文章编号:1671-4598(2025)04-0292-07 DOI:10.16526/j.cnki.11-4762/tp.2025.04.039 中图分类号:TP181 文献标识码:A

基于剪枝的轻量级联邦智能学习算法

张泰宁[1], 汤子娇[2]
(1. 中国电子科技集团 第54研究所, 石家庄 050081;
2. 河北工业大学 亚利桑那工业学院, 天津 300401)

摘要: 针对深度学习模型中工作节点异构性导致的训练效率低下和全局模型准确性无法保证的问题, 提出了一种基于剪枝的轻量级联邦学习框架FedPrune; 采用“教师-学生模型”架构, 通过自适应剪枝方案从全局基础模型中动态生成适应不同工作节点能力的子模型, 实现轻量级的智能算法在资源受限的设备中高效执行, 并提出动态自适应剪枝率学习方法, 使各工作节点在相互不知晓能力的情况下实现相同更新时间; 与两种本地解决方案FedAVG、FedRC和两种全局解决方案FedAsync、SSP算法在CIFAR10、CIFAR100、Tiny-ImageNet数据集上进行对比实验, FedPrune具有更高的准确性和更短的总体时间; FedPrune框架通过动态生成适应不同工作节点能力的子模型, 有效解决了掉队问题, 并在异构环境中保持了高精度和高速度, 证明了其在联邦学习中的效率和适用性。

关键词: 深度学习; 联邦学习; 教师-学生模型; 自适应剪枝; 剪枝率学习

A Lightweight Federated Intelligence Learning Algorithm Based on Pruning

ZHANG Taining[1], TANG Zijiao[2]
(1. The 54th Research Institute of China Eectronics Tchnology Group Corporation, Shijiazhuang 050081, China;
2. Arizona College of Technology, Hebei University of Technology, Tianjin 300401, Chnia)

Abstract: Aiming at the problems of low training efficiency and unguaranteed global model accuracy caused by the heterogeneity of work nodes in deep learning models, a lightweight federated learning framework based on pruning, FedPrune, is proposed. Adopting the “teacher-student model” architecture, the adaptive pruning scheme dynamically generates sub-models adapted to the capabilities of different work nodes from the global base model, realizes the efficient execution of lightweight intelligent algorithms in resource-constrained devices, and proposes a dynamic adaptive pruning rate learning method, so that work nodes achieve the same updating time without knowing each other's capabilities, to realize the same update time without knowing each other's capability. The algorithms are implemented with the two local solutions, FedAVG and FedRC, and the two global solutions, FedAsync and SSP, in the CIFAR10, CIFAR100, and Tiny-ImageNet datasets for comparison experiments, FedPrune has higher accuracy and shorter overall time. The FedPrune framework effectively solves the dropout problem by dynamically generating submodels adapted to the capabilities of different working nodes, and maintains high accuracy and speed in heterogeneous environments, proving its efficiency and applicability in federated learning.

Keywords: deep learning; federated learning; teacher-student modeling; adaptive pruning; pruning rate learning

0 引言

深度学习是一种基于神经网络的机器学习方法[1], 具有精准的特征提取和分类能力, 能够自动从大量数据中学习并提取有用的特征, 实现端到端的学习[2]。然而深度学习模型的效果与用于训练的数据量密切相关, 需要大量标注数据进行训练, 由于数据隐私或数据迁移成本较高, 某些场景无法提供足够的存储资源和计算资源以便支持大规模集中数据的训练, 各个节点间数据不互通, 使得大规模训练效率低下, 诞生数据孤岛问题[3]。

收稿日期:2025-03-20; **修回日期:**2025-03-30。

作者简介:张泰宁(2000-),男,硕士研究生。

引用格式:张泰宁,汤子娇.基于剪枝的轻量级联邦智能学习算法[J].计算机测量与控制,2025,33(4):292-298,305.

经过训练的模型也可以被恶意的第三方利用手段推演出原始隐私数据[4-5]。联邦学习[6]作为一种新机器学习范式，是解决深度神经网络集中学习训练难题的最佳方案。

与传统的深度学习相比，联邦学习具有数据隐私保护、数据利用效率、减少通信成本等方面的优势：1）联邦学习强调数据隐私保护，通过加密机制下的参数交换方式，让数据不出本地，避免了数据泄露的风险，符合各种数据隐私法规；2）联邦学习能够整合分散在不同设备或机构的数据，这些数据在传统深度学习中可能因隐私或合规问题无法使用，联邦学习还能利用边缘设备的计算能力，减轻中央服务器的负担，实现资源的优化配置；3）联邦学习通过传输模型参数而不是原始数据，显著减少了通信开销和存储成本，这在带宽有限或数据传输成本较高的场景下尤为重要。

