中证明了经验上的成功，但它并没有完全解决与异构性相关的潜在挑战。在系统异构的背景下，FedAvg不允许参与设备基于其底层系统约束执行可变数量的本地工作；相反，通常只删除无法在指定时间窗口内计算epoch的设备（Bonawitz等人，2019）。从统计学的角度来看，FedAvg在数据不均匀分布于不同设备的情况下（例如，McMahan等人，2017，第3节）显示出经验上的差异。不幸的是，FedAvg很难在这样的实际场景中进行理论分析，因此缺乏收敛性保证来描述其行为（更多细节见第2节）。*E*

在这项工作中，我们提出了一个联合优化算法，从理论上和实验上解决了异构性的挑战。在开发FedProx的过程中，我们有一个关键的见解：在联合学习中，系统和统计异构性之间存在着相互作用。事实上，不管是丢弃掉队者（如在FedAvg中）还是天真地合并来自散乱者的部分信息（比如在FedProx中，最近项设置为0），都会隐式地增加统计异质性，并可能对收敛行为产生不利影响。为了缓解这一问题，我们建议在目标中添加一个有助于提高方法稳定性的最近项。这个术语为服务器提供了一种解释与部分信息相关联的异构性的原则性方法。理论上，这些修改使我们能够为我们的方法提供收敛保证，并分析异质性的影响。在实验上，我们证明了这些修改提高了异构网络中联合学习的稳定性和整体准确性，在高度异构的环境下，平均提高了22%的绝对测试准确率。

本文的其余部分安排如下。在第2节中，我们将提供联合学习的背景知识和相关工作的概述。然后，我们在第3节中介绍了我们提出的框架FedProx，并在第4节中推导了该框架在统计和系统异构性方面的收敛保证。最后，在第5节中，我们在一组合成的和真实的联邦数据集上对FedProx进行了全面的实证评估。我们的实证结果有助于说明和验证我们的理论分析，并证明在异构网络中FedProx相对于FedAvg的实际改进。

# 2背景及相关工作

在过去的十年中，大规模机器学习，特别是在数据中心环境中，推动了许多分布式优化方法的发展（参见，例如。，

Boyd等人，2010年；Dekel等人，2012年；Dean等人，2012年；

Zhang et al.，2013；Li et al.，2014a；Shamir et al.，2014；Reddi et al.，2016；Zhang et al.，2015；Richtarik&Tak'a&apos;cˇ，2016；Smith等人，2018）。然而，随着诸如电话、传感器和可穿戴设备等计算基础的功率和普及率的增长，与将数据转移到数据中心相比，在分布式设备网络中本地学习统计模型越来越有吸引力。这个问题被称为联合学习，需要解决隐私、异构数据和设备以及大规模分布式网络的新挑战（Li等人，2019年）。

最近提出了一些优化方法，这些方法是针对联邦环境中的特定挑战而定制的。与传统的分布式方法（如ADMM（Boyd et al.，2010）或mini batch methods（Dekel et al.，2012）相比，这些方法显示出了显著的改进，允许不精确的本地更新，以平衡大型网络中的通信与计算，以及在任何通信回合中活跃的设备的一小部分（McMahan等人，2017；Smith等人，2017）。例如，Smith等人。（2017）提出了一种通信高效的原始-对偶优化方法，通过多任务学习框架为每个设备学习独立但相关的模型。尽管所提出的方法具有理论上的保证和实际的有效性，但是这种方法不能推广到非凸问题，例如深度学习，在那里强对偶性不再得到保证。在非凸环境下，联邦平均法（FedAvg）是一种基于在原始数据中平均局部随机梯度下降（SGD）更新的启发式方法，但已被证明在经验上效果良好（McMahan等人，2017）。

不幸的是，FedAvg由于其本地更新方案、每轮活跃的设备很少以及数据在网络中频繁分布的问题，分析起来相当具有挑战性。特别是，当每个设备生成自己的本地数据时，统计异构性与设备之间的数据分布不一致很常见。一些工作已经在更简单的、非联邦的设置中朝着分析FedAvg迈出了一步。例如，在IID环境中研究了使本地更新类似于FedAvg的并行SGD和相关变体（Zhang et al.，2015；Shamir et al.，2014；Reddi et al.，2016；Zhou&Cong，2018；Stich，2019；Wang&Joshi，2018；Woodworth et al.，2018；Lin et al.，2020），已经在IID环境中进行了研究。然而，结果依赖于这样一个前提：每个局部解算器都是同一随机过程的副本（由于IID假设）。这种推理方法不适用于异构环境。

尽管最近的一些研究（Yu et al.，2018；Wang et al.，2019；Hao et al.，2019；Jiang&Agrawal，2018）探索了统计异构环境下的收敛保证，但他们做出了限制性假设，即所有设备都参与每一轮通信，这在现实的联邦网络中通常是不可行的（McMahan等人，2017）。此外，与本文提出的解算框架相比，它们依赖于每个设备（SGD或GD）上使用的特定解算器，并在分析中添加额外的凸性假设（Wang et al.，2019）或一致边界梯度（Yu et al.，2018）。还有一些启发式方法旨在通过共享本地设备数据或服务器端代理数据来解决统计异构性（Jeong等人，2018；Zhao等人，2018；Huang等人，2018）。然而，这些方法可能是不切实际的：除了给网络带宽带来负担外，向服务器发送本地数据（Jeong等人，2018年）违反了联邦学习的关键隐私假设，并向所有设备发送全局共享的代理数据（Zhao等人，2018年；Huang等人。，2018年）需要努力仔细生成或收集此类辅助数据。

除了统计上的异构性之外，系统异构性也是联邦网络中的一个关键问题。由于硬件（CPU、内存）、网络连接（3G、4G、5G、wifi）和电源（电池电量）的变化，联邦网络中每个设备的存储、计算和通信能力可能有所不同。这些系统级特性极大地加剧了诸如分散器缓解和容错等挑战。实践中使用的一种策略是忽略那些无法完成一定量训练的受限设备（Bonawitz等人，2019年）。然而（正如我们在第5节中所展示的那样），这可能会对收敛产生负面影响，因为它限制了有助于训练的有效设备的数量，并且如果丢弃的设备具有特定的数据特性，则可能会导致设备采样过程中的偏差。

在本文中，受FedAvg的启发，我们探索了一个更广泛的框架FedProx，它能够处理异构的联邦环境，同时保持相似的隐私和计算优势。在考虑实际系统约束的同时，通过局部函数间的统计差异性刻画，分析了框架的收敛性。我们的相异性特征是受求解线性方程组的随机Kaczmarz方法启发的

（Kaczmarz，1993；Strohmer&Vershynin，2009），一个类似的假设也被用于分析其他情况下SGD的变体（参见，例如Schmidt&Roux，2013；Vaswani等人，2019；Yin等人，2018）。我们提出的框架为异构联邦网络的优化提供了更好的鲁棒性和稳定性。

最后，在相关工作方面，我们注意到我们提出的工作的两个方面，FedProx中的近端项和我们分析中使用的有界相异性假设-已经在优化文献中进行过研究，尽管通常是在非联合环境下进行的非常不同的动机。为了完整起见，我们在附录B中进一步讨论了这个背景工作。

# 3联邦优化：方法

在本节中，我们将介绍最近的联邦学习方法背后的关键要素，包括FedAvg，然后概述我们提出的框架FedProx。

联邦学习方法（例如，McMahan等人，2017；Smith等人，2017）旨在处理收集数据的多个设备和一个中央服务器，通过网络协调全局学习目标。特别是为了尽量减少：

*N*最小f（w）=（w）=E[Fk（w）]，（1）十*pkFk公司k*

*w k公司*=1

其中是设备数量，≥0，且=1。一般来说，本地目标衡量可能不同数据分布Dk的本地经验风险，即。，*Npk键*P*kpk键*

*Fk公司*（w） ：=E∼Dk[fk（w；xk）]，每个设备上都有样品*xk公司nk公司k*. 因此，我们可以设定，在哪里*n*=

P*knk公司*是数据点的总数。在本文中，我们考虑（w）可能是非凸的。*Fk公司*

为了减少通信量，联邦优化中的一种常用技术是在每个设备上使用基于设备数据的局部目标函数作为全局目标函数的代理。在每个外部迭代中，选择设备的子集，并使用局部解算器优化每个选定设备上的局部目标函数。然后，设备将其本地模型更新通信到中央服务器，中央服务器将其聚合并相应地更新全局模型。在这种情况下，允许灵活执行的关键是每个局部目标都可以不精确地解决。这使得本地计算量与通信量可以根据执行的本地迭代次数进行调整（附加的局部迭代对应于更精确的局部解）。我们在下面正式介绍这个概念，因为它将在整个论文中使用。

