arXiv:2305.04279v1 [cs.DC] 7 May 2023

通过容错传输协议提升分布式机器学习训练性能

陈子轩†、史磊†、刘宣东†、艾欣†、刘森†‡和徐洋†◇*

†中国上海复旦大学计算机科学学院
‡中国上海复旦大学金融科技研究院◇中国深圳鹏城实验室

摘要—**分布式机器学习（DML）系统被用于提高数据中心（DC）和边缘节点中的模型训练速度。参数服务器（PS）通信架构得到了广泛应用，但其面临由多对一“扇入”流量模式引起的严重长尾延迟问题，从而对训练吞吐量产生负面影响。为了解决这一挑战，我们设计了容错传输协议（LTP），该协议允许在同步过程中部分丢弃梯度，以避免不必要的重传，并有助于加快每次迭代的同步速度。LTP 通过** 乱序传输 **和** 乱序确认（***ACK***）**实现容错传输。LTP 采用** 提前关闭**，根据网络状况调整容忍丢包阈值，并利用** 气泡填充 **进行数据校正，以保持训练准确率。LTP 使用 C++ 实现，并集成到 PyTorch 中。在由 8 个工作节点和 1 个参数服务器节点组成的测试平台上进行的评估表明，与传统 TCP 拥塞控制机制相比，LTP 能够将分布式机器学习训练任务的吞吐量最高提升 30 倍，同时不会牺牲最终准确率。**

索引词—**分布式机器学习、传输协议、参数服务器。**

I. 引言

随着机器学习（ML）应用中的数据集和模型规模激增，分布式机器学习（DML）已被广泛采用，以利用多个工作节点的算力进行大规模训练。为实现分布式工作节点之间的同步，人们提出了多种分布式机器学习通信架构，例如参数服务器（PS）架构 [1] 和环形全归约（Ring-AllReduce） [2], ，从而提高了分布式训练系统的效率。

由于简单高效，参数服务器架构已成为分布式机器学习中一种普遍采用的通信架构。在参数服务器架构中，所有计算节点（也称为工作节点）由一个或多个参数服务器管理。工作节点学习训练数据集的一部分，并进行通信。

* 通讯作者：Yang Xu (xuy@fudan.edu.cn) 第一作者：Zixuan Chen (zxchen20@fudan.edu.cn) 本研究得到广东省重点领域研发计划（2021B0101400001）、国家自然科学基金（62150610497、62172108、62002066）、上海市自然科学基金（23ZR1404900）、上海市浦江人才计划（2020PJD005）、鹏城实验室重大重点项目（PCL2021A15）以及之江实验室开放研究项目（2022QA0AB07）的资助。本文将发表于 IWQoS 2023。仅供预览。

979-8-3503-9973-8/23/$31.00 ©2023 IEEE

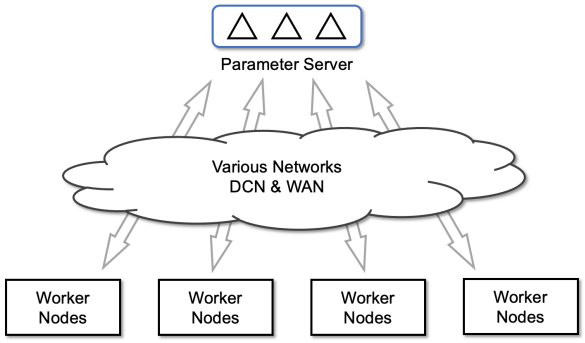


图 1. 分布式机器学习训练中的参数服务器架构。

与关联的参数服务器同步训练结果，以便和其他工作节点进行同步（如图 1 所示）。在每次迭代中，每个工作节点使用一部分数据集训练模型，并将计算得到的梯度发送给参数服务器。参数服务器汇总所有工作节点的梯度，并更新全局模型。最后，参数服务器将最新模型发送给每个工作节点，为下一次迭代做好准备。多个工作节点并行计算，极大地提高了模型训练过程的效率。

随着 DML 系统规模不断扩大，连接工作节点与参数服务器的网络正成为系统的一项主要瓶颈。这主要有两个原因。首先，现有的大多数分布式机器学习训练任务采用 Bulk Synchronous Parallel （ BSP ）同步模型 [3], ，在该模型中，每个训练批次结束时，所有节点都必须与参数服务器充分同步其梯度。这种多对一的汇聚流量模式会导致长尾延迟问题 [4], ，显著缩短通信时间。其次，近来出现了若干基于 DML 的新场景，例如跨数据中心机器学习 [5], 、边缘计算 [6], [7],和联邦学习（FL） [8]。这些场景通常需要通过广域网（WAN）或无线网络传输梯度，因此面临链路不稳定带来的新挑战。因此，这些场景中非拥塞丢包十分常见。例如，数据中心网络中的微突发流量 [9]、[11], 物理链路故障（如光纤故障） [12], [13], 、边缘节点的无线链路 [14], [15],以及广域网中的重新路由 [16] 都可能导致非拥塞数据包

丢失，并降低参数服务器与工作节点之间的通信效率。汇聚流量和非拥塞丢包造成的长尾延迟会严重降低分布式机器学习训练的效率。

当前用于解决 incast 引发的长尾延迟问题的方案，例如 pHost [17] 和 Homa [18],，并未满足分布式机器学习训练的特定需求，也无法有效应对非拥塞丢包。此外，由于数值分析过程的特性，分布式机器学习训练允许一定程度的数据丢失。为改善分布式机器学习的通信，已有多项研究提出了参数量化 [19], 、参数剪枝 [20], 、梯度压缩 [21],和梯度量化 [22]等方法。这些方法旨在通过在应用层优化通信过程，减少每次迭代传输的数据量。然而，这些方案仍有局限：它们只能减少通信数据量，却无法解决同步期间长尾延迟问题的根本原因。

本文提出了一种名为容错传输协议（LTP）的新方案，旨在提升 DML 训练任务的同步效率。该协议在系统同步梯度时允许部分数据丢失（容错传输），有助于缓解分布式机器学习系统中常见的长尾延迟问题。LTP 通过采用乱序传输和乱序确认实现容错传输。为此，LTP 使用提前关闭机制确定容错传输的阈值，并采用气泡填充机制防止数据错误。总而言之，我们的贡献如下：

1) 我们提出了 LTP，这是一种通过允许部分数据丢失来提高分布式机器学习训练同步效率的传输协议。我们设计了两种关键机制：提前关闭和气泡填充。提前关闭可根据预先设定的数据百分比和传输时间阈值（即我们所称的容错传输）提前完成传输，而气泡填充 则能确保机器学习任务的正确性和准确性。基于带宽时延积（BDP）的拥塞控制算法可确保在各种网络条件下实现较高的带宽利用率。

2) 我们使用 C++ 在 Linux 上实现了 LTP，并将其集成到广泛使用的机器学习框架 PyTorch 中。LTP 对机器学习框架透明，因此分布式机器学习程序员无需修改其机器学习代码。若要将 LTP 用作通信协议，程序员只需对套接字接口进行简单修改。

