计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.250100106

自适应梯度稀疏化的深度神经网络训练方法

黄新利　高国举
苏州大学计算机科学与技术学院　江苏 苏州 215006

摘　要　具有误差补偿的 Top-k 稀疏化方法目前是分布式深度神经网络(DNNs)训练中最先进的技术之一,它在每次迭代训练中动态传输部分梯度来减少通信量,传输的梯度总量取决于 k 值的选择。虽然较小的 k 值可以加速训练,但即使在有误差补偿的情况下,也可能降低测试准确性。本文提出了 AdaTopK——一种自适应 Top-k 压缩器,它可以通过动态调整 k 值来权衡训练速度和测试准确性。大量动态网络场景下的实验表明:与不压缩的情况相比,AdaTopK 可以减少 29%的训练时间;同时与已有实验 DC2 相比,AdaTopK 也可以减少 15%的训练时间。

关键词:分布式训练;网络压缩;稀疏化;深度神经网络;误差补偿
中图分类号　TP319

Adaptive Gradient Sparsification Approach to Training Deep Neural Networks

HUANG Xinli and GAO Guoju
School of Computer Science & Technology,Soochow University,Suzhou,Jiangsu 215006,China

Abstract　Top-k Sparsification Method with error compensation is one of the state-of-the-art technologies in the training of distributed deep neural networks(DNNs). This technique aims to reduce the amount of communication by dynamically transmitting only parts of the gradients in each iteration,with the amount of transmitted gradients depending on the value of k. Although a smaller k can speed up training time,it may degrade the test accuracy,even with error compensation,known as the speed-accuracy dilemma. Based on the observation that the increase speed of the training accuracy and test accuracy have a dynamic correlation over time,this paper presents AdaTopK—an adaptive Top-k compressor with convergence guarantees. AdaTopK can dynamically adjust the value of k to accelerate the training speed while keeping or enhancing the test accuracy. Extensive experiments in the static and dynamic network scenarios show that AdaTopK can reduce 29% training time over the baseline without compression, while reducing 15% training time over DC2.

Keywords　Distributed training,Network compression,Sparsification,Deep neural networks,Error compensation

1　引言

深度学习正在成为新一轮经济增长的核心引擎,从自然语言处理[1-2]到图像识别[3-4],都需要在大数据上或包含数千万参数的训练模型上进行训练。为了应对这些挑战,分布式深度学习应运而生。当网络状况不佳时,分布式 DNN 训练中交换梯度的时间可能会因为不稳定的网络带宽和大量的掉队者[5]而增加数倍。此时,分布式工作节点也会因为主导的网络通信开销而减缓加速效果[6-9]。

量化方法能够通过减少每个梯度的位宽,以降低总的通信量的方法来加速训练,但是这样可能会导致测试准确性下降。虽然现有的一些工作[10-12]尝试去保证测试准确性,但这些方法均导致训练时间延长。与量化方法相比,稀疏化方法可以通过减少传输的梯度总量来灵活地减少训练时间。Top-k 作为稀疏化方法的代表,能显著地减少训练时间,但也有可能使测试准确性下降[13-16]。使用固定 k 值的实验[13,15]无法适应训练准确性或测试准确性的动态变化。DC2[17]设计了一组延迟感知的算法,用于动态地改变压缩比率以权衡训练准确性和训练时间。但是,DC2 中使用的大量超参数是预定义的,并且没有得到很好的证明,它们无法捕捉动态网络条件,如不同通信后端的通信延迟变化和时变网络带宽。因此,DC2 中 k 值的动态选择本质上不是最优的,因为解决方案空间因预定义的超参数而缩小,若调整不合适则无法保证测试准确性和训练时间的有效性。

本文首先强调了测试准确性的重要性,具体来说,实验结果表明不同的压缩比值可能对测试准确性产生影响,而这取决于整个训练过程中训练准确性的增长速度和不同的网络场景。当训练准确性增长迅速且网络带宽相对稳定时,可以传输更多的梯度值,否则只能传输较少的梯度值。本文根据这些分析提出了一个目标——动态地确定在训练准确性增长期间哪些梯度将被传输,从而间接影响测试准确性。

本文的主要贡献如下:

1)深入探索并阐明了压缩比与一系列 DNN 指标(如训练准确性、测试准确性、训练损失和训练时间)之间的关系。在探索中不难发现,高训练准确性或低训练损失并不意味着高测试准确性,压缩比值与测试准确性不是线性相关的。这些观察结果为本文算法的设计提供了实验上的指导。