定义1（不精确解）。对于函数*γ*

，且∈[0,1]，*γ*

如果k∇h（w∇；w0）k≤γk∇h（w0；w0）k，其中∇h（w；w0）=∇F（w）+µ（w−w0），则称为minw h（w；w0）的不精确解。请注意，较小的值对应较高的精度。*w*∗*γγ*

我们在我们的分析（第4节）中使用-不精确性来衡量每轮局部解算器的局部计算量。如前所述，由于系统条件的变化，不同的设备在解决局部子问题方面可能会取得不同的进展，因此，允许设备和迭代都有所不同是很重要的。这是我们在下一节讨论的框架的动机之一。为了便于记法，我们首先推导我们的主要收敛结果，假设这里定义了一致性（第4节），然后在推论9中提供带有变量的结果。*γγγγ*

3.1联邦平均法（FedAvg）

在联邦平均法（FedAvg）（McMahan等人，2017）中，设备处全局目标函数的局部代理是（·）*kFk公司*局部求解器采用随机梯度下降法（SGD），在每个设备上使用相同的学习速率和局部epoch个数。在每一轮，选择总设备的一个子集，并在本地运行SGD*E*纪元的数量，然后计算得到的模型更新的平均值。算法1总结了FedAvg的细节。

McMahan等人。（2017）经验表明，正确调整FedAvg的优化超参数至关重要。特别地，FedAvg中局部时间的数目对收敛起着重要的作用。一方面，执行更多的局部epoch可以增加局部计算量，并可能减少通信量，从而大大提高通信约束网络的整体收敛速度。另一方面，对于不同（异构）的局部目标，大量的局部时间可能会导致每个设备朝着其局部算法1联邦平均（FedAvg）的最优方向发展*Fk公司*

输入：，，，，=1，···，N代表=0，···，T−1 do*KTηEw*0*Npk键kt*

随机选择每个设备的一个子集*StKkpk键*

服务器向所有选择的设备发送每个设备∈St更新SGD的时间*重量k重量E*

启用步长*Fk公司η*获得

每个装置∈St*k*发送回服务器

服务器将的聚合为=K1结束*w重量*+1 P*k*∈S*twkt公司*+1



与全局目标相反，可能会影响收敛甚至导致方法发散。此外，在具有异构系统资源的联邦网络中，将本地epoch的数量设置为高可能会增加设备无法在给定通信回合内完成训练的风险，因此必须退出该过程（Bonawitz等人，2019）。

因此，在实践中，重要的是找到一种方法来将本地时间设置为高（以减少通信），同时还允许健壮的收敛。更基本的是，我们注意到，作为本地数据和可用系统资源的函数，本地时间的“最佳”设置可能在每次迭代和每个设备上发生变化。事实上，一种比规定固定数量的本地时间段更自然的方法是允许时间段根据网络的特性而变化，并通过考虑这种异构性仔细地合并解决方案。我们在下面介绍的FedProx中对这个策略进行了正式化。

3.2拟议框架：FedProx

我们提出的框架FedProx（算法2）类似于FedAvg，在每一轮选择设备子集，执行局部更新，然后对这些更新进行平均以形成全局更新。然而，FedProx做了以下简单但关键的修改，这导致了显著的经验改进，也使我们能够为该方法提供收敛性保证。

容忍部分工作。如前所述，联邦网络中的不同设备通常在计算硬件、网络连接和电池电量方面具有不同的资源限制。因此，强制每个设备执行统一数量的工作（即运行相同数量的本地epoch，

*E*)，就像在费达夫。在FedProx中，我们对FedAvg进行了一般化，允许基于可用系统资源的设备在本地执行可变数量的工作，然后聚合从离散器发送的部分解决方案（与丢弃这些设备相比）。换句话说，在整个训练过程中，FedProx不是假设所有设备都是统一的，而是隐式地容纳变量*γγ*用于不同的设备和不同的迭代。我们正式定义了设备的不精确性*k*在下面的迭代中，这是定义1的自然扩展。*t*

定义2-不精确解）。对于函数

，和∈[0,1]，我们*γ*

说*w*∗是…的不精确解minw香港（w；wt）如果，在哪里

∇hk（w；wt）=∇Fk（w）+µ（w−wt）. 请注意，较小的值对应较高的精度。

类似于Definition，它度量在设备上为解决本地子问题而执行的本地计算量*k*在*t*-第四轮。可将可变数量的局部迭代视为的代理。利用更灵活的不精确性，我们可以很容易地推广定义下的收敛结果1（定理4）考虑与系统异质性相关的问题，如离散者（见推论9）。

近端术语。如第3.1节所述，虽然容忍跨设备执行的工作量不均匀有助于减轻系统异构性的负面影响，但过多的本地更新仍可能（潜在地）由于底层异构数据而导致方法分歧。我们建议在局部子问题中加入一个近邻项，以有效地限制变量局部更新的影响。特别是，设备不只是最小化局部函数（·），而是使用其选择的局部解算器来近似最小化以下目标：*Fk公司k香港*

*.* （二）

近端项在两个方面是有益的：（1）它通过限制局部更新更接近初始（全局）模型来解决统计异质性问题，而不需要手动设置局部时间点的数量。（2） 它允许安全地合并由系统异构性引起的可变数量的本地工作。我们总结了算法2中FedProx的步骤。

算法2 FedProx（提议的框架）

输入：，，，，=1，···，N代表=0，···，T−1 do*KTµγw*0*Npk键kt*

服务器随机选择一个子集的设备（每个设备的选择概率）服务器发送到所有选择的设备*StKkpk键重量*

每个选择的器件∈St*k*找到一个不精确的最小值：阿格明

每个设备∈St发送回服务器*kwkt公司*+1

服务器将的聚合为=K1结束*w重量*+1 P*k*∈S*twkt公司*+1



我们注意到，类似上面的术语是整个优化文献中常用的工具；为了完整起见，我们在附录B中对此进行了更详细的讨论。建议使用的一个重要区别是我们建议、探索，为了解决联邦网络中的异构性，分析了这个术语。我们的分析（第4节）在考虑在分布式环境中解决这样一个目标时也是独一无二的：（1）非IID分区数据，（2）使用任何本地解算器，（3）跨设备的变量不精确更新，以及（4）a

在每一轮都处于活动状态的设备子集。这些假设对于在现实的联邦场景中提供这样一个框架的特征至关重要。

在我们的实验（第5节）中，我们证明了在存在系统异构性的情况下，容忍部分工作是有益的，并且我们在FedProx中修改的局部子问题比vanilla FedAvg在异构数据集上的收敛性更为稳健和稳定。在第4节中，我们还看到，使用最近项使FedProx更易于进行理论分析（即，局部目标可能表现得更好）。特别地，如果相应地选择，那么Hessian的可以是半正定的。因此，当为非凸时，将为凸；当为凸时，则为强凸。*µ香港Fk公司香港Fk公司µ*

最后，我们注意到，由于FedProx只对FedAvg进行了轻量级的修改，这使得我们能够对广泛使用的FedAvg方法的行为进行推理，并能够轻松地将FedProx集成到现有的包/系统中，例如TensorFlow federed和LEAF（TFF；Caldas et al.，2018）。特别地，我们注意到FedAvg是FedProx的一个特例，其中（1）=0，（2）本地解算器被特别选择为SGD，（3）一个常数（对应于本地epoch的数量）跨越设备和更新轮（即，没有系统异构性的概念）。事实上，FedProx在这方面更为通用，因为它允许跨设备执行部分工作，并允许在每个设备上使用任何本地（可能是非迭代）解算器。*µγ*

# 4F：收敛分析**EDPROX公司**

FedAvg和FedProx本质上是随机算法：在每一轮中，只有一小部分设备被采样来执行更新，并且在每个设备上执行的更新可能是不精确的。众所周知，为了使随机方法收敛到一个平稳点，需要减小步长。这与非平稳方法（如梯度下降法）相反，后者可以通过采用恒定的步长来找到一个静止点。为了分析常步长方法的收敛性（通常在实际中实现），我们需要量化局部目标函数之间的差异程度。这可以通过假设数据是IID来实现的，也就是说，设备之间是同质的。不幸的是，在现实的联邦网络中，这种假设是不切实际的。因此，我们首先提出一个度量局部函数之间的差异性的度量（第4.1节），然后在考虑变量的前提下分析FedProx（第4.2节）。*γ*

4.1局部差异

在这里，我们引入一个度量联邦网络中设备之间的差异性，这足以证明收敛性。这也可以通过更简单和更严格的梯度有界方差假设（推论10）来满足，我们在第5节的实验中对此进行了探讨。有趣的是，类似的假设（例如，施密特

&Roux，2013；Vaswani et al.，2019；Yin et al.，2018）已在其他地方进行了勘探，但目的不同；我们在附录B中对这些工作进行了讨论。