3）我们在由 8 个工作节点和一个参数服务器节点组成的真实测试平台上，评估了 LTP 在各种网络条件下的性能。评估采用了基于 CIFAR10 数据集的热门机器学习模型（如 ResNet50 和 VGG16）。评估结果表明，与传统拥塞控制算法相比，LTP 可在不损失精度的情况下，将训练速度提升至最高 30 倍。

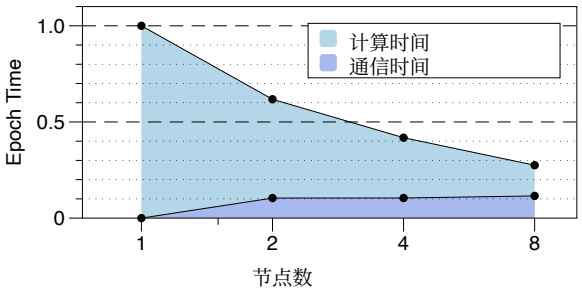


图 2. 分布式机器学习训练任务的可扩展性问题：分布式机器学习训练任务的训练时间和工作节点数量并未按比例减少。

本文其余部分安排如下。第 II 节介绍 LTP 的背景和动机。第 III 节介绍 LTP 的设计及其两项关键机制：气泡填充和提前关闭。第 IV 节介绍 LTP 的实现，第 V 节展示评估结果。第 VI 节进行讨论。第 VII 节总结相关工作，最后第 VIII 节对本文作出总结。

II. 背景与动机

A. 网络成为分布式机器学习扩展的限制因素

尽管近年来网络带宽迅速提升，但它仍然是分布式机器学习训练的一大瓶颈。我们分别在采用参数服务器通信架构的情况下，使用 1、2、4 和 8 台机器对 ResNet50 [23] 模型进行分布式机器学习训练评估。图 2 显示，随着计算节点规模的增加，分布式机器学习的训练效率确实有所提高（每个训练轮次所需的时间逐渐减少），但额外的通信开销也在逐渐增加（通信时间与计算时间之比不断上升）。节点数量的增加与其为整体训练时间带来的优化幅度并不成正比。

我们推测这是由参数服务器架构中的汇聚流量造成的，即多个工作节点并行且突发地与少数几个参数服务器通信。虽然大多数节点几乎可以同时完成传输，但由于长期处于竞争关系，部分节点的拥塞窗口增长缓慢。这些滞后流会拖慢整体训练同步，这种情况在数据中心网络和广域网中十分常见。

我们进行了另一项实验，以说明长尾延迟带来的危害。在默认 TCP 协议参数下，我们使用 8 个工作节点和一个参数服务器构建多对一通信，并统计每个工作节点的流完成时间。图 3 展示了流完成时间（FCT）的概率密度分布。可以看出，大多数流的流完成时间分布相对相似，但仍有一些“饥饿”流的流完成时间相对较长。由于现有流行的 DML 训练任务仍采用整体同步并行模型，系统必须等待所有工作节点完成同步后才能继续，从而降低了整体训练吞吐量。

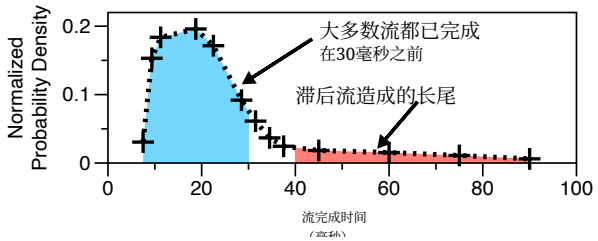


图 3. 汇聚流量模式导致的长尾延迟。

B. TCP 在有损网络中表现不佳

	丢包率	0.00%	0.01%	0.10%	0.50%	1.00%	3.00%	5.00%
1 Gbps 40ms	Cubic	0.00%	0.00%	0.00%	-0.14%	-6.44%	-48.58%	-78.19%
	reno	-0.14%	-0.21%	-0.14%	-0.07%	-3.40%	-43.56%	-88.10%
	DCTCP	-0.21%	-0.14%	-0.14%	-0.28%	-1.84%	-42.71%	-83.29%
	BBR	-0.92%	-0.71%	-0.92%	-3.33%	-5.03%	-7.15%	-13.53%
10 Gbps 1ms	Cubic	-0.16%	-15.06%	-74.95%	-90.73%	-94.28%	-99.11%	-99.80%
	reno	-0.06%	-16.53%	-76.28%	-91.69%	-95.81%	-98.64%	-99.38%
	DCTCP	-0.15%	-20.19%	-75.24%	-92.71%	-95.17%	-98.61%	-99.45%
	BBR	-1.40%	-1.77%	-3.59%	-12.74%	-18.49%	-22.63%	-34.95%

图 4. 存在非拥塞性丢包的网络中，不同的 TCP 拥塞控制算法。

经过数十年的发展，常用的 TCP 拥塞控制算法已具备很强的鲁棒性。然而，在存在非拥塞丢包的网络环境中，这些拥塞控制算法仍有不足。我们开展了一系列实验，以评估 TCP 在不稳定网络中的问题：使用现有流行的 TCP 拥塞控制算法，测试不稳定网络环境下的点对点纯流量性能。我们分别在数据中心网络和广域网中进行了这些实验。

图 4 展示了不同 TCP 拥塞控制算法在各种网络中的带宽利用率下降情况。结果表明，传统 TCP 拥塞控制算法在存在丢包的网络中表现不佳，尤其是在带宽较高、延迟较低的网络中。尽管 BBR [24] 在丢包环境中的表现较好，但其性能下降幅度仍大于非拥塞性丢包率。这是因为传统 TCP 拥塞控制具有保序特性，并使用 3 个重复确认作为瓶颈链路队列已满的信号，以实现拥塞避免。LTP 以乱序方式发送数据包，并采用类似 BBR 的基于带宽时延积的拥塞控制算法来维持链路利用率，因此其性能优于通常使用的拥塞控制算法。

C. 随机数据丢失导致的精度下降可以接受

分布式机器学习是一种数值分析过程，包含数百或数千次迭代，因此，每次迭代中一定程度的数据丢失不会影响模型性能。人们已提出许多方法，利用这一容错特性提高分布式机器学习训练的效率。这些方法分为两大类，包括

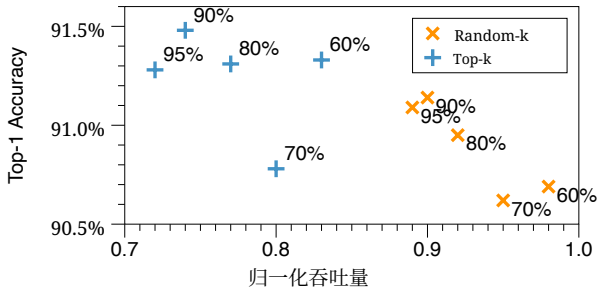


图 5. 在 CIFAR10 数据集上，Top-k 与 Random-k 的 ResNet18 Top-1 准确率和归一化吞吐量对比。

梯度量化和稀疏化 [25]。梯度量化方法使用低比特浮点数存储待传输的数据，例如，可用一个 8 位浮点数近似表示一个 32 位浮点数，从而将通信开销降至四分之一。梯度稀疏化则从梯度向量中选取一部分数据进行传输。Top-k [21] 算法仅传输梯度向量中绝对值最大的前 k 个元素。Random-k [26]则从梯度向量中随机传输一部分数据，与 Top-k 相比可减少排序开销。这两种方法可以结合使用。例如，DGC [25]会将梯度稀疏化与预热训练 [27]、动量校正 [28]等其他训练技巧结合起来。

