arXiv:2304.03589v1 [cs.LG] 7 Apr 2023

论大规模深度学习模型的高效训练：文献综述

Li Shen[§*]、Yan Sun[†*]、Zhiyuan Yu[‡*]、Liang Ding[§]、Xinmei Tian[‡]、Dacheng Tao [†]

[§]京东探索研究院，中国
[†]澳大利亚悉尼大学
[‡]中国科学技术大学，中国

mathshenli@gmail.com, ysun9899@uni.sydney.edu.au, yuzhiyuan@mail.ustc.edu.cn
liangding.liam@gmail.com, xinmei@ustc.edu.cn, dacheng.tao@gmail.com

摘要

近年来，深度学习领域取得了显著进展，尤其是在计算机视觉（计算机视觉）、自然语言处理（自然语言处理）和语音等领域。利用在海量数据上训练的大规模模型进行实际应用，具有巨大的潜力，能够提升工业生产力并促进社会发展。然而，其训练过程极不稳定，并且对计算资源有着严苛的要求。随着对计算能力适应性的需求不断增加，尽管已有大量研究在一定程度上探索了高效训练领域，但针对大规模深度学习模型训练的一般加速技术，仍然迫切需要一份全面的总结与指南。在本文综述中，我们详细回顾了训练加速的一般技术。我们考察了基础更新公式，并从五个主要视角对其基本组成部分进行划分：（1）“以数据为中心”：包括数据集正则化、数据采样和以数据为中心的课程学习技术，这些技术能够显著降低数据样本的计算复杂度；（2）“以模型为中心”：包括基本模块加速、压缩训练、模型初始化和以模型为中心的课程学习技术，重点通过减少参数计算并提供更好的初始化来加速训练；（3）“以优化为中心”：包括学习率的选择、大批量大小的采用、高效目标函数的设计以及模型平均技术，关注训练策略并提升大规模模型的泛化能力；（4）“预算训练”：包括在资源受限情形下采用的一些特色加速方法，例如限制迭代次数总量；（5）“以系统为中心”：包括一些高效的分布式框架和开源库，为上述加速算法的实现提供充足的硬件支持。通过提出这一全面的分类体系，本文综述全面回顾了每个组成部分内部的一般机制及其联合交互，以帮助理解这些机制。同时，我们进一步对一般加速技术的发展进行了详细分析，并讨论了未来的研究方向，这有望启发我们重新思考并设计新颖的高效范式。总体而言，我们希望这份综述能够成为一般高效训练的宝贵指南。

1 引言

随着人工智能技术的快速发展，深度模型的参数规模显著激增，达到数百万乃至数十亿。Kaplan 等人 [110] 研究了模型规模、

*前三位作者（按字母顺序排列）贡献相同。

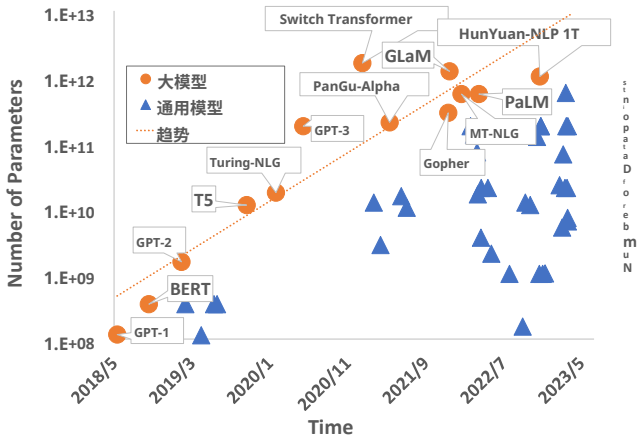


图 1：大规模模型的发展里程碑。

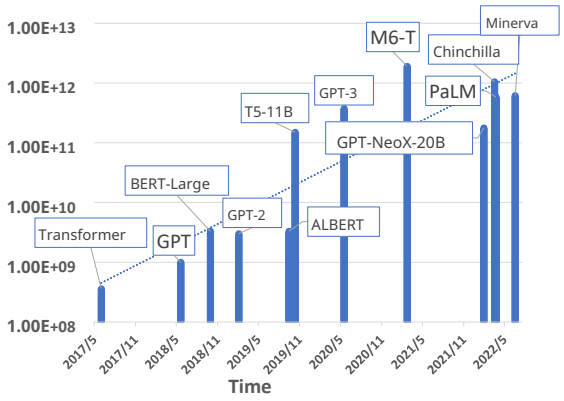


图 2：大型数据集的发展里程碑。

数据集规模以及训练所用计算量之间的幂律关系，并指出，更大的模型天生需要大量数据，而且在学习过程中具有更高的样本效率。大模型的部署也已成为最重要的研究领域之一。例如，Dehghani 等人 [44]提出了 ViT-22B，展示了“类似大型语言模型（LLM）”的扩展在计算机视觉（CV）领域的潜力。GPT-1 [196] 提出了监督微调方法，以驱动拥有 0.1B 个参数的语言模型。两年后，GPT-3 [18] 拥有 175B 个参数，使用 45TB 数据样本进行训练，并在多种自然语言处理任务中成功取得了最先进的成果。Turing-NLG 采用了拥有约 17.2B 个参数的生成式语言模型，仅用一年时间便迅速迭代出参数规模达 530B 的大型模型 MT-NLG [229], , 在若干任务上表现远超 GPT-3。我们在图 1 中总结了模型规模随时间发展的重要里程碑。尽管快速增长带来的收益令人惊叹，但要维持实际应用效率，仍亟待在探索新技术和提升训练能力方面取得实质性进展。目前，训练此类大型模型所需的巨额成本通常难以承受。具体而言，训练 GPT-3 约需 355 个 GPU 年，成本为 4.6M 美元。面对如此庞大的参数规模和数据样本量，传统的从头训练显然无法承担巨额开销，尤其是在扩展到会引入额外架构和过多参数的下游任务时 [30, 138, 187, 188, 211, 243], 。因此，预训练-微调模式日益受到关注，并在深度学习领域大放异彩。

继承迁移学习的范式，预训练的核心是在通用数据集上训练一个通用的基础模型，以获得出色的特征提取器（从优化的角度看，也是一个良好的初始状态），从而帮助在另一项特定任务上实施并稳定训练过程。许多研究领域都受益于预训练方法。具体而言，在计算机视觉领域，通过应用恢复随机遮挡图像的方法，它有助于训练大型 Transformer 模型。掩码自编码器（MAE） [83] 开发了一种非对称编码器—解码器架构，将自监督重建与后端训练相结合，从而在下游任务上实现了出色的迁移性能。同样，图像变换器双向编码器表示（BEiT） [7] 沿用 BERT [48]的方法，通过由分块遮挡图像生成的视觉词元来重建图像。在自然语言处理领域，Radford 等人 [196] 提出生成式预训练（GPT），通过在多样化的无标注文本语料库上采用生成式预训练，实现了显著提升。为进一步提高效率，GPT-2 [197] 大幅提升了模型容量，使其能够作为词向量提取器，并具备丰富的语义知识。GPT-3 [18] 通过双循环策略研究上下文学习，该策略能够显著增强对预训练中知识的理解，并提升实际应用中的流畅性和泛化能力。

场景。Hendrycks等人 [88] 研究发现，使用预训练可以提升模型鲁棒性和不确定性，这表明在海量数据集上训练并实现良好泛化具有显著优势。近期进展表明，具有超大容量的预训练模型带来了显著收益，尤其是在任务无关和少样本场景中。这也为未来发展指明了一个充满前景的方向：在海量数据集上预训练、拥有海量参数的大规模模型，有望超越其他监督训练方法的性能。这一令人振奋的进展有效降低了深度学习对真实值的依赖，并极大推动了大数据的应用。与此同时，它也对计算和训练效率提出了更加严格的要求。高昂的成本，包括时间和金钱成本，严重阻碍了其发展。在此背景下，我们撰写这篇综述，介绍并总结大规模模型训练加速中通用且实用的技术；这些技术已促进训练提速，也有望助力预训练中的大型基础模型。

近年来，许多研究对高效训练技术进行了综述和总结，主要包括高效预训练模型的介绍、为加速而设计的新型组件、先进的优化方法、自然语言处理或计算机视觉社区中的高效训练，以及训练过程中的各种技巧。Qiu等人 [194] 对用于解决各种自然语言处理任务的预训练模型进行了综述，并系统地分类列举了相关模型。他们研究了语言模型的发展历史以及当前预训练领域的进展。Han 等人 [80] 总结了与通用预训练技术相关的研究工作，并对其未来研究提出了一些见解。Bommasani 等人 [15]介绍了高效基础模型，主要从其通用概念、强大能力、底层训练技术和应用等角度展开。他们还总结了预训练的发展历程，以及其在实际场景中面临的当前挑战。Zhou等人 [305]对预训练基础模型（PFM）进行了全面综述，讨论了这些模型在不同领域取得成果方面的最新研究进展，这可能为相关领域的研究带来新的思路。同时，他们系统总结了应用中的一些主要问题和未来挑战。Lin等人 [145] 聚焦于新型 Transformer 模型，并综述了 Transformer 模型的若干变体；这些变体均围绕高效架构修改、预训练技术和训练加速进行设计。Weng [273] 回顾了 Transformer系列的发展历程，详细介绍了 Transformer 模型的演变过程，并系统分析了每种架构的优缺点。Tay 等人 [246] 总结了一些高效Transformer模型的变体。从实际训练的角度出发，他们提出了一些提高 Transformer 模型训练效率的策略，并对未来研究给出了建议。Zhuang 等人 [308]研究了 Transformer 高效训练的概况，包括计算效率、内存效率以及硬件/算法协同设计。与他们不同，我们更加关注基础加速技术，这些技术并不局限于 Transformer 模型。开放机器学习系统社区 [179] 对高效机器学习系统的设计与实现进行了全面研究。他们更加重视数据预处理、前向与反向计算、高效部署与并行通信，以及优化方法的具体实现。He等人 [85] 从泛化保证和优化效率的角度研究了大规模深度学习的最新进展，其中包括新型优化器，以及用于解决训练开销并减少计算设备所需内存的策略。他们还详细阐述了大批量训练的探索。He等人 [84]总结了训练卷积神经网络模型的各种技巧。他们开展了系统实验，并总结了数据增强以及巧妙设计学习率调度器方面的一些有效技术。Treviso等人 [254] 总结了自然语言处理的高效方法，并讨论了这些方法的效率与不足。

近年来，大规模深度学习模型的高效训练已成为机器学习领域的关键研究方向。尽管该领域已取得显著进展，但现有研究大多聚焦于特定的模型架构或服务于特定社群。相比之下，我们的研究全面综述了适用于任何大规模深度学习模型的实用加速技术，不受任务或模型架构的限制。从实际效率的角度来看，我们认为高效训练主要关注两个明确目标：

- 在达到相当测试准确率的情况下，高效训练所需的训练时间更少。
- 在训练成本相近的情况下，高效训练能够取得更高的性能。

我们的综述为深度学习模型的通用训练加速提供了有见地的指导。我们分析了训练加速技术在各种基础骨干架构上的有效性，这些架构构成了许多现代深度学习模型的基础。通过考察深度网络的不同架构，我们的综述有助于实现各类深度学习模型的高效训练。此外，由于我们的综述不依赖特定任务或模型，因此对训练加速技术进行了广泛的泛化，可将其应用于不同的域和模型架构。我们的综述旨在为希望加速大规模深度学习模型训练的研究人员和从业者提供有益的参考。通过理解有效训练加速技术背后的通用原理，研究人员可以开发出更快速、更高效的模型，而不受特定架构或任务的限制。总而言之，我们的研究通过全面综述大规模深度学习模型的通用训练加速技术，为机器学习领域作出了重要贡献。

在我们的综述中，我们关注于解决一般性的基本最小化问题，该问题可以轻松扩展到基础模型训练或预训练任务：

$$\min_{w} \Big\{ F(w) \triangleq \mathbb{E}_{\xi \sim \mathcal{D}} \big[f(w; \xi) \big] \Big\}, \tag{1}$$

其中，w 是参数，ξ 表示受 $\mathcal{D}$约束的数据样本。在实际场景中，我们考虑一个大型训练集（X，Y），并利用经验风险最小化（ERM）损失来近似完整期望 F。在这种情况下，每个单独的数据对（X_i，Y_i）均从训练集中均匀采样。因此，我们求解以下问题：

$$\min_{w} \Big\{ \bar{F}(w) \triangleq \frac{1}{m} \sum_{i=1}^{m} f_i(w) \Big\}, \tag{2}$$

其中，$f_i(w) \triangleq f(w; (X_i, Y_i))$ 表示第 j 个数据样本的损失，m 数据集的大小（X, Y）。¯当 m 足够大时，ERM 将趋近于完整期望，例如对于 $m \to \infty$，$F \to F$。

不同于以往的工作，我们将一般的基于梯度的下降公式分解为本文综述的架构。具体而言，我们考虑公式 (3) 中的所有组成部分，这些部分可以涵盖深度学习中的整个训练过程。我们通过将额外的近端项吸收到 f 中，省略这些项。在不失一般性的情况下，我们使用更新向量 G 替代梯度，以涵盖广泛的方法。我们将基础更新公式定义为：

$$w_T = w_0 - \sum_{t=0}^{T-1} \frac{\gamma_t}{B} \sum_{i=1}^{B} G(w_t; (X_i, Y_i)). \tag{3}$$

在表 1 中，我们总结了符号说明及其对应的研究领域。根据公式（3），通过细化并划分各组件的不同作用，我们依据以往工作的内在启发式见解和理论场景，将其区分为 5 个主要类别。每一类都对应于该类别下计算效率的优化目标。我们进一步细化上述组件，以便在实际可行的前提下，将当前用于训练大规模模型的通用加速技术分类。具体如下：

- **以数据为中心的高效训练**。 在深度学习中，全球预期与训练样本的分布之间通常存在差距。尽管训练早期具有高效性能，但这可能导致训练中期和后期的测试准确率得到提升。为了解决这一问题并提高泛化性能，采用以数据为中心的方法，通过有效的数据增强和正则化策略扩展训练集的样- 本容量。这需要额外的预处理计算来增强多样性，并保持更高稳-定性的潜力，从而在现实应用中带来更好的泛化性能。同时，为了实现高效加速并进一步提高模型的泛化能力，以数据为中心的方法研究有效的采样技术，以便在随机优化过程中选择关键子集。它有效地减少

表 1：公式 (3) 中各术语的符号说明。本文综述的文本结构以此表为纲。我们将训练中的基础更新公式拆解，以介绍当前加速技术如何针对各项进行设计和实现。

组件	定义	研究领域	Text
(X_i, Y_i)	数据集样本	以数据为中心	第 3 节
w	模型参数	以模型为中心	第 4 节
w_0	权重初始化		
B	小批量的大小	以优化为中心	第5节
γ	学习率		
G	梯度估计器		
T	最大迭代次数	预算训练	第6节
$\sum$	求和	以系统为中心	第7节

参与计算梯度所需的样本数量。此外，它还能保护模型，避免在训练过程中对那些不重要的样本或已经得到充分学习的数据发生过拟合。近期研究发现，课程学习提供了一个渐进式过程，从而实现高效训练。在训练早期，它使用正则化程度较低的低分辨率样本，并逐步将其恢复为高质量样本。总而言之，以数据为中心的方法的核心考虑是如何在不影响性能的情况下，减少数据处理需求。

• **以模型为中心的高效训练**。 深度模型是一个从数据域映射到真实值的复杂映射函数。以往的工作探索了多种成熟架构，以构建用于高效训练的网络，例如基于卷积的神经网络（卷积神经网络）、多层感知机和Transformer模型。以模型为中心的方法通过高效的架构近似、压缩和高效初始化，更加关注深度神经网络的计算复杂度，以获得更好的泛化能力。这些方法着重于减小深度神经网络的参数规模，同时保持良好的性能。具体而言，架构近似着重于采用简化的算子组合，以降低训练中的计算成本。它致力于探索基本模块的高表达能力替代方案，以实现通用加速。压缩关注低精度计算和稀疏训练的效率，而这些也需要在硬件实现上得到全面支持。模型初始化着眼于寻找具有更高稳定性和泛化能力的更优初始状态，从而有效加快收敛，并防止训练过程在早期阶段崩溃。总而言之，以模型为中心的方法为降低深度模型的计算复杂度、实现高效训练提供了一种有前景的途径，具有很强的实用性，并且可以轻松地在任何深度学习框架中实现。

• **以优化为中心的高效训练**。 为提高优化效率，我们总结了三个主要因素，即学习率、批量大小和优化目标函数。在深度网络训练的不同阶段，恰当选择学习率和衰减策略是一项关键任务。然而，要找到一种适用于不同模型和优化器的通用方法颇具挑战。因此，以学习率为中心的方法旨在制定高效、灵活的策略，以高效且稳定地训练模型。第二个因素——批量大小——在优化中也发挥着关键作用。借助GPU设备的并行计算能力，增加单个小批量中的样本数量可以提高训练效率，尤其是在计算资源充足时。因此，以批量大小为中心的方法通常着重采用大批量训练来提升优化速度。从优化角度看，我们始终致力于实现稳定性高的目标函数，这正是以目标函数为中心的方法关注的首要问题。这些方法关注的优化目标函数

能够提供对数据分布和模型架构具有鲁棒性的泛化能力。总之，以优化为中心的方法研究训练过程中的高效迭代计算，为高效训练提供坚实的保障。

• **预算约束下的高效训练**。 预算训练是一种在实际训练过程中考虑可用资源的方法。它主要关注资源受限场景下的训练效率，在这些场景中，训练实际耗时或浮点运算次数等计算资源受到限制。预算训练的首要目标是在给定约束条件下，确保训练高效且稳定，同时最大限度地发挥模型潜力。这种方法能够在训练早期阶段带来显著收益。采用预算训练后，研究人员和从业者可以充分利用可用资源，避免将资源浪费在低效的模型或训练流程上。这种方法还有助于开发更实用、更适合真实应用场景的模型，因为这类场景中的资源往往有限。

• **以系统为中心的高效训练**。 以系统为中心的方法着重于在硬件支持下进行实际部署，从而将算法设计转化为可执行的实际项目。大规模模型通常采用多节点、多设备环境来实现并行计算。这类方法主要关注底层逻辑的设计，以解决设备间通信的瓶颈，并高效协调整个训练过程。目前已有多个开源框架，能够显著加速深度网络的训练。为了有效利用分布式训练，训练过程会被拆分为较小的计算任务，并在不同节点或设备上并行执行。这些节点相互通信，以交换梯度更新并同步整个训练过程。借助这种分布式系统，可以训练无法在单台机器上运行的大型数据集和复杂模型。目前已有多个开源分布式训练框架，例如 TensorFlow、PyTorch 和 Horovod。这些框架使多节点、多设备集群上的分布式训练更加高效，并显著缩短了大规模深度学习模型的训练时间。

总而言之，本文综述了用于高效训练的通用训练加速方法。在“以数据为中心”、“以模型为中心”、“以优化为中心”和“预算训练”部分，我们主要从算法设计和方法论的角度关注综合性研究；而在“以系统为中心”部分，我们则从范式创新和硬件支持的角度关注实际实现。本文的主要贡献如下：

• 我们从“数据”“模型”“优化”“预算训练”和“系统”五个视角，综述了训练大规模模型的一般加速技术，总结了各个组件的技术路线与实现，有助于为无任务、无模型的高效训练提供坚实的指导。

• 我们比较训练加速中各组件的优势与劣势，并展示它们的见解和相互作用，这或可启发我们重新思考训练大规模深度学习模型的高效范式设计。

• 我们全面分析了各条技术路线及其在实际场景中面临的主要挑战，可为其未来的发展方向提供参考。

本综述的主要结构安排如下。第2节介绍一些预备知识，包括不同骨干网络中的基本模块、大规模深度学习模型的预训练、数据集，以及本综述采用的详细符号说明。从第3节到第6节，我们基于迭代公式（3），从“以数据为中心”“以模型为中心”“以优化为中心”“预算训练”和“以系统为中心”的角度，介绍其特点和属性的详细内容，并阐述这些方法在训练加速方面采用的不同技术路线。我们还分析并评估了各项实现的优势与不足。这一新颖的分类体系可为现有的高效训练方法提供清晰而全面的指导。第8节将讨论并总结本综述中的技术，并提出未来一些有前景的研究方向。

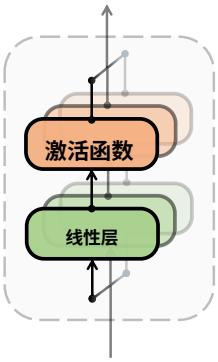


图 3：多层感知机的基础结构。

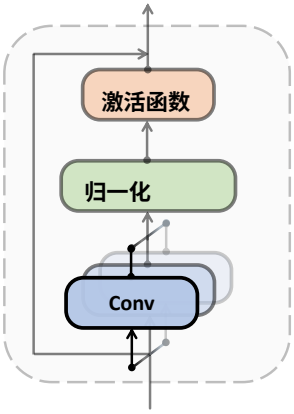


图 4：卷积神经网络的基础结构。

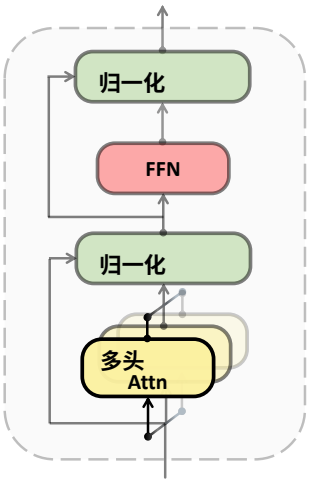


图 5：Transformer 的基础结构。

2 预备知识

本节首先介绍大规模深度学习模型中的若干基本模块。然后介绍一些常用数据集，并说明本综述中使用的若干符号。

2.1 基础骨干网络

在本综述中，我们重点关注通用深度学习骨干网络的高效训练。大多数骨干网络都是由基本经典模块经过复杂组合堆叠而成，主要包括 MLP（线性层模块）、CNN（卷积层模块）和 Transformer（多头注意力模块）。本节仅介绍这些模块的基本概念。在第 4 节中，我们将给出具体公式，说明它们在深度模型中的工作方式，以及如何改进和优化它们以实现高效训练。

MLP。 多层感知机（MLP）是一种广泛使用的经典模型，在深度神经网络的发展过程中发挥了重要作用。MLP 由多个线性层组成，也称为全连接层，并配有逐元素激活函数。在训练过程中，除了输入层和输出层之外，MLP 还会生成大量隐藏神经元。其参数包括投影矩阵和偏置向量。图3 展示了 MLP 的简化示意图，说明了线性计算与非线性激活值的结合。尽管结构简单，但通过整合现代网络结构和训练技术，可以提升 MLP 的性能。例如，*MLP-Mixer*是Tolstikhin 等人提出的一种完全基于MLP 的架构， [250], 包含两种直接应用于图像块或跨图像块应用的 MLP 层，在图像分类基准上取得了当前最先进的性能。

卷积神经网络。卷积神经网络（CNN）是一种强大的深度学习模型，广泛用于视觉图像分析。卷积神经网络可以被视为一种专门用于卷积运算的多层感知器。其参数包括核矩阵（也称为滤波器）和偏置向量。对于图像的每个小区域，卷积神经网络都使用一组相似的核来提取高级特征，这可以看作是将重叠的重构图像相乘。此外，卷积神经网络在训练过程中确保平移不变性，从而充分模拟人类视觉系统。图4 描绘了卷积神经网络的基本构建模块，该模块由卷积层、归一化层和逐元素激活函数组成。一些典型的卷积神经网络结构包括AlexNet [123], VGG [223], Inception [102,

239, 240, 241], 和 ResNet [82]。此外，卷积神经网络通过针对更具体的任务调整其架构得到了优化。例如，全卷积网络（FCN） [152] 是专为语义分割任务设计的卷积神经网络变体。YOLO [207]是一种目标检测网络，通过直接从输入图像中预测边界框和类别概率来实现实时性能。总之，卷积神经网络是深度学习的重要组成部分，并且一直在不断发展，以在各种计算机视觉应用中实现最先进的性能。