定义3（局部差异）。本地功能是*BFk公司B*-在不同的地方k∇f（w）k2B2. 我们进一步定义2 k∇f（w）k6=0。

这里E[·]表示对有质量的器件的期望值*k*

（如方程式1所示）。定义-

第3条可以看作是IID假设的一个推广，具有有界的相异性，同时考虑了统计上的异质性。作为一个健全性检查，当所有的局部函数都相同时，我们的（w）=1代表所有。然而，在联邦环境中，数据分布通常是异构的，并且由于采样差异而为1，即使假设样本是IID。让我们也考虑一下（·）与经验风险目标相关的情况。如果所有设备上的样本都是齐次的，即以IID的方式采样，那么作为mink nk→∞，由于所有的局部函数在大样本极限下收敛到同一个期望风险函数，则得到（w）→1。因此，（w）≥1且（w）的值越大，局部函数之间的差异越大。*BwB>Fk公司BwBB*

使用定义3，我们现在陈述了我们在收敛性分析中使用的形式相异性假设。这只需要定义3中定义的相异性是有界的。正如后面讨论的，我们的收敛速度是

网络中统计异构性/设备差异性的作用。

假设1（有界相异性）。对某些人来说，存在一个*为了所有的要点*

*.*



2

        作为例外，我们定义（w）=1当*B*

，即是静止的所以-*w*

所有地方职能部门一致同意的决议。*Fk公司*

对于大多数实际的机器学习问题，不需要将问题求解为高精度的平稳解，即通常不是很小。事实上，众所周知，解决超过某个阈值的问题甚至可能由于过度拟合而损害泛化性能（Yao等人，2007）。虽然在实际的联合学习问题中，样本不是IID，但它们仍然是从并非完全无关的分布中采样的（如果是这样的话，例如，在设备之间拟合单个全局模型是不明智的）。因此，可以合理地假设局部函数之间的差异在整个训练过程中保持有界。在第5.3.3节中，我们还对真实数据集和合成数据集进行了经验度量，并表明该度量捕获了现实世界的统计异质性，并且与实际性能相关（相异性越小，收敛性越好）。*w*

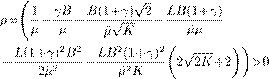
4.2 FedProx分析

利用有界相异性假设（假设1），我们现在分析当执行一步FedProx时目标的期望减少量。我们的收敛速度（定理6）可以直接从每次更新轮的期望减少的结果中推导出来。在下面的分析中，为了便于记谱，我们假设所有这些都是一样的。*γktk、 t*

定理4（非凸FedProx收敛性：局部相异性）。假设*B1 等等。假设功能Fk是非凸的，L-Lipschitz光滑，并且存在*

*我*−*>*0*，这样***我***，与µ*±：=µ−L−>0*.*

*假设如此小波变换不是一个平稳解和局部函数Fk是B-不同，即。B*（重量）≤B*. 如果µ,K、 以及γ在算法中2 被选择为*

*,*

*然后在迭代中算法的t2，我们有以下内容*

*全球目标预期减少：*

*,*

*哪里St是迭代时选择的K个设备t。*

我们让读者参考附录A.1以获得详细的证据。

关键的步骤包括应用我们的概念-不精确*γ*

（定义1）对于每个子问题，使用有界相异性假设，同时只允许设备在每轮活动。最后一步特别全面地介绍了E，一种关于设备选择的期望。我们注意到，在我们的理论中，我们需要0，这是FedProx收敛的一个充分条件，但不是必要条件。因此，正如我们的经验探索（第5节），其他一些（不一定满足0）也可以实现收敛。*KStSttµ >µµ >*

定理4利用定义3中的相异性来确定FedProx在每次迭代中目标值的充分减少。在附录A.2中，我们提供了一个用更常见（尽管稍微有点限制）有界方差假设来描述性能的推论。这种假设通常被采用，例如，在分析SGD等方法时。接下来，我们提供充分（但不是必要的）条件，以确保定理4中的0，以便在每一轮之后可以获得足够的减少。*ρ >*

备注5。对于1*定理中的ρ4 为了积极，我们需要γB<和**. 这些条件有助于量化不同（B）和算法参数之间的权衡(γ,K） 一。*

在上述假设下，我们可以利用差分假设来刻画收敛速度1注意这些结果适用于一般非凸（·）。*Fk公司*

定理6（收敛速度：FedProx）。给一些

*，假设**，和K定理的假设4 在FedProx的每次迭代中保持不变。而且，f*（w0）−f∗=∆*. 然后，之后**FedProx的迭代，我们有**.*

虽然到目前为止的结果对非凸（·）成立，我们也可以刻画凸损失函数在局部目标精确最小化的特殊情况下的收敛性（推论7）。附录A.3提供了证明。*Fk公司*

推论7（收敛：凸情形）。设定理（·）=0的断言*4 等等。另外，让Fk公司是凸的而且γkt对于任何人k、 也就是说，如果**，然后我们可以选择µ*≈6LB2*从中可以看出ρ*≈24*磅*1 2*.*

请注意，假设1中的小值转化为较大值。推论7表明，为了解决使用FedProx的精度越来越高的问题，需要适当提高。在第5.3节中，我们通过经验验证了0导致更稳定的收敛。此外，在推论7中，如果我们在有界方差假设下插入的上界（推论*Bµµ >B*

10)，达到精度所需的步骤数为

. 我们的分析有助于描述FedProx和类似方法在局部函数不同时的性能。

备注8（与新加坡元比较）。我们注意到

*FedProx实现了与SGD相同的渐近收敛保证：在有界方差假设下，如果我们替换B的上界在推论中10 然后选择µ足够大，子问题精确求解时FedProx的迭代复杂度*

*Fk公司*(·)*是凸的**，与新加坡元相同*

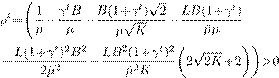
*(加迪米和兰，2013).*

为了给定理6中的速率提供上下文，我们将其与注8中凸情况下的SGD进行比较。总的来说，我们对FedProx的分析并没有产生比经典分布式SGD更好的收敛速度（无需局部更新），尽管FedProx可能在每个通信轮的本地执行更多的工作。事实上，当数据以非同一分布方式生成时，FedProx等本地更新方案的性能可能比分布式SGD差。因此，我们的理论结果并不一定证明FedProx优于分布式SGD；相反，它们为FedProx的收敛提供了充分的（但不是必要的）条件。我们的分析是我们所知道的第一个针对异构环境中的问题（1）分析任何联合（即，本地更新方案和低设备参与）优化方法。

最后，我们注意到前面的分析假设没有系统异构性，并且对所有设备和迭代都使用相同的方法。然而，我们可以扩展它们以允许根据设备和迭代（如定义2所示）而变化，这相当于允许设备执行由本地系统条件确定的可变工作量。我们用下面的变量提供收敛结果。*γγγ*

推论9（收敛性：变量）。假设函数0*γFk是非凸的，L-Lipschitz光滑，并且存在我*−*>，这样***我***，与µ*±：=µ−L−>

0*. 假设如此小波变换不是一个平稳解和局部函数Fk是B-不同，即。B*（重量）≤B*. 如果µ,K、 以及γkt算法2 被选择为*

*,*

*然后在迭代中算法的t2，我们预计全球目标将减少如下：*

*,*

*哪里St是迭代时选择的K个设备t和γt*=最大值∈S*ktγ千吨。*

从定理4的证明中可以很容易地推广证明，注意到E[（1+γkt）k∇Fk（wt）k]≤（1+maxk∈S）E[k∇Fk（wt）k]。*ktγktk*

# 5个实验

现在我们给出了广义FedProx框架的实证结果。在第5.2节中，我们展示了在面对系统异构性时，FedProx容忍部分解决方案的改进性能。在第5.3节中，我们展示了FedProx在具有统计异质性（不考虑系统异构性）的情况下的有效性。我们还研究了统计异质性对收敛性的影响（第5.3.1节），并展示了经验收敛与我们的理论有界相异性假设（假设1）（第5.3.3节）之间的关系。我们在第5.1节和附录C中提供了实验设置的详细信息。所有代码、数据和实验都可以在以下网址公开：github.com/litian96/FedProx.