2)开发了 AdaTopK,它由基于增长速度的算法组成,用

通信作者:黄新利(109838280@qq.com)

于高度动态的网络带宽。其旨在实现测试准确性和训练时间之间的权衡，即测量训练准确性的增长速度和训练时间，并动态调整 k 值和误差补偿。

3)实验在 GPU 集群上开发了基于 PyTorch 的 AdaTopK 原型系统，并使用 Linux tc 调整可用网络带宽以模拟动态网络场景。实验结果表明，与不压缩的情况相比，AdaTopK 可以减少 29%的训练时间；同时与已有实验 DC2 相比，AdaTopK 也可以减少 15%的训练时间，并且可以实现比 DC2 更高的测试准确性。

2 相关工作

本章对最先进的通信压缩方法进行简要回顾，包括量化和稀疏化。

2.1 量化

量化的目的是减少梯度的位宽，即将 64 位或 32 位的梯度量化为更小位宽的梯度。代表性的方法有阈值量化[8]、QSGD[18]、自适应量化[19]，它们都能通过量化梯度来减少梯度值的位宽，以满足不同的压缩需求。例如：1 比特算法[8]中如果原始梯度位大于或等于 0，则将相应的位设置为 1，否则设置为 0。阈值量化与 1 比特算法类似，可以使用更多的固定阈值，而不仅仅使用 1 和 0。一个自适应阈值[19]会对梯度的绝对值进行排序，并对最大的梯度进行量化。QSGD 会量化梯度向量，将它们随机舍入为一组离散值，最后对它们进行编码。量化的一个主要限制是梯度值的减少严重依赖于位宽，即从 32 位浮点数量化到 1 位，梯度值最多可以减少为 1/32[20]。

2.2 稀疏化

稀疏化通过限制要传输的梯度数量来实现有效的通信。Rand-k[21]是稀疏化技术中一种基础的实现方式，它无偏地选择 k 个随机梯度以满足内存限制，其中 k 是待传输梯度数量与总梯度数量的比率。Top-k[13,15]采用了类似的想法，但它选择了最大的 k 个梯度来实现近似且高效的通信。根据稀疏化的范围，Top-k 可分为本地 Top-k 和全局 Top-k。本地 Top-k 将 Top-k 算法应用于每个分布式工作节点的本地梯度，而全局 Top-k 则将 Top-k 算法应用于所有工作节点聚合后的梯度。尽管本地 Top-k 和全局 Top-k 的稀疏化结果并不总是一致，但值得指出的是它们之间的交集部分非常明显[22]，因此可以使用统一方法。DC2[17]通过动态选择压缩比值以适应不同的网络场景，而不是使用上述固定的压缩比值。

3 背景和动机

3.1 背景

分布式 DNN 的工作流程如下：考虑一个包含 P 个工作节点的训练数据集 D，每个工作者包含分区数据集 D_i，其中 $\sum_{i=1}^{P} D_i = D$。DNN 的本地工作节点首先进行前向传播以获取损失值，然后开始反向传播以计算相应的梯度。一旦每个工作节点得到了自己的梯度，就开始梯度聚合阶段。通信系统随机选择一个训练工作节点(或 CPU)作为中心节点，以累积每个本地工作节点发送的梯度。在参数服务器(PS)的中心节点计算出本地梯度的总和后，PS 将总和发送回本地工作节点。PS 可以用批量同步方式或异步方式处理所有工作节点的本地梯度。在本文的实验中，梯度通信系统被设置为批量同步方式，这与文献[23]一致。

x_t 表示第 t 次迭代的本地参数值，每个工作节点减少计算的梯度 $G_i(x_t)$。同步屏障控制着通信的速度，PS 则获取本地梯度的平均值 G，然后本地工作节点根据式(1)描述的学习率 η 更新参数，得到 $t+1$ 次迭代的参数 x_{t+1}。