尽管这些方法能够减少通信数据量，但仍然存在一些挑战：需要经过复杂考量才能确定应丢弃梯度的阈值。例如，保持通信数据量较小（尽可能多地丢弃梯度）可以缩短每个同步轮次的完成时间，尤其是在性能较差的网络中。然而，这样做的缺点是可能引入额外的计算开销，甚至可能超过减少通信所带来的时间优化效果。

在上述方法中，Random-k 丢包和 Top-k 丢包是最常用的两种方法，它们各有优缺点。我们使用 CIFAR10 数据集和 ResNet18 模型 [29] 来探讨这一问题。我们保证 k% 的梯度参与同步，并在 8 个工作节点和 1 个参数服务器的配置下，比较随机丢弃 1-k% 的梯度（Random-k）与保留 Top-k% 梯度（Top-k）在 Top-1 准确率和吞吐量方面的差异。我们使用 CUDA 内置的 topk 函数 [30]，以确保 Top-k 算法足够高效。k 的取值范围为 5 到 40。结果（图 5）表明，Random-k 能够达到相对较高的准确率，并且由于其简单性，训练吞吐量更高。当 $k \leq 70$ 时，Top-k 与 Random-k 的 Top-1 准确率差异仅约为 0.3%，但吞吐量提高了约 25%。这为容忍丢包的传输协议设计提供了指导。

其他研究也提出了类似的结果 [31]。LTP 的行为类似于近似阈值控制的 Random-k，这意味着分布式机器学习训练的最终结果不会

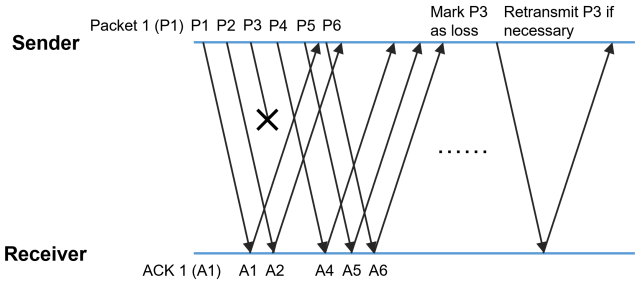


图 6. LTP 中的容损传输。

因有限的和随机的数据丢弃而受到巨大影响。

III. LTP 的设计

LTP 是一种用于各种网络环境下分布式机器学习训练的容错传输协议。我们遵循以下几项基本原则来设计 LTP:

1) 确保引入 LTP 对现有分布式机器学习训练同步机制的影响最小化。2) 减轻汇聚流量模式和非拥塞丢包导致的长尾延迟问题。3) 降低丢包对传输窗口的影响，并避免因非拥塞原因导致的虚假链路拥塞信号。

本节将聚焦于这些基础知识，并介绍 LTP 的设计理念及其原因。

A. LTP 概述

与 TCP 不同，LTP 采用两种核心方案，即**乱序分组传输**和**逐分组确认**，以确保容错传输。对于 TCP，要传输的字节流会被拆分为多个数据段，每个数据段都被视为字节流中一个需要按序传送的独立部分。然而，分布式机器学习中的数据传输无需保序，因此 LTP 采用乱序传输和乱序确认来提升协议性能并支持容错传输。

例如，在图6中，分组3（P3）在收到三个乱序ACK后被标记为丢失。然而，LTP不会实时重传P3，而是等待所有已发送数据包完成后，再考虑是否进行重传。发送端是否在最后重传数据，取决于提前关闭机制所维护的阈值（参见§ III-B）。

要实现容错传输并确保分布式机器学习正常运行，面临着若干挑战。其中一个最重要的挑战是确定应传输多少数据。LTP 提出了基于实时网络质量的提前关闭机制，可动态调整传输阈值，详见 § III-B。另一项挑战是如何处理容错传输导致的数据错误。为应对这一挑战，提出了在接收方进行气泡填充以恢复数据的方法（§ III-C）。

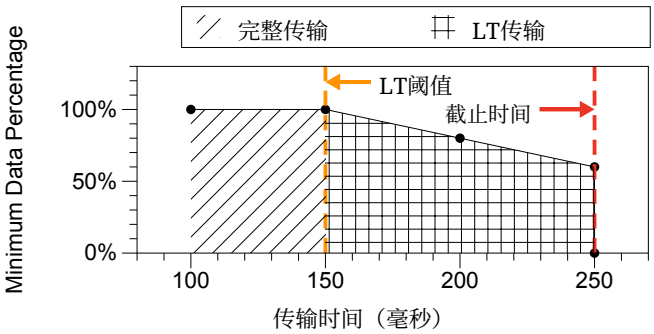


图 7. 提前关闭机制的双重阈值。

本节还讨论了其他设计细节，包括拥塞控制等。

B. 通过提前关闭降低长尾延迟

为避免多对一通信场景中的长尾延迟拖慢全局训练时间，我们引入“提前关闭”机制。提前关闭允许接收方在传输已接收到一定比例的数据，且所有包含元数据的关键分组均已确认送达后，关闭流（§ III-E）。提前关闭机制有两个指标：传输时间和已接收数据的百分比。

受显式拥塞通知（ECN）的双阈值设计启发，我们在传输时间上选择两个阈值，作为一次批次同步传输的上下界。较小的阈值称为容错阈值（LT阈值），较大的阈值称为截止时间（图7）。在LT阈值之前，接收方将等待所有数据包到达。在两个阈值之间，接收方根据已接收数据的百分比提前结束传输。例如，在图7中，当传输进行到200毫秒时，如果接收方（参数服务器）接收到的数据超过所有发送方（工作节点）数据的80%，则传输将结束。超过截止时间后，无论接收到多少数据，接收方都会立即停止接收数据。接收方将主动向发送方广播“停止”消息，以通知其提前关闭。

1) 更新容忍丢包阈值：在提前关闭机制中，阈值配置是一个重要环节。合适的 LT 阈值可以在确保丢弃少量数据的同时，显著减少多对一通信场景导致的流滞后现象。实践中，我们发现将 LT 阈值配置为待传输梯度的预期完成时间（ECT）可以获得更好的性能。ECT 可按 $ECT = RTprop + ModelSize/BtlBw$ 计算。

然而，丢包可能会导致 LTP 的拥塞控制（§ III-D）测得的瓶颈带宽（BtlBw）和往返传播时间（RTprop）出现误差，使得 ECT 小于模型传输所需的时间。虽然不同训练轮次之间可以共享阈值（对于需要更新的固定数量梯度，流大小保持不变），但仍需进行调整，因为网络中的其他竞争流

是动态变化的。综合考虑这些情况，我们将每个训练轮次第一个批次的LT阈值初始值设为 $LTThreshold_{init} = 1.5 * RTprop + ModelSize/BtlBw$ 。