在实际应用中联邦学习也面临着挑战：首先，由于计算和传输能力异构[7]（例如，边缘设备配备不同的计算芯片并位于不同的域），导致联邦学习的工作节点通常受到计算资源和存储资源限制[8]，而且即使使用相同的物理设备，分配给特定任务的计算、存储和带宽资源也是有限的；其次，在联邦学习中，批量同步并行策略的广泛应用导致大量掉队问题出现[9]，由于联邦学习工作节点的异质性，协作最慢的工作节点会拖慢整个神经网络训练过程[10]。出现掉队问题的根本原因是无论节点计算存储转发能力大小，均需要训练相同大小的模型，因此为了避免出现落后者[11]，亟需研究一种能够根据不同节点的计算存储和转发能力，分配与之相适应和匹配的规模的训练模型。

为了在工作节点能力未知的情况下加快全局训练过程，并确保全局准确性，提出了一种新颖高效的联邦学习框架，名为 FedPrune。该方法采用“教师－学生模型”架构[12]，通过定制的自适应剪枝方案从全局基础模型（即教师模型）中动态生成自适应子模型（即学生模型），具有快速训练和高精度的特点且提高全局训练和优选效率，实现轻量级智能算法在资源受限的环境中高效执行的目的。

1　FedPrune 算法原理

1.1　联邦学习

联邦学习是一种新的机器学习范式，与分布式机器学习密切相关。在 FedAVG[13]等联邦学习模型中，每轮训练过程包括以下步骤：1）服务器将当前的全局模型发送给工作节点；2）工作节点对模型进行本地模型更新，并将更新后的本地模型发送至服务器；3）服务器聚合来自工作节点的本地模型，形成新的全局模型。

模型训练和聚合方法的输入包括：剪枝后的子模型参数 θ_ω，工作节点的数据集 D_ω，全局模型参数 θ_g。输出为新的全局模型参数 $\theta_g^{(t+1)}$。本地模型梯度更新公式如式（1）所示。其中 $\theta_w^{t,e}$ 表示第 t 轮第 w 个网元节点在 e 个本地更新后的参数。θ_t^g 是第 t 轮（第 $t-1$ 轮聚合之后）的全局参数，E 是局部更新的次数，η 是学习率：

$$\theta_w^{t,e+1} = \theta_\omega^{t,e} - \eta \nabla f_\omega(\theta_\omega^{t,e}),$$
$$e = 0,1,\cdots,E-1, where \theta_\omega^{t,0} = \theta_g^t \tag{1}$$

第 t 轮局部模型聚合公式如式（2）所示，其中 P_w 是工作节点 w 的模型权重，通常由工作节点中的数据量决定。W 是工作节点数量。

$$\theta_g^{(t+1)} = \sum_{\omega=1}^{w} p_\omega \theta_\omega^{t,e}$$
$$where \sum_{\omega=1}^{w} p_\omega = 1 \tag{2}$$

图 1 是两个不同能力的工作节点在同一联邦学习算法本地模型梯度更新和局部模型聚合的参数变化。

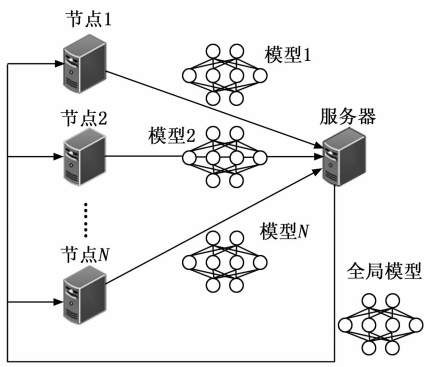

图 1　联邦学习

1.2　网络剪枝

与神经架构搜索[14]等现有技术相比，网络剪枝是实现不同规模模型的最有效方法。因此为了减少联邦学习模型中生成和聚合学生模型带来的压力，采用网络剪枝对教师模型中深度神经网络参数进行适当的剪枝，在保持精度的同时加速模型生成聚合。

网络剪枝包括 3 个阶段：训练、剪枝和微调。根据剪枝对象，网络剪枝可以分为非结构剪枝[15]（只剪掉权重）和结构剪枝[16]（剪掉神经元、滤波器等单元）。

结构剪枝通常会剪掉预定义的低重要性单元，例如零激活的百分比、$l1$ 范数、批量归一化（BN，batch normalization）层的缩放因子、与几何中位数的距离同一层过滤器的数量、特征图的排名或训练中引入的重要性指标。考虑到不同网络层对参数剪枝的敏感度不同，对于给定的剪枝预算，每层的剪枝率应该不同。