5.1实验细节

我们在不同的任务、模型和realworld联合数据集上评估FedProx。为了更好地描述统计异质性并研究其对收敛性的影响，我们还对一组合成数据进行了评估，这使得统计异质性的操作更加精确。我们通过给不同的设备分配不同数量的本地工作来模拟系统的异构性。

合成数据。为了生成合成数据，我们遵循类似于Shamir等人的设置。（2014年），在设备之间增加了异构性。特别地，对于每个器件，我们根据模型=（softmax（Wx+b）），∈R∈R∈R生成样本（Xk，Yk），我们模型∼N（uk，1），∼N（uk，1），∼N（0，α）；∼N（vk，∑），其中协方差矩阵∑与∑j，j=j−1.2对角。平均向量中的每个元素都来自N（Bk，1），Bk∼N（0，β）。因此，控制本地模型彼此之间的差异有多大，并控制每个设备上的本地数据与其他设备的本地数据的差异。我们改变方法来生成三个异构的分布式数据集，表示为Synthetic（），如图2所示。我们还通过在所有设备上设置相同的数据集并设置为遵循相同的分布来生成一个IID数据集。我们的目标是学习全球和。详细信息见附录C.1。*k是的阿格麦克斯十*60*，W*10×60个*，b*10*周黑色英国xk公司vk公司αβα,βα,βW、 bXk公司Wb*

真实数据。我们还研究了四个真实的数据集；统计数据汇总在表1中。这些数据集是根据联合学习的先前工作以及最近的联邦学习基准（McMahan等人，2017年；Caldas等人，2018年）整理而成的。我们用多项式logistic回归研究了一个凸分类问题（LeCun等人，1998）。为了增加统计上的异质性，我们将数据分布在1000台设备中，使得每个设备的样本数只有两位数，并且每个设备的样本数遵循幂律。然后，我们使用相同的模型研究了一个更复杂的62类联邦扩展MNIST（Cohen et al.，2017；Caldas et al.，2018）（FEMNIST）数据集。对于非凸设置，我们考虑使用LSTM分类器对来自感伤140（Go et al.，2009）（Sent140）的tweets执行文本情感分析任务，其中每个twitter帐户对应于一个设备。我们还研究了在莎士比亚全集（McMahan et al.，2017）（莎士比亚）的数据集上进行下一个字符预测的任务。戏剧中的每个说话角色都与不同的装置相关联。数据集、模型和工作负载的详细信息见附录C.1。

*表1。*四个真实联邦数据集的统计。



数据集设备样本/设备

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | 意思是 | 标准偏差 |
| MNIST公司 | 1,000 | 69,035 | 69 | 106 |
| 女性主义者 | 200 | 18,345 | 92 | 159 |
| 莎士比亚 | 143 | 517,106 | 3,616 | 6,808 |
| Sent140型 | 772 | 40,783 | 53 | 32 |

实施。我们在Tensorflow（Abadi等人，2016）中实现了FedAvg（算法1）和FedProx（算法2）。为了与FedAvg进行公平比较，我们采用SGD作为FedProx的局部求解器，并采用了与算法1和算法2稍有不同的设备采样方案：均匀采样设备，然后使用与本地数据点数量成比例的权重平均更新（如McMahan等人最初提出的那样）。（2017年）。虽然我们的分析不支持这种抽样方案，但无论是否采用FedProx和FedAvg，我们都观察到了类似的相对行为。有趣的是，我们还观察到，本文中提出的抽样方案实际上使两种方法的性能更加稳定（见附录C.3.4，图12）。这表明拟议框架还有一个好处。详细信息见附录C.2。

超参数和评估指标。对于每个数据集，我们调整FedAvg上的学习速率（有=1且没有系统异构性），并对该数据集上的所有实验使用相同的学习速率。对于所有数据集的所有实验，我们将所选设备的数量设置为10。对于每次比较，我们会在所有运行中修复随机选择的设备、离散器和小批量订单。我们根据全球目标（w）报告所有指标。请注意，在我们的模拟中（详见第5.2节），我们假设每个通信轮对应于一个特定的聚合时间戳（以真实世界中的全局时钟时间来测量），因此我们报告的结果是以轮次而不是浮点或挂钟时间来表示的。具体参数见附录C。*Ef*

5.2系统异构性：容忍部分工作

为了测量允许发送部分解决方案以使用FedProx处理系统异构性的效果，我们模拟具有不同系统异构性的联邦设置，如下所述。

系统异构仿真。我们假设在训练过程中存在一个全局时钟，每个参与设备决定了作为这个时钟周期及其系统约束的函数的局部工作量。这个指定的局部计算量对应于设备的某个隐式值*k*at the -th iteration. In our simulations, we fix a global number of epochs , and force some devices to perform fewer updates than epochs given their current systems constraints. In particular, for varying heterogeneous settings, at each round, we assign number of epochs (chosen uniformly at random between [1, ]) to 0%, 50%, and 90% of the selected devices, respectively. Settings where 0% devices perform fewer than epochs of work correspond to the environments without systems heterogeneity, while 90% of the devices sending their partial solutions corresponds to highly heterogeneous environments. FedAvg will simply drop these 0%, 50%, and 90% stragglers upon reaching the global clock cycle, and FedProx will incorporate the partial updates from these devices.*tEE x EE*

In Figure 1, we set to be 20 and study the effects of aggregating partial work from the otherwise dropped devices. The synthetic dataset here is taken from Synthetic (1,1) in Figure 2. We see that on all the datasets, systems heterogeneity has negative effects on convergence, and larger heterogeneity results in worse convergence (FedAvg). Compared with dropping the more constrained devices (FedAvg), incorporating variable amounts of work (FedProx, = 0) is beneficial and leads to more stable and faster convergence. We also observe that setting 0 in FedProx can further improve convergence, as we discuss in Section 5.3.*E µ µ >*

We additionally investigate two less heterogeneous settings. First, we limit the capability of all the devices by setting to be 1 (i.e., all the devices run at most one local epoch), and impose systems heterogeneity in a similar way. We show training loss in Figure 9 and testing accuracy in Figure 10 in the appendix. Even in these settings, allowing for partial work can improve convergence compared with FedAvg. Second, we explore a setting without any statistical heterogeneity using an identically distributed synthetic dataset (Synthetic IID). In this IID setting, as shown in Figure 5 in Appendix C.3.2, FedAvg is rather robust under device failure, and tolerating variable amounts of local work may not cause major improvement. This serves as an additional motivation to rigorously study the effect of statistical heterogeneity on new methods designed for federated learning, as simply relying on IID data (a setting unlikely to occur in practice) may not tell a complete story.*E*

|  |
| --- |
| *Figure 1.* FedProx results in significant convergence improvements relative to FedAvg in heterogeneous networks. We simulate different levels of systems heterogeneity by forcing 0%, 50%, and 90% devices to be the stragglers (dropped by FedAvg). (1) Comparing FedAvg and FedProx (= 0), we see that allowing for variable amounts of work to be performed can help convergence in the presence of systems heterogeneity. (2) Comparing FedProx (= 0) with FedProx (0), we show the benefits of our added proximal term. FedProx with 0 leads to more stable convergence and enables otherwise divergent methods to converge, both in the presence of systems heterogeneity (50% and 90% stragglers) and without systems heterogeneity (0% stragglers). Note that FedProx with = 0 and without systems heterogeneity (no stragglers) corresponds to FedAvg. We also report testing accuracy in Figure 7, Appendix C.3.2, and*µ µ µ > µ > µ* |

show that FedProx improves the test accuracy on all datasets.

5.3 Statistical Heterogeneity: Proximal Term

To better understand how the proximal term can be beneficial in heterogeneous settings, we first show convergence can become worse as statistical heterogeneity increases.

*5.3.1 Effects of Statistical Heterogeneity*

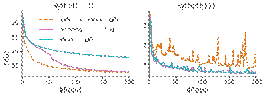
In Figure 2 (the first row), we study how statistical heterogeneity affects convergence using four synthetic datasets without the presence of systems heterogeneity (fixing to be 20). From left to right, as data become more heterogeneous, convergence becomes worse for FedProx with = 0 (i.e., FedAvg). Though it may slow convergence for IID data, we see that setting 0 is particularly useful in heterogeneous settings. This indicates that the modified subproblem introduced in FedProx can benefit practical federated settings with varying statistical heterogeneity. For perfectly IID data, some heuristics such as decreasing if the loss continues to decrease may help avoid the deceleration of convergence (see Figure 11 in Appendix C.3.3). In the sections to follow, we see similar results in our nonsynthetic experiments.*E µ µ > µ*

*5.3.2 Effects of µ >* 0

The key parameters of FedProx that affect performance are the amount of local work (as parameterized by the number of local epochs, ), and the proximal term scaled by . Intuitively, large may cause local models to drift too far away from the initial starting point, thus leading to potential divergence (McMahan et al., 2017). Therefore, to handle the divergence or instability of FedAvg with non-IID data, it is helpful to tune carefully. However, is constrained by the underlying system&apos;s environments on the devices, and it is difficult to determine an appropriate uniform for all devices. Alternatively, it is beneficial to allow for devicespecific &apos;s (variable &apos;s) and tune a best (a parameter that can be viewed as a re-parameterization of ) to prevent divergence and improve the stability of methods. A proper can restrict the trajectory of the iterates by constraining the iterates to be closer to that of the global model, thus incorporating variable amounts of updates and guaranteeing convergence (Theorem 6).*EµE E E E Eγµ Eµ*