LT阈值在每个训练轮次中根据最短的100%梯度传输时间进行更新。每条点对点链路的LT阈值相互独立，并根据当前链路的最优同步时间进行调整。截止时间同时应用于同一接收方的所有接收链路，其取值为 $Max(LTThreshold) + C$。常数 C 由用户定义。我们在数据中心网络（DCN）中将其设置为30毫秒，在广域网（WAN）中将其设置为100毫秒。这些值是通过实验得出的经验值，在评估中能够获得更好的性能。

2）聚集与广播过程中的不同*LT*阈值：LTP通过提前关闭机制避免汇聚流量导致的长尾延迟。在实际应用中，参数服务器架构下的分布式机器学习训练包含两个过程：聚集（工作节点将其训练梯度发送给参数服务器）和广播（参数服务器聚合梯度后再将其发送回工作节点）。这两个过程对数据丢失的容忍度不同。显而易见，聚集过程允许数据丢失，而广播过程则不需要容忍数据丢失，这分别是由1）分布式机器学习本身的特性，以及2）参数服务器架构所导致的汇聚流量模型决定的。

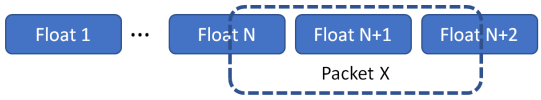
从分布式机器学习训练的特点来看，其带来的加速效果是通过多个计算节点划分数据集，并同时进行并行计算。因此，我们认为，不同机器上的机器学习模型应保持一致，以避免全局模型混乱。例如，使用异步并行同步模型（ASP）的分布式机器学习任务 [32] 无法保证工作节点之间训练模型的同步，因此会出现最终训练准确率较低，甚至无法收敛的问题。因此，容错传输仅在聚集阶段发挥作用，但能确保广播阶段的所有数据均得到传输。

从流量模式的角度来看，广播阶段的流量并非资源竞争激烈的汇聚模式，而是一对多模式。这种流量模式不会导致长尾延迟，因此不需要提前关闭机制。

C. 使用气泡填充避免数据错误

由于 LTP 的设计目标是支持容错传输，因此可以接受丢弃数据包而不进行重传。这些丢失的数据包可能导致严重的数据错误，为此我们提出气泡填充机制，以修复容错传输造成的问题。具体而言，气泡是由全零组成的可变长度字符串；为确保正确性，我们使用两种气泡，分别称为“数据包气泡”和“填充气泡”。

对于 LTP接收方认为不应（重新）传输的数据包，接收方会用数据包气泡填充。在这种情况下，这些气泡的长度与数据包相同。由于最大传输单元（最大传输单元）将



(a) 浮点数被拆分所带来的问题。



(b) 填充气泡可以避免浮点数被拆分。

图 8. 气泡填充可用于纠正因浮点数被拆分而导致的严重数据错误。

在传输过程中保持不变，接收方可以根据上下文数据包推断数据包气泡的长度。

然而，在某些情况下，这种操作可能导致接收方收到错误的分段。例如，当一个浮点数被拆分为两个分段，且其中只有一个分段用 0 填充时，接收方获得的数值可能会出错，如图 8(a) 所示。好消息是，分布式机器学习训练中待传输的大多数梯度都采用相同的数据类型，例如 float32（占用相同的内存空间）。因此，LTP 使用填充气泡（图 8(b)），以确保不会出现错误分段。

D. 拥塞控制

在数据中心网络和广域网中，分布式机器学习训练任务都要与其他流争用链路带宽、交换机队列和网卡缓冲区。需要探讨不同流量之间公平性与效率的权衡。通常会提出不同的拥塞控制算法来解决这一问题。传统的 TCP 拥塞控制通过对拥塞窗口大小进行加性增、乘性减（AIMD）来调整发送速率，并利用 3 个重复确认作为丢包信号，确保数据完整且按序到达。然而，广域网中的非拥塞丢包可能会触发错误的拥塞信号，从而阻碍链路利用率达到最大。相比之下，BBR 是一种新型拥塞控制算法，它将带宽时延积作为在途数据包数量的上限，并通过发送节奏控制避免缓冲区溢出，因此在存在非拥塞丢包的网络中表现良好。

LTP 的拥塞控制算法基于带宽时延积，并在发送端生效；该算法不会将识别到的数据包丢失用作调整拥塞窗口的信号。LTP 分别通过定期探测往返传播时间和瓶颈带宽来估算带宽时延积，并将其作为在途数据包数量的上限。我们无法直接使用 BBR，因为它构建于 TCP 之上，且更难修改。因此，由于 LTP 在用户空间（UDP）运行，无法精确实施发送节奏控制。我们采用近似的发送节奏控制方案：当同时待发送的数据包数量超过 20 个时（10G

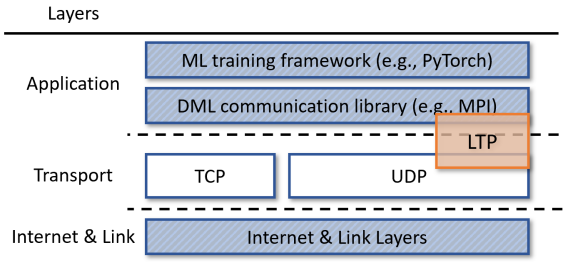


图 9. LTP 基于 UDP 设计。因此，它主要位于传输层，并包含一些应用层功能。

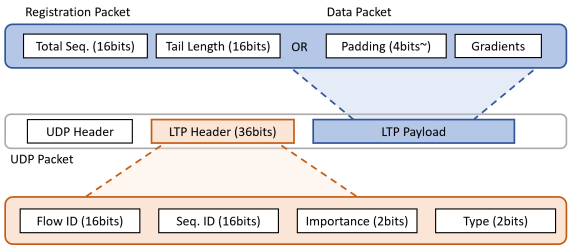


图 10. LTP 数据包结构。

链路，最大传输单元为 1500 字节，30KB）（发送窗口稳定时，这种情况并不常见）。

E. 数据包优先级

许多分布式机器学习中的梯度压缩方法都讨论了如何对不同数据进行排序的重要性。在 LTP 中，数据的重要性同样值得探讨。LTP 将数据包分为两种优先级，以不同的可靠性传输：1）关键分组；2）普通数据包。LTP 确保所有关键分组在传输过程中都能被接收方 100% 接收，而普通数据包则可能被部分丢弃。LTP 允许用户自定义关键分组的选择，同时仅将最少数量的消息标记为关键消息。

LTP 仅标记矩阵中不可或缺的字节（例如，矩阵比特流首尾部分的若干字节）。LTP 不会遍历矩阵数据所代表内容的具体细节，例如每个数据的绝对值，因为 1）遍历整个矩阵耗时较长；2）根据数据的绝对值来判断重要性并不理想（参见 § II-C）。

IV. LTP 的实现

我们使用 C++ 在 Linux（Ubuntu 18.04.2）上部署 LTP 协议，并将其集成到现有的、广泛使用的深度学习框架 PyTorch 中。LTP 基于 UDP 部署，并对分布式机器学习通信库进行了修改（图 9）。本节介绍 LTP 实现的设计细节，包括 LTP 数据包的数据结构以及如何进行队列管理。