直接应用独立的剪枝技术是不合适的。为了保证训练模型的全局效率，针对联邦学习算法面向的分布式场景设计适合本算法的定制化剪枝方法。网络剪枝方法的输入包括：全局模型参数 θ_g，工作节点的剪枝率 P_w，单元重要性度量 X。输出为剪枝后的子模型参数 θ_w。剪枝时每个工作节点根据重要性度量 X 计算其模型中的单元重要性并将单元重要性发送到服务器，随后服务器对从工作节点接收到的单元重要性进行平均，将所有单元按重要性升序排列以形成剪枝顺序，最后在确保剪枝后的子模型与全局模型的结构相似性下每个工作节点根据剪枝顺序和剪枝率 剪掉不重要的单元。

1.3 训练模型

训练模型的效率是影响联邦学习模型全局效率的重要因素，而导致训练模型效率低的原因可以分为局部原因和全局原因两个方面。局部原因是模型太大，导致模型训练和传输极其耗时。全局原因是工作节点之间的异质性，导致掉队，进而影响同步效率。

局部问题解决方案集中在加速模型传输，主要包括梯度量化和稀疏化。梯度量化是通过将梯度量化为低精度值实现；梯度稀疏化通常是通过选择重要参数传输或添加约束以获得稀疏模型。除了减少传输参数加速单次传输之外，还可以通过调整参数聚合频率以提高效率。

与局部解决方案相比，全局解决方案从全局角度考虑效率问题并关注异构性。因此，除了并发计算（BSP，bulk synchronous parallel）策略之外，其他服务器同步策略也被广泛研究以解决落后问题。异步并行（ASP，asynchronous serial parallel）策略与 BSP 相反。在 ASP 中，服务器收到更新就更新全局模型。过时同步并行（SSP，stale synchronous parallel）策略是 BSP 和 ASP 之间的权衡。当最快的工作线程领先最慢的工作线程预定义的阈值 s 轮时，最快的工作线程需要停止并等待。

FedPrune 方案选择的是将学生模型引入联邦学习中，根据工作节点的能力获得自适应的学生模型。学生模型可以更快地进行训练和传输，解决局部问题；自适应大小模型可以调整工作节点的更新时间，消除掉队者，从而解决全局问题。

2 FedPrune 框架

本章节首先对 FedPrune 框架进行整体概述，然后详细说明该框架的关键组成部分，包括剪枝率学习方法、网络剪枝方法和模型训练聚合。1）剪枝率学习方法确定剪枝率，以实现与最快更新时间的一致性；2）网络剪枝方法确定以确定的剪枝率剪枝哪些单元，以尽量减少对全局模型的影响；3）模型训练和聚合是在定制剪枝方案下进行的，目的是尽量减少对全局模型的影响。

2.1 概述

FedPrune 框架流程如图 2 所示。FedPrune 框架包括用于模型聚合的服务器，以及多台具有不同能力、保存各种数据的工作节点。工作节点直接与服务器通信，但带宽不同。该任务旨在利用每个工作节点的数据获得卓越的全局模型。

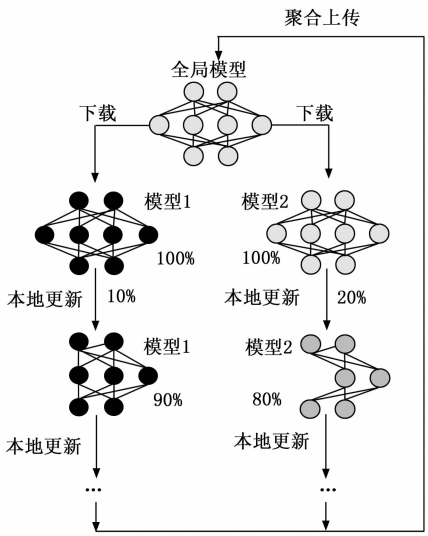

图 2 FedPrune 框架

训练过程如下：

1）训练初始阶段，由于工作节点的能力未知，参数服务器基于同一规则将相同模型 θ_g 分配给每个工作节点；

2）服务器获得工作节点能力信息，即更新时间后，为工作节点生成自适应剪枝率（例如 10%、20%）；

3）工作节点接收剪枝率，然后在本地剪枝其模型；

4）在此基础上。工作节点获得不同大小的学生模型，例如 90%和 80%。

接下来，工作节点使用学生模型进行训练，并将学生模型发送至服务器进行聚合。

上述训练过程攻台执行，直到更新时间趋于相同。服务器端采用同步方法聚合训练模型，直到收到所有工作节点的更新。更新包括局部模型的参数 θ_w^t 和全局索引 I_w^t，其中 I_w^t 是在局部模型被剪枝的情况下对齐局部模型和全局模型中的单元。同时，服务器计算节点 w 的更新时间 φ_w^t，即服务器发送模型和接收模型之间的时间差。φ_w^t 用于对工作节点能力进行建模并获得剪枝率。