Transformer。 经典 Transformer [256]最初采用编码器—解码器架构，旨在从自然语言中提取信息。其基本构建单元称为一个单元，由两个模块组成：多头注意力（MHA）和前馈神经网络（FFN）。MHA 是一种模块，它并行运行多个相互独立的自注意力层，以捕捉输入在不同特征层级上的高级语义。这使模型能够联合关注来自不同表示子空间以及序列不同部分的信息。FFN 是一种特征提取器，可将不同 MHA 模块提取的高级语义映射到同一特征空间。Transformer 架构也为计算机视觉领域作出了重要贡献。例如，它启发了视觉变换器（ViT） [53], DeiT [252], 和 Swin Transformer [150]等模型的开发。这些模型在计算机视觉任务中取得了最先进的性能，包括目标检测、图像分类和图像描述生成。基于 Transformer 架构，还出现了 BERT [48] 和 GPT [196]等经过改进的预训练基础模型。仅编码器的 BERT 和仅解码器的 GPT 在问答、语言推理和文本分类等自然语言处理任务中取得了卓越的成果。总之，Transformer 架构能够捕捉并提取输入在不同特征层级上的高级语义，因此为自然语言处理和计算机视觉领域作出了重要贡献。它的发展促成了新型模型的诞生，这些模型在众多任务中实现了最先进的性能。

2.2 大规模模型和数据集

不断增加的参数量和日益扩大的数据集规模，使大规模深度学习模型的训练过程更加耗时，也因此需要开发更先进的框架和系统，以进一步加快训练过程并降低成本。下面，我们首先介绍几种常用的大规模深度学习模型及其对应的训练数据集。

大规模模型。 自 Vaswani 等人于 2017年提出 Transformer 以来，它已成为自然语言处理领域的主导模型，推动了大型语言模型（LLM）和视觉Transformer（ViT）的蓬勃发展。若干（预训练）Transformer 模型展现了高效训练的发展，无论是在模型结构改进方面，还是在高效训练技术的应用方面。对于自然语言处理，自 GPT（拥有 117M 个参数） [196], ——首个由 Transformer 解码器构成、在 BookCorpus 上进行无监督预训练的预训练 Transformer——问世以来，模型参数数量呈现快速增长的趋势。与 GPT 的结构不同，BERT（340M） [48] 由多个双向 Transformer 编码器层组成，能够捕捉不同位置之间的关系。GPT-2（1.5B） [197] 的参数数量远超 GPT-1，并在更大的数据集上训练。事实证明，使用大规模网络训练语言模型是一种非常有效的策略；数据集的增长、高效的模型结构以及软硬件领域的突破使其成为可能，并促使 Turing-NLG（17.2B） [35], 、GPT-3（175B） [18], 、Megatron-Turing NLG（530B） [229]和 Switch Transformer（1.6T） [60] 等模型进一步增加参数数量。至于计算机视觉，ViT（ViT-Base 版本有 86M 个参数） [53]是 Transformer 应用于这一领域的里程碑。近期，受扩展 Transformer 推动大型语言模型取得突破的启发，Dehghani 等人 [44]提出了一种高效且稳定的训练方案，将 ViT 扩展至 22B 个参数（ViT-22B）。在跨模态或多模态领域，CLIP [198]通过联合训练图像编码器和文本编码器，将分类任务转化为图文匹配任务；其视觉骨干采用 ResNet-50 [82]或 ViT，语言骨干则采用 Transformer。它还随附发布了 DALL-E [204], ，这是 GPT-3 的一个 12B 参数版本，能够根据图文对的预训练，从输入文本生成图像

表 2：本综述中使用的符号说明。

组成部分	定义
A	矩阵
d	参数维度，其中 $w \in \mathbb{R}^d$
m	数据集大小（X，Y）
H	中间层的输入/输出
ϕ	激活函数
Φ	归一化
n	词元数量
h, b	特征图的高度和宽度
L	交叉熵损失
N	非嵌入模型参数数量
D	数据集词元数量
$C_{\min}$	达到给定损失所需的最少计算量

预训练而成。在上述模型中，大多数属于或基于 Bommasani 等人 [15], 所定义的基础模型，即通常利用扩展的自监督方法在广泛的数据上训练、并可适配（例如微调）以用于广泛下游任务的模型。Wang 等人 [265], 提出通用型大规模模型 BEiT-3，并指出，不同域的大规模模型研究正逐渐趋向“重大收敛”：首先，Transformer 已成为通用的骨干架构；其次，生成式预训练已成为最重要的自监督学习方法；第三，扩展数据和模型规模能够进一步提升大规模模型的性能。尽管模型形式多样，即便是最庞大的模型，也由线性层、卷积层和注意力层等基本模块构成。因此，我们总结了大规模深度学习模型训练的一般加速方法与技术，这些内容借鉴自主流预训练基础模型及其他一些不局限于自监督训练的研究。

数据集。 大型语言模型需要在海量数据集上进行预训练，以学习适用于下游任务的通用语言表示。例如，BERT [48] 模型在两个大型数据集上训练：BooksCorpus（8亿词） [307] 和英文维基百科（25亿词）。当前模型使用了更新、更大的数据库，例如 C4（超大规模清洗爬取语料库）、 [200], Pile、 [67], OSCAR（开放式超大规模爬取聚合语料库）、 [232], mC4（C4 的扩展多语言版本）、 [284], BigScience ROOTS 语料库（3500 亿词元）； [125] 例如，LLaMA（Touvron 等人） [253] 混合使用了来自 C4、GitHub、维基百科、古腾堡计划（一个收录公版图书的项目）、Books3（Pile 的一个部分）、arXiv 和 Stack Exchange 的资源。最大的两个多语言数据集是 OSCAR，截至2023年1月包含152种语言，规模为9.4TB；以及 mC4，包含101种语言，规模为 27T。Thompson [248]总结了从 GPT-1 [196]到 Gopher [199]等部分语言模型所使用的数据集。

大型数据集可能包含经过筛选的网页、书籍、文章、维基百科、新闻、源代码和社交媒体对话。模型可以基于对数据集中不同部分进行按数量和质量加权的采样 [18, 253]来学习。在计算机视觉领域，ImageNet [45] 是一个广为人知的大型标注照片数据集。其最常用的子集 ILSVRC（ImageNet 大规模视觉识别挑战赛） [212], 包含超过 100 万张图像，涵盖 1,000 个物体类别。作为预训练模型所用数据集的一个示例，近期的 ViT-22B [44]是在 JFT [234] 的一个版本上训练的，该版本包含约 40 亿张图像 [292]。

符号说明。 表2展示了本综述中使用的一些符号规范。我们使用大写字母表示矩阵。 d 是模型的大小。 m 是数据集的大小。 H 是训练过程中的特征图（中间变量）。 $\phi(\cdot)$ 是激活函数， $\Phi(\cdot)$ 是归一化函数。

n 是Transformer模型中的词元数量。 h， b 是输入样本（图像）的高度和宽度。L 是交叉熵损失。 N 是非嵌入模型参数的数量。 D 是以 $D = nm$表示的数据集词元数量。 $C_{\min}$ 是达到目标损失或目标准确率所需的最小计算量。

3 以数据为中心的高效训练

近年来，大规模模型取得了显著进展，与此同时，对数据集的需求也急剧增加。训练过程需要海量数据样本驱动，才能取得卓越性能。因此，研究以数据为中心的方法对于实际加速至关重要。数据处理的关键作用在于无需额外标注即可有效提高数据样本的多样性。数据标注往往成本过高，难以承担，这凸显了以数据为中心领域的研究的重要性。同时，这类研究也致力于提高数据样本的并行加载效率。

本节将所有这些高效的数据处理方法统称为“以数据为中心”的方法，它们能够显著提升大规模模型训练的性能。我们从以下几个角度回顾并研究相关技术：

- **数据正则化**。 数据正则化是一种预处理技术，通过一系列变换增强原始数据样本的多样性。它无需额外的标注信息，即可在特征空间中等效地改善训练样本的表示。高效的数据正则化已广泛应用于训练过程，并显著提升了大规模模型的泛化性能。

- **数据采样**。 数据采样是一种从大批次中选取样本子集来执行更新的有效方法。它能够忽略当前批次中不重要或质量较差的样本，从而采用小批大小进行训练。通常，采样得到的数据更为重要，因此能够取得与使用全批次训练相当的性能。每次迭代的概率会随着训练过程逐步调整，以确保无偏采样。

- **以数据为中心的课程学习**。 课程学习研究训练过程不同阶段的渐进式训练设置，以降低总计算成本。在开始阶段，使用低质量数据集进行训练足以学习低层特征。然后采用高质量数据集（更多数据增强和复杂的预处理方法）逐步帮助学习高级特征，并达到与完整训练相同的精度。

3.1 数据正则化

在大型高质量数据集上进行训练，其性能将优于在小型、组织不佳的数据集上进行训练。对于个人研究者而言，手动收集和处理大量数据通常并不现实，因此需要选择一些开源数据集，并考虑结合多领域、多语言数据集，以提高数据多样性，进而提升模型的泛化能力。因此，可能需要应用相应的预处理和清洗方法，例如去重 [127]（通常使用局部敏感哈希，LSH [222]）、移除词元数量较少的文档以进行去噪等。此外，研究高效的数据集正则化有助于提升大规模模型的训练性能和泛化能力。数据正则化的目的是减少过拟合，使模型更易于适应，并在不过多影响训练过程的情况下提升模型泛化能力。然而，它通常会引入额外的计算成本，有时还会成为训练过程中的瓶颈。为了实现高效训练，从一系列技术中进行选择和测试，并决定采用哪一种技术至关重要。

Szegedy等人 [240]提出了标签平滑正则化，通过在标签分布与另一种固定分布（例如均匀分布）之间进行插值来替换标签。这种方法缓解了过拟合问题，并使模型具有更强的适应性。He等人 [84] 表明，标签平滑

使分布集中于理论值附近，并减少极端值。此外，M¨uller 等人 [171] 提出，标签平滑还能改善模型的校准效果。他们还观察到，如果教师网络使用标签平滑进行训练，那么将知识蒸馏到学生网络中的效果会大幅降低。为解释这些现象，他们通过可视化表明，标签平滑会促使训练实例的表示在正确类别中形成紧密的簇，但也会导致不同类别样本之间的相似性信息丢失，从而对模型蒸馏的效果产生负面影响。

数据增强（DA）是另一种重要的正则化策略，有助于实现高效训练。通过更好地利用原始数据集，例如对图像进行变换，数据增强可以人为且有效地扩充训练数据，从而提升模型的泛化能力。甚至一些看似违反直觉的数据增强方法，也能有效提升模型性能。在计算机视觉领域，AlexNet [123], ResNet [82], 和 EfficientNetV2 [244] 等此前最先进的模型在训练中都采用了图像数据增强技术。基础数据增强方法有很多，大多相对简单，例如旋转、平移、错切、翻转、裁剪和缩放，调整对比度和亮度、添加噪声、高斯模糊、颜色抖动、格式转换（例如将 RGB 颜色模型转换为 HSV 表示，其中 HSV 分别代表色相、饱和度和值）、滤波等。近期研究证明，基础擦除方法也很有效，例如 CutOut [49] 会在训练过程中随机从输入中裁去一个正方形区域；Random Erasing [304] 也会擦除图像中的矩形区域，并用随机值或均值填充。许多数据增强技术都基于 Zhang等人提出的 Mixup 思想 [294],，即混合特征及其对应的标签。还有一类 Mixup 扩展方法，例如 Yun 等人提出的 CutMix，它会裁剪图像中的一个小块并用另一张图像的对应区域替换。较新的研究方法包括 SaliencyMix [255],、Manifold Mixup [257], 、StyleMix [94], 、TokenMix [146], 、Co-mixup [117], 、Supermix [38], 和 TransMix [23]。

更先进的研究探讨了各种数据增强算法的组合使用，并进入了自动数据增强领域，其中 AutoAugment [36]是一个重要代表。它使用强化学习中以控制器 RNN 实现的搜索算法来寻找数据驱动的数据增强策略。然而，这会带来高昂的计算成本，因此只能应用于小型数据集。其他先进算法，如基于种群的数据增强（Population Based Augmentation，PBA） [91], 、快速自动增强（Fast AutoAugment） [144], 和对抗性自动增强（Adversarial AutoAugment） [299] ，进一步降低了成本。随机组合也可能很有效。例如，Hendrycks等人 [89]提出了一种名为 AugMix 的数据增强技术，该技术随机选择不同的数据增强方法，然后混合增强后的图像。Cubuk 等人 [37] 提出了 RandAugment，它从多种图像增强方法中随机选择一定数量的方法，设定一个通用的变换幅度，并对每个样本依次应用这些方法。他们研究了不同数据增强方法的影响，并且表明，他们的方法能够以较低的成本将学习到的数据增强策略扩展到更大的数据集和模型上，并获得更好的性能。

在自然语言处理领域，数据增强通常在训练过程开始前进行，其方式与图像数据增强相比有所不同。根据 Feng 等人 [62]，的研究，一些数据增强技术包括：基于规则的技术，例如 简易数据增强 ，Wei 和 Zou [271], 对其进行了总结，包括随机插入、交换、删除和同义词替换。其中，同义词替换方法包括 Zhang 等人 [298] 提出的方法：根据语义相似度排序，随机用词语或短语的同义词替换原词语或短语；以及 Wang 和 Yang [266] 提出的方法：在嵌入词汇表中搜索某个词语的 k 近邻（KNN），并用搜索结果替换该词语，从而创建新的实例。以 Mixup 为先驱的示例插值技术，例如 SeqMix [76]。基于模型的技术，例如回译 [56, 219], ，即先将文本翻译成另一种语言，再翻译回来。随着大语言模型展现出强大的语言理解能力，一些近期研究开始使用最先进模型进行数据增强，例如 ChatAug [39] 使用 ChatGPT 作为数据增强工具，将输入句子改写为更多新句子。

我们注意到，先进的数据增强方法可以提升鲁棒性，但也会带来额外负担；与较小的模型相比，这些方法通常更适用于参数更多、规模更大、训练时间更长的模型。特别是，当训练过程的其他部分存在瓶颈，且

执行数据增强计算不会影响最终耗时，那么就可以加入数据增强方法，以增强效果并提高吞吐量。Steiner 等人 [231] 开展了一项实证研究，结果表明，数据增强（他们使用了 Mixup 和 RandAugment）与模型正则化合称为*AugReg*，对于数据量相对较少的情况是有效的。

3.2 数据采样

训练过程中，模型会基于样本批次进行更新，并将批次中的所有样本一视同仁。然而，这可能导致训练过程花费更多时间处理常规或信息量较少的样本。为加快训练并避免这一问题，可以采用数据采样等方法，包括重要性采样、设计新颖的目标函数等，以提高数据效率。

重要性采样是一种蒙特卡洛方法，通过从不同的分布中采样，帮助评估某个分布的属性并降低方差。在随机梯度下降中，重要性采样着重关注对模型参数影响最大的样本，从而降低梯度估计的方差。当计算成本固定或批量大小增加时，这种方法有助于降低计算成本。选择合适的重要性采样分布至关重要；该分布应准确反映原始数据分布，以避免引入偏差。Katharopoulos 和 Fleuret [113] 推导出梯度范数的上界，并提出一种利用重要性采样降低方差的估计器。Katharopoulos 和 Fleuret [112] 提出，可以使用损失值而非梯度范数作为重要性采样的指标。Jiang 等人 [108] 提出了选择性反向传播方法，通过跳过低损失训练样本的反向传播过程来加速训练，从而减少样本使用量和反向传播计算。他们表明，该方法比传统标准 SGD 收敛更快。采样后的批次能够提高数据吞吐量，从而加快训练过程。

Chen 等人 [29] 提出在参数上生成掩码，以冻结输入的反向传播。该方法保留前向图，并且归一化等层中所有批次的统计特征保持整个数据集的映射关系。Zhang等人 [296]关注训练期间表现较差的样本。他们利用选择器突出显示大批次中梯度较大的训练样本，并将其作为有效近似。梯度较小的样本在这一轮迭代中被视为“好点”，而网络需要对“坏点”给予额外关注。Mindermann 等人 [167]建立了一种新颖的 *RHO* 损失函数，从大批次中选择一个子集，并根据伯努利采样的估计更新损失。*RHO* 依据自身的评估标准，使用不同损失值衡量输入的重要性，然后选择重要样本来更新模型。Kawaguchi 和 Lu [114], 、Ni 等人 [176]实现了类似的思想。在 [176], 中，他们提出 *K-SAM* ，根据梯度计算选择两个样本子集，从而有效减少反向传播的开销。Kawaguchi 和 Lu [114] 研究了 *Ordered-SGD* ，该方法直接选择小批量中损失函数值最高的前 q 个样本，以在每次迭代中执行更新。在每个小批量中，根据损失值选择 k 个样本进行训练，使得小批量能够达到与大批次相当的训练效果。Liu等人 [151]采用低秩和稀疏输入，以加速图神经网络中的聚合。他们将输入图划分为若干稀疏块，以降低稠密矩阵的计算开销，其复杂度为 $\mathcal{O}(N^2)$。He 和 Dube [81]使用计算能力不同的边缘设备模拟去中心化框架。他们在训练过程中交替移除速度最慢的设备，以直接减少延迟时间成本，这可以看作动态批大小方法的一种变体。Xie 等人 [279] 在缩减后的特征空间中工作，通过 *KL* 缩减，使文本空间上的重要性权重估计变得可处理，并利用重要性采样高效训练语言模型。Sujit 等人 [233] 应用类似思想，对强化学习的输入样本进行裁剪。Cai 等人 [19] 舍弃不相似的图像块，以在 Transformer模型版本中实现同样的数据减少效果。

基于采样的加速方法侧重于直接减少参与计算的数据总量。这些研究间接验证了训练中的数据冗余。换言之，

训练中涉及的数据量与整个数据集中的数据量之比，已成为衡量样本利用效率的重要指标之一。如何设计利用率高、计算成本低的算法，将继续作为加速领域的研究方向。

3.3 以数据为中心的课程学习

课程学习 [9, 57] 基于这样一种直觉：先从简单任务开始学习，然后逐渐增加任务的复杂度。这种方法也以多种方式应用于预训练，并且在许多研究中展现出了良好的加速效果。Wu 等人 [275] 表明，当训练时间预算有限或存在噪声数据时，课程学习可以提升性能。在应用课程学习时，更重要的挑战之一是找到合适的标准来判断样本的难度。

渐进式缩放 [95, 111] 是一种将小型下采样图像逐步训练到大型完整图像的技术，通过训练过程提升模型性能并缩短收敛时间。例如，一组建议的参数是：初始图像缩放为完整图像的一半，在训练过程的前半段每次增加4个像素，最后 20% 的过程用于微调。Touvron等人 [251]提出了FixRes，并表明，以较小分辨率训练网络、再以较高分辨率进行推理，可以实现更好的泛化。使用低分辨率图像进行训练还有助于节省训练成本：使用 224进行训练、使用 384进行测试，比直接使用384进行训练的效果更好。Wang 等人 [269] 在输入图像的傅里叶频谱中引入裁剪操作，使模型能够首先从低频成分中学习，作为一种课程学习形式，从而减少训练时间。Koc¸yi˘git 等人 [121] 采用渐进式分辨率训练 [244], 1与循环学习调度技术 [226], 以及困难增强选择技术来训练自监督学习任务，并实现训练加速。对于自然语言处理任务，短序列/长序列对应于低分辨率/高分辨率图像。Press 等人 [191] 表明，先使用较短的子序列训练 Transformer，再切换到较长的子序列，可以实现加速。Ding等人 [51] 发现，按照词-短语-句子的数据学习顺序训练机器翻译模型，可以有效促进跨语言建模，从而获得更好的翻译效果。在训练过程中，使用低分辨率图像时，我们应保持词元数量不变，并增加批量大小。Li 等人 [131] 提出了序列长度预热，在训练过程的早期阶段线性增加序列长度。在复现 GPT-3模型（125M）的实验中，他们表明，该方法能够以更大的批量大小和更高的学习率稳定训练，并且使用更少的数据和更短的时间，性能超过原始的 GPT-3 训练方案。Li 等人 [133]提出了一种基于课程学习的自适应长度调节技术，以稳定GPT的训练过程，从而获得一种能够实现实际时间加速的高效大批量训练调度方案。Nagatsuka 等人 [172] 提出在 BERT训练过程中逐步增加输入文本的块大小，结果表明，与RoBERTa[149]基线相比，该方法具有更快的收敛速度和更高的训练效率。

3.4 讨论

在本节中，我们从以数据为中心的视角回顾高效训练技术。我们重点关注数据集正则化的数据效率，包括通过数据增强和预处理来提升其多样性；从大批次（或整个数据集）中采样有效子集，以提高训练效率；以及课程学习方法，以缓解训练早期阶段高昂的消耗。

数据正则化是一种无需额外标注即可扩展数据样本多样性的有效方法。由于采用了复杂且随机的变换，数据集相当于被多次扩展，这有助于训练具有更佳泛化性能的大规模模型。然而，过度使用数据正则化会引入巨大的偏差，导致质量下降。随着可用正则化方法越来越多，选择有效的组合变得十分重要。在数据采样方面，它能够通过小批量训练实现相同甚至更好的性能。通过精心设计的算法进行数据采样，可以有效提升训练模型的容量。通常，在训练过程开始时，

模型从表现较差的样本中获益更多。忽略优质样本有助于加快对不同带标签样本特征的学习平衡。此外，课程学习为处理数据集提供了渐进式流程。它允许使用较少的数据增强来输入低分辨率数据，以提取粗粒度特征，并逐步提升数据质量以捕获细粒度特征。在保持高性能的同时，实现了切实可行的加速。

未来，我们认为，从以数据为中心的视角出发，有前景的研究包括：

• 高效正则化实现。值得研究更多任务特定的数据正则化方法。基于先验知识设计新颖的数据正则化方法，可以进一步提升大型网络的泛化性能。同时，采用数据正则化的并行处理也能大幅加快训练过程。

• 高效数据采样。期待出现能够同时考虑训练加速和泛化保证的新颖采样方法 [167]。

4 以模型为中心的高效训练

设计高效的模型架构始终是深度学习领域最重要的研究方向之一。优秀的模型是有效的提取器，并且可以映射为易于分离的高级特征。不同于其他额外关注高效新颖结构的研究，我们以模型为中心的研究更加关注通用模块的等价替代方案，这些方案能够在性能相当的情况下实现更高的效率。几乎所有大模型都由小型模块或层组成。因此，我们的综述可以为高效训练大规模模型提供良好的指导。

在本节中，我们 将从以下几个角度探讨这些高效的“以模型为中心”的技术 ：

• **架构效率**。 随着深度模型的参数量迅速增加，其计算开销也随之大幅上升。因此，设计高效的替代方案来近似原始架构，已成为一个热门话题。这种替代不仅是对数值计算的近似，还包括深度模型的结构简化与融合。在本部分，我们根据不同架构对现有加速技术进行分类，并阐述其原理。

• **压缩训练效率**。 压缩一直是计算加速的研究方向之一，在数字信号处理（多媒体计算／图像处理）中发挥着关键作用。传统压缩主要分为两个分支：量化和稀疏性。我们将详细介绍它们在深度训练中的现有成果与贡献。

• **初始化效率**。 模型参数的初始化是现有理论分析和实际应用场景中的一个非常重要的因素。不良的初始化状态甚至可能导致整个训练过程崩溃，并在训练早期停滞；而良好的初始状态则有助于在平滑的损失景观中加快整体收敛。在本部分，我们主要从模型初始化的角度研究评估方法和算法设计。

• **以模型为中心的课程学习**。 从以模型为中心的视角来看，课程学习通常从训练小模型或大规模模型中的部分参数开始，再逐步恢复到完整架构。事实证明，这种方法能够加速训练过程，且不会产生显著的负面影响。本部分将回顾其在训练过程中的实现方式与效率。

4.1 架构效率

我们回顾的这项技术，是在现有模型模块上采用高效的替代方案来加速计算。作为大模型的基石，每个基础模块都具有不同的属性，应分别进行研究。本节主要研究以下模块的效率，这些模块广泛应用于大模型的构建。我们的回顾清晰地指出，从以模型为中心的视角来看，通用加速技术具有较高的效率。