$$x_{t+1} = x_t - \eta \frac{1}{P} \sum_{i=1}^{P} G_i(x_i) \tag{1}$$

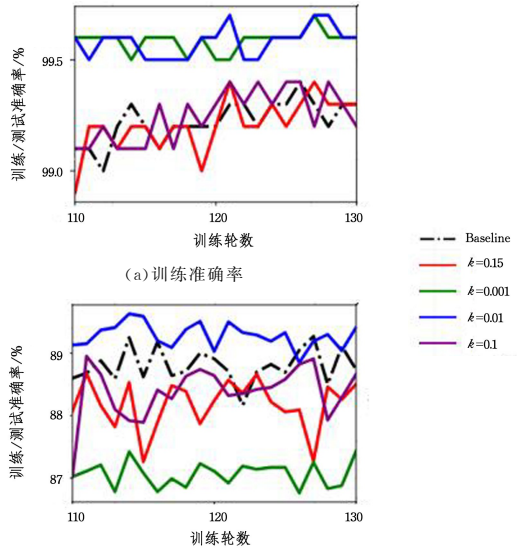
3.2 动机

具有误差补偿的 Top-k 稀疏化方法通过设定固定的压缩比值来筛选梯度，其目的是减少通信量，只传输那些对梯度下降过程更有影响的梯度。当压缩比值较小时，训练时间也会相应缩短。然而，Top-k 方法在训练大型深度神经网络(DNN)模型时可能会使测试准确性不可避免地降低。因此有学者设置了一项实验来评估在不同 DNN 模型和不同压缩比下的测试准确性和加速比，实验基于 PyTorch 的分布式 RPC 框架，带宽为 1 Gbps。在实验中，使用了 ResNet-18 和 VGG-19 在 CIFAR-10 数据集上进行测试，以及 ResNet-34 和 ResNet-50 在 CIFAR-100 数据集上进行测试。测试准确性的平均值以百分比形式展示，将加速结果与未压缩(即 k 等于 100%)的情况进行比较。

表 1 不同 DNN 模型在 k 分别等于 100%，15%和 0.1%时的测试准确率和加速比

Table 1 Test accuracy and speedup ratio of different DNN models at k=100%, 15%, and 0.1%

DNN models	k=1.00(100%)	k=0.15(15%)	k=0.001(0.1%)
ResNet18	87.428%(1.00)	88.309%(1.26)	86.039%(1.59)
VGG19	88.775%(1.00)	88.245%(1.25)	87.074%(1.62)
ResNet34	67.153%(1.00)	66.303%(1.26)	58.432%(1.66)
ResNet50	63.525%(1.00)	63.862%(1.25)	54.247%(2.05)



(a)训练准确率

(b)测试准确率

注：k 值代表压缩比，基准线指无压缩的情况。

图 1 在 Cifar-10 数据集上，VGG-19 在不同压缩比下的训练准确率和测试准确率对比

Fig. 1 Comparison of training accuracy and test accuracy for VGG-19 under different compression ratios on CIFAR-10 dataset

从图 1(a)可以看出，所有情况下训练准确率最后都达到了很高的水平：当 $k=0.15$ 和 $k=0.1$ 时，准确率超过 0.99；而当 $k=0.01$ 时甚至可以达到 100%准确率。因此，可以总结出观察结果 1：

观察结果 1　在合适的学习率下，不同固定压缩比下的训练准确率可以达到很高的水平甚至完全准确。

备注：先前的研究，特别是 DC2，致力于在训练准确性和训练时间之间找到平衡，但发现盲目追求训练准确性并没有意义。这主要是随机梯度下降(SGD)算法的鲁棒性所致。虽然经过压缩后每个工作节点保留的梯度相比未压缩情况下只有一部分，但 SGD 仍然可以采取最优路径以达到高训练准确性，尽管这可能导致测试准确性降低。

根据表 1，可以观察到一个违背一般认知的现象：当 $k=0.15$ 时，ResNet-18 和 ResNet-50 的测试准确性优于未压缩的情况。在图 1(b)中，最终的测试准确率与低压缩比通常会导致性能下降的认识不符：$k=0.01$ 时的测试准确性高于 $k=0.15$ 和 $k=0.1$ 的情况。根据以上现象，能够总结出观察结果 2。

观察结果 2　降低压缩比(即减小 k 的值)并不总会使最终测试准确性恶化。

备注：这种违背一般认知现象的原因是 Top-k 可以约等同于 DNN 模型中的剪枝。因此，当 k 的值适当时，可以打破“速度-准确性”困境。

从图 1 中可以明显看出，训练准确性和测试准确性在 VGG-19 中的表现可能不一致。例如，当 $k=0.001$ 时，图 1(a)所示的高训练准确性并不意味着图 1(b)所示的良好测试准确性，其测试准确性明显是最低的，因此可以总结出观察结果 3。