如果剪枝轮到来，则使用剪枝率学习方法获取每个工作节点的剪枝率 P_w^{t+1}。FedPrune 采用迭代剪枝，在每个剪枝间隔（P_I）之后进行剪枝，并从训练一开始就进行剪枝，使得整个训练过程节省时间，留下更多轮次恢复模型。最后，服务器通过提取位置 I_w^t 的 θ_g^t 参数，得到工作节点子模型的参数 θ_w^{+1}，并将参数和剪枝率发送给工作节点。

在工作节点端，工作节点在接收参数后用其数据集 D_w 训练部分。如果不需要剪枝，工作节点会继续在之前的模型上训练纪元的另一部分；当需要剪枝时，按照特定的剪枝顺序剪枝网络。

2.2　剪枝率学习

作为 FedPrune 框架中的关键组成部分之一，剪枝率学习为每个工作节点提供自适应剪枝率，以实现大致相同的更新时间，避免掉队问题的发生。工作节点的更新时间由两部分组成：模型传输时间和模型训练时间。

剪枝率学习方法输入包括：历史数据（r_i,t_i），其中是模型保留率 r_i,t_i 是对应的更新时间；当前最小平均更新时间 $\Phi_{\min}$；约束条件，最小模型保留率 $\gamma_{\min}$，最大剪枝率 $\rho_{\max}$，最小剪枝率 $\rho_{\min}$；输出为新的剪枝率 p_ω^{t+1}。

在提出的剪枝率学习方法中，首先将当前最小平均更新时间作为下一轮剪枝的目标时间 $\Phi_{\min}$。利用已经收集到的数据，即每轮的模型保留率和平均更新时间。平均更新时间避免了一些随机因素的影响，使容量估算更加准确。在许多插值方法中，牛顿插值法稳定且快速，因此在 FedPrune 中采用牛顿插值法。如果之前没有进行过剪枝，保守假设更新时间随模型保留率线性变化。剪枝率学习的目标是使所有工作节点的更新时间趋于一致，同时确保模型的精度尽可能高。目标函数如公式（3）所示，其中 P_w 是工作节点 w 的剪枝率，ψ_ω（P_ω）是工作节点 w 在剪枝率 P_w 下的更新时间，$\Phi_{\min}$ 是当前最小的平均更新时间，作为目标时间。

$$\min_{P_w} \sum_{\omega=1}^{w} \mid \psi_\omega(P_\omega) - \Phi_{\min} \mid \tag{3}$$

牛顿插值法用于根据历史数据（模型剪枝率和更新时间）预测新的剪枝率。假设已经收集了 n 组历史数据（r_i，t_i），其中 r_i 是模型保留率，t_i 是对应的更新时间，牛顿插值多项式表示为式（4）：

$$t(r) = a_0 + a_1(r - r_0) + a_1(r - r_0)(r - r_1) + \cdots + a_n(r - r_0)(r - r_1)\cdots(r - r_n - 1) \tag{4}$$

其中：a_n 是通过差商计算得到的系数。$a_n = f[r_0, r_1, \cdots, r_n]$。

差商是一种递归计算方法，用于构造插值多项式的系数，给定一组数据点（r_0，t_0），（r_1，t_1），…，（r_n，t_n），其中 r_n 是模型保留率，t_n 是对应的更新时间，差商定义如公式（5）所示：

$$f[r_i, r_j, \cdots, r_k] = \frac{f[r_j, r_j + 1, \cdots, r_k] - f[r_i, r_i + 1, \cdots, r_k - 1]}{r_k - r_i} \tag{5}$$

如果之前没有进行过剪枝，可以假设更新时间随模型保留率线性变化，如公式（6）所示：

$$t(r) = t_{\text{base}} + k(1 - r) \tag{6}$$

其中：t_{base}是基线更新时间，k 是线性斜率。

为了防止过度剪枝或剪枝不足，引入以下约束条件：最小模型保留率 $\gamma_{\min}$，确保模型保留率不低于某个阈值，设置为 0.5 或更高，确保模型保留率不低于 50％以维持模型的基本性能；最大剪枝率 $\rho_{\max}$，限制剪枝率的最大值，设置为 0.5 或更低，限制剪枝率不超过 50％防止过度剪枝导致模型性能大幅下降；最小剪枝率 $\rho_{\min}$，设置为 0.05 或更高，防止过于频繁的小剪枝，减少计算开销。