A. 数据结构

LTP 重新设计了数据包的数据结构（图 10）。LTP 运行于 UDP 之上，与大多数自定义传输协议（例如，QUIC [33], RoCEv2 [34]）类似。

为了优化数据包传输效率，LTP 仅增加了一个额外的 68 位首部（约 9B ）。LTP 首部包含流标识符、序列标识符、数据包的重要性、数据包类型、往返传播时间和瓶颈带宽等字段。每轮传输都被视为一个独立的流。接收方通过记录发送端的地址和流标识符来区分不同的流。序列标识符表示 Jigsaw 片段的顺序。

（数据段）。重要性字段用于标记数据包的重要程度；在初始设计中，重要性字段仅包含两类：关键（11）和非关键（00）。类型字段用于标记数据包的属性，包括注册数据包（00）、数据包（01）、确认（10）和结束（11）。LTP 将拥塞控制信息发送给接收方，其中包含往返传播时间和瓶颈带宽。

有效载荷字段在不同类型的消息中承担不同的作用。在注册数据包中，有效载荷字段存放特定流的数据段总数。在数据包中，该字段用于保存分段后的数据。在 ACK 数据包中，有效载荷字段为空。

B. 队列实现

为了确保不同优先级的数据包采用不同的传输顺序，我们设计了 3 个队列。其中两个是先进先出（FIFO）队列，用于传送数据包，分别是**关键队列（CQ）**（图 11(a)）和**普通队列（NQ）**（图 11(b)）。另一个队列是**重传队列（RQ）**，它采用随机入队、先进先出的方式。

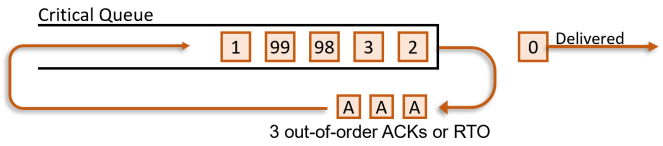
LTP 将待传输的字节流拆分成数据包，然后根据规则将其放入两个队列之一。NQ 中的数据包在 CQ 中的数据包之后发送，而被识别为丢失的数据包会重新插入 CQ（原本在 CQ 中的数据包）或 RQ（原本在 NQ 中的数据包），等待重传。RQ 用于存储被识别为丢失的数据包；当 CQ 和 NQ 中的所有消息都传输完毕后，RQ 才会开始发送数据包。

LTP 通过三个乱序确认来识别丢包。LTP 在发送端维护一个实际的数据包发送序列队列，并以此作为判断确认到达顺序的依据。

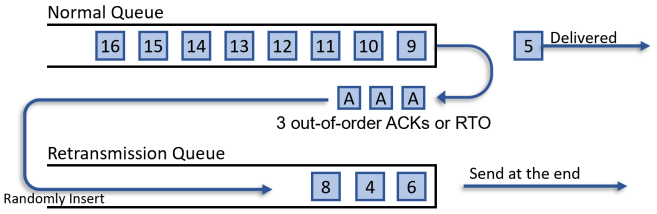
V. 评估

我们从以下几个方面通过测试平台实验评估 LTP：

1) LTP 能为现有 DML 训练任务带来多大的吞吐量提升？（§ V-B）2) 在存在非拥塞丢包的网络中，LTP 的表现如何？（§ V-C）3) 气泡填充是否会导致最终收敛精度较低？（§ V-B）4) 与其他传输协议共存时，LTP 的公平性表现如何？（§ V-D）



(a) CQ 保证所有数据包均能到达（类似于经典 TCP）。



(b) NQ 中的数据包只传输一次。检测到数据包丢失时，将通过 RQ 重传。

图 11. LTP 中的详细队列管理。

A. 实验设置

1) 集群配置：我们在由 9 台机器组成的集群上评估 LTP。其中一台机器用作参数服务器（PS），其余 8 台为工作节点。所有节点位于同一机架中，并连接到一台 HUAWEI CE6881-48S6CQ 三层 ToR 交换机。每台机器配备 2 颗 Intel(R) Xeon(R)Silver 4116CPU @2.10GHz（24 核，48 线程）、256GB 内存，以及一块 Intel(R) Ethernet Network Adapter X722。每台机器还配备一块 16GB NVIDIA Tesla T4 GPU，NVIDIA 驱动程序版本为 460.91.03，CUDA 版本为 11.2 [30]。每台机器的操作系统均为 Ubuntu 18.04.2，内核版本为 4.15.0。

2) 基线：由于我们只修改传输协议，不更改分布式机器学习训练的其他逻辑，为控制变量，我们为所有评估设计了自己的基于参数服务器的分布式机器学习框架。我们将 Linux 内核中采用默认参数的 TCP 拥塞控制算法作为基线，与 LTP 进行比较（LTP 使用 UDP，不受内核拥塞控制算法的影响）。我们将 LTP 与现有的广泛使用的 TCP 拥塞控制算法进行比较评估，包括 BBR [24], Cubic [35], 和 New Reno [36]。最大传输单元设为 1500，并调整 ToR 交换机上的 QoS 策略，以确保其对 TCP 和 UDP 一视同仁。

3) 工作负载：我们在集群上运行 PyTorch，并使用上述框架进行评估。我们使用包含 6 万张图像的数据集 CIFAR-10 [29],训练常用的机器学习模型，例如 ResNet18、ResNet50、ResNet152 [23]和 VGG 16 [37]，。在每组实验中，我们保持随机种子固定，学习率初始设为 0.1，并且每经过 10 个训练轮次乘以 0.8，以确保变量相同。本节展示 ResNet50 和 VGG16 的结果，其他模型也取得了类似的改进。

4) 性能指标：评估 LTP 时使用了三个指标：

- 训练吞吐量，即 DML 训练任务的训练速度（图像/秒）。
- 达到准确率的时间（TTA），即在测试数据集评估中达到该准确率所需的时间。
- 批次同步时间（BST），即每个批次在聚集和广播上花费的时间总和。它可以反映同步过程中所使用通信协议的效率。

除了这三个指标外，我们还评估 LTP 与其他常用传输协议共存时的公平性。

B. 训练吞吐量和达到准确率的时间

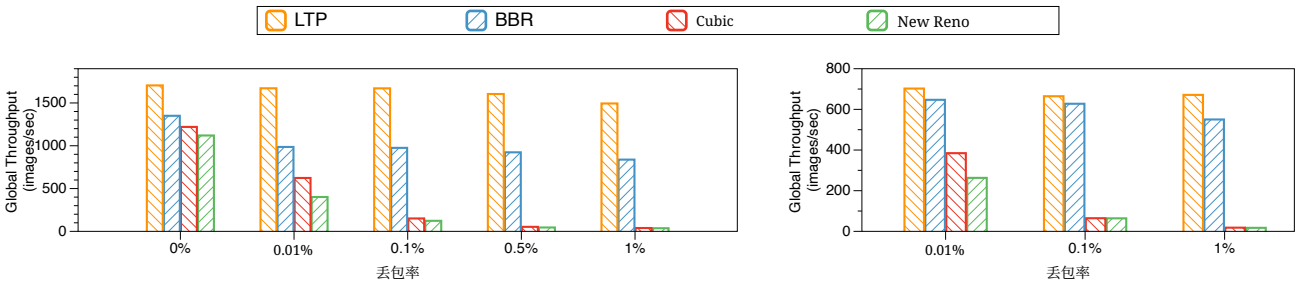
LTP 的主要优化集中于传输协议。在存在丢包的网络中执行分布式机器学习训练任务并不罕见（§ II-B）。因此，我们在具有不同丢包率（0%、0.01%、0.1%、0.5%、1%）的有损网络中，将 LTP 与常用的拥塞控制算法进行比较。所选择的丢包率也用于 [38], 的评估，并且我们同样会在 BST 的评估中使用这些丢包率（§ V-C）。