观察结果 3　高训练准确性并不一定意味着高测试准确性。

基于这些观察结果，本文提出了在 DNN 模型训练过程中动态改变 k 值的方法，其目标是打破“速度-准确性”困境。

4　AdaTopK 概述

本章首先介绍 AdaTopK 的概念。在实验的整个训练过程中，训练准确性的增长似乎是不均匀的。训练准确性在训练过程的早期会出现一波指数级的增长，这种现象在使用不同数据集的其他 DNN 模型中也普遍存在。这样来看，使用测试准确性作为指标来设计自适应 Top-k 算法似乎是一个更好的方法，但在该实验中，并没法获得即时的测试准确性，因为测试阶段要在整个训练过程结束后才开始，所以只能使用即时的训练准确性来调整自适应 Top-k 算法。DNN 模型在训练准确性快速增长时对梯度更新更加敏感，这个阶段被定义为敏感期。而与之相对应是另一个缓慢变化的阶段，被定义为钝化期，在这个阶段，训练准确性以一种微小的方式增长，如图 1 所示。此现象也给予了我们一定启示：可以在敏感期传输更多的梯度，而在钝化期传输更少的梯度。为了精确理解自适应 Top-k，首先需要从数学角度定义一个 AdaTopK 的度量。

$$\arg\max_{k_s \in k_{all}} \alpha N(\mu-\mu_{\min})+\beta N(v_{\max}-v) \tag{2}$$

其中，μ 表示每次迭代的训练准确性，v 表示发送压缩梯度和接收聚合梯度之间的时间间隔，$\mu_{\min}$ 和 $v_{\max}$ 分别是可接受的最小训练准确性和最大通信时间，k_s 表示所有候选压缩比 k_{all} 中的一个可能压缩比，$N(\cdot)$ 是用于表示训练准确性和通信时间的归一化增益的函数，即 $N(x)=(x-x_{\text{rand}})(x_{\max}-x_{\min})$。为了在训练准确率和通信时间上给予不同权重，引入了两个权重系数 α 和 β。训练准确率和通信时间在每次迭代中都比较容易获得，因此可以通过 AdaTopK 来改变 k 的值。具体来说，AdaTopK 可以确定一个压缩比 k_s，使得式(2)生成的归一化增益最大化。

实验不使用训练损失作为评估指标，是因为训练准确性更加直观和精确，特别是 DNN 模型中的图像分类任务。AdaTopK 包含用于动态网络场景的自适应 Top-k 算法，其基本原理源自式(2)。

本文的核心任务是确定哪些因素会影响测试准确性和通信时间，以及如何在每次迭代中动态确定 k 的最佳值。在动态网络场景中，传输的梯度总量和带宽变化均会影响通信时间。在这种情况下，如果当前训练准确性低且带宽足够小，通信时间就会延长。根据式(2)，k 的值若减少，可能会进一步降低测试准确性。因此，本文设计了一种概率方法来确定 k 的增减，这样可以联合处理带宽变化、测试准确性和通信时间。该方法测量训练准确性和通信时间的增加速度，并用它们来替代式(2)的两个部分。这里不使用式(2)的求和来表示增益，因为带宽变化无法包含在内。通过比较这两个部分来确定对压缩比 k 的具体操作，即如果训练准确性的增加速度快于通信时间的增加速度，便以预定义的概率增加 k 的值；否则，k 的值保持不变。

本文将为动态网络场景设计基于增加速度的算法，用于动态调整压缩比，将在后文对其进行讨论。

5　AdaTopK 设计

本章针对动态网络场景设计了基于增加速度的优化算法，以确定压缩比的值，即 k 值。

本文为 AdaTopK 开发了一个在动态网络条件下应用的细粒度算法，同时考虑了训练准确性、通信时间和拥塞控制。为了在动态网络场景中保持高测试精度的同时节省训练时间，实验中尝试从衡量训练准确性和通信时间增加速度的角度调整压缩比。因此，提出了动态网络条件下 AdaTopK 的算法，如算法 1 所示。

算法 1　基于速度的优化

输入：第 i−1 次迭代的压缩比 k_{i-1}

输出：第 i 次迭代的压缩比 k_i

1. 在每个工作节点上放置增量差异代理以进行各自的选择
2. for t=0 to T do：
3.　for i=0 to I do：
4.　dratio=|delay[w−1]−delay[w−2]|/rdelay
5.　aratio=|accur[w−1]−accur[w−2]|/raccur
6.　if α * aratio≥β * dratio：
7.　if rdiffer≤0：
8.　$k_i=m_{inc}*k_{i-1}+(1-m)*k_{max}$
9.　else：
10.　$k_i=m_{inc}*k_{i-1}+(1-m)*k_{max}$(概率 1−$\rho$)
11.　else：
12.　if rdiffer>0：
13.　$k_i=m_{dec}*k_{i-1}-(1-m)*k_{dec}$