通过牛顿插值法预测剪枝率，并结合约束条件动态调整剪枝率。具体步骤如下：1）初始化，设置初始剪枝率 $P_\omega^{(0)}$；2）插值预测，使用牛顿插值法根据历史数据预测新的剪枝率 $P_\omega^{(t+1)}$；3）约束检查，确保预测的剪枝率满足约束条件，见公式（7）；4）更新模型保留率，根据预测的剪枝率计算模型保留率，见公式（8）；5）迭代优化，重复上述步骤，直到更新时间趋于一致。

$$P_\omega^{(t+1)} = max\left[\rho_{\min}, min\left(\rho_{\max}, P^{(t+1)}\right)_\omega\right] \tag{7}$$

$$r_\omega^{(t+1)} = 1 - P_\omega^{(t+1)} \tag{8}$$

2.3　网络剪枝

在确定网络剪枝率后，网络剪枝试图为工作节点找到单元的剪枝顺序。直观的方法是每个工作节点根据学习到的剪枝率单独剪枝他们的模型。这样，之前的网络剪枝工作就可以单独应用于每个工作节点以获得剪枝顺序。然而，以前的网络剪枝工作只能保证获得的子模型的准确性，即每个工作节点本地模型的准确性，而不能保证聚合后全局模型的准确性。分布式剪枝的本质是在兼顾全局模型精度的情况下通过剪枝获得子模型。

在 HeteroFL[17] 采用了一种剪枝方法，忽略了模型中某个任务的单元重要性，但也取得了良好的性能。被剪枝的单元在单元索引中是相邻的，并且每个工作点之间的剪枝顺序是相同的，每轮之间是恒定的。为探究其具有良好性能的根本原因，并找到保证全局模型准确性的关键。良好的性能可能与 3 个特征有关，即相邻（剪枝单元相邻）、相同（所有工作共享剪枝顺序）和恒定（所有轮次共享剪枝顺序）。其中相同和恒定是保证全局模型准确性的两个原则。考虑到不同层表达不同的语

义，引入了一个原则：单位重要性需要是全局的。因此设计了一个恒定、相同和全局的剪枝方案。

2.4 模型训练和聚合

模型训练和聚合是 FedPrune 框架一轮训练最后的关键步骤。在结构剪枝，即剪掉单元实现对任意软硬件的加速后，使用稀疏训练方法减轻切断单元造成的影响。模型训练的每轮都使用损失函数进行稀疏训练，随着组套索正则化的引入，与一个单元相关的参数被视为一个组 g。

如果忽略网元节点之间数据量的差异，参数聚合系数 p_w 为 $1/W$。然而，由于网络剪枝的原因，某些参数可能只存在于 w'（$w'<W$）局部模型中。以图 3 为例，节点 1 的本地模型（以下简称本地模型 1）没有 j 列参数，即 j 列参数只存在于两个本地模型中（$w'=2$）。设置 j 列参数的聚合系数有两种方法，即在某些网元节点处剪枝参数，如图 3 所示。

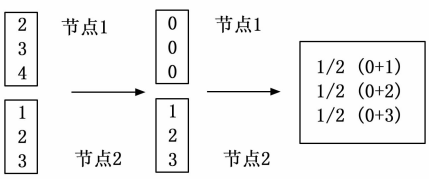

图 3 剪枝参数设置

一种方法是 By-unit，系数为 $1/w'$，即使用 HeteroFL 的方法。By-unit 相当于将剪枝后的参数（例如局部模型 1 中的第 j 列参数）视为其他局部模型中相应位置的参数的平均值。本文提出另一种方式，将剪枝后的参数视为 0。例如，在图 3 中的第一个权重为 $1/2*1=1/2*(0+1)$。考虑到剪枝后的参数比较小，将其视为 0 更接近真实值。此外，表现良好的原因是掩码操作将某些值冻结为零，并且可以使这些值更快地到达优化过程的末尾。总之，聚合时将剪枝后的参数设为 0 不仅更接近原始值，而且还可以加快优化速度。因此采用该方法在 FedPrune 中进行模型聚合。

3 实验结果与分析

3.1 实验配置

本实验采用 Ubuntu 20.04 LTS 操作系统，CPU 型号为 Intel Xeon W-2225，主频 4.1 GHz，4 核心，8 线程，GPU 型号为 NVIDIA RTX A2000-6 G，内存 32 G，硬盘容量为 4 TB，使用 PyTorch1.10.0 深度学习框架，利用编程语言 Python3.8.10 进行仿真实验并对实验结果进行了分析。实验基于数据集 CIFAR10、CIFAR100、Tiny-ImageNet4 评估 FedPrune。CIFAR10 和 CIFAR100 分别由 10 个类别和 100 个类别的 50 000 个训练数据和 10 000 个验证数据组成。TinyImageNet 有 200 个类，每个类包含 500 条训练数据，50 条验证数据。MNLI 分为 3 个类别。实验时需要进行如下数据预处理：数据增强，对训练数据进行随机水平翻转、随机裁剪；归一化，使用数据集的均值和标准差对图像进行归一化处理；数据划分，将数据集划分为训练集、验证集和测试集，比例分别为 70%、15%和 15%。