训练吞吐量提升。 吞吐量评估使用了两种模型：ResNet50（计算密集型，模型大小为 98MB）和 VGG16（通信密集型，模型大小为 500+MB）。吞吐量结果如图 12 所示，表明 LTP 在有损网络中能够实现更高的吞吐量。与 ResNet50 中的 BBR 相比，当网络不存在非拥塞丢包时，LTP 可实现 1.26 倍的吞吐量提升；当非拥塞丢包率介于 0.01% 至 1% 时，吞吐量提升更为明显（最高可达 2 倍）。由于 Cubic 和 New Reno 的拥塞发现机制，它们对丢包非常敏感。因此，LTP 可带来最高 31 倍的性能提升。使用 VGG16 模型时，训练吞吐量的提升并不显著。我们认为，这是因为 VGG16 的模型大小是 ResNet50 的 5 倍。因此，在 VGG16 的评估中，通信数据量也是其 5 倍。大象流会降低长尾延迟带来的风险，因此与 BBR 在 VGG16 评估中的表现相比，LTP 的吞吐量提升有限。

根据我们的分析，这一性能提升源于：1）引入提前关闭机制，避免长尾延迟对 BST 的影响；2）LTP 基于带宽时延积的拥塞控制机制，能够在存在非拥塞丢包的网络中保持较高的带宽利用率。

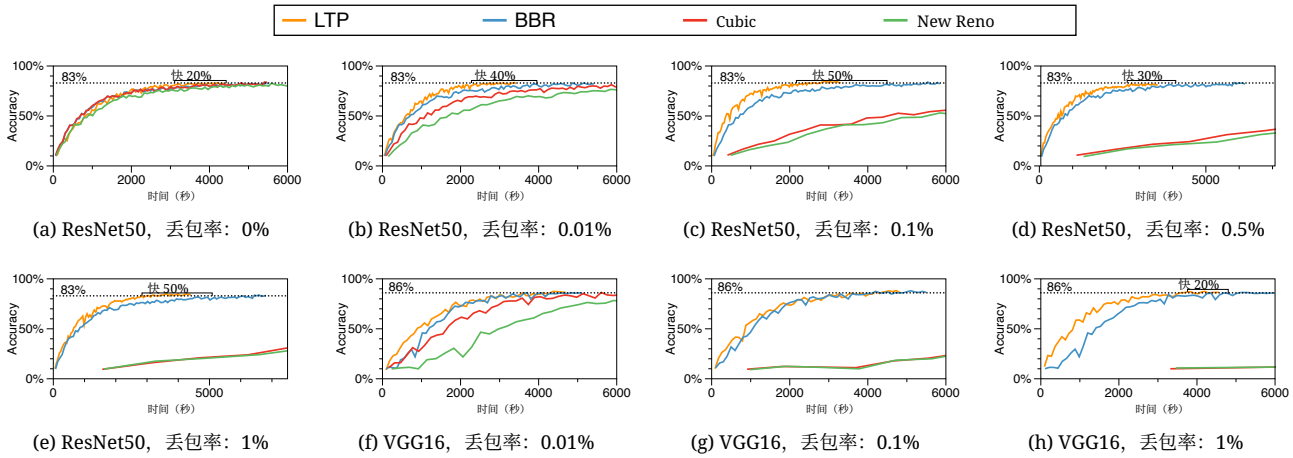
精度损失。引入气泡填充可能会导致传输过程中出现部分数据丢失。因此，我们通过 TTA 评估这类部分数据丢失对精度收敛的影响。图 13 展示了 LTP 与其他拥塞控制算法的 TTA 对比结果，表明在典型丢包率范围内，LTP 不会导致 Top-1准确率下降。

值得注意的是，梯度随机丢失对于DML训练任务来说是一把双刃剑。每个模型都有其



（a）ResNet50　　（b）VGG16

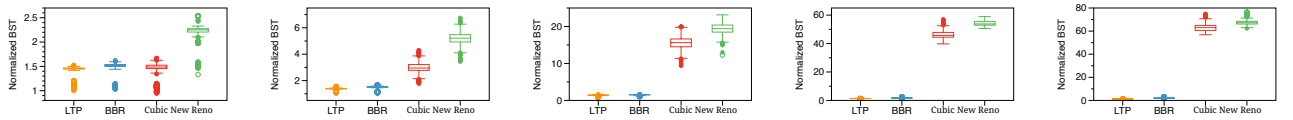
图 12. 不同非拥塞丢包率下不同拥塞控制算法的吞吐量



(a) ResNet50，丢包率：0%　(b) ResNet50，丢包率：0.01%　(c) ResNet50，丢包率：0.1%　(d) ResNet50，丢包率：0.5%

(e) ResNet50，丢包率：1%　(f) VGG16，丢包率：0.01%　(g) VGG16，丢包率：0.1%　(h) VGG16，丢包率：1%

图 13. 不同非拥塞丢包率下不同拥塞控制算法之间的 TTA。



(a) ResNet50，丢包率：0%　(b) ResNet50，丢包率：0.01%　(c) ResNet50，丢包率：0.1%　(d) ResNet50，丢包率：0.5%　(e) ResNet50，丢包率：1%

图 14. 不同非拥塞丢包率下，ResNet50 在不同拥塞控制算法下的 BST（归一化至 LTP）。

独特的数据丢失容忍度；在容忍范围内，部分梯度丢失可以增强模型的泛化能力，甚至最终使模型达到更高的准确率。当丢失的梯度数量超过模型的容忍度时，训练收敛会受到影响。不同模型的容忍度各不相同，需要分别讨论。

C. 批次同步时间

为分析 LTP 带来吞吐量提升的根本原因，我们收集了分布式机器学习训练期间的 BST 分布，并以箱线图展示（图 14）。结果表明，LTP 的 BST 表现远优于传统 TCP 拥塞控制算法（Cubic 和 New Reno）。与 BBR 相比，LTP 的平均 BST 也降低了约 30%。

在评估中，随着丢包率升高，长尾延迟问题会变得更加严重。由于

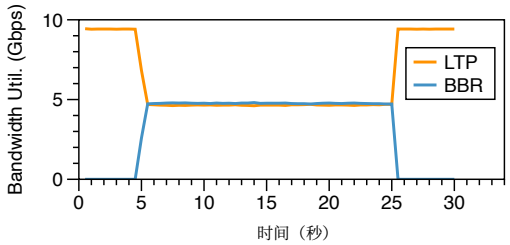


图 15. BBR 与 LTP 之间的公平性。

引入了提前关闭机制，这种现象并未出现在 LTP 的评估中。该机制以部分数据丢失为代价改善流完成时间，并显著优化分布式机器学习训练的整体吞吐量。

D. 不同拥塞控制算法之间的公平性

作为一种采用定制拥塞控制算法的传输协议，有必要评估其与其他协议

共存时的公平性。我们通过点对点通信，评估 LTP 与其他拥塞控制算法（包括 BBR、Cubic 和 New Reno）共存时的带宽分配情况。评估结果表明，在带宽消耗方面，LTP 的表现略逊于 BBR（约为其 97%，见图 15）。我们推测，这一轻微的性能损失是由 LTP 额外的数据包头（9B）造成的。此外，由于这两种方法的性能接近，LTP 与其他拥塞控制协议共存时应能展现出与 BBR 相似的性能。