4.1.1 高效线性层

线性层是 t 传统模型中的基本模块，也称为全连接层 r
在多层感知机中。它可能 可视为：

$$H_{out} = HW + b, \tag{4}$$

其中 W 是权重， b 是偏置。它本质上是矩阵乘法，因此许多针对乘法的基础优化都能节省资源并加速其计算。此外，在深度模型训练中，大量研究工作对其近似方法进行了探讨，这些方法在实践中可以减少参数量和计算复杂度。Li 等人 [140] 提出了一种 GPU 友好的蝴蝶因子分解，将乘法拆分为两个递归块对角矩阵，仅使用 $1/r$ 个参数（r 是秩控制器），同时取得了可比的性能。从理论上讲，它将一个 $N \times N$ 矩阵化简为 $\mathcal{O}$ 个（log N）矩阵的乘积，仅需 $\mathcal{O}$（Nlog N）次运算和相应内存。Meng 等人 [165] 基于蝴蝶因子提出了可高效计算且可逆的蝴蝶层，其雅可比行列式的复杂度更低。为了将这一方法扩展到计算设备上，他们将 *ButterflyFlow* 的架构整合为用于训练深度模型的独立模块。Song 等人 [230] 研究了 MLP 层中一种特定的 GPU 友好稀疏性，以加速训练。该方法采用近似随机丢弃，并结合规则的在线按行或按块丢弃。这种敏感性感知的丢弃方法有助于减少前向和反向过程中所需的输入特征图。线性层加速主要关注低秩和稀疏矩阵乘法，以保持可比的性能。在当前的大型模型中，它并不是时间消耗的主要部分，与其他模块相比，其计算相对简单。基础性研究可以启发我们设计更高效的结构，以实现加速训练。

4.1.2 高效卷积

卷积始终是为下游任务提取高级特征的有效技术，在若干计算机视觉任务中尤为如此。它能保持图像的平移不变性和旋转不变性，从而高效地捕捉细节特征。深度模型中的一般卷积计算可写为：

$$H_{out}(x,y) = \sum_{i=0}^{h-1}\sum_{j=0}^{b-1} H(x-i, y-j) W^c(i,j), \tag{5}$$

其中，(h， b) 是输入 H 的高度和宽度（我们考虑单个特征图）， W^c 是卷积核。

可以看出，卷积计算具有相当大的重复性，其中大量的矩阵乘法使其极其耗时。因此，卷积加速始终是深度网络中的一个重要研究课题。Tai 等人 [242] 利用张量的低秩分解来评估卷积滤波器中的冗余，并针对参数提出了一种新的低秩约束。与原始卷积相比，该方法实现了显著的训练加速和更好的性能。大量实证实验验证了其高效性，同时测试准确率具有可比性。Zhao等人 [303]提出了一种混合方法，将 Winograd最小滤波与 Strassen算法相结合，从而大幅降低计算复杂度。该方法同时减少了网络所需的浮点运算次数和卷积次数，并且在视觉几何组网络（一种纯卷积结构）的测试中几乎没有准确率损失。
Maji 等人 [161] 更加关注高效实现

面向内存和预算资源有限的嵌入式 CPU设备。它填补了相关空白，并将神经网络迁移到移动设备中，以支持本地应用。Cheng和Parhi [31] 研究了一种新颖的快速卷积算法，用于移除 1D 和 2D 卷积中受控单元上的冗余乘法运算。该算法还处理输入特征，并通过一种独立于卷积核大小的灵活块大小生成输出。此外，交替计算神经元也大幅提升了内存访问效率。Wang等人 [270] 提出了一种新方法，研究深度网络中的 1D 卷积，称为*ODLS* （一维深度低秩与稀疏）。它简化了架构，使深度网络更易于在加速磁共振成像任务中进行训练和泛化。Ghimire 等人 [69]总结了加速卷积计算和实现硬件高效化的主流技术，包括分解、压缩、低秩近似和稀疏性。尽管Transformer模型取得了最先进水平的结果，并影响了卷积层的使用，但当前的一些高效架构考虑将Transformer与卷积模块相结合，以进一步提取用于计算机视觉任务的细粒度特征 [139, 283]。未来，卷积仍将在架构设计中发挥巨大作用。作为一种高效的特征提取器，加速其实现同样是值得探索的重要课题。

4.1.3 高效激活函数

激活函数是一种手工设计的函数，用于对输入进行逐元素缩放，帮助深度模型学习复杂模式和高级特征。它类似于我们大脑中基于神经元的机制，最终决定哪些信号会传递给下一个神经元。它可以表示为：

$$H_{out} = \phi(H), \tag{6}$$

通常， $\phi(\cdot)$ 是一个非线性函数。

激活函数为深度网络构建非线性映射，有效提升了其在各种场景中拟合复杂目标函数的能力。自*ReLU* [78, 123] 被提出用于模拟生物机制以来，深度神经网络已在许多领域取得长足进展。它能防止梯度在层间反向传播时消失或爆炸。一系列相关的优化型激活函数也得到应用，例如*LReLU*、*ALReLU*和*LeLeLU*。随着人们逐步设计出能够解决大型任务的大模型，也开始采用更高效的激活函数。Clevert 等人 [34] 提出了*ELU* 激活函数，可将其视为具有双侧饱和特性的*Tanh* 。它通过指数乘法控制负值输出，以解决梯度消失和神经元死亡问题，从而加快训练过程。Hendrycks 和 Gimpel [87]研究了*GeLU* ，它根据输入的具体值而非符号对输入进行加权，有助于改善各种大模型的平滑性。训练早期的平滑性可有效减少训练轮次，从而加快训练。为进一步提升网络中数据流的稳定性，Klambauer 等人[120] 结合数据归一化技术，开发了*SeLU* 激活函数，使所有层的输出都能呈正态分布。输出的一致性有助于在反向传播中维持有效梯度，并加快学习。Carlile 等人 [20] 着眼于均值在零值附近偏移的优势，设计了*ISRLU* ，以节省零值左侧指数函数带来的高昂计算成本。其多项式式复杂度远低于指数函数，并能在 MLP 网络中实现高效训练。Raffel 等人 [200] 验证了 *SwiGLU* 激活函数在前馈网络模块中的效率，而前馈网络模块是 Transformer 的重要组成部分。微调系数可以提升不同层输出分布的相似性。在预训练任务中，它往往能减少所需的训练轮次。激活函数通常按元素进行计算，并由并行计算单元加速。在实际的前向传播和反向传播过程中，这些层的时间开销通常较小（低于 10%）。恰当的激活函数可在不同阶段提高网络的平滑性，使其在训练期间更快地转向损失值较小的区域。

4.1.4 高效注意力

注意力机制成功推动了 Transformer 模型的发展，并在多个任务上取得最先进水平的结果。目前通用的注意力模块是多头注意力（MHA），它采用多个自注意力分支来关注不同特征。我们将 H 记作输入向量，整个推理过程分为两部分，

$$Q_i, K_i, V_i = HW_i^Q, HW_i^K, HW_i^V, \tag{7}$$

分别是投影查询（Q）、键（K）和值（V）。然后，自注意力计算过程如下：

$$H_{out} = softmax\left(\frac{Q_i K_i^\top}{\sqrt{d}}\right) \cdot V_i, \tag{8}$$

其中，d 是维度。在每个头中，每个词元都会与其他词元计算内积，以生成关系值，然后将其归一化为 $[0, 1]$ 范围内的比例，从而公平地计算注意力强度。因此，如果词元数量为 n，则需要 $\mathcal{O}(n^2)$的计算复杂度，这也使得自注意力模块通常非常耗时。

在Transformer架构 引入之前，注意力机制是通过复杂的基于RNN的编码器-解码器模块实现的，其复杂度为 $\mathcal{O}(n)$，并且采用顺序操作。Transformer模型 [256] 研究了替代性的缩放点积模块，以摆脱递归计算，同时实现类似的效果。在小型任务中，计算问题并不那么重要 [68]。当数据的维度显著增加时，自注意力模块将发挥关键作用，并成为训练Transformer模型的主要成本。因此，围绕如何高效计算注意力模块，开展了一系列研究。Kitaev等人 [119] 采用局部敏感哈希与可逆残差层，替代传统的点积注意力，使Reformer模型中的复杂度从$\mathcal{O}(n^2)$降低至 $\mathcal{O}(n\log\ n)$。Choromanski等人 [32] 通过正交正随机特征方法，在具有可证明性质的Performer中估计全秩注意力，在不依赖任何先验知识的情况下保持线性空间与时间复杂度。Wang 等人 [264]提出Linformer，通过低秩矩阵近似自注意力，在时间和空间上的复杂度均为 $\mathcal{O}(n)$。Xiong等人 [281]在Nystr¨omformer中，通过Nystr¨om方法研究线性化自注意力近似。Peng等人[190] 基于随机特征方法提出线性时间与空间注意力RFA，通过可选的门控提供直接的解决方案。Pan 等人 [182] 专注于视觉任务，并提出新颖的IA-RED^2 框架，在不同阶段移除不相关的词元，从而有效缩减推理计算。Ma 等人 [159] 使用额外的固定序列作为输入，使 $Luna$ 框架能够在线性时间内执行注意力操作，同时存储充足的上下文信息。Wu等人 [274] 为每个词元表示建模全局上下文，而非对所有词元之间的成对交互进行建模，并在理论线性复杂度下实现了高效的Fastformer。Ren等人 [208] 通过将自注意力模块视为每个位置嵌入上的条件期望，以替换方式在每个注意力头中实现完整能力，同时保持较低的计算和内存复杂度。Chen 等人 [22]将经典的鲁棒PCA融入注意力矩阵乘法，以统一稀疏计算和低秩计算。Rabe和Staats[195] 提供了一种实用且稳定的实现，将自注意力模块的内存成本加速降低至 $\mathcal{O}(\sqrt{n})$。Liu等人 [147] 通过核化哈希定制高维注意力，在EcoFormer范式下，将传统的查询向量和键向量映射为汉明空间中的低维二进制信息，该范式由真实相似关系学习得到。Li等人 [141]制定阈值，以选择经过集成到全局目标函数中的软可微正则化器训练出的最相关词元，从而高效减少计算，同时保持相同的测试误差。Pan 等人 [186]引入HiLo注意力，根据频率区分图像信息，将大型词元簇分割为不同的小组，并执行分组自注意力编码器。Zhang等人 [300] 重新组合混合多头注意力机制，以整合每个词元在不同注意力头之间的共享信息，这类似于在注意力头维度上的分组方法。Hua等人 [97] 开发了门控注意力单元，以允许使用较弱的单头注意力

具有质量损失的模块，随后为这一新颖变体提供线性近似。Qin等人 [193]总结了以往线性设计注意力模块中的主要瓶颈，即无界梯度和注意力稀释，这些问题导致了性能差距。为进一步解决这些问题，应更加关注注意力矩阵的扩展。Dao 等人 [40] 考虑了通过硬件协同设计的方法，并提出了 *FlashAttention* 实现，以在实际运行时间方面加速自注意力计算，该实现已集成到许多机器学习框架中，例如 TensorFlow 和 PyTorch。Transformer 模型中的注意力多尺度可视化展示了不同的注意力头，并分析了多头注意力，发现除了一些专用注意力头外，其他注意力头可以被剪枝。未来研究中，如何高效且等效地实现注意力机制仍然是探索的主要方向，尤其是面对那些具有庞大模型和海量数据样本的任务时。作为高效Transformer 领域的综合性综述，Tay 等人 [246]总结并分类了线性注意力方法。

4.1.5 高效归一化

归一化是减轻模型中协方差偏移的重要技术。当初始状态不佳时，训练初期会受到不平衡特征的显著影响，进而导致梯度不稳定。因此，引入归一化来规范特征的均值和方差。卷积神经网络（CNN）和 Transformer 均受益于统一的动态特性。我们可以将其记为：

$$\Phi(H) = \left(\frac{H - \mu_H}{\sigma_H} \right) \odot W + b, \tag{9}$$

其中， W 和 b 分别是权重和偏置， μ_H 和 σ_H 是输入 H的统计均值和方差。它们的统计计算由归一化方法决定。

传统归一化采用批归一化（批量归一化）来避免反向传播过程中梯度消失，并在多个计算机视觉任务中取得了显著进展。它能够平衡一个批次中样本的均值和方差，并在训练过程中保持稳定性。由于自然语言处理任务中的批大小较小，因此采用层归一化（层归一化）。目前最先进的 Transformer 模型仍沿用这一技术。此外，还有实例归一化（实例归一化）和组归一化（组归一化）。它们的区别在于对数据样本进行归一化时所选用的维度。 [92] 提出了*GhostNorm* （幽灵批归一化），这是一种批归一化技术，可将批次划分为更小的批次，并行地分别进行归一化。该技术减少了每次通信所需的页锁定内存，并充分利用缓冲区缓存。 [50] 提出*SeqNorm* （序列归一化），对输入特征图的两个维度上的输入进行归一化。它巧妙地结合了*GhostNorm* 和*BatchNorm* ，在分类任务中的表现优于其他方法。Zhang 和 Sennrich[293] 将层归一化扩展为均方根归一化（均方根层归一化），该方法假设 LN 中的重新居中不变性并非必要，并通过输入平方均值的平方根对求和后的输入进行正则化。与原始 LN 相比，该方法计算更简单、效率更高。在分布式系统中使用时，均值和方差的计算实际上属于归约操作，而这受到设备内存与寄存器之间通信瓶颈的限制。为节省有限的计算资源，研究者提出融合层归一化，将多个归约操作合并到一个函数中，从而在减少内存分配的同时充分提升 GPU 利用率。Singh 和 Singh [224] 研究了一种新颖的特征级归一化方法，以实现快速运行并获得最佳性能。该方法通过集体响应将输入特征作为独立单元进行归一化，从而选出可优化的更佳组合。Xia 和 Ding [276]提出协同归一化，以消除域差异并加速无监督域自适应任务的训练过程。该方法衡量不同域中特征的差异，并学习特定统计量，以挖掘相似的知识和表征。它能够提升泛化准确率，并且易于嵌入深度模型，无需额外开销。

4.2 压缩与训练效率

压缩也是目前最主流的加速方法之一。从整个训练过程的耗时来看，大部分时间都花在推理（约 30%）和反向传播（约 60%）上。提高计算效率能够显著加速整个训练过程。量化是一种有效的方法。减少原始数据所占的位数，可以缩短计算设备上的推理时间。另一种方法是稀疏训练。大量研究表明，大型神经网络在训练过程中，其参数和梯度存在冗余，这意味着可以适当降低部分参数的更新频率。在这一部分，我们重点关注那些在整个训练过程中保持参数总数不变的方法。

4.2.1 量化效率

量化加速训练的基本原理是：如果将参数和输入转换为低精度存储，深度网络所需的内存和计算量就会减少。然而，量化与准确率之间始终存在权衡。一系列研究探讨了量化与准确率之间的联系，并提出了在实践中尽可能保留训练准确率的方法。

目前，常用算法默认使用标准 FP32 浮点类型来进行推理和参数反向传播。这种数据类型能够满足当前深度训练的高精度要求，尤其适用于大型数据集和庞大模型。相应地，半精度数据类型 FP16 保留 5 位指数和 10 位尾数，现已成为当前训练中 FP32 的替代方案。在一些评估中，其耗时不到常规训练的一半，计算结果甚至可接近基于单精度的计算结果。然而，精度受限往往会导致梯度溢出，尤其是在训练初期，从而引发带有 NaN 的核心转储。此外，低精度计算对训练过程后期的影响更大。为平衡效率和准确率，自动混合精度 （automatic mixed precision，AMP）应运而生，并在深度训练中得到广泛应用。它依据经验，对部分不重要的参数使用 FP16，对其他参数使用 FP32，以实现最佳性能；目前，这是最高效的加速技术之一。在此基础上，Wang 等人 [262] 提出了 *HAQ* 框架，利用强化学习在硬件加速器上自动确定量化策略。该框架可实现全自动化，并为网络中的不同模块选择量化策略。Huang 等人 [98]为深度模型的芯片架构师提供了指南，帮助他们选择对硬件友好的设计方案，例如在实际场景中恰当选择量化方法和最优映射策略。Ryu 等人 [213] 提出了一种高效的精度可扩展加速结构，采用新颖的逐位求和和逐通道对齐方案。该结构显著降低了扩展带来的面积开销，并能从片上 SRAM 缓冲区正确读取输入和参数。Dettmers 等人 [47] 尝试采用结合 8 位优化器的分块量化，取得了可比的性能。分块量化将输入拆分为若干块，各块独立并行量化，从而实现更快的优化和更高精度的量化。这是一种简化计算的直接方法。如何实现能够在硬件设备计算中得到有效支持的量化方法，仍是未来最重要的研究方向之一。

4.2.2 稀疏效率

稀疏计算需要硬件支持，否则无法在实际场景中实现加速。Brock 等人 [17] 提出逐步冻结前面的层，并在训练过程中将它们排除在反向传播之外。通过在 CIFAR 上进行实验，他们以经验结果表明，其算法 *FreezeOut* 可使 DenseNet 提速 20%（准确率损失 3%），或使 ResNet 提速 20%（准确率无损失），但对 VGG 网络没有提升。Raghu 等人[201]也提出了类似的思路。Xiao 等人 [277] 提出了一种方法，通过计算归一化梯度差异来确定层的冻结比例，从而智能地冻结层。该实现表明，这种方法在训练中期效率很高。Liu 和 Mozafari [148] 提出了一种新颖的梯度压缩方法 *JOINTSPAR* ，用于大型

通过逐层自适应学习率进行批量优化。它通过最小化梯度范数与参数选择的组合，从这些阈值中丢弃原始梯度的坐标。Ding 等人[52] 提出了一种用于卷积网络的有效方法，以在优化器上实现重新参数化。该方法结合并行残差结构中不同分支之间的卷积参数，以实现等效训练。重新参数化的结构可以在保留原始网络特征的同时，将更宽的网络压缩为更窄的网络。Kong等人 [122] 指出了百万规模训练数据集中的冗余，并提出了端到端实用框架 *Tri-LevelE-ViT*。该框架同时利用数据样本的重要性以及图像中图像块上的稀疏连接来加速视觉变换器训练。Huo 等人 [101] 利用延迟梯度解耦反向传播，并通过将网络拆分为多个模块来消除反向锁定。这种解耦并行机制节省了梯度反向传播链上的等待时间。Goli和 Aamodt [70]观察到，在训练过程中，超过 90% 的梯度可以重复使用。利用这一现象，他们提出了 *ReSprop* 方法，通过减少反向传播计算来指定梯度向量。为了进一步匹配 GPU设备的细粒度计算，*ReSprop* 引入了一种通用稀疏卷积加速器。Lym 等人 [156]提出了 *PruneTrain* ，以逐步降低训练成本。该方法采用结构化组套索正则化方法，将参数约束为较小的值。它引入重配置技术来实现局部剪枝，并减少浮点运算次数和内存方面的计算。Nguyen 等人 [175]关注多头注意力中的冗余。他们提出高斯键混合（*MGK*）来替代原始多头架构。Dun 等人[55] 考虑了分块训练方法，并提出*ResIST* 来加速ResNet。在每一轮中，它训练一个随机生成的子网络，并通信一小部分参数，最后在末阶段合并结果。Fan 等人 [59] 使用稀疏蝴蝶乘法加速器来节省由硬件实现支持的计算复杂度。这类矩阵可以轻松应用于注意力和前馈网络模块，并可节省大量训练实际耗时。由于蝴蝶矩阵的特性，性能下降极其轻微。Han 等人 [79] 提出了*Turbo* 训练，用于动作分类任务和视频-语言表征学习中的 Transformer模型。他们允许在训练中丢弃词元，以指定输入向量，并能够将长时间安排的视频帧拆分为片段组。Li 等人 [136] 开放了*Magicube* 库，用于在张量核心上进行稀疏矩阵计算。每个值都可以在A100设备上表示为低精度整数。它支持 SpMM（稀疏矩阵乘法）和稀疏-稠密-稠密矩阵乘法，可在稀疏Transformer推理中加速稀疏内核，同时不降低精度。Bolya 等人 [14] 受到稀疏输入思想的启发，并引入 *ToMe* （令牌合并）技术来实现输入的稀疏化。不同于直接移除词元，*ToMe* 允许选择若干词元，然后将其余词元合并为一个词元，作为 Transformer模型的输入向量。

总之，用于加速的稀疏训练主要侧重于以下方面：**稀疏梯度**。 梯度决定了每次迭代中参数的偏移量。对于在一次更新中梯度值较小的参数，其偏移量也较小。与其他参数相比，暂停对这些参数的更新不会损害训练性能。因此，通过更新稀疏梯度来减少反向传播的计算，可以有效加速训练。

稀疏参数。 参数冗余是深度模型中非常常见的问题。许多研究指出，对于不同的数据样本，选择性更新往往具有更好的泛化能力。因此，在训练中冻结部分参数的剪枝和重配置技术，可以大幅节省前向和反向过程所需的时间和内存成本。

稀疏输入。输入样本中存在大量冗余。通常，在一个数据小批量中，不同类别的样本接受的训练程度各不相同。通过一些独特的采样方法（例如重要性采样），可以在批次维度上实现降维。这使得支持更大的批大小成为可能，从而加速整个训练过程。

稀疏特征。特征是中间层对输入样本进行函数映射的结果，会消耗大量计算资源和内存。特征中也存在大量冗余。通过聚合或额外的注意力机制来聚合中间层的特征，可以有效地

加速深度网络的训练。

4.3 初始化效率

关于模型初始化的研究表明，随意采用初始化策略可能导致收敛缓慢，甚至在训练开始时完全停滞 [169], , 这种情况在大规模模型中尤为明显。因此，恰当的初始化有助于实现事半功倍的高效训练。此前的多项研究从不同角度揭示了初始化与训练之间的关系，并提出了一系列高效方法，以更好地初始化模型。预训练技术的最新进展也表明，更好的初始状态有助于理解与特定任务无关的通用知识，并提升下游任务的表现。本部分主要聚焦于高效初始化方法和预训练技术。

从头开始训练的初始化。 模型初始化对于高效地从头开始训练模型起着重要作用。Sutskever 等人 [237] 发现，模型初始化和动量更新对于训练都很重要。他们指出，如果模型从初始化不佳的状态开始，采用动量更新可能会导致训练过程失败。先前研究表明，随机初始化导致的失败可能源于深层中特征的方差不断缩小。为克服这一困难，Saxe 等人 [214] 提出，两个线性层之间参数矩阵 W 初始条件的适当缩放，相当于在正交矩阵下从均值为零、方差为$1/d$的高斯分布中选择初始参数，其中 d 是矩阵W的维度。即使在深度模型中，这种初始化也能保持输入的范数，从而有效避免训练过程中梯度不稳定带来的负面影响。Sussillo 和 Abbott [236] 表明，对每一层的随机矩阵进行适当缩放，会使参数范数的对数产生意外偏移，其方差会随着模型深度增加。他们提出增大模型宽度，并采用额外的校正标量来调整模型初始化。Kumar [124]研究了激活层影响的实证证据。双曲正切和 sigmoid 等在零点可微的函数，在随机初始化时会比在零点不可微的函数表现出更强的敏感性。Zhang等人 [295] 探索了带跳跃连接的残差网络的专用初始化方法。Huang 等人 [99] 将权重初始化扩展到 Transformer架构，并提供理论校正保证。该方法使模型无需预热和层归一化即可进行训练。可学习的初始化在模型初始化方面也展现出巨大潜力。Dauphin 和 Schoenholz [42]提出*MetaInit* ，以搜索近似线性损失景观下的良好状态。它通过最小化二阶效应的曲率信息，采用若干次梯度下降来调整参数范数。Zhu 等人 [306] 通过若干经典数据样本，找到能使损失函数最小化的更优初始状态。Zhao 等人 [301] 提出零初始化 ，仅用零或一来初始化模型。它采用 Hadamard 变换消除初始化中的全部随机性，并且可以轻松扩展到稀疏训练。高效初始化在大规模模型训练中表现出色，潜力巨大。它也启发我们思考如何寻找一个训练好的模型，以便在复杂的下游任务上进一步微调参数，这正是预训练技术。