14. else:
15. $k_i = m_{dec} * k_{i-1} - (1-m) * k_{dec}$(概率 $1-\rho$)
16. if $k_i > k_{max}$:
17. $k_i = k_{max}$
18. if $k_i < k_{min}$:
19. $k_i = k_{min}$
20. $k_0 = k_I$

算法 1 借鉴了 SGD 中动量的思想作为更新第 i 次迭代压缩比的方法。m_{inc} 和 m_{dec} 表示缩放因子，这些因子类似于动量。$delay$ 数组包含过去 w 个样本的通信时间；$accuracy$ 数组包含过去 w 个样本的训练精确度；$rdelay$ 表示过去 w 个样本通信时间差的绝对值之和($|delay[i-1]-delay[i-2]|, \cdots, |delay[i-w]-delay[i-w-1]|$)；$raccur$ 表示过去 w 个样本训练精度差的绝对值之和($|accuracy[i-1]-accuracy[i-2]|, \cdots, |accuracy[i-w]-accuracy[i-w-1]|$)；$rdiffer$ 表示过去 w 个样本通信时间差之和($delay[i-1]-delay[i-2], \cdots, delay[i-w]-delay[i-w-1]$)；$aratio(dratio)$ 表示 $|accuracy[i-1]-accuracy[i-2]|$ 和 $u=w|accuracy[i-u]-accuracy[i-1-u]|$(通信时间)在当前迭代中的影响比例，并乘以不同的系数。$rdiffer$ 为网络拥塞的一项指标，即如果 $rdiffer>0$，则意味着拥塞可能发生，过去 w 个样本的通信时间变长，因此需要减少 k 的值。尽管测试精确度和不同压缩比之间的关系不能很好地用数学公式表示(第 3 章)，但当训练精确度的增加速度主导式(2)的值时，增加 k 的值总能达到较高的测试精度。

算法 1 的基本流程如下：在获取第 $i-1$ 次迭代的 dratio 和 aratio(第 4—5 行)之后，需要在两种情况下比较训练精确度和通信时间的增加速度(第 6 行)。

情况 1 训练精确度的增加速度主导式(2)的值。当拥塞指示器 $rdiffer \leqslant 0$ 时，增加 k_i 的值；否则，以概率 $1-\rho$ 增加 k_i，或以概率 ρ 保持 k_i 不变(第 7—10 行)，其中 ρ 是一个预定义的概率。

情况 2 通信时间的增加速度主导式(2)的值。当拥塞指示器 $rdiffer>0$ 时，减少 k_i 的值；否则，以概率 $1-\rho$ 减少 k_i，或以概率 ρ 保持 k_i 不变(第 11—15 行)。对于大于 k_{max} 或小于 k_{min} 的 k_i，AdaTopK 将 k_i 设置为 k_{max} 或 k_{min}(第 16—19 行)，并在一个训练周期结束后，使用该周期的最终迭代中的压缩比率 k_I 来替换下一个周期的初始压缩比率 k_0(第 20 行)。为便于展示，在表 2 中总结了关键符号。

表 2 关键符号

Table 2 Key symbols

符号	含义	值
α	训练准确度的系数	3
β	通信时间的系数	2
ω	先前样本的数量	18
k_{max}	最大压缩比	0.29 或 0.26
k_{min}	最小压缩比	0.001
ϵ	概率阈值	0.9
ρ	概率阈值	0.78
m_{inc}	增加权重	0.98
m_{dec}	减少权重	0.97

6 实验评估

为了验证 AdaTopK 在动态网络场景中的有效性，本章在一个 GPU 集群上进行了实验。该实验主要关注以下几个要点：测试准确率、加速比和压缩比的变化。实验选取了以下基准进行比较：固定 $k=0.15$ 和 $k=0.001$ 的 Top-k，DA2[17]，DA4[17]，DA5[17] 和 AdaTopK。同时，加入了没有进行压缩的案例作为基线。