We show the effects of the proximal term in FedProx

(0) in Figure 1. For each experiment, we compare the results between FedProx with = 0 and FedProx with a best (see the next paragraph for discussions on how to select ). For all datasets, we observe that the appropriate can increase the stability for unstable methods and can force divergent methods to converge. This holds both when there is systems heterogeneity (50% and 90% stragglers) and there is no systems heterogeneity (0% stragglers). 0 also increases the accuracy in most cases (see Figure 6 and Figure 7 in Appendix C.3.2). In particular, FedProx improves absolute testing accuracy relative to FedAvg by 22% on average in highly heterogeneous environments (90% stragglers) (see Figure 7).*µ > µ µ µµ µ >*



|  |
| --- |
| Sy theti II 3 Sy theti (0 0) Sy theti (0 5 0 5) Sy theti (1 1)                                    # Rou d # Rou d # Rou d # Rou d  *Figure 2.* Effect of data heterogeneity on convergence. We remove the effects of systems heterogeneity by forcing each device to run the same amount of epochs. In this setting, FedProx with = 0 reduces to FedAvg. (1) Top row: We show training loss (see results on testing accuracy in Appendix C.3, Figure 6) on four synthetic datasets whose statistical heterogeneity increases from left to right. Note that the method with = 0 corresponds to FedAvg. Increasing heterogeneity leads to worse convergence, but setting 0 can help to combat this. (2) Bottom row: We show the corresponding dissimilarity measurement (variance of gradients) of the four synthetic datasets. This metric captures statistical heterogeneity and is consistent with training loss — smaller dissimilarity indicates better convergence.*µ µ µ >* |

*Figure 3.* Effectiveness of setting adaptively based on the current model performance. We increase by 0.1 whenever the loss increases and decreases it by 0.1 whenever the loss decreases for 5 consecutive rounds. We initialize to 1 for Synthetic IID (in order to be adversarial to our methods), and initialize to 0 for Synthetic (1,1). This simple heuristic works well empirically.*µ µ µ µ*

Choosing . One natural question is to determine how to set the penalty constant in the proximal term. A large may potentially slow the convergence by forcing the updates to be close to the starting point, while a small may not make any difference. In all experiments, we tune the best from the limited candidate set {0.001,0.01,0.1,1}. For the five federated datasets in Figure 1, the best values are 1, 1, 1, 0.001, and 0.01, respectively. While automatically tuning is difficult to instantiate directly from our theoretical results, in practice, we note that can be adaptively chosen based on the current performance of the model. For example, one simple heuristic is to increase when seeing the loss increasing and decreasing when seeing the loss decreasing. In Figure 3, we demonstrate the effectiveness of this heuristic using two synthetic datasets. Note that we start from initial values that are adversarial to our methods. We provide full results showing the competitive performance of this approach in Appendix C.3.3. Future work includes developing methods to automatically tune this parameter for heterogeneous datasets, based, e.g., on the theoretical groundwork provided here.*µµ µ µ µ µ µ µ µ µ µ*

*5.3.3 Dissimilarity Measurement and Divergence*

Finally, in Figure 2 (the bottom row), we demonstrate that our B-local dissimilarity measurement in Definition 3 captures the heterogeneity of datasets and is therefore an appropriate proxy of performance. In particular, we track the variance of gradients on each device, [k∇Fk(w)−∇f(w)k2], which is lower bounded by (see Bounded Variance Equivalence Corollary 10). Empirically, we observe that increasing leads to smaller dissimilarity among local functions , and that the dissimilarity metric is consistent with the training loss. Therefore, smaller dissimilarity indicates better convergence, which can be enforced by setting appropriately. We also show the dissimilarity metric on real federated data in Appendix C.3.2.*EkB µ Fkµ*

# 6 CONCLUSION

In this work, we have proposed FedProx, an optimization framework that tackles the systems and statistical heterogeneity inherent in federated networks. FedProx allows for variable amounts of work to be performed locally across devices, and relies on a proximal term to help stabilize the method. We provide the convergence guarantees for FedProx in realistic federated settings under a device dissimilarity assumption, while also accounting for practical issues such as stragglers. Our empirical evaluation across a suite of federated datasets has validated our theoretical analysis and demonstrated that the FedProx framework can significantly improve the convergence behavior of federated learning in realistic heterogeneous networks.

# ACKNOWLEDGEMENTS

We thank Sebastian Caldas, Jakub Konecnˇ y, Brendan´

McMahan, Nathan Srebro, and Jianyu Wang for their helpful discussions. AT and VS are supported in part by DARPA FA875017C0141, the National Science Foundation grants IIS1705121 and IIS1838017, an Okawa Grant, a Google Faculty Award, an Amazon Web Services Award, a JP Morgan A.I. Research Faculty Award, a Carnegie Bosch Institute Research Award, and the CONIX Research Center, one of six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of DARPA, the National Science Foundation, or any other funding agency.

# REFERENCES

Tensorflow federated: Machine learning on decentralized data. URL https://www.tensorflow.org/ federated.

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean,

J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M. K., Levenberg, J., Monga, R., Moore, S., Murray,

D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., and Zheng, X. Tensorflow: A system for large-scale machine learning. In Operating Systems Design and Implementation, 2016.

Allen-Zhu, Z. How to make the gradients small stochastically: Even faster convex and nonconvex sgd. In Advances in Neural Information Processing Systems, 2018.

Bonawitz, K., Eichner, H., Grieskamp, W., Huba, D., Ingerman, A., Ivanov, V., Kiddon, C., Konecny, J., Mazzocchi, S., McMahan, H. B., Overveldt, T. V., Petrou, D., Ramage, D., and Roselander, J. Towards federated learning at scale: system design. In Conference on Machine Learning and Systems, 2019.

Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. Distributed optimization and statistical learning via the alternating direction method of multipliers. Foundations and Trends in Machine Learning, 2010.

Caldas, S., Wu, P., Li, T., Konecnˇ y, J., McMahan, H. B.,` Smith, V., and Talwalkar, A. Leaf: A benchmark for federated settings. arXiv preprint arXiv:1812.01097, 2018.

Cohen, G., Afshar, S., Tapson, J., and van Schaik, A. Emnist: an extension of mnist to handwritten letters. arXiv preprint arXiv:1702.05373, 2017.

Dean, J., Corrado, G., Monga, R., Chen, K., Devin, M., Le, Q. V., Mao, M., Ranzato, M., Senior, A., Tucker, P., Yang, K., and Ng, A. Large scale distributed deep networks. In Advances in Neural Information Processing Systems, 2012.

Dekel, O., Gilad-Bachrach, R., Shamir, O., and Xiao, L. Optimal Distributed Online Prediction Using Mini-Batches. Journal of Machine Learning Research, 2012.

Ghadimi, S. and Lan, G. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. SIAM Journal on Optimization, 2013.

Go, A., Bhayani, R., and Huang, L. Twitter sentiment classification using distant supervision. CS224N Project Report, Stanford, 2009.

Goldblum, M., Reich, S., Fowl, L., Ni, R., Cherepanova,

V., and Goldstein, T. Unraveling meta-learning: Understanding feature representations for few-shot tasks. arXiv preprint arXiv:2002.06753, 2020.

Hao, Y., Rong, J., and Sen, Y. On the linear speedup analysis of communication efficient momentum sgd for distributed non-convex optimization. In International Conference on Machine Learning, 2019.

Huang, L., Yin, Y., Fu, Z., Zhang, S., Deng, H., and Liu, D. Loadaboost: Loss-based adaboost federated machine learning on medical data. arXiv preprint arXiv:1811.12629, 2018.

Jeong, E., Oh, S., Kim, H., Park, J., Bennis, M., and Kim, S.L. Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data. arXiv preprint arXiv:1811.11479, 2018.

Jiang, P. and Agrawal, G. A linear speedup analysis of distributed deep learning with sparse and quantized communication. In Advances in Neural Information Processing Systems, 2018.

Kaczmarz, S. Approximate solution of systems of linear equations. International Journal of Control, 1993.

Khodak, M., Balcan, M.-F. F., and Talwalkar, A. S. Adaptive gradient-based meta-learning methods. In Advances in Neural Information Processing Systems, 2019.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradientbased learning applied to document recognition. Proceedings of the IEEE, 1998.

Li, M., Andersen, D. G., Smola, A. J., and Yu, K. Communication efficient distributed machine learning with the parameter server. In Advances in Neural Information Processing Systems, 2014a.

Li, M., Zhang, T., Chen, Y., and Smola, A. J. Efficient minibatch training for stochastic optimization. In Conference on Knowledge Discovery and Data Mining, 2014b.