预训练-微调初始化。 预训练-微调模式是一种高效范式，可在下游任务中取得出色的性能。它源自迁移学习，允许在大规模数据集上训练大规模模型，以学习良好的特征提取器。随后，冻结骨干网络并微调部分参数，从而取得良好性能。预训练阶段可视为为下游任务寻找更优初始化状态的过程。近期进展为预训练带来了耀眼的曙光。Radford 等人 [196]提出了一种用于语言任务的半监督方法，采用通过无监督预训练得到的初始化状态。结果表明，这种生成式模型优于监督训练。Radford 等人 [197] 大幅度

在预训练阶段通过联合多任务增加模型参数和数据样本。他们指出，在数据样本充足的情况下，大规模模型在下游任务的高效训练方面潜力巨大。此外，Brown等人 [18] 使用一个超大规模基础模型进行预训练，并在其他任务上进行测试。该模型在一些少样本甚至零样本场景中取得了最先进水平的性能。He等人 [83] 也采用类似的训练策略对计算机视觉模型进行预训练，并在计算机视觉领域的下游任务中取得了巨大成功。预训练为模型的高效训练提供了出色的初始化。当前趋势倾向于将大规模模型与极其丰富的数据样本相结合。因此，如何实现预训练阶段的高效训练，是未来一个很有前景的研究方向。

4.4 以模型为中心的课程学习

以模型为中心的课程学习主要关注规避大规模模型中的训练困难。它探究大规模模型在训练早期较小的容量，并建议相应地减少参与计算的参数数量。随着训练的进行，模型的容量逐渐增加，所有参数得以恢复，从而完成整个训练过程。这在高效训练方面展现出巨大潜力，尤其是在大规模模型上。

Bengio等人 [8]提出了一种贪心方法，以实现渐进式的逐层学习。这启发了在模型层面进行渐进式训练的潜力。Karras等人 [111]研究了生成对抗网络训练中的课程学习。该方法同步地逐步增加生成器和判别器的层数。新层被平滑地加入训练，以避免对已充分训练层产生突然而剧烈的冲击。Smith 等人[227] 提出了 *DropIn* ，从浅层子网络开始训练深度模型，然后逐渐添加新层，以保持其效率。他们证明，以模型为中心的渐进式训练可作为参数上的正则-化项。Wang 等人 [261]专注于半监督学习，并设计了深度渐进学习。他们将训练形式化为类似 EM的过程，使大规模模型能够在自动增长层之间交替迭代。他们指出，渐进式训练可能有助于大规模模型保持更高的稳定性和泛化能力。Gong 等人 [72] 研究了无监督预训练技术，并提出了 堆叠算法，将已充分训练的浅层模型中的基础知识迁移到扩展后的模型中。在迁移过程中，采用逐步增加的堆叠来加速BERT 的训练。Li 等人在神经机器翻译（NMT）任务中提出了类似的思想。Zhang 和 He [297]通过层丢弃学习一种渐进式调度策略，为训练好的模型增加额外的稳定性。在训练过程中，为所有ST块设置一个全局丢弃率列表，并针对不同层进行更新。Gu 等人 [75]观察到，在模型的多个维度上采用联合增长是有益的。他们提出探索每个维度中的交替增长，而不是仅沿深度方向增加层数。Li 等人 [130] 将渐进式学习的优化问题放宽为子网络优化过程，该过程可以通过自动化的一次性估计求解。它还通过复用子网络的参数，最大限度地减少了搜索开销。以模型为中心的课程学习在大规模模型训练中取得了巨大成功。未来，设计与数据样本相匹配的复杂渐进式调度策略可能是一项有前景的研究。

4.5 讨论

本节重点讨论如何在特定的大规模模型中实现高效训练加速。我们从基本训练过程的角度，考虑加速训练的一般方法，包括采用替代性的高效架构以降低计算复杂度，压缩参与训练过程的张量以减少内存占用和计算成本，以及采用高效的模型初始化方法以实现快速收敛。

就模型架构而言，我们的重点是评述基础模块的高效替代方案，而不是探索设计高效模型，包括基本组件线性层、卷积、注意力、激活函数和归一化。简单结构通常往往对

参数。采用低参数量替代方案时，线性层和卷积层等基础模块的性能容易受到影响。这限制了实际加速比的上限。与减少参数相比，采用高效的替代计算在泛化能力和稳定性方面表现更好。然而，设计等效替代方案是一项非常困难的任务，其重点通常在于近似这些计算。另一种主流方案是压缩训练。我们重点关注已在实践中证明有效、如今广泛用于大模型训练或作为基础设计的量化和稀疏方法。不同模块对量化影响的敏感程度各不相同。因此，在选择低精度存储和计算时，可以优先考虑误差不明显的模块，例如激活层。稀疏性是加速训练的一种有效实现方式，但要提高实际效率，需要严格的硬件支持。GPU 的并行单元无法直接执行稀疏计算，这是训练过程中存在的主要缺陷。在模型初始化方面，良好的初始状态能够确保大规模模型训练的稳定性和效率。近期进展也表明，预训练在提升性能方面潜力巨大。为缓解预训练带来的巨大消耗，模型级课程学习发挥着关键作用。它使训练过程能够从浅层模型或部分参数开始，并逐步恢复完整模型。这项技术有望推动其他领域实现更高效的训练。

展望未来，我们推测，加速训练的重要应用场景包括：

- 复杂模块的高效替代方案。由若干独立层构成的加速替代方案，其累积误差容易造成较大的偏差。将一些复杂模块视为一个整体来设计替代方案，能够减小输出误差。

- 高效算子融合。将相邻步骤中的算子组合起来，能够大幅提高计算效率和精度，例如*FlashAttention* [40]。这些设计充分利用了硬件特性，提高了训练吞吐量，从而加快训练速度。

- 高效模型初始化。类似预训练的可训练高效初始化，是未来一个颇具前景的研究方向。探索模型初始化对不同模块的相互作用，也是一个值得研究的方向，有助于为理解超大规模模型训练的本质提供有效指导。

- 高效渐进式调度。设计新颖的渐进式调度方案，引导大模型在各个维度上进行训练，可能是未来一个颇具前景的研究方向。一个关键问题是探索模型规模与训练所需数据样本之间的关系，这或许能为理解课程学习提供扎实的见解。

5 以优化为中心的高效训练

优化方法的加速方案一直是机器学习领域的重要研究方向。在实现最优条件的同时降低复杂度，一直是学术界追求的目标。近年来，高效且强大的优化方法在深度神经网络训练方面取得了重要突破。

作为机器学习中广泛使用的基础优化器，*SGD*类优化器成功帮助深度模型实现了各种实际应用。然而，随着问题变得更加复杂，模型总是更容易陷入局部最小值，并且无法稳定地实现泛化。为解决这些困难，*Adam* 及其变体被提出，以在更新过程中引入自适应性。这种做法在大规模网络训练中取得了良好效果，例如用于训练 *BERT*、Transformer 和视觉变换器模型。除了所设计优化器本身的性能之外，结合加速训练技术也十分重要。我们从优化的角度出发，将当前关于加速训练的思考总结为以下几个方面：

• **学习率**。学习率是非凸优化中的一个重要超参数，对于当前深度网络的训练也至关重要。Adam及其变体等自适应方法已在深度模型上取得了出色的进展。一些基于高阶梯度调整学习率的策略也能有效实现加速训练。学习率衰减的实现也会影响训练过程中的性能。

• **大批量大小**。 采用大批量大小将有效提高训练效率。它直接减少了完成一轮训练所需的迭代次数。同时，由于额外的内存占用和通信瓶颈，处理一个大批次所花费的成本少于处理总样本数量相同的几个小批次。

• **高效目标函数**。原始 ERM在最小化问题中发挥着关键作用，使许多任务能够得到实际应用。随着对大型网络的深入研究，许多工作更加关注优化与泛化之间的差距，并提出高效目标函数以降低测试误差。他们从不同角度诠释泛化的重要性，并在训练过程中对其进行联合优化，从而显著提高测试准确率。

• **平均权重**。 权重平均是一种增强模型泛化能力的实用技术。它使用一组固定或可学习的系数，对历史状态进行加权平均，从而显著加快训练过程。

5.1 学习率

学习率是控制训练中更新的最重要超参数之一。应结合考虑优化器和批大小进行精细选择。从自适应策略的实现角度来看，我们将其分为三类：缩放系数、逐元素自适应性和基于矩阵的预条件器。它们分别采用一个单一值来衰减学习率、一个向量来平衡梯度，以及一个矩阵来搜索最优点。

5.1.1 学习率调度器

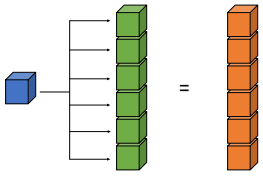
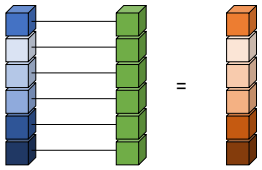
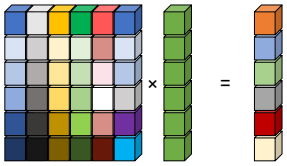
通用学习率调度器采用单一衰减系数，逐步降低学习率，以学习训练数据集中的细粒度知识。通常，我们以轮次为单位调整学习率的变化，可将其表示如下：

$$\gamma = \gamma_0 \cdot \rho(t), \tag{10}$$

其中， γ_0 是初始学习率， $\rho(t)$是一个标量函数。

常见方案是采用带有恒定上限的分阶段衰减，其中 $\rho(t) = \rho_0^{\lfloor t/t_s \rfloor}$ 。 t_s 是预定义的衰减阶段，在许多任务中表现良好。然而，它在大模型上未能取得更优性能。简单的微调过程使整个训练过程缺乏捕捉细节的能力，从而导致模型停留在一个平庸的局部最小值。Loshchilov 和Hutter [153] 证明了简单指数式阶段衰减学习率的负面影响。训练中期学习率的快速下降，最终导致模型无法逃离局部最小值。他们提出了余弦衰减，其函数为 $\rho(t) = 0.5\,(\gamma_{\max} - \gamma_{\min})(1 + \cos(t\pi/t_s))$，其中 t_s 是用于重启学习率的预定义计划。该方案在当前Transformer模型的训练中发挥着关键作用。为进一步提升其性能，提出了*CLR* （周期性学习率） [225] 策略，在最大上限和最小下限之间交替更新。通过周期性调整，它大幅提升了泛化性能。与余弦 学习率不同，*CLR* 采用固定窗口的周期模式。通过减少所需的训练轮数，它显著加快了大数据集上的训练速度。Smith和Topin [226]通过扩展目标损失的二次近似，研究了超收敛现象。海森矩阵近似有助于构建其学习率。Pan 等人 [183, 184]提出了一种改进的*ESD*（弹性步长衰减）学习率调度器，以改进 BERT 的训练。它采用了一种宽松的

表 3：学习率调整方法。

方法	表述方式	引言
	$\Delta w = \gamma\rho\ (t)\ \ g$	乘以一个标量
	$\Delta w = \gamma v(w) \odot g$	乘以一个向量
	$\Delta w = \gamma P(w)\ \cdot g$	乘以一个矩阵

学习率的选择并推动集中式训练。张力步长在训练过程中提供更具泛化能力的模型，并创建一个稳定学习的连续过程，以避免陷入局部最小值。Koc.yi˘git 等人 [121] 采用缩放后的单周期学习率，使训练过程与经过调整的动量系数 $\beta_t = \beta_t + 0.5(\beta_h - \beta_t)$ (cos($t\pi/L$) + 1)相匹配，其中 β_h 是 $beta$ 的最大上限， L 是用于缩放它的预定义常数。这种组合在基于动量的优化器中效果更好，并能有效提升测试性能。Ni 等人. [177] 提出了一种深度模型组装技术，用于加速基础模型的训练。该技术分别训练全局模型的各个部分，合并后的全局模型则可通过本地训练权重快速微调。他们表明，Deit-L 和 Deit-H 的训练速度都可以提升。

5.1.2 自适应性

自适应学习率侧重于对梯度所有维度进行逐元素自适应。它认为，梯度中的每个单独元素与目标函数的重要性各不相同，因此应被视为一个局部标量。这有助于更准确、精确地控制学习率，并适用于若干大型任务。人们从不同角度设计了多种基于自适应性的优化器，以提高其效率。我们将简要介绍相关工作以及近期进展中的一些新颖想法。其一般计算形式可以写为：

$$\gamma = \gamma_0 \odot v(t), \tag{11}$$

其中， $v(t)$ 是一个与梯度维度相同的向量。

传统的自适应方法包括*AdaGrad* [54], *Adadelta* [291], *RMSprop* [249], 和*Adam* [118],等。它们利用一种 $v(t) = 1/\sqrt{\beta v(t-1) + (1-\beta)g_i^2}$型函数来调整学习率。 $v(t)$函数认为，如果某个参数在过去曾被大幅更新，那么在当前阶段对它的更新幅度就应小于其他参数，这有助于平衡局部梯度，避免其发生剧烈变化

。这有助于稳定训练过程，并根据历史梯度搜索最优学习率。*Adam* 结合了动量项和自适应学习率，从而提高效率，因此被广泛应用于许多任务。

近年来，Loshchilov 和 Hutter [154] 提出了*AdamW*，该优化器将梯度和权重衰减正则化都视为动量项。它在 Transformer 训练中取得了巨大成功。You 等人 [288] 采用逐层正则化*LARS*，以便为每一层获得更准确的标量。由于反向传播中的链式法则，梯度具有明显的分层特征，这导致某些层的梯度范数与权重范数之间的不平衡，从而产生不稳定性。因此，它将权重范数共同纳入标量，以进一步微调学习率。类似地，You 等人 [287] 将该技术应用于以*Adam* 为基础的优化器，并取得了更高的性能。为了进一步规范化 l_2 正则化项，提出了*AdamW*，以减小每层参数之间差异的影响。它在 Transformer 和 ViT 模型的训练中取得了巨大成功。Xie 等人 [280] 提出了*Adan*，即用于加速原始*Adam*优化器的 Nesterov 动量，并实现了更快的训练。它将原始梯度转换为惯性型梯度，以近似下一状态的方向。前瞻性方法进一步对梯度进行正则化，使更新能够在早期阶段有效实现加速训练。最近，Chen 等人 [28]提出了*Lion* （进化符号动量）优化器，大幅提升了训练速度。*Lion*源自于在若干预定义数学函数池中进行策略搜索。它首先搜索 21 个函数的复杂版本。经过经验处理和近似后，*Lion* 最终利用带动量的符号梯度构成更新。这种实现可能会成为未来一个重要的研究方向。

自适应优化器在大模型的训练中取得了显著进展。此前的若干工作基于通过Hadamard乘积采用逐元素标量。它能够有效稳定大批量大小下的训练过程，并且通常比原始的基于梯度的方法收敛更快。未来，结合考虑阶段级和层级的冻结技术，将为优化器的设计提供新的见解。

5.1.3 预条件

预条件器通过矩阵微调学习率。不同于占主导地位的一阶梯度方法，预条件器涉及高阶导数，主要包括海森矩阵和Fisher 矩阵（FIM）。经典的牛顿法应用海森矩阵来实现最优收敛，从而有效加速训练。目前，尽管此类方法在大模型训 练中并非主流，但其中一些技术和见解值得借鉴。

为了最小化目标函数 F，我们考虑对学习率进行如下调整:

$$w = w - \gamma P(w)\nabla F(w), \tag{12}$$

其中 $P(w)$ 是 w 处的预条件矩阵。

经典牛顿法采用海森矩阵的逆作为 $P(w) = H^{-1}$。然而，由于计算复杂度极高，很难直接从曲率中获益。因此，研究者提出了拟海森矩阵方法 [10, 46]。这些方法利用牛顿函数构造海森矩阵的更新，以迭代拟海森矩阵，从而降低大部分计算复杂度。为了实现高效的预条件器，研究者更加关注其替代方案 FIM。可以证明，FIM 是对数似然目标函数上的海森矩阵的期望，这表明曲率信息可以指导更新。此外，它只需要与梯度相同的计算量，如 $P(w) = \nabla l \cdot \nabla l^\top$ 其中 l 是输出分布的对数似然。这有助于在多个实际场景中加速训练。Gupta 等人 [77] 提出了*Shampoo* 方法，将FIM张量近似为三个矩阵的乘积。该方法使用两个 FIM 信息的克罗内克积，这些信息可以按块结构进行计算。与自适应性相比，它在描述特征方面具有更大的张力，直到

状态 w。类似的工作，如*K-FAC* [163]，提出分别使用输入和梯度的期望的乘积，而不是它们乘积的期望。另一个重要的研究方向聚焦于无矩阵近似。Frantar 等人 [65] 根据定义研究了这种近似方法，该定义记录梯度的相邻差值并计算其乘积。通过预计算过程和逆对齐，预条件矩阵被估计为若干秩一矩阵之和，其计算开销低于非无矩阵方法。Pauloski 等人 [189] 提出*KAISA* 框架，以实现通信和计算效率方面的内存占用优化及硬件支持。它重构了 FIM 的计算过程，并减少了曲率信息的更新频率，从而进一步稳定训练。

研究可被视为学习率矩阵投影、以实现最优收敛的预条件器具有重要意义。由于计算的复杂性，目前这类方法在大模型上尚不能高效运行。设计具有一阶计算复杂度的预条件矩阵，以加速大模型的训练，是未来的重要目标。此外，平衡优化与泛化的目标函数可能带来更多新颖的见解。

5.2 大批量大小

适当采用大批量训练可以减少时间消耗、提高存储利用率、生成平滑的梯度，并维持稳定的训练过程。然而，使用过大的批大小进行训练会适得其反，可能严重破坏训练的稳定性，甚至导致发散，尤其是在大规模模型上。Keskar 等人 [115] 表明，小批量训练倾向于收敛到远离初始状态的平坦极小值，而大批量训练倾向于收敛到初始状态附近的尖锐极小值，并且在测试数据集上的泛化能力更差。因此，研究如何在确保性能的同时使用大批量大小进行训练，是一项重要课题。

为了讨论方程中的批量大小项，我们将主要关注如何实现大批次训练，因为这是大多数最先进模型采用的一种策略。首先简要介绍随机梯度下降（SGD） [210], ，这是一种常用于深度学习的重要方法。小批量梯度下降的迭代步骤如下所示:

$$w_{t+1} = w_t - \frac{\eta_t}{B_t} \sum_{i=1}^{B_t} \nabla f_i(w_t; (X_i, Y_i)), \tag{13}$$

其中， $B_t \subset B$ 表示从整个训练数据集 B中，在第 t步采样得到的一个小批量（通常采用均匀采样）。克服 SGD 的两个不足——采样效率低下和并行化困难——可以实现加速。

为实现大批次训练并避免较差的泛化性能，一些现有解决方案专注于联合微调学习率和批大小，包括随着批量大小扩展对学习率进行线性缩放、在初始阶段采用学习率预热，以及逐步增加训练批大小。此外，对数据进行额外的数据增强以及采用所提出的高效优化器，也能够支持大批量训练，从而提升泛化性能并加速训练过程。Smith 等人 [228]表明，增大批量大小自然会产生与快速衰减学习率相似的结果，这可以减少参数更新中的意外误差，促进其并行化，并实现更高的训练效率。Hoffer等人 [92] 表明，在训练中使用特定的归一化来控制协方差偏移，有助于稳定训练，并缩小泛化与优化之间的差距。McCandlish 等人 [164] 提出了一种称为梯度噪声尺度的经验统计量，用于预测模型的最大批量大小。考虑一个参数化为$w \in \mathbb{R}^d$的模型，令 G 表示真实梯度， H 表示 w处的真实海森矩阵， Σ 表示逐样本协方差矩阵，梯度噪声尺度 $\mathcal{B}_{\text{noise}} = \text{tr}(H\Sigma)\, G^\top HG$；如果假设优化具有良好的条件数（海森矩阵是单位矩阵的倍数），则可将其简化为 $\mathcal{B}_{\text{simple}} = \text{tr}(\Sigma)\, |G|^2$ ，并据此预测临界批量大小 $\mathcal{B}_{\text{crit}}$。梯度噪声尺度被定义为用于平衡权衡的标量。

对于高效的分布式系统，采用了与梯度累积类似的思想，通过在多台机器之间并行划分梯度计算来减少训练时间，同时还需要解决学习率、优化器和通信等问题，从而实现跨设备训练中的高效并行化。Goyal 等人 [73] 使用256 个GPU 训练 ResNet-50，并将小批量大小从256增加到 8192，在仅 1 小时的训练时间内取得相同的精度（原始工作使用 8个 TeslaP100 GPU 训练耗时 29小时）。他们还采用逐步学习率预热，并提出了一种随机打乱方法：每个周期使用单个分布，再按照工作节点数量进行划分。You 等人 [288] 使用层级缩放 [287]将批量大小增加到 32K，并利用 1024个 CPU 进行 ImageNet 训练。他们仅用11分钟就训练 AlexNet完成100 个周期，并取得了可比的性能。Jia 等人 [107] 展示了一个训练系统，该系统包括以下特性：(a) 一种混合精度训练方法，在不损失精度的情况下提高单个GPU 的训练吞吐量；(b)一种针对极大小批量大小的优化方法；以及(c) 高度优化的全归约算法，与基于NCCL的训练相比，可加速反向过程。Li 等人 [133] 研究了当前技术难以克服的稳定性与效率困境。他们提出了一种序列长度预热策略，使用更大的批大小训练 GPT模型，以避免这些负面影响。Xue等人 [285]指出，利用大批量训练会给密集视觉预测的复杂流水线带来严峻挑战，并提出了一种高效的自适应梯度方差调节器，使得模型能够在非常大的批大小（即 $10K$）下实现稳定的泛化性能。

大批量训练是一种广泛使用的加速技术，但需要选择学习率等细粒度超参数，以防止陷入意外的局部最小值 [85]。正如Golmant 等人所提到的 [71], , 在达到特定状态（临界批次）后增加批大小会导致计算效率低下，并且无法加快收敛速度。同时，如果参与训练的各设备之间聚合后的批大小过小，则会在通信过程中产生额外的等待冲突 [229]。因此，随着训练数据和规模不断增加，为了进一步在大规模模型上实现高效训练，大批大小的适配仍然是一个值得进一步探索的热门话题。

5.3 高效目标函数

目标函数始终是优化中最重要的因素之一，而优化是我们希望针对任务加以改进的原则。优化目标直接决定最终目标和训练效率。在本部分中，我们回顾以优化为中心的加速技术，这些技术受益于设计新型目标函数时的相关考量。我们不局限于狭义定义的目标函数，而是总结旨在构建方程（3）中的更新向量 G的方法，以进行全面回顾。

经验风险最小化（ERM）是统计学习理论中的一项原则，它定义了一系列学习算法，并用于提供其性能的理论界限。在求解最小化机器学习任务时，各种优化方法得到了显著发展。然而，在平衡优化效率与泛化误差方面，它并不占优势。由于优化目标与实践中的预期目标存在偏差，ERM 的张力较低，并且在表示泛化能力方面缺乏精度。因此，研究人员提出了一系列修正后的目标函数，将效率与稳定性 [216]共同纳入全局目标。除 SGD、Adam 等广泛使用的优化器外，逐层自适应速率缩放（*LARS*）也是一种旨在使用大批量进行训练的通用方法。它允许不同的模块或层采用自适应学习率，而不是固定的学习率。细粒度调整能够有效缓解大批量带来的局部过拟合。Khosla 等人 [116] 在不同模块上采用监督式对比学习的 *LARS* 选择机制，以稳定训练过程；与三元组、最大间隔和神经网络对损失等传统对比损失相比，该方法在现代批量对比方法上取得了更好的性能。 [27]中也可以发现类似的思想。Goyal 等人 [74]在未经整理的 ImageNet 数据集上尝试了一种自监督预训练机制。*LARS* 技术使大型模型超越了最佳的自监督预训练模型。它有助于处理不同分辨率的输入，并避免训练早期容易出现的参数坍塌。Foret 等人 [63] 提出了*SAM* （锐度感知最小化），