将 FedPrune 训练结果与 3 种效率提升解决方案进行比较，包括两种局部解决方案（FedRC[18]、FedGen[19]）和两种全局解决方案（FedAsync[20]、SSP[21]）。

异质性 H 是根据工作器更新时间 ψ_w 的分布定义的，如下面公式（9）所示。该定义保证 H 在 0 和 1 之间，越接近 1 异质性越高：

$$H=1-\frac{1}{W-1}\sum_{\omega=1}^{W-1}\frac{\varphi w}{\varphi \omega}$$
$$Assume\varphi w=Min(\varphi 1,\cdots,\varphi w) \tag{9}$$

通过让不同的工作节点运行在不同的计算芯片上，即 CPU 或 GPU 上实现异构计算能力，并通过设置不同的带宽实现异构传输能力。

3.2 异构环境实验

为了验证 FedPrune 在不同异构环境下的适应性和鲁棒性，将数据在 IID 条件下，设置工作的 CPU 的频率为 3.6 GHz，1 Gbps 的网络带宽，添加 10 ms 的网络延迟作为基准，分别与数据在 Non-IID 条件下、3.2 GHz 的 CPU 频率、800 Mbps 的网络带宽、20 ms 网络延迟的情况下进行对比实验，同时将 FedAVG 和 FedRC 算法在相同的条件下进行对比实验，结果如图 4 所示。

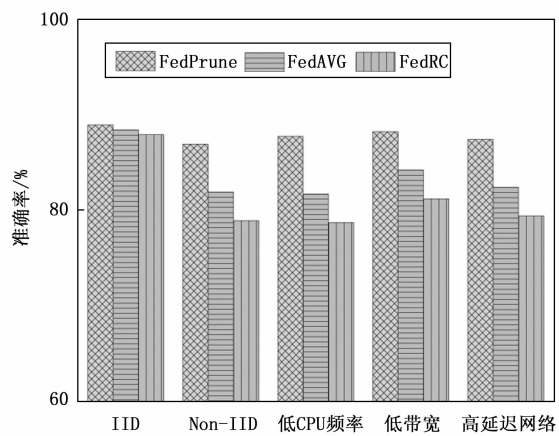

图 4 异构环境对比

可以看到，数据分布由 IID 转为 Non-IID 时，FedPrune 准确率下降了约 2%，FedAVG 和 FedRC 准确率分别下降了 5%和 8%；在较低的 CPU 频率环境下，FedPrune 准确率保持在 87.8%，FedAVG 和 FedRC 准确率分别下降了 6%和 9%；在低带宽环境下，Fed-

Prune 准确率保持在 88.3%，FedAVG 和 FedRC 准确率分别下降了 4%和 7%；最后在高延迟网络下，FedPrune 准确率保持在 87.5%，FedAVG 和 FedRC 准确率分别下降了 5%和 8%。通过在不同异构环境下的实验验证，剪枝率学习方法在计算能力、传输能力和数据分布变化时仍能准确地为每个工作节点提供合适的剪枝率，保证训练过程的高效性和全局模型的准确性。与 FedAVG 和 FedRC 两个算法相比，FedPrune 在异构环境下的适应性和鲁棒性表现更为出色。

3.3　与局部解决方案的比较

将初始学习率设置为 0.1，以便能够在训练初期快速收敛。批量大小设定为 128，在计算效率和模型收敛稳定性之间取得平衡。总共训练 100 个 epoch，确保模型充分训练，同时避免过拟合。

对于 FedRC，使算法能够改变每轮的本地 epoch 数量，以通过设置不同的总时间限制来实现不同的加速。对于 FedGen，通过设置 sketch 数据结构的大小实现不同的加速。对于 FedPrune，调整最小保留率 $\gamma_{\min}$、最大剪枝率 $\rho_{\max}$ 和剪枝间隔 P_I 以实现不同的加速。较大的 $\gamma_{\min}$、较小的 $\rho_{\max}$ 和较大的 P_I 都可以带来更高的精度和更长的总时间。