Li, T., Sahu, A., Talwalkar, A., and Smith, V. Federated learning: Challenges, methods, and future directions. arXiv preprint arXiv:1908.07873, 2019.

Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., and Smith, V. Feddane: A federated newton-type method. arXiv preprint arXiv:2001.01920, 2020.

Lin, T., Stich, S. U., and Jaggi, M. Don&apos;t use large minibatches, use local sgd. In International Conference on Learning Representations, 2020.

McMahan, H. B., Moore, E., Ramage, D., Hampson, S., and Arcas, B. A. y. Communication-efficient learning of deep networks from decentralized data. In International Conference on Artificial Intelligence and Statistics, 2017.

Pennington, J., Socher, R., and Manning, C. Glove: Global vectors for word representation. In Empirical Methods in Natural Language Processing, 2014.

Reddi, S. J., Konecnˇ y, J., Richt` arik, P., P´ ocz´ os, B., and´ Smola, A. Aide: Fast and communication efficient distributed optimization. arXiv preprint arXiv:1608.06879, 2016.

Richtarik, P. and Tak´ a´c, M. Distributed coordinate descentˇ method for learning with big data. Journal of Machine Learning Research, 2016.

Schmidt, M. and Roux, N. L. Fast convergence of stochastic gradient descent under a strong growth condition. arXiv preprint arXiv:1308.6370, 2013.

Shamir, O., Srebro, N., and Zhang, T. Communicationefficient distributed optimization using an approximate newton-type method. In International Conference on Machine Learning, 2014.

Smith, V., Chiang, C.-K., Sanjabi, M., and Talwalkar, A. S.

Federated multi-task learning. In Advances in Neural Information Processing Systems, 2017.

Smith, V., Forte, S., Ma, C., Takac, M., Jordan, M. I., and Jaggi, M. Cocoa: A general framework for communication-efficient distributed optimization. Journal of Machine Learning Research, 2018.

Stich, S. U. Local sgd converges fast and communicates little. In International Conference on Learning Representations, 2019.

Strohmer, T. and Vershynin, R. A randomized kaczmarz algorithm with exponential convergence. Journal of Fourier Analysis and Applications, 2009.

Vaswani, S., Bach, F., and Schmidt, M. Fast and faster convergence of sgd for over-parameterized models (and an accelerated perceptron). In International Conference on Artificial Intelligence and Statistics, 2019.

Wang, J. and Joshi, G. Cooperative sgd: A unified framework for the design and analysis of communication-efficient sgd algorithms. arXiv preprint arXiv:1808.07576, 2018.

Wang, S., Tuor, T., Salonidis, T., Leung, K. K., Makaya, C., He, T., and Chan, K. Adaptive federated learning in resource constrained edge computing systems. IEEE Journal on Selected Areas in Communications, 2019.

Woodworth, B. E., Wang, J., Smith, A., McMahan, B., and Srebro, N. Graph oracle models, lower bounds, and gaps for parallel stochastic optimization. In Advances in Neural Information Processing Systems, 2018.

Yao, Y., Rosasco, L., and Caponnetto, A. On early stopping in gradient descent learning. Constructive Approximation, 2007.

Yin, D., Pananjady, A., Lam, M., Papailiopoulos, D., Ramchandran, K., and Bartlett, P. Gradient diversity: a key ingredient for scalable distributed learning. In International Conference on Artificial Intelligence and Statistics, 2018.

Yu, H., Yang, S., and Zhu, S. Parallel restarted sgd for non-convex optimization with faster convergence and less communication. In AAAI Conference on Artificial Intelligence, 2018.

Zhang, S., Choromanska, A. E., and LeCun, Y. Deep learning with elastic averaging sgd. In Advances in Neural Information Processing Systems, 2015.

Zhang, Y., Duchi, J. C., and Wainwright, M. J.

Communication-efficient algorithms for statistical optimization. Journal of Machine Learning Research, 2013.

Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D., and Chandra, V. Federated learning with non-iid data. arXiv preprint arXiv:1806.00582, 2018.

Zhou, F. and Cong, G. On the convergence properties of a -step averaging stochastic gradient descent algorithm for nonconvex optimization. In International Joint Conference on Artificial Intelligence, 2018.*k*

Zhou, P., Yuan, X., Xu, H., Yan, S., and Feng, J. Efficient meta learning via minibatch proximal update. In Advances in Neural Information Processing Systems, 2019.

# A COMPLETE PROOFS

A.1 Proof of Theorem 4

*Proof.* Using our notion of -inexactness for each local solver (Definition 1*γ*), we can define such that:

∇F(w+1) + µ(w+1 − w) − e+1 = 0,*kktktttk*

|  |  |
| --- | --- |
| ketk+1k ≤ γk∇Fk(wt)k. | (3) |

Now let us define. Based on this definition, we know

*.* (4)

Let us define ¯ = µ − L− > 0 *µ*and = argminw hk(w;wt). Then, due to the ¯-strong convexity of , we have*µhk*

*.* (5)

Note that once again, due to the ¯-strong convexity of *µhk*, we know that. Now we can use

the triangle inequality to get

*.* (6)

Therefore,

*,*

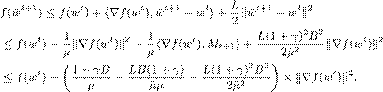
where the last inequality is due to the bounded dissimilarity assumption.

Now let us define *Mt*+1 such that, i.e..

We can bound kMt+1k:



where the last inequality is also due to bounded dissimilarity assumption. Based on the L-Lipschitz smoothness of and Taylor expansion, we have*f*

(8)

From the above inequality it follows that if we set the penalty parameter large enough, we can get a decrease in the objective value of ( ¯wt+1) − f(wt) which is proportional to k∇f(wt)k2. However, this is not the way that the algorithm works. In the algorithm, we only use devices that are chosen randomly to approximate ¯t*µ fK w*. So, in order to find the , we use local Lipschitz continuity of the function *f*.

*f*(w*t*+1) ≤ f( ¯w*t*+1) + L0kw*t*+1 − w¯*t*+1k, (9)

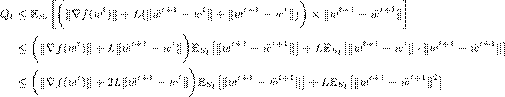
where is the local Lipschitz continuity constant of function and we have*L*0 *f*

*L*0 ≤ k∇f(wt)k + Lmax(kw¯t+1 − wtk,kwt+1 − wtk) ≤ k∇f(wt)k + L(kw¯t+1 − wtk + kwt+1 − wtk).

Therefore, if we take expectation with respect to the choice of devices in round we need to bound*t*

*,* (10)

where. Note that the expectation is taken over the random choice of devices to update.

(11)

From (7), we have that. Moreover,

(12)

and



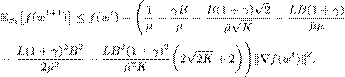
(from (6))

(13)

where the first inequality is a result of devices being chosen randomly to get and the last inequality is due to bounded dissimilarity assumption. If we replace these bounds in (11) we get*K wt*

(14)

Combining (8), (10), (9) and (14) and using the notation we get



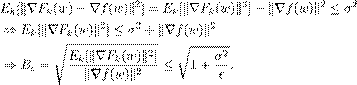


A.2 Proof for Bounded Variance

Corollary 10 (Bounded variance equivalence). Let Assumption *1 hold. Then, in the case of bounded variance, i.e.,*

*, for any* *it follows that**.*

Proof. We have,



With Corollary 10 in place, we can restate the main result in Theorem 4 in terms of the bounded variance assumption.

Theorem 11 (Non-convex FedProx convergence: Bounded variance). Let the assertions of Theorem *4 hold. In addition, let the iterate wt be such that* *, and let* *hold instead of the dissimilarity*

*condition. If µ, K and γ in Algorithm 2 are chosen such that*

*,*

*then at iteration t of Algorithm 2, we have the following expected decrease in the global objective:*

*,*

*where St is the set of K devices chosen at iteration t.*

The proof of Theorem 11 follows from the proof of Theorem 4 by noting the relationship between the bounded variance assumption and the dissimilarity assumption as portrayed by Corollary 10.

A.3 Proof of Corollary 7

In the convex case, where = 0 and√ µ¯ = µ, if = 0, i.e., all subproblems are solved accurately, we can get a decrease√*L*− *γ*

proportional to k∇f(wt)k2 if . In such a case if we assume 1 << B ≤ 0.5 K, then we can write*B < K*

*.* (15)

In this case, if we choose ≈ 6LB2 we get*µ*

*.* (16)

Note that the expectation in (16) is a conditional expectation conditioned on the previous iterate. Taking expectation of both sides, and telescoping, we have that the number of iterations to at least generate one solution with squared norm of gradient less than.