6.1 实验设置

6.1.1 硬件

分布式机器学习的环境配置为 4 节点的 Tesla v100 集群，动态网络场景的带宽在 1.15 Gbps 到 1.45 Gbps 之间。

6.1.2 软件

所有 GPU 均安装了 Nvidia 驱动程序(版本 455.32.00)和 CUDA 11.1，通过 PyTorch 的分布式 RPC 框架部署 PS 架构。虽然 PyTorch 本身不支持传统的 PS 架构，但 RPC 的内置功能可以通过设置一个 GPU 作为主节点来模拟参数服务器。

6.1.3 模型和数据集

在不同的数据集上部署了 4 个 DNN 模型：

(1)CIFAR-10：包含 50 000 个训练样本和 10 个分类对象。

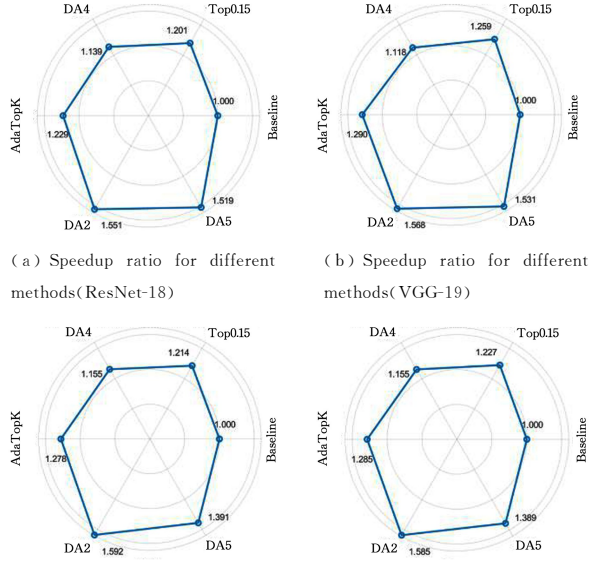
(2)CIFAR-100：包含 50 000 个训练样本和 100 个分类对象。

在 CIFAR-10 上使用 ResNet-18 和 VGG-19，在 CIFAR-100 上使用 ResNet-34 和 ResNet-50。

6.2 动态网络场景下的实验结果分析

6.2.1 加速比

根据图 2 所示的加速比结果可知 DA4 的加速比最小，而 AdaTopK 则有很高的加速比，其在所有 DNN 模型中的加速比均大于 Top-0.15 和 DA4。再结合图 3，可以得出以下结论：DA2 和 DA5 以测试准确率的一定程度下降为代价获得了更高的加速比，其中 DA2 在所有 DNN 模型中的测试准确率几乎是最低的。



(a) Speedup ratio for different methods(ResNet-18)

(b) Speedup ratio for different methods(VGG-19)

(c) Speedup ratio for different methods(ResNet-34)

(d) Speedup ratio for different methods(ResNet-50)

图 2 动态网络场景下，使用 4 个工作节点进行分布式深度神经网络训练的加速比

Fig. 2 Speedup ratio of distributed deep neural network training with 4 workers in dynamic network scenarios

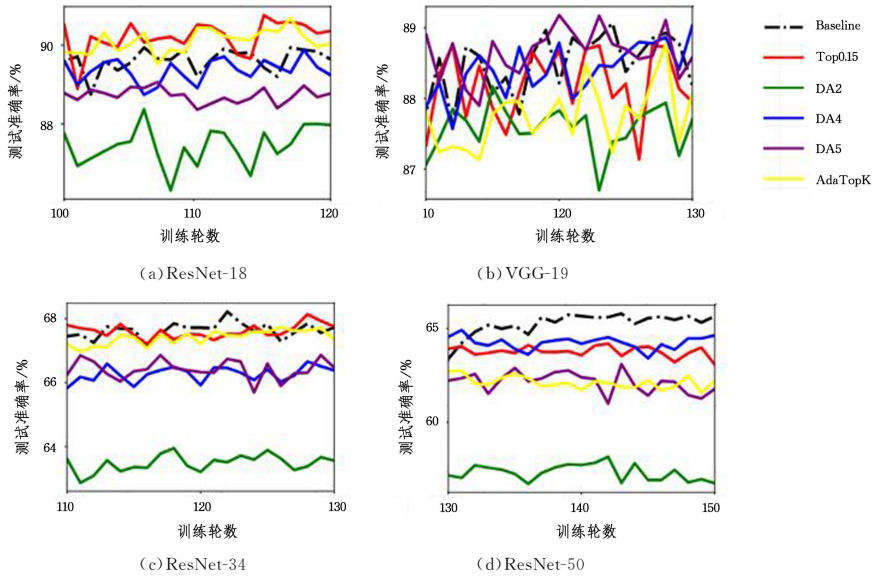


图 3　动态网络场景下，使用 4 个工作节点进行分布式深度神经网络训练的测试准确率

Fig. 3　Test accuracy of distributed deep neural network training with 4 workers in dynamic network scenarios