对于 CIFAR，根据定义的异质性设置了一个异质环境，其 $H\approx 0.8$。

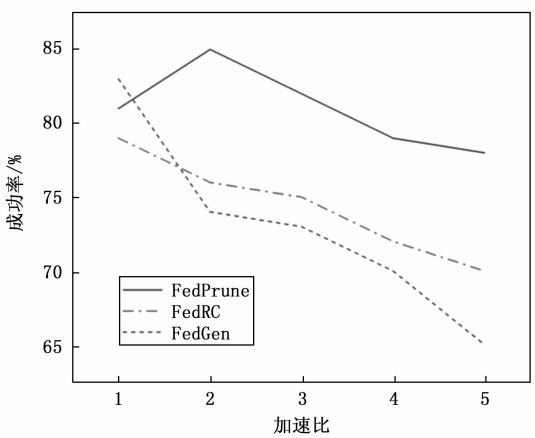

图 5　CIFAR10 数据集下准确度

对于 CIFAR10，所有方法都可以在较低的加速下实现精度提高。虽然 FedGen 达到了最高的准确率，但消耗的时间更多，即加速比小于 1。准确率高可能与 FedGen 内部的知识蒸馏有关。FedPrune 行中最左边的点显示了执行稀疏训练但没有修剪的结果。经过适度剪枝后，FedPrune 的准确率相比没有剪枝的准确率有一定的提升。这种改进可能与修剪后的参数在聚合中被视为零有关，并且大多数参数也朝着参数零的方向进行了优化。因此剪枝加速了大多数剪枝参数的优化。随着加速比的增加，基线的精度开始显著下降。FetchSGD 的加速比受到其无法减少训练时间的限制，尽管它减少了传输时间，但训练时间在总时间中占主导地位。对于 FedRC 和 FedGen，与最小加速比下的精度相比，它们的精度分别下降了 10%和 18%。FedPrune 可以实现约 5 倍的加速，但与最小加速比相比，精度略有下降。

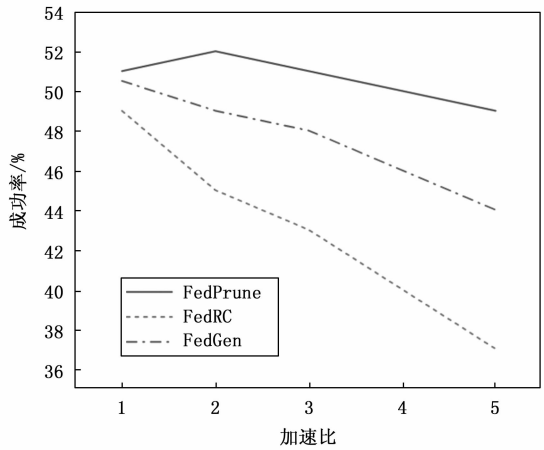

图 6　CIFAR100 数据集下准确度

从图 6 中可以看出，基线的准确性并没有显著提高，并且在保持准确性的同时可以实现的加速非常低。对于 FedRC 和 FedGen，与最小加速比下的精度相比，它们的精度分别下降了约 15%和 1%。这与任务难度的增加有关，即从 10 路分类任务到 100 路分类任务。相比之下，FedPrune 仍然保持着较好的性能。

对于 Tiny-ImageNet，设置了一个异构环境，其中 $H\approx 0.8$。ResNet50 在 Tiny-ImageNet 上的结果如图 7 所示。随着任务难度再次增加（从 100 路分类任务到 200 路分类任务），FedPrune 仍然表现出较高的整体性能，展现了其鲁棒性。当加速比增加时，FedRC 和 FedGen 的准确率仍然显著下降，分别约为−15%和−10%。FetchSGD 的准确率较低，推断 NonIID 任务对它来说太困难了，而 FedPrune 的准确度有所下降。

3.4　与全局解决方案的比较

全局解决方案关注异构环境，并尝试通过同步策略缓解掉队问题。FedAsync_S 是一个使用稀疏训练方法的异步 FedAVG，其中局部和全局模型的聚合权重以多项式方式设置，超参数 a 设置为 0.6，SSP 中的聚合系数设置为 $1/W$，SSP 中的阈值 s 分别设置为 2、4 和 6，选择最好的结果作为 SSP 的结果。对于 FedAsync 和 SSP，每个工作线程运行 T 轮，从而产生 $W*T$ 聚合，报告 $W*T$ 聚合的最佳准确性以及该轮相应的完成时间。将工作线程全部放在 GPU 上，并以不同的方式设