该方法同时最小化损失值和损失尖锐度，以提升泛化能力。这表明，原始 ERM 会导致尖锐极小值，并带来较低的泛化性能。因此，*SAM* 考虑在每次迭代中，于当前状态的邻域内优化最差点。结合损失尖锐度进行考虑后，它会搜索优化状态的平坦景观。Andriushchenko 和 Flammarion [3]提出了对 SAM 的理解，并提出了一种同时使用 SGD 梯度和 SAM 梯度进行更新的变体。SAM 优化器有望改进的一点，是减少可能在训练中造成额外开销的双重梯度计算。Mindermann 等人 [167] 引入了*RHO-LOSS* （可约简留出损失选择），以加速网络规模数据上的训练。他们关注留出数据集的优化性能，该数据集与全部样本具有相同的分布。为了在原始训练样本上复用交叉熵损失，可以将其写成训练集与留出集之间损失差异的最终形式。第二部分被近似为不包含训练分布的训练模型，从而能够有效节省计算量。因此，该方法扩大了批大小，并在单次迭代中选择那些较差的样本进行训练，从而加快训练过程。

目标函数的改进会考虑更多洞见和实际问题，以完善传统的经验风险最小化（ERM）。如何表征泛化差距和泛化误差，正逐渐成为一个值得未来研究的课题。结合经验评估方法，在合理的搜索空间内进行大规模搜索以找到最优优化器，展现出显著的效率。

5.4 平均权重

模型平均是一种重要技术，通过整合多个历史状态来生成泛化性能高效的模型。生成的模型通常比训练模型具有更高的测试准确率。平均模型无需训练，仅用于测试，因此不会增加计算和内存使用方面的额外开销。

模型平均源自*Anderson* 加速 [1, 258] ，并在随机非凸优化方面取得了良好进展。Scieur 等人 [217] 提出*RNA* （正则化非线性加速），通过一组经优化的权重聚合历史状态，这些权重通过最小化正则化项求得。该方法为每个状态寻找缩放系数，并将这些状态合并为下一个初始点。*EMA* （指数移动平均）是一种简单的模型合并技术，它使用指数衰减权重，对每个先前训练步骤的参数进行加权平均。在训练中期，它能显著提高测试准确率。Izmailov 等人 [103] 提出*SWA* （随机权重平均），这是一种模型平均技术，根据恒定或周期性学习率获取模型权重，并维护权重的平均值以降低泛化误差。与*EMA*相比，*SWA* 会保留并平均所有历史状态，从而得到一个平坦极小值。他们展示了，在 ImageNet 上使用 ResNet-50、ResNet-152 和 DenseNet-161 时，该方法使 Top-1 准确率比预训练模型提高约 0.6–0.9%。Li 等人 [137] 提出*TWA* （可训练权重平均），以进一步提升泛化性能；该方法基于一组正交基的投影。它在由若干历史状态张成的降维子空间中运行，这些状态通常在训练过程开始时被冻结。该方法大幅降低了*SWA*中的近似误差，易于实现，并且能够灵活地应用于训练过程的不同环节。Kaddour[109] 研究了一种固定大小的截断移动窗口，用于对历史状态求平均，并将其称为*LAWA* （最新权重平均）。研究表明，只有最后几个状态有助于实现良好的泛化。所选状态不必是彼此相邻的点；该方法会在每经过若干轮训练后选取一个状态。当模型方差较大时，这种模型平均在训练早期并不起关键作用。模型趋于稳定后，*LAWA* 的性能提升十分显著，通常还能大幅加快训练。

模型平均有助于在训练过程中趋近于一组泛化能力更强的状态。尽管此类方法并不直接参与优化更新，但其在泛化能力方面的显著改进有助于在参数相当的情况下提前停止训练。它大幅减少了所需的训练轮次。在高效训练领域，这是一个具有实际应用前景的研究方向。

5.5 讨论

本节从以优化为中心的视角回顾通用加速技术。我们主要关注以下实际应用：选择高效的学习率以加速训练、采用大批量训练、设计高效的目标函数，以及利用模型平均技术提升泛化能力。这些技术体现了两种主流场景：加速优化过程和提升泛化性能。我们将对各个组成部分进行全面分析。

近期在学习率选择方面的进展表明，自适应性是以可接受的额外计算成本维持高效率的有效手段，这得益于梯度不同维度上的逐元素标量。需要注意的一个主要问题是，平方动量的标量可能导致学习率急剧衰减，从而使收敛速度大幅变慢。采用预条件器是一种很有前景的方法，可引入曲率校正，从而实现快速收敛。然而，在实际训练中，更新预条件器的成本很高，尤其是在大规模模型上。此外，不准确的曲率方向可能会将优化过程引入局部最小值。大批量训练对超参数的选择表现出很强的依赖性。高效大批量训练通常通过调整其他超参数或设计新的目标函数来实现。这两种途径都着眼于加速优化过程，从而促进更好的收敛。

另一种情形则更加关注提升泛化效率。许多既有研究探讨了优化与泛化之间的差距，并提出寻找具有平坦损失曲面的合适状态。研究者基于有效的平坦度测量策略提出了若干目标函数，以增强大规模模型的泛化能力，并取得了显著成效。尽管一些额外的计算开销不容忽视，这种基于目标函数的改进仍然十分有效。与目标函数设计相比，模型平均是一种直接而有效的提升泛化能力的方法。它无需引入任何额外的计算成本，即可实现有效加速，并在训练中期展现出巨大潜力。随着训练完成，其带来的改进最终会消失。

总而言之，我们认为，从以优化为中心的视角来看，未来的发展包括以下方向：

- 优化的自适应学习率。进一步平衡训练大规模模型时由不稳定梯度造成的偏差，有助于提高效率。值得探索利用曲率信息构建低成本预条件矩阵，以进行细粒度调整，这或许有助于在训练中适应更大的批大小。

- 关于训练模型泛化能力的有效测量。采用了多种测量方法来评估局部稳定性和平坦性，例如海森矩阵的最大特征值、梯度范数、邻域中的最差状态，并在训练加速方面实现了很高的效率。探索低成本的表示有助于通过高泛化性能改进训练过程。同时，利用合理的奖励和策略来搜索更新后公式的最佳组合，展现出提高训练效率的巨大潜力 [28]。

- 平均模型参与训练。具有良好泛化性能的平均模型可以被视为下一次迭代的更优初始状态。设计更好的优化算法，以充分利用这一聚合状态，将是一个值得探索的方向。

6 预算高效训练

近期，有若干研究致力于以更少资源训练深度学习模型，同时尽可能提高准确率。这类问题被定义为预算训练，即在给定预算（对可衡量成本的限制）下进行训练，以实现最高的模型性能 [134, 183]。为系统地考虑硬件支持，使研究更贴近实际情况，我们将预算训练定义为

使用给定设备并在有限时间内进行训练，例如在一台低端深度学习服务器上训练一天 [68,104], , 以获得性能最佳的模型。对预算训练的研究有助于探索如何制定预算训练方案，包括确定模型规模、模型结构、学习率计划及其他影响性能的可调因素的配置，以及结合适合可用预算的高效训练技术。本节主要回顾预算训练领域的最新进展。

6.1 预算训练技术

由于预算带来的影响，研究人员开发了更多针对性的高效预算训练技术。本节从数据、模型和优化三个角度总结预算高效训练的方法。随后，我们介绍如何制定预算训练方案，并总结为预算训练提出或改编的技术。

数据。 从数据角度来看，预算训练关注数据集的预算分配，以及与数据相关的算法。增加训练过程中处理的数据量或扩大模型规模，都可以提升训练性能。Kaplan 等人 [110] 指出，模型性能与以下三个因素分别呈幂律关系：非嵌入模型参数的数量 N、以词元数量计的训练数据集规模 D，以及非嵌入计算量 C。然而，在计算量相同的情况下 C， N 与 D之间存在权衡，例如，规模更大的模型在单位计算量内处理的数据更少。Kaplan 等人 [110] 提出，模型规模与训练过程中处理的数据量之间存在最优的预算分配。他们证明，在固定计算预算下（“计算预算”指计算总量），通过训练非常大的模型并在收敛前提前停止，可以获得最优模型性能。Hoffmann 等人 [93] 得出了不同结论，并提出在给定计算预算的情况下，训练过程中处理的数据词元数量应与模型规模同步扩展，而不应像以往工作那样只扩大模型规模、同时保持训练数据量不变。他们表明，经过充分训练的小模型可以胜过训练不足的大模型，并通过训练语言模型 Chinchilla （70B 个参数， $4\times$ 更多数据）检验了这一假设；在相同计算预算下，该模型的得分优于另一个大语言模型 Gopher （ 280B 个参数）。上述研究总结了在固定预算下确定数据集规模的经验规律。此外，Steiner 等人 [231] 研究了训练数据量、*AugReg* （数据增强和模型正则化）、模型规模与计算预算之间的相互作用。他们表明，采用*AugReg*训练、使用较小数据集但投入更多计算预算的模型，能够达到与使用更多数据训练的模型相同的性能。针对数据有限的场景，Chen 等人 [25] 提出了一种数据高效的生成对抗网络训练方法，该方法使用少量真实图像训练集，从原始生成对抗网络中找出可训练的稀疏子网络（“中奖彩票” [64]）。随后，使用同一有限数据集，并结合数据级和特征级增强技术，进一步训练该子网络。他们表明，该框架可以与真实图像数据增强方法互为补充。此外，Zhao 等人 [302]提出了用于生成对抗网络训练的可微增强（DiffAugment），对真实样本和生成样本均应用可微增强技术。他们表明，这种方法能够提升稳定性和收敛性能，在训练数据有限时尤其有效。需要指出的是，若干数据处理技术（例如数据增强和重要性采样）计算开销较大，因此在给定计算预算下使用这些技术时需要权衡。Arazo 等人 [4]研究了在固定迭代次数的预算训练中，重要性采样与数据增强之间的相互作用；结果表明，适当的数据增强比重要性采样方法效果更好，并能加快训练。

模型。正如我们提到的，给定计算预算，可以推导出模型参数的最佳数量和数据量。Kaplan 等人 [110] 建议在适量数据上训练非常大的模型，并在收敛前提前停止训练。Geiping 和 Goldstein [68],Izsak 等人 [104]也证明，在计算预算较低的情况下，训练大模型而非小模型更为合适。类似的发现还包括。

Li 等人 [142], 提出了一种训练策略，为此提供了支持：该策略训练一个更大的模型（更宽且更深），并提前停止训练，然后通过量化和剪枝对其进行大幅压缩，而不是在相同预算下将一个较小的模型训练至收敛。他们还表明，较大的 Transformer 模型比更小的 Transformer 收敛快得多，并且对压缩更加稳健。此外，训练大模型还有其他优势，例如Wei 等人 [272]表明，扩展规模可以为语言模型带来额外的（涌现）能力。在确定模型的参数数量后，仍然可以针对模型寻找并优化有效的架构调整，以加速预算训练或提升性能。Izsak等人 [104] 研究了如何使用一台低端深度学习服务器，在 24小时内完成类似 BERT的掩码语言模型的预训练。他们提出了一套训练方案，包括训练大模型、使用稀疏词元预测（仅预测被掩码的词元） [149]。在 RTX2080Ti上训练 24 小时的预算与 Izsak 等人设定的预算相近。 [104], Geiping 和Goldstein [68] 证明，移除注意力块中QKV 投影的偏置并禁用所有线性层偏置，可以提升计算效率。他们还发现，在注意力块和前馈网络块之前应用层归一化， [5], 并使用门控线性单元 [221] ，可以提升性能。最近，Pan 等人 [185] 从模型结构的角度观察到，Transformer模型在不同训练阶段表现出不同程度的冗余。他们提出了一种预算训练框架，在固定预算下，通过控制模型结构的激活率（主要包括注意力头的数量、多层感知机的隐藏维度以及训练期间视觉词元的数量），调整每个模型复杂度等级所占用的时长。

优化。Izsak 等人 [104] 指出，在预算训练中，为了获得更高性能，有几个重要的超参数需要优化，包括批量大小、峰值学习率、学习率的预热比例以及训练天数。预算训练中的超参数配置与大规模框架中的配置不同，需要进行调整以适应固定预算。为了找到最佳超参数配置，可能需要在估算的超参数范围内，使用不同的超参数进行多次测试。此外，还有多项研究致力于高效训练计划。Li 等人 [134] 提出了预算训练设置，并指出可以采用一种新的学习率调度器（原始线性调度器）来提升性能，尤其是在预算较小时。Chen 等人 [24]提出将新的学习率计划函数与采样率相结合，通过从给定函数中采样来更新学习率。他们表明，在低预算设置下，其计划的表现优于线性计划。

预算训练的总体方案。 预算训练的总体方案应包括配置关键超参数，并建立一条整合多种高效训练技术的训练流水线。开始预算训练时，重要的是确定若干会显著影响最终性能的超参数。可以根据有限的训练时间和所用设备的计算能力，估算总计算预算。随后，可依据经验规律 [93, 110] 或现有方案确定模型规模和训练数据词元数量。为确定最优参数配置，可以通过实验观察训练早期的结果，并选出最佳参数组合。在数据层面，确定数据集规模后，可以收集适量的高质量数据，并进行必要的处理，例如采用数据增强技术。在模型层面，虽然缩放定律 [110] 表明，在模型规模固定后，改变模型结构只能带来有限的性能提升，但通过某些模型结构改动 [68]仍可提高计算效率。在优化器层面，也有一些超参数需要确定，例如峰值学习率和学习率计划的预热比例；这些参数应依据现有方案或通过超参数搜索来确定。此外，还应引入高效优化器，以调整学习率 [24] 或模型结构 [185]。最后，从系统角度来看，可以将适用于给定预算的高效技术整合到现有方案中。可以采用针对特定硬件优化的技术来提升系统性能，例如，将数据集分片后离线处理并加载到 RAM [104] 中，以减少磁盘输入/输出瓶颈；也可以使用磁盘序列化 [184] 来节省 RAM。这些技术有助于提高系统吞吐率，并

表 4：提升预算训练速度和可用性的技术。

中心型	方面	方法	优势
Data	数据集	短序列	预算化的计算和内存
	数据加载器	磁盘序列化 [184]	预算内存，节省 RAM
		分片后离线加载到 RAM [104]	避免磁盘输入/输出
模型	模型架构	禁用 QKV 和线性层偏置 [68]	计算效率
		预归一化 [104, 222]	稳定性，大学习率
		门控线性单元 [68, 221]	提升性能
		移除非线性注意力头 [68]	计算和内存效率
		复杂度调度 [185]	效率
	模型规模	训练大模型	效率
优化器	学习率计划	线性层衰减至零 [134]	快速预算收敛
		单周期学习率 [226]	快速收敛
		弹性阶梯衰减 [183, 184]	预训练加速
	大批次大小	梯度累积	计算效率
		梯度检查点	预算内存
		激进的批量大小调度 [68]	早期训练进展
		重要性采样	计算效率
	权重平均	最新权重平均 [109]	加速
	其他	稀疏词元预测 [68, 104, 149]	内存效率
		梯度裁剪	稳定性
		在预训练期间禁用 dropout [68]	计算效率
		提前停止	效率

在不损失准确性的情况下实现整体加速。此外，正如我们将在第7节中描述的那样，可以将 DeepSpeed [205] 等框架添加到模型中，以实现 ZeRO 优化器 [202]，自动混合精度训练和融合实现。综合来看，这些技术可以提供一种高效的训练方案。除超参数配置外，我们总结并列出了表4，将其作为由各种技术组成的预算训练方案。

6.2 讨论

在本节中，我们从预算训练的角度回顾了高效训练技术。我们分别从数据、模型和优化三个方面总结了高效预算训练技术，并提出了一套全面的技术方案。

权衡，即牺牲一方以换取另一方的收益，在预算训练中发挥着重要作用。例如，人们开发了通过空间—时间权衡缓解内存限制的技术。此外，鉴于训练预算有限，影响模型性能的不同技术和超参数之间往往需要权衡。当单独调整这些参数的规模，或单独使用高效技术时，通常可以提升模型性能。但在计算预算固定的情况下，需要在这些超参数之间，以及计算密集型技术之间做出权衡。若干考察不同训练配置的系统性研究，为实践中如何决定权衡提供了经验建议，并提出了有效的预算分配方案 [93, 110]。问题在于，这些结论可能仍有片面性，即仅适用于某一范围内的参数；或者这些结论会得出有偏的规律，因为它们只是从实验中归纳出的经验规律。要在新的设置中验证这些规律，未来可能仍需训练更多模型，这会比沿用现有模型训练配置来取得良好结果耗费更多成本。相对可行的做法似乎是从小规模、低预算的训练设置入手，确定高效的预算训练方案，再逐步扩展到更大规模。另一个可能很有价值的思路是设计更具预算感知能力的训练组件，即能够根据剩余预算调整参数的训练组件，例如一些近期研究提出的学习率调度器 [24]。

总而言之，我们建议未来关于预算训练的研究重点关注以下方向：

• 实用基准测试。这里指的是可以研究一个统一且实用的预算基准，例如在低端 GPU 服务器上训练一天 [68, 104]。对于研究人员而言，在不同配置下（例如 [93, 110, 231]）开展系统性研究的成本过高。相反，在相同的实用预算下，未来的工作可以遵循现有的训练方案，并进一步引入优化。

• 预算感知调度器。高效学习率计划等算法 [24] 可以根据剩余预算进行调整，从而促进快速收敛并提升性能。仍有待开发更多能够动态响应剩余预算的技术。

7 以系统为中心的高效训练

从系统中心视角出发的研究，为所设计的算法提供了具体的实现方法。它研究硬件有效且实用的执行方式，从而真正实现高效训练。在本综述中，我们关注多节点集群中通用计算设备（例如 CPU 和 GPU设备）的实现。以硬件视角解决所设计算法中潜在的冲突，是核心关注点。本节主要回顾现有框架和第三方库中的硬件实现技术，这些技术能够高效支持数据、模型和优化的处理。随后，我们介绍一些现有的开源平台，它们为模型构建、高效利用数据进行训练、混合精度训练和分布式训练提供了坚实的框架。本节安排如下：

• **以系统为中心的数据效率**。 高效的数据处理和数据并行是系统实现中的两个重要考量。随着数据集迅速增大，低效的数据处理逐渐限制了训练效率，尤其是在多节点的大规模训练中。设计更适合硬件的计算方法并实现并行化，可以有效避免训练过程中的时间浪费。

• **以系统为中心的模型效率**。 随着模型参数数量急剧增加，从模型角度出发的系统效率已成为重要瓶颈之一。大规模模型的存储和计算效率给硬件实现带来了巨大挑战。在这一部分，我们主要回顾如何实现高效的部署 I/O 和精简的模型并行实现，从而加速实际训练。

• **以系统为中心的优化效率**。 优化过程涵盖每次迭代中的反向传播和更新，这是训练中最耗时的计算。因此，以系统为中心的优化实现直接决定了训练效率。为清晰阐释系统优化的特点，我们聚焦于不同计算阶段的效率，并回顾各个过程的改进。

• **开源框架**。高效的开源框架有助于开展训练，充当衔接算法设计与硬件支持的桥梁。我们研究了一系列开源框架，并分析了每种设计的优势与不足。

7.1 以系统为中心的数据效率

尽管数据处理在训练中的时间成本占比并不高，但低效的处理方式仍会导致训练时间大幅增加。尤其是在大规模模型上训练大批次时，合理利用无冲突进程来加速数据处理，可以有效减少额外开销。系统层面已经提出了许多工作，以实现高效的数据处理。在这一部分，我们主要回顾数据处理和数据并行的高效实现。随后，我们介绍一些可在实际场景中采用的高效库。

高效的数据处理可为加速训练带来显著收益，实现高效训练并减少系统中的冗余等待时间。Bai 等人 [6] 提出了 *PaGraph* ，用于支持在多GPU设备上进行通用且高效的基于采样的图神经网络训练。该方法利用可用的GPU设备，通过缓存存储维护频繁访问的图数据。Qi 等人 [192] 推出了*SpeeChain*，这是一款开源 PyTorch 工具包，用于开发大规模机器语音链应用。它同时支持多数据加载器批生成和在线数据选择技术。Jia 等人 [106]设计了 *A-Dloader* ，用于动态分配 CPU资源以并发运行多个程序，并采用数据加载器分配作为本地阀门，以缩短访问 GPU设备的间隔，从而提高整体效率。Svogor 等人 [238] 评估了多个基准，以梳理数据加载器流水线特定步骤中的性能问题，并修改了 *ConcurrentDataloader* ，进一步提高 GPU 利用率。它支持对云数据集进行高效训练。Xie 等人 [278] 关注由多个训练任务并行执行的重叠数据集。他们提出了一种依赖采样方法*JOADER* ，该方法采用领域专用缓存策略和一种用于加载数据的新型树结构。Dash 等人 [41] 为超级计算机开发了一种解决方案，以缓解超高分辨率图像中小图像块加载效率低、符号说明极度稀疏以及磁盘存储I/O延迟等问题。

在使用多个 GPU 时，可以通过数据并行来加速计算。采用数据并行时，每个节点都持有一份模型副本，处理不同的数据分片，然后执行前向与反向计算以获得梯度。在集中式架构中，这些节点会将计算出的梯度发送到参数服务器，由参数服务器聚合参数、更新模型，再将参数发送回各个节点。相比之下，在去中心化架构中（深度神经网络训练中更为常见），每个节点在进行多节点通信后更新模型权重中对应的部分。随着模型规模不断扩大，分布式训练会成为一项工程难题，训练人员需要优化分布策略，以及网络通信和内存。例如，在多个 GPU 上训练 PyTorch模型，只需修改代码以使用*torch.nn.DataParallel*即可。然而，在大批次上训练语言模型等大模型时，必须平衡 GPU 负载，例如进一步分配损失准则计算，以免收集结果的服务器节点负载过重。

高效库。 高效库为训练过程提供了坚实的支持。当数据集规模较小时，所有数据样本都可以在训练前存储到计算设备中，无需额外花费进行数据I/O。然而，随着深度神经网络的迅速发展，数据集规模呈几何级数增长。如此海量的数据必须存储在本地存储中，并在参与训练时以批次交替读取。在实际应用中，数据加载器并非主要瓶颈（通常占总耗时的 10% 以下）。各框架中的传统数据加载器通过多线程在 CPU 上执行数据I/O和预处理，因此需要确保有足够数量的核心。Leclerc 等人 [126] 设计了一种新颖的结构，将图像样本存储在*FFCV* 中，采用.*beton*文件格式，并将其划分为文件头、数据表、堆存储和分配表四个主要部分。在整个过程中，他们对各部分进行重新组合和编译，以充分利用缓存结构和机制。即使在磁盘性能特别昂贵的情况下，也会采用准随机采样技术，尽可能减轻底层存储的负担。另一项核心原则是利用环形缓冲区，避免流水线中不必要的内存分配。这一改进使其在处理图像和视频等数字数据源时效率极高。若干数据增强方法涉及各种基础矩阵计算，由此也带来了数据增强方面的另一项创新加速方案。英伟达推出数据加载库（*DALI*），以支持快速、可移植且灵活的数据加载。*DALI* 是一个开源库，用于对图像或语音信号进行解码和数据增强。它为开发者提供外部流水线，以便在并行单元上利用GPU设备处理原始数据样本。在实际应用中，这种GPU友好型数据加载器能够有效缩短训练耗时，并稳定 CPU 与 GPU 之间的通信频率和间隔。它可以集成到 Python 以及许多广泛使用的深度学习框架中，例如 MXNet、PyTorch 和 TensorFlow。持续学习是另一项深度学习领域的基础研究，

与离线训练的基本场景不同，持续学习允许训练过程建立在动态模型架构和不固定的数据集流之上。这种训练模式会使当前大多数数据加载器面临严重拥塞。为解决这些难题，Carta 等人 [21] 基于 PyTorch 提供并维护了开源库*Avalanche*，以拓展持续学习和增量训练的应用场景。*AvalancheDatasets* 可以通过子采样和拼接来定义和操控本地训练中的数据流与回放缓冲区。它还提供了灵活的设置，可控制多个数据集之间的平衡并进行交叉采样。

7.2 以系统为中心的模型效率

硬件设计与算法设计是相辅相成的过程。研究人员可以基于对模型的理解，根据 CPU、GPU 和其他硬件架构的内存、计算和通信特性，优化算法的计算与内存使用。在本节中，我们重点关注大规模模型部署的高效存储，以及通过模型并行化降低其实际训练开销。