6.2.2　压缩比变化

AdaTopK 在所有 DNN 模型中表现出相似的压缩比变化趋势，这与其他自适应 Top-k 算法类似，因此这里仅展示 ResNet-50 模型中自适应 Top-k 算法的轨迹。

如图 4 所示，DA2 的压缩比值在训练初期较高，但随后保持在较低水平。DA4 的压缩比值整体高于 DA5，这主要归因于两种算法的不同阈值控制。AdaTopK 的轨迹符合预期，在训练的早期阶段，由于训练准确率较低，AdaTopK 的压缩比值较高。并且随着训练的进行，AdaTopK 的压缩比值不断降低。

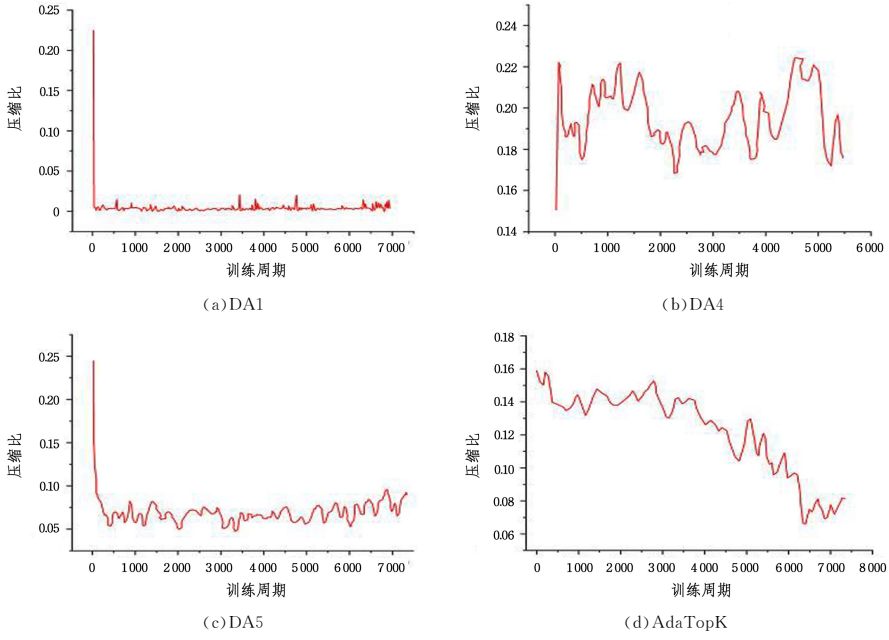


图 4　动态网络场景下，使用 4 个工作节点进行分布式训练时 ResNet-50 模型的压缩比

Fig. 4　Compression ratio of ResNet-50 model in distributed training with 4 workers under dynamic network scenarios

6.2.3　测试准确率

图 3 显示了动态网络场景中所有算法的测试准确率结果。在图 3(a)和图 3(c)中，AdaTopK 和 Top-0.15 获得了相似的测试准确率，ResNet-18 的最后 20 个 epoch 中(100～120)，两者的最终测试准确率均约为 90%，且均高于其他算法。同时，在图 3(c)中，AdaTopK、Top-0.15 和无压缩的方法具有较高的准确率，ResNet-34 的最后 20 个 epoch 中(110～130)，这三者的最终测试准确率均约为 68%。虽然 DA4 通过较高的压缩比值保留了高测试准确率，但如图 3(b)所示，DA5 在较短的训练时间内也能获得良好的测试准确率，在 VGG-19 的最后 20 个 epoch 中(110～130)，其测试准确率高于 Top-0.15，DA2，DA4 和 AdaTopK。这一现象在动态场景中再次出现，促进了对压缩比值与测试准确率之间关系的进一步探索。在图 3(d)中，Top-0.15 和 DA4 达到了较高的测试准确率，而无压缩方法则得到最佳的测试准确率：在 ResNet-50 的最后 10 个 epoch 中(140～150)，Top-0.15 和 DA4 的测试准确率均低于 65%，而无压缩方法的测试准确率高于 65%。再结合图 2(d)，可以得出结论：AdaTopK 和 DA5 在 ResNet-50 中以一定程度的测试准确率的下降为代价来得到了较高的加速比。