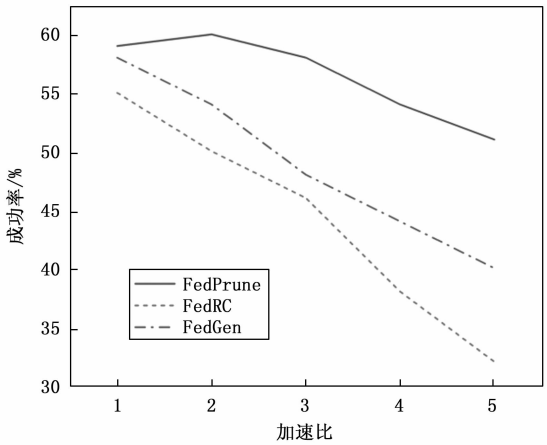

图 7 Tiny-ImageNet 数据集下准确度

置工作线程的带宽以实现所需的异构性。

为了确保所有工作节点的更新时间一致，避免梯度过时问题，因此选择同步策略。剪枝间隔设置为每 10 个 epoch 进行一次，以便在模型精度和训练效率之间取得平衡。

CIFAR10 和 CIFAR100 上的结果如图 8 和图 9 所示。FedAsync 和 SSP 总体时间较短，但精度较低。FedAsync 和 SSP 精度较低是由异步方式中常见的梯度过时问题引起的。在最慢的工作节点上训练的模型落后最新的全局模型很多轮，并且从最慢的工作节点获得的更新可能会损坏最新的全局模型，甚至导致不收敛的困境。随着任务难度（从 CIFAR10 到 CIFAR100）、数据 Non-IID 程度（从 IID 到 Non-IID），梯度陈旧的影响逐渐增大。在某些情况下，SSP 的总体时间也会增加，这是因为服务器需要进行更多的聚合，即 $W * T$ 聚合。如果服务器繁忙，则工作节点在提交更新的模型后必须等待更长时间才能获得新的全局模型。相比之下，FedPrune 具有更高的准确性和更短的总体时间。FedPrune 仍然采用同步方式，因此不存在梯度陈旧问题。

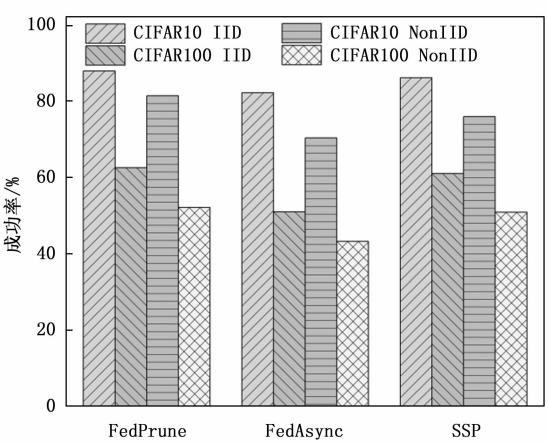

图 8 不同条件下三种算法的准确度

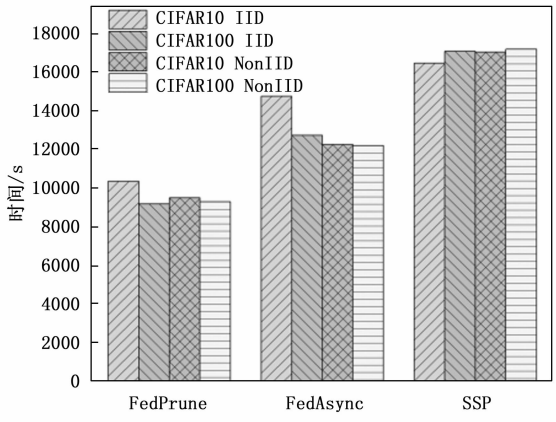

图 9 不同条件下三种算法的训练时间

4 结束语

提出了一种基于剪枝的轻量级联邦学习框架 FedPrune，旨在解决深度学习模型中工作节点异构性导致的训练效率低下和全局模型准确性无法保证的问题。FedPrune 将网络剪枝技术引入联邦学习，通过自适应剪枝动态生成子模型，有效解决了异构环境下的模型训练和聚合问题；通过动态调整剪枝率，确保所有工作节点几乎同步提交模型更新，避免了因节点异构性导致的训练延迟，提高了训练效率和模型准确性。然而 FedPrune 在处理极端异构环境（如计算能力差异极大的设备共存）时，可能面临无法准确估计极端异构节点的更新时间，从而影响整体训练效率的挑战，此外，FedPrune 尚未深入探索与先进隐私保护技术（如差分隐私、同态加密）的集成，这在某些对数据隐私要求极高的应用场景中是一个需要解决的问题。未来的研究将结合联邦神经架构搜索，以自动设计适应不同工作节点能力的最优模型架构，进一步提升模型性能和效率；研究更高效、更智能的剪枝策略，如基于重要性评分的动态剪枝、渐进式剪枝等，以在保证模型精度的同时进一步减少模型大小和计算复杂度。

参考文献：

[1] 刘 萌，齐孟津，詹圳宇，等. 基于深度学习的图像－文本匹配研究综述 [J]. 计算机学报，2023，46 (11)：2370－2399.

[2] LI X，LI M，YAN P，et al. Deep learning attention mechanism in medical image analysis：basics and beyonds [J]. International Journal of Network Dynamics and Intelligence，2023：93－116.

[3] LI Q，DIAO Y，CHEN Q，et al. Federated learning on non-IID data silos：an experimental study [C] //2022 IEEE 38th International Conference on Data Engineering (ICDE). IEEE，2022：965－978.

(下转第 305 页)