# B CONNECTIONS TO OTHER SINGLE-MACHINE AND DISTRIBUTED METHODS

Two aspects of the proposed work—the proximal term in FedProx, and the bounded dissimilarity assumption used in our analysis—have been previously studied in the optimization literature, but with very different motivations. For completeness,

we provide a discussion below on our relation to these prior works.

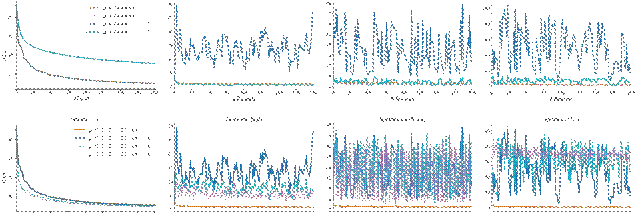
Proximal term. The proposed modified objective in FedProx shares a connection with elastic averaging SGD

(EASGD) (Zhang et al., 2015), which was proposed as a way to train deep networks in the data center setting, and uses a similar proximal term in its objective. While the intuition is similar to EASGD (this term helps to prevent large deviations on each device/machine), EASGD employs a more complex moving average to update parameters, is limited to using SGD as a local solver, and has only been analyzed for simple quadratic problems. The proximal term we introduce has also been explored in previous optimization literature with different purposes, such as Allen-Zhu (2018), to speed up (mini-batch) SGD training on a single machine, and in Li et al. (2014b) for efficient SGD training both in a single machine and distributed settings. However, the analysis in Li et al. (2014b) is limited to a single machine setting with different assumptions (e.g., IID data and solving the subproblem exactly at each round).

In addition, DANE (Shamir et al., 2014) and AIDE (Reddi et al., 2016), distributed methods designed for the data center setting, propose a similar proximal term in the local objective function, but also augment this with an additional gradient correction term. Both methods assume that all devices participate at each communication round, which is impractical in federated settings. Indeed, due to the inexact estimation of full gradients (i.e., ∇φ(w(t−1)) in Shamir et al. (2014, Eq (13))) with device subsampling schemes and the staleness of the gradient correction term (Shamir et al., 2014, Eq (13)), these methods are not directly applicable to our setting. Regardless of this, we explore a variant of such an approach in federated settings and see that the gradient direction term does not help in this scenario—performing uniformly worse than the proposed FedProx framework for heterogeneous datasets, despite the extra computation required (see Figure 4). We refer interested readers to Li et al. (2020) for more detailed discussions.

Finally, we note that there is an interesting connection between meta-learning methods and federated optimization methods (Khodak et al., 2019), and similar proximal terms have recently been investigated in the context of meta-learning for improved performance on few-shot learning tasks (Goldblum et al., 2020; Zhou et al., 2019).

                          Sy theti II Sy theti (0 0) Sy theti (0 5 0 5) Sy theti (1 1)



       0               25            50            75            100          125           150          175          200               0               25            50            75            100          125           150          175          200               0               25            50            75            100          125           150          175          200               0              25             50            75            100          125           150          175                 200

                           # Rou d # Rou d # Rou d # Rou d

*Figure 4.* DANE and AIDE (Shamir et al., 2014; Reddi et al., 2016) are methods proposed in the data center setting that use a similar proximal term as FedProx as well as an additional gradient correction term. We modify DANE to apply to federated settings by allowing for local updating and low participation of devices. We show the convergence of this modified method, which we call FedDane, on synthetic datasets. In the top figures, we subsample 10 devices out of 30 on all datasets for both FedProx and FedDane. While FedDane performs similarly as FedProx on the IID data, it suffers from poor convergence on the non-IID datasets. In the bottom figures, we show the results of FedDane when we increase the number of selected devices in order to narrow the gap between our estimated full gradient and the real full gradient (in the gradient correction term). Note that communicating with all (or most of the) devices is already unrealistic in practical settings. We observe that although sampling more devices per round might help to some extent, FedDane is still unstable and tends to diverge. This serves as additional motivation for the specific subproblem we propose in FedProx.

Bounded dissimilarity assumption. The bounded dissimilarity assumption we discuss in Assumption 1 has appeared in different forms, for example in Schmidt & Roux (2013); Yin et al. (2018); Vaswani et al. (2019). In Yin et al. (2018), the bounded similarity assumption is used in the context of asserting gradient diversity and quantifying the benefit in terms of scaling of the mean square error for mini-batch SGD for IID data. In Schmidt & Roux (2013); Vaswani et al. (2019), the authors use a similar assumption, called strong growth condition, which is a stronger version of Assumption 1 with. They prove that some interesting practical problems satisfy such a condition. They also use this assumption to prove optimal and better convergence rates for SGD with constant step-sizes. Note that this is different from our approach as the algorithm that we are analyzing is not SGD, and our analysis is different in spite of the similarity in the assumptions.

# C SIMULATION DETAILS AND ADDITIONAL EXPERIMENTS

C.1 Datasets and Models

Here we provide full details on the datasets and models used in our experiments. We curate a diverse set of non-synthetic datasets, including those used in prior work on federated learning (McMahan et al., 2017), and some proposed in LEAF, a benchmark for federated settings (Caldas et al., 2018). We also create synthetic data to directly test the effect of heterogeneity on convergence, as in Section 5.1.

• Synthetic: We set (α,β)=(0,0), (0.5,0.5) and (1,1) respectively to generate three non-identical distributed datasets (Figure 2). In the IID data (Figure 5), we set the same ∼ N(0,1) on all devices and to follow the same distribution N(v,Σ) where each element in the mean vector is zero and Σ is diagonal with Σj,j = j−1.2. For all synthetic datasets, there are 30 devices in total and the number of samples on each device follows a power law.*W,b Xk v*

• MNIST: We study image classification of handwritten digits 0-9 in MNIST (LeCun et al., 1998) using multinomial logistic regression. To simulate a heterogeneous setting, we distribute the data among 1000 devices such that each device has samples of only 2 digits and the number of samples per device follows a power law. The input of the model is a flattened 784-dimensional (28 × 28) image, and the output is a class label between 0 and 9.

• FEMNIST: We study an image classification problem on the 62-class EMNIST dataset (Cohen et al., 2017) using multinomial logistic regression. To generate heterogeneous data partitions, we subsample 10 lower case characters (&apos;a&apos;-&apos;j&apos;) from EMNIST and distribute only 5 classes to each device. We call this federated version of EMNIST FEMNIST. There are 200 devices in total. The input of the model is a flattened 784-dimensional (28 × 28) image, and the output is a class label between 0 and 9.

• Shakespeare: This is a dataset built from The Complete Works of William Shakespeare (McMahan et al., 2017). Each speaking role in a play represents a different device. We use a two-layer LSTM classifier containing 100 hidden units with an 8D embedding layer. The task is next-character prediction, and there are 80 classes of characters in total. The model takes as input a sequence of 80 characters, embeds each of the characters into a learned 8-dimensional space and outputs one character per training sample after 2 LSTM layers and a densely-connected layer.

• Sent140: In non-convex settings, we consider a text sentiment analysis task on tweets from Sentiment140 (Go et al., 2009) (Sent140) with a two layer LSTM binary classifier containing 256 hidden units with pretrained 300D GloVe embedding (Pennington et al., 2014). Each twitter account corresponds to a device. The model takes as input a sequence of 25 characters, embeds each of the characters into a 300-dimensional space by looking up Glove and outputs one character per training sample after 2 LSTM layers and a densely-connected layer.

C.2 Implementation Details

(Implementation) In order to draw a fair comparison with FedAvg, we use SGD as a local solver for FedProx, and adopt a slightly different device sampling scheme than that in Algorithms 1 and 2: sampling devices uniformly and averaging updates with weights proportional to the number of local data points (as originally proposed in McMahan et al. (2017)). While this sampling scheme is not supported by our analysis, we observe similar relative behavior of FedProx vs. FedAvg whether or not it is employed (Figure 12). Interestingly, we also observe that the sampling scheme proposed herein results in more stable performance for both methods. This suggests an added benefit of the proposed framework.

(Machines) We simulate the federated learning setup (1 server and devices) on a commodity machine with 2 IntelXeon*N* R R E5-2650 v4 CPUs and 8 NVidia 1080Ti GPUs.

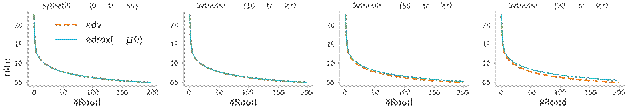
(Hyperparameters) We randomly split the data on each local device into an 80% training set and a 20% testing set. We fix the number of selected devices per round to be 10 for all experiments on all datasets. We also do a grid search on the learning rate based on FedAvg. We do not decay the learning rate through all rounds. For all synthetic data experiments, the learning rate is 0.01. For MNIST, FEMNIST, Shakespeare, and Sent140, we use the learning rates of 0.03, 0.003, 0.8, and 0.3. We use a batch size of 10 for all experiments.