VI. 讨论与未来工作

A. 与其他梯度压缩算法的协作

梯度压缩方法与 LTP 的理念相似，但优化目标不同。LTP 旨在优化多种网络条件下分布式机器学习训练的同步时间，而梯度压缩方法则致力于减少分布式机器学习训练的通信数据量。LTP 可以与梯度压缩相关工作协作，进一步提升分布式机器学习任务的吞吐量。这种协作会引入二次压缩，从而导致更多梯度丢失。我们的未来工作之一是评估这些算法协同工作所带来的性能提升。

B. 不同模型的最低数据百分比

在评估中，当在丢包率大于 5% 的网络下训练 ResNet50 模型时，精度损失问题很少发生。然而，在 VGG16 模型的训练任务中并未出现同样的问题。我们推测，这主要是因为 ResNet50 模型对数据丢失的容忍度有限，而气泡填充所丢弃的数据超出了该容忍度，进而触发了最终精度问题。因此，我们正在针对具有不同目标的不同模型进行测试，以获取每个模型的大致数据丢失容忍范围，例如 InceptionV3 [39], 、LSTM [40] 和 NCF [41]。

我们正在评估，在传输时间达到截止时间时立即关闭连接是否过于激进。一种可能的解决方案是，在达到截止时间后仍要求维持最低到达速率，而不是直接结束传输。这个最低到达速率可能与模型对数据丢失的容忍度有关，我们正在进行进一步实验。

C. 异构网络中的分布式机器学习系统中的 *LTP*

在数据中心的联邦学习和分布式机器学习系统中，网络异构性十分常见。LTP 通过 LT Thresh- old 保证在不同质量的网络下实现不同的数据到达率。在异构网络场景中，不同工作节点可能具有不同的梯度到达率，从而导致不同工作节点的梯度贡献存在偏差。本文讨论的网络环境均为同构网络，因此工作节点发送的梯度到达率相对均衡。我们正在努力减少异构网络中不同工作节点之间梯度贡献的偏差，以提升模型的泛化能力。

VII. 相关工作

近年来，分布式机器学习受到了业界和学术界的广泛关注。然而，工作节点与参数服务器之间的通信开销一直是制约分布式机器学习训练效率的严重瓶颈。为此，人们提出了许多优化方法。

1) 梯度压缩： 梯度压缩是一种已得到广泛研究的方法。该方法可分为两大类。第一类是梯度稀疏化，即从梯度向量中选择一部分数据进行传输。Top-k [21]算法仅传输梯度向量中绝对值最大的前 k 个元素。Random-k [26] 随机传输梯度向量中的一部分数据，与 Top-k 相比，可以减少排序开销。Threshold-v [42]传输梯度向量中绝对值大于某一阈值的元素，但这一阈值难以合理设定。第二类方法是梯度量化。8 位量化 [43]将每个 32 位浮点数转换为 8 位。Seide [44] 等人提出了一种极端量化形式：根据梯度元素的取值范围，将所有梯度值转换为 0 和 1。然而，这些方法都有可能引入精度损失，而误差补偿技术 [45] 已被证明是解决这一问题的有效方法。GRACE [31] 对梯度压缩进行了全面分析和详细比较。

如第 VI-A 节所述，LTP 可以与这些方法协同工作，进一步降低同步开销。我们正在评估在这些方法中使用 LTP 是否会导致精度损失。

2）分布式机器学习的网络加速：提高同步效率的另一种有效方法是优化或调整传输网络。Xia等人 [46] 证实，分布式机器学习任务能够容忍部分数据包丢失，并呼吁业界开发支持容忍丢包的传输协议。基于相同的发现，DGT [47] 根据重要性将梯度分为两类。必要的梯度通过可靠的 TCP 传输，而不重要的梯度则通过不可靠的 UDP 传输。这可以提高分布式机器学习训练的吞吐量，但缺少对到达阈值的讨论，也没有针对 UDP 通道的拥塞控制的详细设计。SwitchML [48]和 ATP [38]使用可编程交换机实现网络内聚合，显著提高了模型训练效率。

VIII. 结论

我们设计了一种名为 LTP 的传输协议，用于加速采用参数服务器架构的分布式机器学习训练任务。LTP 旨在减轻参数服务器架构通信模式所产生的 incast 流量模式的影响。通过启用容错传输，LTP 允许在分布式机器学习训练期间部分丢失同步梯度，从而加快每次迭代中的同步速度。LTP 使用提前关闭机制来调整容错传输的丢失阈值

并利用气泡填充来纠正丢失的数据。我们实现了 LTP 的原型，并将其集成到广泛使用的机器学习框架 PyTorch 中。在配备 9 台机器的真实测试平台上进行评估，结果表明，在常见机器学习模型上，与传统 TCP 拥塞控制算法相比，LTP 可将分布式机器学习训练吞吐量最高提升 30 倍；与 BBR 相比，最高可提升 2 倍，且不会损失准确率。