通道后置 是一种高效的存储格式，在训练中可作为 4D *NCHW* 张量的替代方案。这种方法逐像素存储张量，将通道置于最后一个维度。在混合精度训练下，使用通道后置格式并在张量核心上训练模型，可提高训练吞吐量 [61]。另一种高效方法是采用量化低精度格式来存储和计算张量。混合精度训练 [166] 是一种适用于单GPU和多GPU场景的技术，通过混合使用单精度和半精度浮点格式来加速网络训练过程。使用 FP16（即半精度浮点数）存储权重和梯度时，可通过提高吞吐量并减少内存使用与访问来加速训练。同时，还会存储一份 FP32 权重主副本，并在优化器步骤中根据权重梯度进行更新，以维持单精度训练所达到的网络精度。使用混合精度可以在给定硬件上训练更大的模型或采用更大的批量大小。BF16 [263] 是一种不同于 FP16 的半精度格式，可表示范围更大的浮点数以提高稳定性；用它替代 FP16 可能会取得更好的结果 [33, 125]。稀疏训练也是实现高效训练的良好方案。关于模型稀疏性的研究很多，但在实际场景中，稀疏计算需要硬件支持，否则很难实现加速。幸运的是，先进的硬件设计已为稀疏性提供了更多支持，例如英伟达的 Ampere GPU架构在其矩阵运算单元（即张量核心）中引入了对稀疏性的支持，并采用 2:4（50%）的稀疏模式 [170]，从而使数学运算吞吐量达到稠密运算的两倍。基于结构化剪枝的方法利用 CPU 和 GPU 架构中的硬件稀疏性，有助于扩大模型规模 [105]。为实现适配硬件的结构化稀疏性，Ma等人 [157] 提出了面向基于块剪枝的硬件友好型重组（HRBP），该方法在反向传播过程中保留提取出的稠密块。

对于极大规模的模型，模型并行是一种预期采用的方法，可通过增加计算资源有效缩短训练时间。对于无法放入单张 GPU 卡、但可以放入多 GPU 服务器的大模型，或者对于模型中会消耗大量 GPU 内存的操作（例如类别数量很多时的 Softmax），可以使用张量并行（也称为层内并行或模型并行）来减少内存使用并分摊计算。在张量并行中，模型的部分特定权重、梯度和优化器状态会被拆分到多个设备上，原张量的每个切片放置在指定的 GPU 上，并在分布式环境中进行计算。以 [174]，中给出的 Transformer 层的多层感知机块为例，它由两个通用矩阵乘法和一个 GELU 函数组成，$Y = GeLU(XA)$ $Z = YB$。给定两个处理器，我们将 A 按列划分为 $[A_1 A_2]$，使输出可以独立馈入 $GeLU$。每个处理器上分别计算 $Y_i = GeLU(XA_i)$，得到 $[Y_1\ Y_2] = [GeLU(XA_1)\ GeLU(XA_2)]$。这称为列并行方法。当第二个线性层 $Z = YB$ 接在列并行层之后时，我们将 B 按行划分为 $[B_1 B_2]^\top$，以匹配 Y 的形状并避免额外通信，这称为行并行。要计算$Z = [Y_1 Y_2][B_1 B_2]^\top$，我们先计算 $Y_i B_i$，然后使用 all- 将结果聚合到 $Z = Y_1 B_1 + Y_2 B_2$。

归约方法。我们注意到，列并行线性层需要在反向传播过程中跨处理器应用全归约，以聚合输入张量的拆分梯度。例如，对于 $Y = XA$，应用全归约以获得 $\dot{X} = \dot{Y}A^\top = \dot{Y}_1 A_1^\top + \dot{Y}_2 A_2^\top$。然而，当模型规模极大，必须跨多个多GPU服务器进行拆分时，这种并行化存在两个问题 [174, 229]: (a)张量并行需要节点之间的高带宽通信，例如在单个 DGX A100服务器内使用NVLink，但对于多个服务器之间的通信，服务器间链路速度较慢；(b) 更多节点会导致更小的矩阵乘法（通用矩阵乘法），这可能会降低 GPU 利用率。

从以模型为中心的视角来看，硬件支持主要针对存储和计算。高效存储可使模型在设备间传输和分配时节省大量时间并实现加速。此外，使用混合精度数据类型也能在维持效果的同时，有效避免等待时间的冗余。基于模块特定或层特定优化的一些底层方法也能带来显著收益，为大规模模型的高效训练奠定坚实基础。

7.3 以系统为中心的优化效率

优化对应于反向传播和参数更新过程，这是训练中最耗时的计算。它对于加速至关重要。受反向传播机-制的影响，传统范式在优化过程中可操作的空间相对有限。因此，训练效率的主要决定因素之一，是能否实现高效的底层优化过程。在本节中，我们主要回顾实际实现，包括通过计算图重建进行重物化、流水线并行、融合算子、高效的设备间通信以及零冗余优化（ZeRO）。

重物化是一种重建计算图中实体的技术，可以有效避免特定冗余。它通常会改变梯度计算。梯度检查点 [26] （重物化）可用于重建计算图；在反向传播期间，该方法会释放中间输出，并重新计算前向结果，以恢复已释放的张量。重物化也可以手动实现，如 Touvron 等人 [253] 在 LLaMA 的高效实现中所示。为提高训练效率，他们通过重建计算图重新实现反向传播，而不是使用 Autograd，从而存储线性层的输出。他们的方法降低了激活值的开销，因为这些激活值在反向传播过程中重新计算的计算成本很高。

流水线并行是一种高效的并行技术，它将输入的批次分解为小批次（从而减少流水线气泡），并将模型的层划分到多个 GPU 上，由此创建一条流水线，其中每个阶段，即一组连续的层，以前一阶段的结果作为输入，并将结果传递到下游，使不同的GPU 能够并行参与计算过程。它的通信量最低，并且可以跨节点调度。最先进的系统 [229]使用减少通信以及增加局部性和并行化的组合来提高训练效率。Huang 等人 [100] 提出了 Gpipe 调度，其中只有在一个批次的完整前向传播过程完成后才执行反向传播过程（先前向后反向）。Narayanan 等人 [173] 提出了一种一前向一反向（1F1B）的调度机制，Narayanan等人 [174] 展示了具有交错阶段的 1F1B 调度，其中每个设备被分配多个层子集（称为模型块），而不是一组连续的层；与先前向后反向相比，这种方式在时间和空间方面都更加高效。单个 GPU的有限内存限制了其所能容纳的最大模型规模，也限制了GPU利用率。为了进一步增大模型规模以获得更高的准确率，现有的开源框架通常提供模型并行库，以帮助在多个节点上高效地分布和训练模型，从而提高准确率和吞吐量，实现最优的分布式训练。这些框架集成了并行训练和集合通信库（例如 NCCL），使研究人员能够通过在现有训练设置中进行调优来解决训练瓶颈。

融合实现通过重复利用计算单元来实现高效率。这些基础优化可以通过组合操作来减少内存 I/O，并提高计算效率。内核

融合例如指将多个操作组合到单个内核中，以提升性能和数值稳定性。借助面向英伟达 GPU 的 nvFuser 等深度编译器，研究人员可以将多个神经网络层融合到单个 CUDA 内核中，并重构 CUDA 算子，以高效利用 GPU 的计算单元，进一步减少内存 I/O。这些方法显著提升了整体速度。例如，英伟达 Apex [178] 实现了一些常见优化器（如 Adam）的融合版本，避免了多次读写 GPU 内存，速度也快于 PyTorch 中的实现。PyTorch 最近发布的 2.0 版本提供了*torch.compile*，它可以使用 OpenAI Triton 编译器生成代码，其性能可媲美手写内核和专用 CUDA 库。

高效通信对于训练加速极其重要，尤其是在大规模训练中。跨多节点的通信总线效率成为主要瓶颈。即使在单个节点内，CPU 与 GPU 之间的大量通信仍会导致严重阻碍。在大规模分布式深度学习训练中，重点关注 GPU 之间的通信效率非常重要：在单台机器上，GPU 通过 PCIe 或 NVLink 进行通信；在多台机器之间，则通过 Socket（以太网）或 InfiniBand（配合 RDMA）进行通信，且节点内的通信带宽高于节点间带宽。现有算法和系统可以通过优化来提高通信效率。例如，*1* 位压缩 [218]通过将梯度量化为每个值最低 1 位来减少通信。Tang 等人 [245] 提出了*1* 位 *Adam* ，该方法适用于基于非线性梯度的优化器*Adam*。Lu 等人[155] 提出*0/1 Adam* ，用于解决*Adam* 在采用激进的 1 位量化和本地步时面临的非线性挑战；与 *1* 位 *Adam* 基线相比，该方法实现了更高的训练吞吐量，并减少了端到端训练时间。Markov 等人 [162] 提出了一种名为 CGX 的通信框架，支持数据并行训练中的压缩通信。此外，还可以探索更适合高效通信的并行策略。数据并行、张量并行和流水线并行是三种常用的并行策略。例如，Narayanan 等人 [174]展示了如何组合数据并行、张量并行和流水线并行，将规模扩展至数千个 GPU。

Rajbhandari 等人 [202] 提出了零冗余优化器（ZeRO），这是目前训练大规模模型的重要技术。ZeRO通过在处理器之间分三个相应阶段对冗余模型状态（即优化器状态、梯度和参数）进行分区并优化通信，从而优化内存中的冗余模型状态，最终使模型状态均匀分布在每个节点上。在同时划分优化器状态和梯度的第二阶段（ZeRO-2）中，构建了ZeRO-Offload [209] ，它将这两类状态的数据和计算卸载到 CPU，从而利用CPU 节省GPU 内存。然而，由于 GPU内存有限，存储在GPU 内存中并复制到所有设备上的参数数量仍然受限；次优的数据分区和有限的 PCIe 带宽也要求采用大批量训练才能提高效率。此外，Rajbhandari 等人 [203]展示了 Zero-infinity，这是一种异构系统技术，可在多个设备之间并行利用 CPU 和 NVMe内存（其成本低、速度慢但容量巨大），为当前 GPU 集群聚合高效带宽。当前的大规模深度学习模型通常达到 10B 甚至 100B个参数，例如 GPT3 [18] （175 B）和 MT-NLG [229]（530 B）。训练如此大规模的模型通常需要综合使用数据并行、张量并行、流水线并行、混合精度训练以及类似 ZeRO的分布式优化器，也称为 3D 混合并行。通常，具有最大通信量的张量并行会优先部署在单个节点内。如果可能，数据并行也会部署在节点内，以加快梯度通信；否则则会映射到邻近节点。

7.4 Open SourceFrameworks

对于希望简单地应用并行技术来训练大模型的研究人员来说，各种现有的开源框架都提供了便捷的实现。要从单GPU扩展到多GPU，甚至多机器，通常只需导入库、定义训练策略和环境，并封装优化器即可。下面我们介绍一些其他平台、开源框架和库，以及它们如何使（分布式）训练大规模模型受益。

表5：高效训练框架。

托管方	库	优势
PyTorch	PyTorch 2.0	*torch.compile*、混合精度、DDP 和 FSDP
百度	飞桨	3D 并行、混合精度、GroupSharded（类似 ZeRO）、模型实现
字节跳动	LightSeq	混合精度、融合算子、GEMM 优化
谷歌	JAX	Autograd、XLA
微软	DeepSpeed	3D 并行、ZeRO、混合精度、数据效率库
	TorchScale	扩展 Transformer
元研究	FairScale	FSDP、流水线并行、通用技术、Adascale
	Fairseq	FSDP、通用技术、模型实现
	xFormers	Transformer 模型实现
MosaicML	Composer	多GPU训练、加速方法的实现
拥抱脸	Transformer	模型实现
	加速	分布式训练、混合精度、支持 FSDP、DeepSpeed 和 Megatron-LM。
HPC-AI 技术	Colossal-AI	3D（尤其是张量）并行化、序列并行、ZeRO
OneFlow	LiBai	3D 并行, 混合精度, ZeRO, 通用技术
英伟达	Megatron-LM	分布式预训练，模型实现
	Apex	自动混合精度、分布式训练、融合优化器、融合层和层归一化

JAX [16] 是一个用于机器学习的库，它结合了Autograd（一个自动微分库）和 XLA（加速线性代数，用于线性代数的编译器），并提供与NumPy 相同的 API，同时兼容 GPU 和 TPU。对于深度学习，尤其是在使用 TPU 的情况下，基于 JAX的库（如 Flax [86], 和Haiku [90]）提供了深度学习特征，而 JAX 对海森矩阵的高效计算也值得考虑。JAX 和 Haiku 中有一个用于并行 Transformer的库 Mesh-Transformer-JAX [259] 。此外，谷歌最近使用基于 JAX 的库 Scenic [43]发布了 ViT-22B。DeepSpeed [205] 是一个用于大模型训练的开源PyTorch 库，提供 ZeRO（零冗余优化器） [202, 203, 209], 3D-并行化等功能，并用于BLOOM(176B) [215] 和 MT-NLG (530B) [229] 的训练。DeepSpeed 的数据效率库 [132] 结合了课程学习和随机逐层令牌丢弃技术（随机-LTD） [286]。最近，微软推出了用于扩展Transformer 的 TorchScale 开源工具包 [158] 。Meta Research的 FairScale [58] 是一个 PyTorch扩展库，提供分布式训练技术，例如模型并行化（包括完全分片数据并行FSDP）、激活检查点、梯度累积、梯度裁剪等，以改善内存使用并加快训练过程。FairScale 还包括一些优化器和调度器，例如 Adascale（大批量优化器）、FusedAdam、FusedSGD 等，以提高大模型训练的稳定性和数据效率。另一个库 Fairseq [181] 提供了一系列序列到序列和语言模型实现，例如 Transformer、RoBERTa [149], 等，同时也提供了一些用于数据预处理和训练的工具。此外，还有 xFormers [128], , 这是一个 Transformer构建模块库，支持以可组合的方式构建Transformer。飞桨平台 [160]是一个专为并行分布式深度学习设计的框架，支持各种深度学习硬件。相关开发套件以及文心 [235] 后续开展的一系列工作涵盖了大量训练技术。

有些框架提供覆盖整个流程的全面操作。Megatron-LM 是英伟达开发的大型 Transformer 库，并配备高效的训练框架。此外，英伟达还维护着 Apex [178] 工具库，用于混合精度和分布式训练。MosaicML 的 Composer [247], 是一个用于高效训练的 PyTorch 库，它为训练过程的各个环节提供接口，支持将内置或自定义的多种加速方法组合到训练流水线中，从而制定高效的训练方案。字节跳动的 LightSeq [267, 268], 是一个用于 Transformer 的高效推理与训练引擎。Bagua [66] 由快手和苏黎世联邦理工学院开发，是一个面向 PyTorch 的深度学习训练加速框架。Colossal-AI [11], 提供一系列并行组件，用于构建和训练分布式人工智能模型。除了上文提到的一维张量并行外，Colossal-AI [11] 平台还集成了二维、2.5 维、三维

张量并行策略 [12, 260, 282] 和序列并行 [135]，可分配计算与内存负载，并进一步优化通信开销。EleutherAI GPT-NeoX [2], 是一个用于训练大规模语言模型的库，GPT-NeoX-20B [13]这一拥有 200 亿参数的模型便基于该库构建。Ray [206],是一个开源框架，用于扩展人工智能和 Python 应用，例如机器学习应用。Howard 和 Gugger 构建的 fastai 深度学习库 [96], 为研究人员提供高层和底层组件，帮助他们构建最先进或可配置的方法。OpenBMB BMTrain [180], 是一个用于高效训练大型模型的大模型训练工具包。OneFlow-LiBai [143], 是一个大规模开源训练工具箱，提供三维并行分布式训练及其他训练技术。华为 MindSpore [168], 是一个人工智能计算框架。Horovod [220] 由 LF AI & Data Foundation 托管，是一个分布式深度学习框架。

总之，我们通过表5概述各框架库的优势，并清晰展示它们对高效训练的主要贡献。正如前文所述，训练大模型需要综合运用各种通用技术。多个深度学习框架已专门设计用于为训练加速提供高效支持。我们重点关注能够提高效率的有效实现。随着相关工作持续推进，更多技术仍在不断涌现并得到探索。

7.5 讨论

本节从系统中心视角回顾训练加速。系统实现承载着算法设计，而算法设计决定了实际场景中的训练效率。所设计算法与技术的效率也高度依赖可靠的硬件支持，以实现最佳性能。本节在系统层面研究当前的数据处理、模型训练和优化执行技术，并总结现有方法的优缺点如下。

从以系统为中心的数据处理角度来看，关键方向包括高性能数据处理和数据并行。随着数据集规模迅速增长，训练数据样本的预处理变得复杂而繁重，这极大阻碍了大批量训练的发展。当数据处理时间与训练消耗无法匹配时，训练过程便会进入等待状态，导致效率极低。越来越多的研究致力于减少通信并保持数据处理本地化，从而实现高效的数据处理与并行化。同样的思路也应用于模型层面。与数据处理不同，模型通常存储在计算设备上。因此，我们必须应对大规模模型部署的挑战。多种基于量化压缩的技术和混合精度模型有效保障了训练加速。此外，在大规模训练中，分布式架构的优化也有助于提高模型的计算效率。除了数据处理和模型部署之外，硬件对优化过程的支持是实际训练中最重要的环节。当前主流技术路线围绕重构计算图上的高效计算、用于分阶段训练的流水线并行、对高效优化器的融合实现，以及多节点和多设备间的高效通信展开。计算图和流水线决定了硬件计算中的简化计算流程，而通信效率则消除了训练中系统层面的实际运行时间冗余。在硬件系统的可靠支持下，所设计算法的效果便能得到充分发挥。

未来，我们看好以下研究方向：

• 局部化数据处理。我们可以充分利用 GPU设备中的高效并行计算，在大批量训练中实现数据正则化等预处理。有一些高效库可以实现这一技术，但它尚未得到充分普及。仍有许多操作有待发现。

• 垂直模型部署。随着模型规模不断增大，计算设备的局限性将愈发明显。同时，由于梯度的链式法则，大规模模型中各个模块的计算无法完全并行化。关于垂直训练的研究cuntegn

多节点、多设备正成为值得探索的有前景方向。垂直训练的优化算法也有望成为后续研究方向。

- 新颖的高效范式。开发新颖且高效的范式，可以进一步加快大模型的训练。设计时应兼顾各个独立模块的加速技术，以及它们之间可能存在的相互作用和冲突，从而为高效训练框架的实施提供有力支持。例如，针对大规模边缘计算设备，在去中心化设置中协同训练模型也是未来值得探索的方向 [289]。

8 结论

在本综述中，我们回顾了用于高效训练大规模深度学习模型的通用训练加速技术。我们考察了深度学习领域中涵盖整个训练过程的基于梯度的更新公式（3）中的所有组成部分。我们提出了一种新颖的分类体系，将这些技术归纳为五个主要方向：“以数据为中心”“以模型为中心”“以优化为中心”“预算训练”和“以系统为中心”。在前四个部分中，我们主要从算法设计和方法论的角度进行全面探讨。在“以系统为中心的高效训练”部分，我们从范式创新和硬件支持的角度总结了实际实现方式。我们回顾并总结了各个组成部分中常用或最新提出的技术、这些技术的优势与权衡，并讨论了它们的局限性以及高效训练领域中值得探索的未来方向。

在提供全面的技术综述与指导的同时，我们的调研也介绍了高效训练领域当前的突破与瓶颈。我们希望，这项调研不仅能够帮助研究人员高效实现通用的训练加速，还能为高效训练的未来发展提供有意义且前景广阔的启示。除了各节末尾提到的一些潜在进展之外，我们还提出以下更广泛且前景广阔的观点：**(i) 高效配置搜索**。高效训练可以从数据增强组合、模型架构、优化器设计等角度，为模型设计预构建且可定制的配置搜索策略。这方面的研究已取得了一些进展。此外，新的模型结构与压缩模式、新的预训练任务，以及对“模型-边缘” [80] 知识的利用，也值得进一步探索。未来，这一领域必将取得更加辉煌的成果。**(ii) 自适应调度器**。通过使用经过优化的调度器，例如用于课程学习、学习率和批量大小以及模型复杂度的调度器，可以获得更好的性能。能够根据剩余预算动态调整的预算感知调度器，可降低这些方面的人工设计成本。自适应调度器可用于探索更强大的并行化与通信方法，同时兼顾更普遍且实际的场景，例如在广域异构网络和数据中心上进行大规模去中心化训练。

参考文献

[1] Donald G Anderson。非线性积分方程的迭代求解过程。美国计算机协会学报（*JACM*），12(4):547–560，1965年。[2] Alex Andonian、Quentin Anthony、Stella Biderman、Sid Black、Preetham Gali、Leo Gao、Eric Hallahan、Josh Levy-Kramer、Connor Leahy、Lucas Nestler、Kip Parker、Michael Pieler、Shivanshu Purohit、Tri Songz、WangPhil 和 Samuel Weinbach。GPT-NeoX：基于 PyTorch 的大规模自回归语言建模，8月 2021年。URL https://www.github.com/eleutherai/gpt-neox。[3] Maksym Andriushchenko和Nicolas Flammarion。迈向对尖锐度感知小批量最小化的理解。在国际机器学习会议，第 639–668 页。PMLR，2022年。

[4] Eric Arazo、Diego Ortego、Paul Albert、Noel E O'Connor 和 Kevin McGuinness。重要性采样对于深度预算训练有多重要？*arXiv*预印本 *arXiv:2110.14283*，2021年。[5] Alexei Baevski 和 Michael Auli。用于神经语言建模的自适应输入表示。*arXiv*预印本 *arXiv:1809.10853*，2018年。[6] Youhui Bai、Cheng Li、Zhiqi Lin、Yufei Wu、Youshan Miao、Yunxin Liu 和 Yinlong Xu。用于大型图上快速基于采样的图神经网络训练的高效数据加载器。*IEEE*并行与分布式系统汇刊，32(10):2541–2556，2021年。[7] Hangbo Bao、Li Dong、Songhao Piao 和 Furu Wei。BEiT：图像变换器的BERT预训练。*arXiv*预印本 *arXiv:2106.08254*，2021年。[8] Yoshua Bengio、Pascal Lamblin、Dan Popovici 和 Hugo Larochelle。深度网络的贪心逐层训练。神经信息处理系统进展，19，2006年。[9] Yoshua Bengio、J´erˆome Louradour、Ronan Collobert 和 Jason Weston。课程学习。载于第*26*届机器学习国际年会论文集，第41–48页，2009年。[10] Albert S Berahas、Majid Jahani、Peter Richt´arik 和 Martin Tak´aˇc。机器学习的拟牛顿方法：忘掉过去，只需采样。优化方法与软件，37(5):1668–1704，2022年。[11] Zhengda Bian、Hongxin Liu、Boxiang Wang、Haichen Huang、Yongbin Li、Chuanrui Wang、Fan Cui 和 Yang You。Colossal-AI：用于大规模并行训练的统一深度学习系统。*arXiv*预印本 *arXiv:2110.14883*，2021年。[12] Zhengda Bian、Qifan Xu、Boxiang Wang 和 Yang You。最大化超大型神经网络分布式训练中的并行化。*arXiv*预印本 *arXiv:2105.14450*，2021年。[13] Sid Black、Stella Biderman、Eric Hallahan、Quentin Anthony、Leo Gao、Laurence Golding、Horace He、Connor Leahy、Kyle McDonell、Jason Phang、Michael Pieler、USVSN Sai Prashanth、Shivanshu Purohit、Laria Reynolds、Jonathan Tow、Ben Wang 和 Samuel Weinbach。GPT-NeoX-20B：开源自回归语言模型。载于*ACL*大型语言模型创建挑战与展望研讨会论文集，2022年。网址：https://arxiv.org/abs/2204.06745。[14] Daniel Bolya、Cheng-Yang Fu、Xiaoliang Dai、Peizhao Zhang、Christoph Feichtenhofer 和 Judy Hoff- man。令牌合并：让你的视觉变换器运行得更快。*arXiv*预印本 *arXiv:2210.09461*，2022年。[15] Rishi Bommasani、Drew A Hudson、Ehsan Adeli、Russ Altman、Simran Arora、Sydney von Arx、Michael S Bernstein、Jeannette Bohg、Antoine Bosselut、Emma Brunskill 等。基础模型的机遇与风险。*arXiv*预印本 *arXiv:2108.07258*，2021年。[16] James Bradbury、Roy Frostig、Peter Hawkins、Matthew J ames Johnson、Chris Leary、Dougal Maclaurin、George Necula、Adam Paszke、Jake VanderPlas、Skye Wanderman-Milne 和 Qiao Zhang。JAX：Python+NumPy 程序的可组合变换，2018年。网址：http://github.com/google/jax。[17] Andrew Brock、Theodore Lim、James M Ritchie 和 Nick Weston。Freezeout：通过逐步冻结层来加速训练。*arXiv*预印本 *arXiv:1706.04983*，2017年。[18] Tom Brown、Benjamin Mann、Nick Ryder、Melanie Subbiah、Jared D Kaplan、Prafulla Dhariwal、Arvind Neelakantan、Pranav Shyam、Girish Sastry、Amanda Askell 等。语言模型是少样本学习者。神经信息处理系统进展，33:1877–1901，2020年。[19] Han Cai、Chuang Gan 和 Song Han。EfficientViT：用于高分辨率低计算量视觉识别的增强型线性注意力。*arXiv*预印本 *arXiv:2205.14756*，2022年。[20] Brad Carlile、Guy Delamarter、Paul Kinney、Akiko Marti 和 Brian Whitney。利用逆平方根线性单元（ISRLU）改进深度学习。*arXiv*预印本 *arXiv:1710.09967*，2017年。