结束语 本文首先对分布式 DNN 模型的 3 个指标(训练准确率、训练时间和测试准确率)之间的关系进行了 3 项观察结果。基于这些观察结果,本文设计了用于动态网络场景的算法,以动态确定压缩比 k 的值。广泛的实验评估表明,AdaTopK 在训练速度和测试准确率权衡方面有显著的效果。

参 考 文 献

[1] 朱永伟. 基于深度学习和注意力机制的文本分类关键技术研究[D]. 南京信息工程大学,2024.

[2] CHENG Z T, HUANG H R, XUE H, et al. Event Causality Identification Model Based on Prompt Learning and Hypergraph[J]. Computer Science, 2025, 52(9): 303-312.

[3] LYU Y F, ZHANG X L, GAO W N, et al. The Application of Deep Learning in Customs Image Recognition Technology[J]. China Port Science and Technology, 2024, 6(Z2): 4-12.

[4] 井煜. 基于深度学习的局部遮挡人脸图像识别方法研究[J]. 互联网周刊, 2024, (21): 56-58.

[5] VOGELS T, KARIMIREDDY S P, JAGGI M. Powersgd: Practical low-rank gradient compression for distributed optimization[C]// NeurIPS. 2019: 14236-14245.

[6] 李诗琪. 分布式深度学习模型训练中梯度稀疏方法的改进[D]. 北京:北京邮电大学,2021.

[7] SAPIO A, CANINI M, HO C Y, et al. Scaling distributed machine learning with in-network aggregation[J/OL]. https://www.usenix.org/system/files/nsdi21-sapio.pdf.

[8] SEIDE F, FU H, DROPPO J, et al. 1-bit stochastic gradient descent and its application to data-parallel distributed training of speech dnns[C]// INTERSPEECH. 2014: 1058-1062.

[9] 欧阳硕. 基于梯度压缩的分布式深度学习通信优化技术研究[D]. 长沙:国防科技大学,2021.

[10] ESSER S K, MCKINSTRY J L, BABLANI D, et al. Learned step size quantization[C]// ICLR. 2020.

[11] LI R, WANG Y, LIANG F, et al. Fully quantized network for object detection[C]// CVPR. 2019: 2810-2819.

[12] 刘松伟. 高性能二值卷积神经网络的研究与实现[D]. 杭州:浙江大学,2021.

[13] AJI A FHEAFIELD K. Sparse communication for distributed gradient descent[C]// EMNLP. 2017: 1440-445.

[14] HORVÁTH S, RICHTÁRIK P. A better alternative to error feedback for communication-efficient distributed learning[C]// ICLR. 2021.

[15] LIN Y, HAN S, MAO H, et al. Deep gradient compression: Reducing the communication bandwidth for distributed training[C]// ICLR. 2018.

[16] 牛丽玲. 基于深度学习处理器的Top-K算法实现及应用[D]. 北京:中国科学院大学(中国科学院大学人工智能学院),2020.

[17] ABDELMONIEM A M, CANINI M. Dc2: Delay-aware compression control for distributed machine learning[C]// INFOCOM. 2021.

[18] ALISTARH D, GRUBIC D, LI J, et al. Qsgd: Communication-efficient sgd via gradient quantization and encoding[C]// NIPS. 2017: 1709-1720.

[19] DRYDEN N, MOON T, JACOBS S A, et al. Communication quantization for data-parallel training of deep neural networks[C]// MLHPC@SC. 2016: 1-8.

[20] SHI S, TANG Z, WANG Q, et al. Layer-wise adaptive gradient sparsification for distributed deep learning with convergence guarantees[C]// ECAI. 2020: 1467-1474.

[21] ALISTARH D, HOEFLER T, JOHANSSON M, et al. The convergence of sparsified gradient methods[C]// NeurIPS. 2018: 5977-5987.

[22] CHEN C Y, NI J, LU S, et al. Gopalakrishnan. Scalecom: Scalable sparsified gradient compression for communication-efficient distributed training[C]// NeurIPS. 2020.

[23] BRADLEY J K, KYROLA A, BICKSON D, et al. Parallel coordinate descent for l1-regularized loss minimization[C]// ICML. 2011: 321-328.

HUANG Xinli, born in 2004, undergraduate. His main research interests include computer networks and machine learning.