参考文献

[1] M. Li、L. Zhou、Z. Yang 等。分布式机器学习的参数服务器。载于 NIPS 大规模学习研讨会，第 6 卷，第 2 页，2013年。[2] A. Gibiansky。将高性能计算技术引入深度学习。百度研究院，技术报告，2017年。[3] A. V. Gerbessiotis 和 L. G. Valiant。直接批同步并行算法。《并行与分布式计算杂志》，22(2):251–267，1994年。[4] D. Zats、T. Das、P. Mohan 等。Detail：缩短数据中心网络中的流完成时间长尾。载于 ACM SIGCOMM 2012 计算机通信应用、技术、架构与协议会议论文集，第 139–150 页，2012年。[5] K. Hsieh、A. Harlap、N. Vijaykumar 等。Gaia：$Geo-Distributed$ 机器学习接近 LAN 极限速度。载于第 14 届 USENIX网络系统设计与实现研讨会（NSDI 17），第 629–647 页，2017年。[6] W. Shi、J. Cao、Q. Zhang 等。边缘计算：愿景与挑战。IEEE《物联网期刊》，3(5):637–646，2016年。[7] X. Lyu、C. Ren、W. Ni 等。无线网络边缘中地理分布式机器学习的最优在线数据分区。IEEE《通信选定领域期刊》，37(10):2393–2406，2019年。[8] B. McMahan、E. Moore、D. Ramage 等。利用去中心化数据进行深度网络的通信高效学习。载于人工智能与统计学，第 1273–1282 页。PMLR，2017年。[9] D. Shan、F. Ren、P. Cheng 等。数据中心网络中微突发流量的观测与缓解。IEEE/ACM《网络学报》，28(1):98–111，2019年。[10] D. Shan、F. Ren、P. Cheng 等。数据中心中的微突发：观测、分析与缓解。载于 2018年 IEEE第26届国际网络协议会议（ICNP），第 88–98 页。IEEE，2018年。[11] Y. Zhang、J. Bi、Z. Li 等。虚拟机监控器：基于数据中心网络虚拟交换机层的负载均衡。IEEE《通信选定领域期刊》，38(6):1176–1190，2020年。[12] C.-S. Li 和 R. Ramaswami。透明全光网络中的自动故障检测、隔离与恢复。《光波技术杂志》，15(10):1784–1793，1997年。[13] H. Choi、S. Subramaniam 和 H.-A. Choi。波分复用光网络中的双链路故障恢复。载于 IEEE计算机与通信学会第二十一届年度联合会议论文集，第 2 卷，第 808–816 页。IEEE，2002年。[14] N. Abbas、Y. Zhang、A. Taherkordi 和 T. Skeie。移动边缘计算：综述。IEEE《物联网期刊》，5 465，2017年 1 450( ): – 。[15] D. G¨und¨uz、P. de Kerret、N. D. Sidiropoulos 等。空中机器学习。IEEE《通信选定领域期刊》，37(10):2184–2199，2019年。[16] S. S. Lor、R. Landa 和 M. Rio。数据包循环利用：消除网络故障导致的数据包丢失。载于第 9 届 ACM SIGCOMM网络热点问题研讨会论文集，第 1–6 页，2010年。[17] P. X. Gao、A. Narayan、G. Kumar 等。pHost：基于商用网络架构的分布式近最优数据中心传输。载于第 11 届 ACM新兴网络实验与技术会议论文集，第 1–12 页，2015年。[18] B. Montazeri、Y. Li、M. Alizadeh 和 J. Ousterhout。Homa：一种由接收方驱动、利用网络优先级的低延迟传输协议。载于 2018年 ACM数据通信特别兴趣小组会议论文集，第 221–235 页，2018年。[19] L. Lai 和 N. Suda。在物联网边缘实现深度学习。载于 2018年 IEEE/ACM国际计算机辅助设计会议（ICCAD），第 1–6 页。IEEE，2018年。[20] S. Yao、Y. Zhao、A. Zhang 等。DeepIoT：利用压缩器-评论器框架压缩感知系统的深度神经网络结构。载于第 15 届 ACM嵌入式网络传感系统会议论文集，第 1–14 页，2017年。

[21] A. F. Aji 和 K. Heafield。用于分布式梯度下降的稀疏通信。arXiv 预印本 arXiv:1704.05021，2017年。[22] D. Alistarh、D. Grubic、J. Li 等。Qsgd：通过梯度量化与编码实现通信高效的 SGD。《神经信息处理系统进展》，第 30 卷，2017年。[23] K. He、X. Zhang、S. Ren 和 J. Sun。用于图像识别的深度残差学习。载于 IEEE 计算机视觉与模式识别会议论文集，第 770–778 页，2016年。[24] N. Cardwell、Y. Cheng、C. S. Gunn 等。BBR：基于拥塞的拥塞控制——测量瓶颈带宽和往返传播时间。Queue，14(5):20–53，2016年。[25] Y. Lin、S. Han、H. Mao 等。深度梯度压缩：降低分布式训练的通信带宽。载于国际学习表征会议，2018年。[26] S. U. Stich、J.-B. Cordonnier 和 M. Jaggi。带记忆的稀疏化 SGD。《神经信息处理系统进展》，第 31 卷，2018年。[27] P. Goyal、P. Doll´ar、R. Girshick 等。精确的大批量 SGD：在 1 小时内训练 ImageNet。arXiv 预印本 arXiv:1706.02677，2017年。[28] N. Qian。梯度下降学习算法中的动量项。《神经网络》，12(1):145–151，1999年。[29] A. Krizhevsky、G. Hinton 等。从微小图像中学习多层特征。2009年。[30] NVIDIA。CUDA，版本：11.2，2020年。[31] H. Xu、C.-Y. Ho、A. M. Abdelmoniem 等。Grace：用于分布式机器学习的压缩通信框架。载于 2021年 IEEE第41届分布式计算系统国际会议（ICDCS），第 561–572 页。IEEE，2021年。[32] B. Recht、C. Re、S. Wright 和 F. Niu。Hogwild!：一种并行化随机梯度下降的无锁方法。《神经信息处理系统进展》，第 24 卷，2011年。[33] J. Iyengar 和 M. Thomson。QUIC：一种基于 UDP 的多路复用安全传输协议。互联网工程任务组，互联网草案 draft-ietf-quic-transport-27，2020年。[34] I. T. Association。RoCEv2 7781，2014年。https://cw.infinibandta.org/document/dl/ 。[35] S. Ha、I. Rhee 和 L. Xu。Cubic：一种新的 TCP 友好型高速 TCP 变体。ACM SIGOPS 操作系统评论，42(5):64–74，2008年。[36] S. Floyd、T. Henderson 和 A. Gurtov。TCP 快速恢复算法的 NewReno 改进。技术报告，2004年。[37] K. Simonyan 和 A. Zisserman。用于大规模视觉识别的超深卷积网络。arXiv 预印本 arXiv:1409.1556，2014年。[38] C. Lao、Y. Le、K. Mahajan 等。 {ATP}：面向多租户学习的网络内聚合。载于第 18 届 USENIX网络系统设计与实现研讨会（NSDI 21），第 741–761 页，2021年。[39] C. Szegedy、V. Vanhoucke、S. Ioffe 等。重新思考用于计算机视觉的 Inception 架构。载于 IEEE计算机视觉与模式识别会议论文集，第 2818–2826 页，2016年。[40] R. Jozefowicz、O. Vinyals、M. Schuster 等。探索语言建模的极限。arXiv 预印本 arXiv:1602.02410，2016年。[41] X. He、L. Liao、H. Zhang 等。神经协同过滤。载于第 26 届国际万维网会议论文集，第 173–182 页，2017年。[42] A. Dutta、E. H. Bergou、A. M. Abdelmoniem 等。分布式深度学习压缩通信理论分析与实际实现之间的差异。载于 AAAI人工智能会议论文集，第 34 卷，第 3817–3824 页，2020年。[43] T. Dettmers。用于深度学习并行化的 8 位近似。arXiv 预印本 arXiv:1511.04561，2015年。[44] F. Seide、H. Fu、J. Droppo 等。1 位随机梯度下降及其在语音深度神经网络数据并行分布式训练中的应用。载于国际语音通信协会第十五届年度会议。Citeseer，2014年。[45] H. Tang、C. Yu、X. Lian 等。DoubleSqueeze：采用双遍误差补偿压缩的并行随机梯度下降。载于国际机器学习会议，第 6155–6165 页。PMLR，2019年。[46] J. Xia、G. Zeng、J. Zhang 等。重新思考分布式机器学习的传输层设计。载于 2019年第 3 届亚太网络研讨会论文集，第 22–28 页，2019年。[47] H. Zhou、Z. Li、Q. Cai 等。Dgt：一种面向贡献度的分布式机器学习差分梯度传输机制。《未来一代计算机系统》，121:35–47，2021年。

[48] A. Sapio、M. Canini、C.-Y. Ho 等人：《利用网络内聚合扩展分布式机器学习》。*arXiv* 预印本 *arXiv:1903 06701*，2019年。。