[21]Antonio Carta、Lorenzo Pellegrini、Andrea Cossu、Hamed Hemati 和 Vincenzo Lomonaco。Avalanche：用于深度持续学习的 PyTorch 库。*arXiv* 预印本 *arXiv:2302.01766*，2023年。[22] Beidi Chen、Tri Dao、Eric Winsor、Zhao Song、Atri Rudra 和 Christopher R´e。Scatterbrain：统一稀疏注意力与低秩注意力。神经信息处理系统进展，34：17413–17426，2021年。[23] Jie-Neng Chen、Shuyang Sun、Ju He、Philip HS Torr、Alan Yuille 和 Song Bai。Transmix：视觉Transformer的注意力混合。载于*IEEE/CVF*计算机视觉与模式识别会议论文集，第12135–12144页，2022年。[24] John Chen、Cameron Wolfe 和 Tasos Kyrillidis。Rex：通过改进训练计划重新审视预算训练。机器学习与系统会议论文集，4：64–76，2022年。[25] Tianlong Chen、Yu Cheng、Zhe Gan、Jingjing Liu 和 Zhangyang Wang。超越（单纯）数据增强的数据高效生成对抗网络训练：彩票奖券视角。神经信息处理系统进展，34：20941–20955，2021年。[26] Tianqi Chen、Bing Xu、Chiyuan Zhang 和 Carlos Guestrin。以次线性内存开销训练深度网络。*arXiv* 预印本 *arXiv:1604.06174*，2016年。[27] Ting Chen、Simon Kornblith、Mohammad Norouzi 和 Geoffrey Hinton。用于视觉表征对比学习的简单框架。载于国际机器学习会议，第1597–1607页。PMLR，2020年。[28] Xiangning Chen、Chen Liang、Da Huang、Esteban Real、Kaiyuan Wang、Yao Liu、Hieu Pham、Xuanyi Dong、Thang Luong、Cho-Jui Hsieh 等。优化算法的符号发现。*arXiv* 预印本 *arXiv:2302.06675*，2023年。[29] Yuzhong Chen、Zhenxiang Xiao、Lin Zhao、Lu Zhang、Haixing Dai、David Weizhong Liu、Zihao Wu、Changhe Li、Tuo Zhang、Changying Li 等。用于少样本学习的掩码引导视觉变换器（mg-vit）。*arXiv* 预印本 *arXiv:2205.09995*，2022年。[30] Bowen Cheng、Ishan Misra、Alexander G Schwing、Alexander Kirillov 和 Rohit Girdhar。掩码注意力变换器：通用图像分割方法。载于*IEEE/CVF*计算机视觉与模式识别会议论文集，第1290–1299页，2022年。[31] Chao Cheng 和 Keshab K. Parhi。卷积神经网络的快速二维卷积算法。*IEEE*电路与系统汇刊*I*：常规论文，67(5)：1678–1691，2020年。doi：10.1109/TCSI.2020.2964748。[32] Krzysztof Choromanski、Valerii Likhosherstov、David Dohan、Xingyou Song、Andreea Gane、Tamas Sarlos、Peter Hawkins、Jared Davis、Afroz Mohiuddin、Lukasz Kaiser 等。重新思考 Performer 中的注意力。*arXiv* 预印本 *arXiv:2009.14794*，2020年。[33] Aakanksha Chowdhery、Sharan Narang、Jacob Devlin、Maarten Bosma、Gaurav Mishra、Adam Roberts、Paul Barham、Hyung Won Chung、Charles Sutton、Sebastian Gehrmann 等。PaLM：通过 pathways 扩展语言建模。*arXiv* 预印本 *arXiv:2204.02311*，2022年。[34] Djork-Arn´e Clevert、Thomas Unterthiner 和 Sepp Hochreiter。利用指数线性单元（ELU）实现快速、准确的深度网络学习。*arXiv* 预印本 *arXiv:1511.07289*，2015年。[35]Rosset Corby。Turing-NLG：微软推出的170亿参数语言模型。
h
t
t
p
s://
w
w
w.
microsoft.com/en-us/research/blog/turing-nlg-a-17-billion-parameter-language-model-by-microsoft/，2020年。

[36] Ekin D Cubuk、Barret Zoph、Dandelion Mane、Vijay Vasudevan 和 Quoc V Le。AutoAugment：从数据中学习数据增强策略。*arXiv* 预印本 *arXiv:1805.09501*，2018年。[37] Ekin D Cubuk、Barret Zoph、Jonathon Shlens 和 Quoc V Le。RandAugment：缩小搜索空间的实用自动数据增强方法。载于*IEEE/CVF*计算机视觉与模式识别会议研讨会论文集，第702–703页，2020年。[38] Ali Dabouei、Sobhan Soleymani、Fariborz Taherkhani 和 Nasser M Nasrabadi。Supermix：监督数据增强混合。载于*IEEE/CVF*计算机视觉与模式识别会议论文集，第13794–13803页，2021年。[39] Haixing Dai、Zhengliang Liu、Wenxiong Liao、Xiaoke Huang、Zihao Wu、Lin Zhao、Wei Liu、Ninghao Liu、Sheng Li、Dajiang Zhu 等。ChatAug：利用 ChatGPT 进行文本数据增强。*arXiv* 预印本 *arXiv:2302.13007*，2023年。[40] Tri Dao、Daniel J Fu、Stefano Ermon、Atri Rudra 和 Christopher R´e。FlashAttention：具备IO感知能力的快速、节省内存的精确注意力机制。*arXiv* 预印本 *arXiv:2205.14135*，2022年。[41] Sajal Dash、Benjam´ın Hern´andez、Aristeidis Tsaris、Folami T Alamudun、Hong-Jun Yoon 和 Feivi Wang。面向领导级高性能计算系统的十亿像素全切片图像分析可扩展流水线。载于*2022*年*IEEE*国际并行与分布式处理研讨会（*IPDPSW*），第1266–1274页。IEEE，2022年。[42] Yann N Dauphin 和 Samuel Schoenholz。MetaInit：通过学习如何初始化来学习初始化。神经信息处理系统进展，32，2019年。[43] Mostafa Dehghani、Alexey Gritsenko、Anurag Arnab、Matthias Minderer 和 Yi Tay。Scenic：用于计算机视觉研究及其他领域的 JAX 库。载于*IEEE/CVF*计算机视觉与模式识别会议论文集（*CVPR*），第21393–21398页，2022年。[44] Mostafa Dehghani、Josip Djolonga、Basil Mustafa、Piotr Padlewski、Jonathan Heek、Justin Gilmer、Andreas Steiner、Mathilde Caron、Robert Geirhos、Ibrahim Alabdulmohsin 等。将视觉Transformer扩展至220亿参数。*arXiv* 预印本 *arXiv:2302.05442*，2023年。[45] Jia Deng、Wei Dong、Richard Socher、Li-Jia Li、Kai Li 和 Li Fei-Fei。ImageNet：大型分层图像数据库。载于*2009*年*IEEE*计算机视觉与模式识别会议，第248–255页。IEEE，2009年。[46] John E Dennis, Jr 和 Jorge J Mor´e。拟牛顿方法：动机与理论。*SIAM*评论，19 (1)：46–89，1977年。[47] Tim Dettmers、Mike Lewis、Sam Shleifer 和 Luke Zettlemoyer。通过分块量化实现8位优化器。*arXiv* 预印本 *arXiv:2110.02861*，2021年。[48] Jacob Devlin、Ming-Wei Chang、Kenton Lee 和 Kristina Toutanova。BERT：用于语言理解的深度双向Transformer预训练。*arXiv* 预印本 *arXiv:1810.04805*，2018年。[49] Terrance DeVries 和 Graham W Taylor。利用 Cutout 改进卷积神经网络的正则化。*arXiv* 预印本 *arXiv:1708.04552*，2017年。[50] Neofytos Dimitriou 和 Ognjen Arandjelovi´c。序列归一化：拥抱较小的归一化样本量。信息，13(7)：337，2022年。[51] Liang Ding、Longyue Wang、Xuebo Liu、Derek F Wong、Dacheng Tao 和 Zhaopeng Tu。面向非自回归机器翻译的渐进式多粒度训练。载于计算语言学协会论文集精选，第2797–2803页，2021年。

[52] Xiaohan Ding、Honghao Chen、Xiangyu Zhang、Kaiqi Huang、Jungong Han 和 Guiguang Ding。重新参数化优化器，而非架构。*arXiv* 预印本 *arXiv:2205.15242*，2022年。[53] Alexey Dosovitskiy、Lucas Beyer、Alexander Kolesnikov、Dirk Weissenborn、Xiaohua Zhai、Thomas Unterthiner、Mostafa Dehghani、Matthias Minderer、Georg Heigold、Sylvain Gelly 等。一张图像胜过 16×16 个词：用于大规模图像识别的 Transformer。*arXiv* 预印本 *arXiv:2010.11929*，2020年。[54] John Duchi、Elad Hazan 和 Yoram Singer。用于在线学习和随机优化的自适应子梯度方法。机器学习研究期刊，12(7)，2011年。[55] Chen Dun、Cameron R Wolfe、Christopher M Jermaine 和 Anastasios Kyrillidis。ResIST：用于分布式训练的 ResNet 分层分解。载于 人工智能中的不确定性，第 610–620 页。PMLR，2022年。[56] Sergey Edunov、Myle Ott、Michael Auli 和 David Grangier。理解大规模反向翻译。*arXiv* 预印本 *arXiv:1808.09381*，2018年。[57] Jeffrey L Elman。神经网络中的学习与发展：从小规模开始的重要性。认知，48(1)：71–99，1993年。[58] FairScale 作者。FairScale：用于高性能和大规模训练的通用模块化 PyTorch 库。https://github.com/facebookresearch/fairscale，2021年。[59] Hongxiang Fan、Thomas Chau、Stylianos I Venieris、Royson Lee、Alexandros Kouris、Wayne Luk、Nicholas D Lane 和 Mohamed S Abdelfattah。通过硬件与算法协同设计实现适用于基于注意力的神经网络的可适配蝴蝶加速器。载于 *2022* 年第 *55* 届 *IEEE/ACM* 微体系结构国际研讨会（*MICRO*），第 599–615 页。IEEE，2022年。[60] William Fedus、Barret Zoph 和 Noam Shazeer。Switch Transformer：利用简单高效的稀疏性扩展至万亿参数模型。*J. Mach. Learn. Res*，23：1–40，2021年。[61]Vitaly Fedyunin。（测试版）PyTorch 中的 channels-last 内存格式，2022年。URL https://pytorch.org/tutorials/intermediate/memory_format_tutorial.html#beta-channels-last-memory-format-in-pytorch。[62] Steven Y Feng、Varun Gangal、Jason Wei、Sarath Chandar、Soroush Vosoughi、Teruko Mitamura 和 Eduard Hovy。自然语言处理数据增强方法综述。*arXiv* 预印本 *arXiv:2105.03075*，2021年。[63] Pierre Foret、Ariel Kleiner、Hossein Mobahi 和 Behnam Neyshabur。锐度感知最小化：高效提升泛化能力。*arXiv* 预印本 *arXiv:2010.01412*，2020年。[64] Jonathan Frankle 和 Michael Carbin。彩票券假设：寻找稀疏且可训练的神经网络。*arXiv* 预印本 *arXiv:1803.03635*，2018年。[65] Elias Frantar、Eldar Kurtic 和 Dan Alistarh。M-FAC：二阶信息的高效无矩阵近似。神经信息处理系统进展，34：14873–14886，2021年。[66] Shaoduo Gan、Xiangru Lian、Rui Wang、Jianbin Chang、Chengjun Liu、Hongmei Shi、Shengzhuo Zhang、Xianghong Li、Tengxu Sun、Jiawei Jiang、Binhang Yuan、Sen Yang、Ji Liu 和 Ce Zhang。Bagua：通过系统松弛扩展分布式学习，2021年。[67] Leo Gao、Stella Biderman、Sid Black、Laurence Golding、Travis Hoppe、Charles Foster、Jason Phang、Horace He、Anish Thite、Noa Nabeshima 等。Pile：用于语言建模的 800GB 多样化文本数据集。*arXiv* 预印本 *arXiv:2101.00027*，2020年。

[68] Jonas Geiping 和 Tom Goldstein。填鸭式训练：在一天内于单个 GPU 上训练语言模型。*arXiv* 预印本 *arXiv:2212.14034*，2022年。[69] Deepak Ghimire、Dayoung Kil 和 Seong-heum Kim。高效卷积神经网络与硬件加速综述。电子学，11(6)：945，2022年。[70] Negar Goli 和 Tor M Aamodt。ReSprop：复用稀疏化反向传播。载于 *IEEE/CVF*计算机视觉与模式识别会议论文集，第 1548–1558 页，2020年。[71]Noah Golmant、Nikita Vemuri、Zhewei Yao、Vladimir Feinberg、Amir Gholami、Kai Rothauge、Michael W Mahoney 和 Joseph Gonzalez。随机梯度下降中大批次计算低效问题。*arXiv* 预印本 *arXiv:1811.12941*，2018年。[72] Linyuan Gong、Di He、Zhuohan Li、Tao Qin、Liwei Wang 和 Tieyan Liu。通过渐进式堆叠高效训练 BERT。载于 国际机器学习会议，第 2337–2346 页。PMLR，2019年。[73] Priya Goyal、Piotr Doll´ar、Ross Girshick、Pieter Noordhuis、Lukasz Wesolowski、Aapo Kyrola、Andrew Tulloch、Yangqing Jia 和 Kaiming He。准确的大批量 SGD：一小时训练 ImageNet。*arXiv* 预印本 *arXiv:1706.02677*，2017年。[74] Priya Goyal、Mathilde Caron、Benjamin Lefaudeux、Min Xu、Pengchao Wang、Vivek Pai、Mannat Singh、Vitaliy Liptchinsky、Ishan Misra、Armand Joulin 等。开放环境中的视觉特征自监督预训练。*arXiv* 预印本 *arXiv:2103.01988*，2021年。[75] Xiaotao Gu、Liyuan Liu、Hongkun Yu、Jing Li、Chen Chen 和 Jiawei Han。渐进式 BERT 训练中的 Transformer 增长问题。*arXiv* 预印本 *arXiv:2010.12562*，2020年。[76] Demi Guo、Yoon Kim 和 Alexander M Rush。序列级混合样本数据增强。*arXiv* 预印本 *arXiv:2011.09039*，2020年。[77] Vineet Gupta、Tomer Koren 和 Yoram Singer。Shampoo：预条件随机张量优化。载于 国际机器学习会议，第 1842–1850 页。PMLR，2018年。[78] Richard HR Hahnloser、Rahul Sarpeshkar、Misha A Mahowald、Rodney J Douglas 和 H Sebastian Seung。数字选择与模拟放大共存于受皮层启发的硅电路中。自然，405(6789)：947–951，2000年。[79] Tengda Han、Weidi Xie 和 Andrew Zisserman。使用词元丢弃进行 Turbo 训练。*arXiv* 预印本 *arXiv:2210.04889*，2022年。[80] Xu Han、Zhengyan Zhang、Ning Ding、Yuxian Gu、Xiao Liu、Yuqi Huo、Jiezhong Qiu、Yuan Yao、Ao Zhang、Liang Zhang 等。预训练模型：过去、现在与未来。*AIOpen*，2：225–250，2021年。[81] Haoze He 和 Parijat Dube。通过非阻塞小批量加速并行随机梯度下降。*arXiv* 预印本 *arXiv:2211.00889*，2022年。[82] Kaiming He、Xiangyu Zhang、Shaoqing Ren 和 Jian 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。[83] Kaiming He、Xinlei Chen、Saining Xie、Yanghao Li、Piotr Doll´ar 和 Ross Girshick。掩码自编码器是可扩展的视觉学习器。载于 *IEEE/CVF*计算机视觉与模式识别会议论文集，第 16000–16009 页，2022年。

[84] Tong He、ZhiZhang、Hang Zhang、ZhongyueZhang、JunyuanXie和 Mu Li。使用卷积神经网络进行图像分类的实用技巧汇编。载于*IEEE/CVF*计算机视觉与模式识别会议论文集，第558–567页，2019年。[85] Xiaoxin He、FuzhaoXue、Xiaozhe Ren 和 Yang You。大规模深度学习优化：全面综述。*arXiv*预印本*arXiv:2111.00856*，2021年。[86] Jonathan Heek、Anselm Levskaya、Avital Oliver、Marvin Ritter、Bertrand Rondepierre、Andreas Steiner 和MarcvanZee。Flax：用于 JAX 的神经网络库及生态系统，2023年。网址http://github. com/google/flax。[87]DanHendrycks 和 Kevin Gimpel。高斯误差线性单元（GELU）。*arXiv*预印本*arXiv:1606.08415*，2016年。[88] Dan Hendrycks、Kimin Lee和 Mantas Mazeika。使用预训练可以提升模型鲁棒性和不确定性。载于国际机器学习会议，第2712–2721页。PMLR，2019年。[89]Dan Hendrycks、Norman Mu、Ekin D Cubuk、Barret Zoph、Justin Gilmer 和 BalajiLakshmi- narayanan。AugMix：一种提升鲁棒性和不确定性的简单数据处理方法。*arXiv*预印本*arXiv:1912.02781*，2019年。[90] Tom Hennigan、Trevor Cai、TamaraNorman 和 Igor Babuschkin。Haiku：JAX 的 Sonnet，2020年。网址http://github.com/deepmind/dm-haiku。[91] Daniel Ho、Eric Liang、XiChen、Ion Stoica 和PieterAbbeel。基于群体的数据增强：高效学习数据增强策略计划。载于国际机器学习会议，第2731–2741页。PMLR，2019年。[92] Elad Hoffer、ItayHubara 和DanielSoudry。训练更久，泛化更好：缩小神经网络大批量训练中的泛化差距。神经信息处理系统进展，30，2017年。[93] Jordan Hoffmann、Sebastian Borgeaud、Arthur Mensch、Elena Buchatskaya、Trevor Cai、ElizaRuther- ford、Diego de Las Casas、Lisa Anne Hendricks、Johannes Welbl、Aidan Clark 等。训练计算最优的大型语言模型。*arXiv*预印本*arXiv:2203.15556*，2022年。[94] Minui Hong、JinwooChoi 和 Gunhee Kim。StyleMix：分离内容与风格以增强数据增强。载于*IEEE/CVF*计算机视觉与模式识别会议论文集，第14862–14870页，2021年。[95]Jeremy Howard。Fastai——渐进式缩放。https://www.fast.ai/2018/04/30/ dawnbench-fastai/，2018年。[96] JeremyHoward和 Sylvain Gugger。Fastai：面向深度学习的分层 API。信息，11(2):108，2020年。[97] Weizhe Hua、ZihangDai、Hanxiao Liu和Quoc Le。线性时间内的 Transformer 质量。载于国机器学习会议，第9099–9117页。PMLR，2022年。[98] ShanshiHuang、HongwuJiang 和Shimeng Yu。面向硬件的存算一体加速器量化/映射策略。电子系统设计自动化 *ACM* 汇刊，2022年。[99] Xiao Shi Huang、FelipePerez、Jimmy Ba 和 Maksims Volkovs。通过更好的初始化改进 Transformer 优化。载于国际机器学习会议，第4475–4483页。PMLR，2020年。

[100] Yanping Huang、Youlong Cheng、Ankur Bapna、Orhan Firat、Dehao Chen、Mia Chen、HyoukJoong Lee、Jiquan Ngiam、Quoc V Le、Yonghui Wu 等。Gpipe：使用流水线并行高效训练巨型神经网络。神经信息处理系统进展，32，2019年。[101] Zhouyuan Huo、Bin Gu、Heng Huang 等。具有收敛保证的解耦并行反向传播。载于国际机器学习会议，第2098–2106页。PMLR，2018年。[102] Sergey Ioffe 和 Christian Szegedy。批量归一化：通过减少内部协变量偏移加速深度网络训练。载于国际机器学习会议，第448–456页。PMLR，2015年。[103] Pavel Izmailov、Dmitrii Podoprikhin、Timur Garipov、Dmitry Vetrov 和 Andrew Gordon Wilson。权重平均可带来更宽的最优点和更好的泛化。*arXiv* 预印本 *arXiv:1803.05407*，2018年。[104] Peter Izsak、Moshe Berchansky 和 Omer Levy。如何以学术预算训练 BERT。载于*2021*年自然语言处理实证方法会议论文集，第10644–10652页，2021年。[105] Sebastian Jaszczur、Aakanksha Chowdhery、Afroz Mohiuddin、Lukasz Kaiser、Wojciech Gajewski、Henryk Michalewski 和 Jonni Kanerva。扩展 Transformer 时，稀疏性已足够。神经信息处理系统进展，34：9895–9907，2021年。[106] Danlin Jia、Geng Yuan、Xue Lin 和 Ningfang Mi。一种可调节的数据加载器参数，可缩短分布式深度学习中的 GPU 空闲时间。载于*2022*年第*15*届*IEEE*云计算国际会议（*CLOUD*），第449–458页。IEEE，2022年。[107] Xianyan Jia、Shutao Song、Wei He、Yangzihao Wang、Haidong Rong、Feihu Zhou、Liqiang Xie、Zhenyu Guo、Yuanzhou Yang、Liwei Yu 等。采用混合精度的高可扩展深度学习训练系统：四分钟内训练 ImageNet。*arXiv* 预印本 *arXiv:1807.11205*，2018年。[108] Angela H Jiang、Daniel L-K Wong、Giulio Zhou、David G Andersen、Jeffrey Dean、Gregory R Ganger、Gauri Joshi、Michael Kaminksy、Michael Kozuch、Zachary C Lipton 等。聚焦最大输家以加速深度学习。*arXiv* 预印本 *arXiv:1910.00762*，2019年。[109] Jean Kaddour。别再浪费我的时间！通过最新权重平均，节省数天的 ImageNet 和 BERT 训练时间。载于 *Has it Trained Yet? NeurIPS 2022*年研讨会，2022年。网址 https://openreview.net/ forum?id=0 OrABUHZuz。[110] Jared Kaplan、Sam McCandlish、Tom Henighan、Tom B Brown、Benjamin Chess、Rewon Child、Scott Gray、Alec Radford、Jeffrey Wu 和 Dario Amodei。神经语言模型的缩放定律。*arXiv* 预印本 *arXiv:2001.08361*，2020年。[111] Tero Karras、Timo Aila、Samuli Laine 和 Jaakko Lehtinen。通过 GAN 渐进式增长提升质量、稳定性和多样性。*arXiv* 预印本 *arXiv:1710.10196*，2017年。[112] Angelos Katharopoulos 和 François Fleuret。用于深度神经网络训练的有偏重要性采样。*arXiv* 预印本 *arXiv:1706.00043*，2017年。[113] Angelos Katharopoulos 和 François Fleuret。并非所有样本都生而平等：使用重要性采样进行深度学习。载于国际机器学习会议，第2525–2534页。PMLR，2018年。[114] Kenji Kawaguchi 和 Haihao Lu。有序随机梯度下降：用于经验风险最小化的一种新型随机优化框架。载于国际人工智能与统计会议，第669–679页。PMLR，2020年。