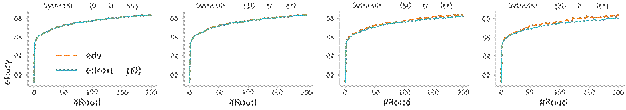
(Libraries) All code is implemented in Tensorflow Version 1.10.1 (Abadi et al., 2016). Please see github.com/litian96/FedProx for full details.

C.3 Additional Experiments and Full Results

*C.3.1 Effects of Systems Heterogeneity on IID Data*

We show the effects of allowing for partial work on a perfect IID synthetic data (Synthetic IID).



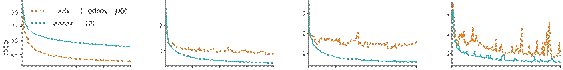


*Figure 5.* FedAvg is robust to device failure with IID data. In this case, whether incorporating partial solutions from the stragglers would not have much effect on convergence.

*C.3.2 Complete Results*

In Figure 6, we present testing accuracy on four synthetic datasets associated with the experiments shown in Figure 2.

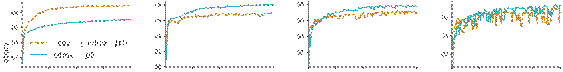
                                Sy theti II 3 Sy theti (0 0) Sy theti (0 5 0 5) Sy theti (1 1)



                     0           50         100         150         200               0           50          100         150         200               0           50         100         150         200               0            50         100         150        200

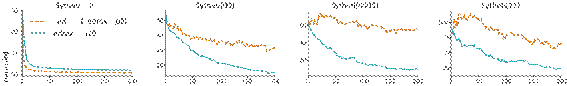
                                  # Rou d # Rou d # Rou d # Rou d

                                Sy theti II Sy theti (0 0) Sy theti (0 5 0 5) Sy theti (1 1)



                     0           50         100         150         200               0           50          100         150         200               0           50         100         150         200               0            50         100         150        200

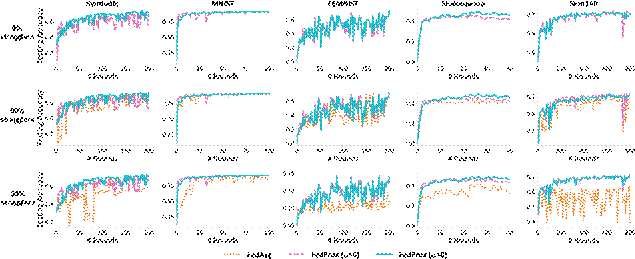
                                  # Rou d # Rou d # Rou d # Rou d



                                  # Rou d # Rou d # Rou d # Rou d

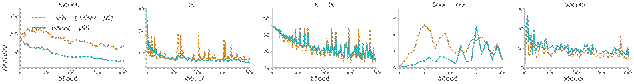
*Figure 6.* Training loss, test accuracy, and dissimilarity measurement for experiments described in Fig. 2.

In Figure 7, we show the testing accuracy associated with the experiments described in Figure 1. We calculate the accuracy improvement numbers by identifying the accuracies of FedProx and FedAvg when they have either converged, started to diverge, or run sufficient number of rounds (e.g., 1000 rounds), whichever comes earlier. We consider the methods to converge when the loss difference in two consecutive rounds |ft − ft−1| is smaller than 0.0001, and consider the methods to diverge when we see − ft−10 greater than 1.*ft*



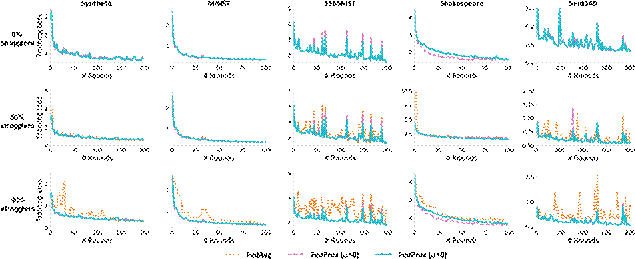
*Figure 7.* The testing accuracy of the experiments in Figure 1. FedProx achieves on average 22% improvement in terms of testing accuracy in highly heterogeneous settings (90% stragglers).

In Figure 8, we report the dissimilarity measurement on five datasets (including four real datasets) described in Figure 1. Again, the dissimilarity characterization is consistent with the real performance (the loss).

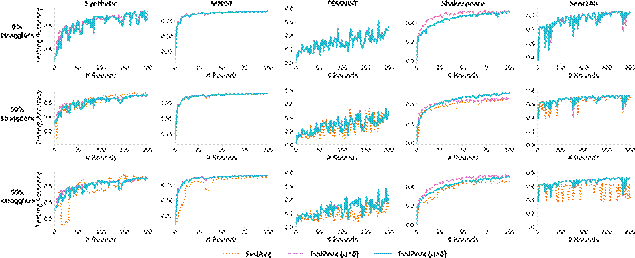


*Figure 8.* The dissimilarity metric on five datasets in Figure 1. We remove systems heterogeneity by only considering the case when no participating devices drop out of the network. Our dissimilarity assumption captures the data heterogeneity and is consistent with practical performance (see training loss in Figure 1).

In Figure 9 and Figure 10, we show the effects (both loss and testing accuracy) of allowing for partial solutions under systems heterogeneity when = 1 (i.e., the statistical heterogeneity is less likely to affect convergence negatively).*E*



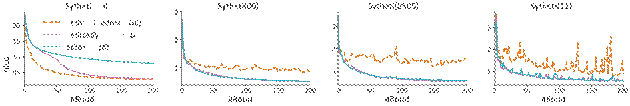
*Figure 9.* The loss of FedAvg and FedProx under various systems heterogeneity settings when each device can run at most 1 epoch at each iteration (= 1). Since local updates will not deviate too much from the global model compared with the deviation under large &apos;s, it is less likely that the statistical heterogeneity will affect convergence negatively. Tolerating for partial solutions to be sent to the central server (FedProx, = 0) still performs better than dropping the stragglers (FedAvg).*E Eµ*



*Figure 10.* The testing accuracy of the experiments shown in Figure 9.

*C.3.3 Adaptively setting µ*

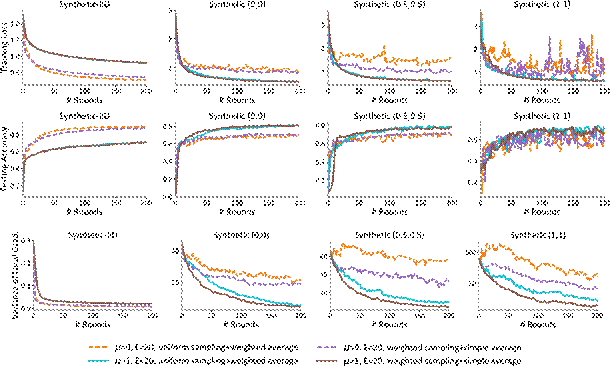
One of the key parameters of FedProx is . We provide the complete results of a simple heuristic of adaptively setting on four synthetic datasets in Figure 11. For the IID dataset (Synthetic-IID), starts from 1, and for the other non-IID datasets, starts from 0. Such initialization is adversarial to our methods. We decrease by 0.1 when the loss continues to decrease for 5 rounds and increase by 0.1 when we see the loss increase. This heuristic allows for competitive performance. It could also alleviate the potential issue that 0 might slow down convergence on IID data, which rarely occurs in real federated settings.*µµ µ µ µ µ µ >*



*Figure 11.* Full results of choosing adaptively on all the synthetic datasets. We increase by 0.1 whenever the loss increases and decreases it by 0.1 whenever the loss decreases for 5 consecutive rounds. We initialize to 1 for the IID data (Synthetic-IID) (in order to be adversarial to our methods), and initialize it to 0 for the other three non-IID datasets. We observe that this simple heuristic works well in practice.*µ µ µ*

*C.3.4 Comparing Two Device Sampling Schemes*

We show the training loss, testing accuracy, and dissimilarity measurement of FedProx on a set of synthetic data using two different device sampling schemes in Figure 12. Since our goal is to compare these two sampling schemes, we let each device perform the uniform amount of work (= 20) for both methods.*E*



*Figure 12.* Differences between two sampling schemes in terms of training loss, testing accuracy, and dissimilarity measurement. Sampling devices with a probability proportional to the number of local data points and then simply averaging local models performs slightly better than uniformly sampling devices and averaging the local models with weights proportional to the number of local data points. Under either sampling scheme, the settings with = 1 demonstrate more stable performance than settings with = 0.*µ µ*