[115] Nitish Shirish Keskar、Dheevatsa Mudigere、Jorge Nocedal、Mikhail Smelyanskiy 和 Ping Tak Peter Tang。深度学习的大批量训练：泛化差距与尖锐极小值。*arXiv* 预印本 *arXiv:1609.04836*，2016年。[116]Prannay Khosla、Piotr Teterwak、Chen Wang、Aaron Sarna、Yonglong Tian、Phillip Isola、Aaron Maschinot、Ce Liu 和 Dilip Krishnan。监督式对比学习。神经信息处理系统进展，33:18661–18673，2020年。[117] Jang-Hyun Kim、Wonho Choo、Hosan Jeong 和 Hyun Oh Song。Co-mixup：结合超模多样性的显著性引导联合混合。*arXiv* 预印本 *arXiv:2102.03065*，2021年。[118] Diederik P Kingma 和 Jimmy Ba。Adam：一种随机优化方法。*arXiv* 预印本 *arXiv:1412.6980*，2014年。[119] Nikita Kitaev、Łukasz Kaiser 和 Anselm Levskaya。Reformer：高效Transformer。*arXiv* 预印本 *arXiv:2001.04451*，2020年。[120] G¨unter Klambauer、Thomas Unterthiner、Andreas Mayr 和 Sepp Hochreiter。自归一化神经网络。神经信息处理系统进展，30，2017年。[121] Mustafa Taha Koc¸yi˘git、Timothy M Hospedales 和 Hakan Bilen。通过高效训练策略加速自监督学习。载于*IEEE/CVF*冬季计算机视觉应用会议论文集，第5654–5664页，2023年。[122] Zhenglun Kong、Haoyu Ma、Geng Yuan、Mengshu Sun、Yanyue Xie、Peiyan Dong、Xin Meng、Xuan Shen、Hao Tang、Minghai Qin 等。剥洋葱：通过分层约简数据冗余，实现高效的视觉Transformer训练。*arXiv* 预印本 *arXiv:2211.10801*，2022年。[123] Alex Krizhevsky、Ilya Sutskever 和 Geoffrey E Hinton。利用深度卷积神经网络进行ImageNet分类。*ACM*通信，60(6):84–90，2017年。[124]Siddharth Krishna Kumar。深度神经网络中的权重初始化。*arXiv* 预印本 *arXiv:1704.08863*，2017年。[125] Hugo Laurenc¸on、Lucile Saulnier、Thomas Wang、Christopher Akiki、Albert Villanova del Moral、Teven Le Scao、Leandro Von Werra、Chenghao Mou、Eduardo Gonz´alez Ponferrada、Huu Nguyen 等。BigScience ROOTS 语料库：一个 1.6 TB 的复合多语言数据集。载于第三十六届神经信息处理系统进展会议数据集与基准赛道，2022年。[126] Guillaume Leclerc、Andrew Ilyas、Logan Engstrom、Sung Min Park、Hadi Salman 和 Aleksander Madry。FFCV：通过消除数据瓶颈加速训练。2022年。[127] Katherine Lee、Daphne Ippolito、Andrew Nystrom、Chiyuan Zhang、Douglas Eck、Chris Callison-Burch 和 Nicholas Carlini。训练数据去重能让语言模型变得更好。*arXiv* 预印本 *arXiv:2107.06499*，2021年。[128] Benjamin Lefaudeux、Francisco Massa、Diana Liskovich、Wenhan Xiong、Vittorio Caggiano、Sean Naren、Min Xu、Jieru Hu、Marta Tintore、Susan Zhang、Patrick Labatut 和 Daniel Haziza。xFormers：一个模块化且易于修改的Transformer建模库。https://github.com/facebookresearch/ xformers，2022年。[129] Bei Li、Ziyang Wang、Hui Liu、Yufan Jiang、Quan Du、Tong Xiao、Huizhen Wang 和 Jingbo Zhu。神经机器翻译的由浅入深训练。*arXiv* 预印本 *arXiv:2010.03737*，2020年。[130] Changlin Li、Bohan Zhuang、Guangrun Wang、Xiaodan Liang、Xiaojun Chang 和 Yi Yang。用于高效训练视觉变换器的自动渐进式学习。载于*IEEE/CVF*计算机视觉与模式识别会议论文集，第12486–12496页，2022年。

[131] Conglong Li、Minjia Zhang 和 Yuxiong He。课程学习：一种用于高效且稳定的十亿规模 GPT模型预训练的正则化方法。*arXiv* 预印本*arXiv:2108.06084*，2021年。[132] Conglong Li、Zhewei Yao、Xiaoxia Wu、Minjia Zhang和 Yuxiong He。DeepSpeed 数据效率：通过高效的数据采样和路由提升深度学习模型质量与训练效率。*arXiv* 预印本*arXiv:2212.03597*，2022年。[133] ConglongLi、Minjia Zhang 和 YuxiongHe。稳定性与效率困境：研究用于训练 GPT 模型的序列长度预热。在神经信息处理系统进展，2022年。[134] Mengtian Li、ErsinYumer和Deva Ramanan。预算训练：在资源约束下重新思考深度神经网络训练。在国际学习表征会议，2019年。[135] Shenggui Li、Fuzhao Xue、Yongbin Li 和 Yang You。序列并行：实现 4D并行。在*arXiv* 预印本*arXiv:2105.13120*，2021年。[136] Shigang Li、KazukiOsawa和Torsten Hoefler。张量核心上的高效量化稀疏矩阵运算。*arXiv* 预印本 *arXiv:2209.06979*，2022年。[137] Tao Li、Zhehao Huang、Qinghua Tao、YingwenWu和 Xiaolin Huang。用于快速收敛和更佳泛化的可训练权重平均。*arXiv* 预印本*arXiv:2205.13104*，2022年。[138] Wenhao Li、Hong Liu、HaoTang、PichaoWang和LucVan Gool。MHFormer：用于三维人体姿态估计的多假设变换器。在*IEEE/CVF*计算机视觉与模式识别会议论文集，第 13147–13156页，2022年。[139] Yawei Li、Kai Zhang、JiezhangCao、RaduTimofte和Luc Van Gool。LocalViT：为视觉变换器引入局部性。*arXiv* 预印本*arXiv:2104.05707*，2021年。[140] YingzhouLi、Haizhao Yang、Eileen R Martin、Kenneth L Ho 和 LexingYing。蝴蝶因子分解。多尺度建模与仿真，13(2)：714–732，2015年。[141] ZhengLi、Soroush Ghodrati、Amir Yazdanbakhsh、HadiEsmaeilzadeh和 Mingu Kang。通过基于梯度学习的运行时剪枝加速注意力。在第*49* 届年度国际计算机体系结构研讨会论文集，第902–915 页，2022年。[142] Zhuohan Li、Eric Wallace、Sheng Shen、Kevin Lin、Kurt Keutzer、Dan Klein和 Joey Gonzalez。先训练大模型，再进行压缩：重新思考 Transformer 高效训练与推理的模型规模。在国际机器学习会议，第 5958–5968 页。PMLR，2020年。[143] Xingyu Liao、Peng Cheng、Tianhe Ren、Depeng Liang、Kai Dang、Yi Wang 和 Xiaoyu Xu。LiBai。https://github.com/Oneflow-Inc/libai，2021年。[144] Sungbin Lim、Ildoo Kim、TaesupKim、Chiheon Kim 和 Sungwoong Kim。快速自动增强。神经信息处理系统进展，32，2019年。[145] Tianyang Lin、YuxinWang、XiangyangLiu 和 Xipeng Qiu。Transformer 综述。*AI Open*，2022年。[146] Jihao Liu、Boxiao Liu、Hang Zhou、Hongsheng Li 和 Yu Liu。TokenMix：重新思考视觉 Transformer中用于数据增强的图像混合。在计算机视觉——*ECCV 2022*：第 *17* 届欧洲会议，以色列特拉维夫，*2022*年*10* 月 *23–27* 日，论文集，第二十六部分，第 455–471 页。施普林格，2022年。[147] Jing Liu、Zizheng Pan、Haoyu He、JianfeiCai和 BohanZhuang。EcoFormer：具有线性复杂度的节能注意力。*arXiv* 预印本*arXiv:2209.09004*，2022年。[148] Rui Liu和Barzan Mozafari。面向大批次优化的通信高效的分布式学习。在国际机器学习会议，第 13925–13946 页。PMLR，2022年。

[149] Yinhan Liu、Myle Ott、Naman Goyal、Jingfei Du、Mandar Joshi、Danqi Chen、Omer Levy、Mike Lewis、Luke Zettlemoyer 和 Veselin Stoyanov。RoBERTa：一种经过稳健优化的 BERT 预训练方法。*arXiv* 预印本 *arXiv:1907.11692*，2019年。[150] Ze Liu、Yutong Lin、Yue Cao、Han Hu、Yixuan Wei、Zheng Zhang、Stephen Lin 和 Baining Guo。Swin Transformer：使用移位窗口的分层视觉变换器。载于*IEEE/CVF*国际计算机视觉会议论文集，第10012–10022页，2021年。[151] Zirui Liu、Shengyuan Chen、Kaixiong Zhou、Daochen Zha、Xiao Huang 和 Xia Hu。RSC：通过随机稀疏计算加速图神经网络训练。*arXiv* 预印本 *arXiv:2210.10737*，2022年。[152] Jonathan Long、Evan Shelhamer 和 Trevor Darrell。用于语义分割的全卷积网络。载于*IEEE*计算机视觉与模式识别会议论文集，第3431–3440页，2015年。[153] Ilya Loshchilov 和 Frank Hutter。SGDR：采用热重启的随机梯度下降。*arXiv* 预印本 *arXiv:1608.03983*，2016年。[154]Ilya Loshchilov 和 Frank Hutter。解耦权重衰减正则化。*arXiv* 预印本 *arXiv:1711.05101*，2017年。[155] Yucheng Lu、Conglong Li、Minjia Zhang、Christopher De Sa 和 Yuxiong He。通过 0/1 Adam 最大限度地提升大规模训练的通信效率。*arXiv* 预印本 *arXiv:2202.06009*，2022年。[156] Sangkug Lym、Esha Choukse、Siavash Zangeneh、Wei Wen、Sujay Sanghavi 和 Mattan Erez。Prune-train：通过动态稀疏模型重配置实现快速神经网络训练。载于国际高性能计算、网络、存储与分析会议论文集，第1–13页，2019年。[157] Haoyu Ma、Chengming Zhang、lizhi xiang、Xiaolong Ma、Geng Yuan、Wenkai Zhang、Shiwei Liu、Tianlong Chen、Dingwen Tao、Yanzhi Wang、Zhangyang Wang 和 Xiaohui Xie。HRBP：面向硬件的重组方法，助力稀疏训练中的块级剪枝，2023年。网址：https://openreview. net/forum?id= OSS-yWzE9Yu。[158] Shuming Ma、Hongyu Wang、Shaohan Huang、Wenhui Wang、Zewen Chi、Li Dong、Alon Benhaim、Barun Patra、Vishrav Chaudhary、Xia Song 和 Furu Wei。TorchScale：大规模 Transformer。*CoRR*，abs/2211.13184，2022年。[159] Xuezhe Ma、Xiang Kong、Sinong Wang、Chunting Zhou、Jonathan May、Hao Ma 和 Luke Zettlemoyer。Luna：线性统一嵌套注意力。神经信息处理系统进展，34：2441–2453，2021年。[160] Yanjun Ma、Dianhai Yu、Tian Wu 和 Haifeng Wang。飞桨：源自工业实践的开源深度学习平台。数据与计算前沿，1(1)：105–115，2019年。[161] Partha Maji、Andrew Mundy、Ganesh Dasika、Jesse Beu、Matthew Mattina 和 Robert Mullins。在广泛使用的移动CPU上高效实现 Winograd 或 Cook-Toom 卷积核。载于*2019*年第二届嵌入式应用能效机器学习与认知计算研讨会（*EMC2*），第1–5页，2019年。doi：10.1109/EMC249363.2019.00008。[162] Ilia Markov、Hamidreza Ramezani 和 Dan Alistarh。CGX 项目：在商用 GPU 上实现可扩展的深度学习。*arXiv* 预印本 *arXiv:2111.08617*，2021年。[163] James Martens 和 Roger Grosse。利用克罗内克因子近似曲率优化神经网络。载于国际机器学习会议，第2408–2417页。PMLR，2015年。

[164] Sam McCandlish、Jared Kaplan、Dario Amodei 和 OpenAI Dota 团队。大批次训练的经验模型。*arXiv* 预印本 *arXiv:1812.06162*，2018年。[165] Chenlin Meng、Linqi Zhou、Kristy Choi、Tri Dao 和 Stefano Ermon。Butterflyflow：利用蝴蝶矩阵构建可逆层。载于国际机器学习会议，第15360–15375页。PMLR，2022年。[166] Paulius Micikevicius、Sharan Narang、Jonah Alben、Gregory Diamos、Erich Elsen、David Garcia、Boris Ginsburg、Michael Houston、Oleksii Kuchaiev、Ganesh Venkatesh 等。混合精度训练。*arXiv* 预印本 *arXiv:1710.03740*，2017年。[167] S¨oren Mindermann、Jan M Brauner、Muhammed T Razzak、Mrinank Sharma、Andreas Kirsch、Winnie Xu、Benedikt H¨oltgen、Aidan N Gomez、Adrien Morisot、Sebastian Farquhar 等。优先训练那些可学习、值得学习且尚未学会的样本点。载于国际机器学习会议，第15630–15649页。PMLR，2022年。[168] MindSpore-AI。MindSpore，2021年。网址：https://github.com/mindspore-ai/mindspore。[169] Dmytro Mishkin 和 Jiri Matas。你所需要的只是良好的初始化。*arXiv* 预印本 *arXiv:1511.06422*，2015年。[170]Asit Mishra、Jorge Albericio Latorre、Jeff Pool、Darko Stosic、Dusan Stosic、Ganesh Venkatesh、Chong Yu 和 Paulius Micikevicius。加速稀疏深度神经网络。*arXiv* 预印本 *arXiv:2104.08378*，2021年。[171] Rafael M¨uller、Simon Kornblith 和 Geoffrey E Hinton。标签平滑何时有帮助？神经信息处理系统进展，32，2019年。[172] Koichi Nagatsuka、Clifford Broni-Bediako 和 Masayasu Atsumi。通过增大输入文本块大小，利用课程学习对 BERT 进行预训练。载于自然语言处理最新进展，2021年。[173] Deepak Narayanan、Aaron Harlap、Amar Phanishayee、Vivek Seshadri、Nikhil R Devanur、Gregory R Ganger、Phillip B Gibbons 和 Matei Zaharia。Pipedream：用于 DNN 训练的通用化流水线并行。载于第*27*届 *ACM* 操作系统原理研讨会论文集，第1–15页，2019年。[174] Deepak Narayanan、Mohammad Shoeybi、Jared Casper、Patrick LeGresley、Mostofa Patwary、Vijay Ko- rthikanti、Dmitri Vainbrand、Prethvi Kashinkunti、Julie Bernauer、Bryan Catanzaro 等。利用 Megatron-LM 在 GPU 集群上高效进行大规模语言模型训练。载于国际高性能计算、网络、存储与分析会议论文集，第1–15页，2021年。[175] Tam Minh Nguyen、Tan Minh Nguyen、Dung DD Le、Duy Khuong Nguyen、Viet-Anh Tran、Richard Baraniuk、Nhat Ho 和 Stanley Osher。利用概率注意力键改进 Transformer。载于国际机器学习会议，第16595–16621页。PMLR，2022年。[176] Renkun Ni、Ping-yeh Chiang、Jonas Geiping、Micah Goldblum、Andrew Gordon Wilson 和 Tom Gold- stein。K-SAM：以 SGD 的速度进行锐度感知最小化。*arXiv* 预印本 *arXiv:2210.12864*，2022年。[177] Zanlin Ni、Yulin Wang、Jiangwei Yu、Haojun Jiang、Yue Cao 和 Gao Huang。深度模型组装。*arXiv* 预印本 *arXiv:2212.04129*，2022年。[178] 英伟达。Apex。https://github.com/NVIDIA/apex，2022年。[179] 开放机器学习系统社区。机器学习系统：设计与实现。GitHub，2022年。网址：https://github.com/openmlsys/openmlsys-zh。

[180] OpenBMB。BMTrain，2021年。网址 https://github.com/OpenBMB/BMTrain。[181] Myle Ott、Sergey Edunov、Alexei Baevski、Angela Fan、Sam Gross、Nathan Ng、David Grangier 和 Michael Auli。fairseq：一种快速、可扩展的序列建模工具包。载于*NAACL-HLT 2019*会议论文集：演示，2019年。[182] Bowen Pan、Rameswar Panda、Yifan Jiang、Zhangyang Wang、Rogerio Feris 和 Aude Oliva。IA-RED：面向可解释性的视觉Transformer冗余缩减。神经信息处理系统进展，34：24898–24911，2021年。[183] Rui Pan、Haishan Ye 和 Tong Zhang。Eigencurve：针对海森矩阵谱偏斜的二次目标函数，SGD的最优学习率计划。载于国际学习表征会议，2021年。[184] Rui Pan、Shizhe Diao、Jianlin Chen 和 Tong Zhang。ExtremeBERT：加速定制BERT预训练的工具包。2022年。网址 https://arxiv.org/abs/2211.17201。[185] Xuran Pan、Xuan Jin、Yuan He、Shiji Song、Gao Huang 等。视觉变换器的预算训练。载于第十一届国际学习表征会议。[186] Zizheng Pan、Jianfei Cai 和 Bohan Zhuang。采用HiLo注意力的快速视觉Transformer。*arXiv*预印本 *arXiv:2205.13213*，2022年。[187] Ziqi Pang、Zhichao Li 和 Naiyan Wang。SimpleTrack：理解并重新思考三维多目标跟踪。载于计算机视觉——*ECCV 2022*研讨会：以色列特拉维夫，*2022*年*10*月*23*日至*27*日，论文集，第一部分，第680–696页。施普林格，2023年。[188] Aarthi Paramasivam 和 S Jaya Nirmala。基于文本蕴含的问答综述。《沙特国王大学学报：计算机与信息科学》，34(10)：9644–9653，2022年。[189] J Gregory Pauloski、Qi Huang、Lei Huang、Shivaram Venkataraman、Kyle Chard、Ian Foster 和 Zhao Zhang。Kaisa：一种用于深度神经网络的自适应二阶优化器框架。载于国际高性能计算、网络、存储与分析会议论文集，第1–14页，2021年。[190] Hao Peng、Nikolaos Pappas、Dani Yogatama、Roy Schwartz、Noah A Smith 和 Lingpeng Kong。随机特征注意力。*arXiv*预印本 *arXiv:2103.02143*，2021年。[191] Ofir Press、Noah A Smith 和 Mike Lewis。Shortformer：通过缩短输入改进语言建模。载于计算语言学协会第*59*届年会暨自然语言处理国际联合会议第*11*届会议论文集（第一卷：长篇论文），第5493–5505页，2021年。[192] Heli Qi、Sashi Novitasari、Andros Tjandra、Sakriani Sakti 和 Satoshi Nakamura。Speechain：用于大规模机器语音链的语音工具包。*arXiv*预印本 *arXiv:2301.02966*，2023年。[193] Zhen Qin、XiaoDong Han、Weixuan Sun、Dongxu Li、Lingpeng Kong、Nick Barnes 和 Yiran Zhong。线性Transformer中的魔鬼。*arXiv*预印本 *arXiv:2210.10340*，2022年。[194] Xipeng Qiu、Tianxiang Sun、Yige Xu、Yunfan Shao、Ning Dai 和 Xuanjing Huang。自然语言处理预训练模型综述。《中国科学：技术科学》，63(10)：1872–1897，2020年。[195] Markus N Rabe 和 Charles Staats。自注意力不需要 $o(n^2)$内存。*arXiv*预印本 *arXiv:2112.05682*，2021年。[196] Alec Radford、Karthik Narasimhan、Tim Salimans、Ilya Sutskever 等。通过生成式预训练提升语言理解能力。2018年。

[197] Alec Radford、Jeffrey Wu、Rewon Child、David Luan、Dario Amodei、Ilya Sutskever 等。语言模型是无监督多任务学习器。*OpenAI*博客，1(8)：9，2019年。[198] Alec Radford、Jong Wook Kim、Chris Hallacy、Aditya Ramesh、Gabriel Goh、Sandhini Agarwal、Girish Sastry、Amanda Askell、Pamela Mishkin、Jack Clark 等。通过自然语言监督学习可迁移视觉模型。载于国际机器学习会议，第8748–8763页。PMLR，2021年。[199] Jack W Rae、Sebastian Borgeaud、Trevor Cai、Katie Millican、Jordan Hoffmann、Francis Song、John Aslanides、Sarah Henderson、Roman Ring、Susannah Young 等。语言模型扩展：Gopher训练方法、分析与见解。*arXiv*预印本 *arXiv:2112.11446*，2021年。[200] Colin Raffel、Noam Shazeer、Adam Roberts、Katherine Lee、Sharan Narang、Michael Matena、Yanqi Zhou、Wei Li 和 Peter J Liu。通过统一的文本到文本转换器探索迁移学习的极限。《机器学习研究期刊》，21(1)：5485–5551，2020年。[201] Maithra Raghu、Justin Gilmer、Jason Yosinski 和 Jascha Sohl-Dickstein。SVCCA：用于深度学习动态与可解释性的奇异向量典型相关分析。神经信息处理系统进展，30，2017年。[202] Samyam Rajbhandari、Jeff Rasley、Olatunji Ruwase 和 Yuxiong He。ZeRO：面向万亿参数模型训练的内存优化。载于*SC20*：国际高性能计算、网络、存储与分析会议，第1–16页。IEEE，2020年。[203] Samyam Rajbhandari、Olatunji Ruwase、Jeff Rasley、Shaden Smith 和 Yuxiong He。Zero-infinity：突破极大规模深度学习的GPU内存墙。载于国际高性能计算、网络、存储与分析会议论文集，第1–14页，2021年。[204] Aditya Ramesh、Mikhail Pavlov、Gabriel Goh、Scott Gray、Chelsea Voss、Alec Radford、Mark Chen 和 Ilya Sutskever。零样本文本到图像生成。载于国际机器学习会议，第8821–8831页。PMLR，2021年。[205] Jeff Rasley、Samyam Rajbhandari、Olatunji Ruwase 和 Yuxiong He。DeepSpeed：系统优化使超过1000亿参数的深度学习模型训练成为可能。载于第*26*届*ACM SIGKDD*知识发现与数据挖掘国际会议论文集，第3505–3506页，2020年。[206] Ray Project。Ray：面向新兴人工智能应用的分布式框架。https://github.com/ ray-project/ray，2023年。[207] Joseph Redmon、Santosh Divvala、Ross Girshick 和 Ali Farhadi。You Only Look Once：统一的实时目标检测。载于*IEEE*计算机视觉与模式识别会议论文集，第779–788页，2016年。[208] Hongyu Ren、Hanjun Dai、Zihang Dai、Mengjiao Yang、Jure Leskovec、Dale Schuurmans 和 Bo Dai。Combiner：稀疏计算成本的全注意力转换器。神经信息处理系统进展，34：22470–22482，2021年。[209] Jie Ren、Samyam Rajbhandari、Reza Yazdani Aminabadi、Olatunji Ruwase、Shuangyan Yang、Minjia Zhang、Dong Li 和 Yuxiong He。ZeRO-Offload：让十亿级模型训练普惠化。载于*USENIX*年度技术会议，第551–564页，2021年。[210] Herbert Robbins 和 Sutton Monro。一种随机逼近法。《数学统计年刊》，第400–407页，1951年。

[211] Anna Rogers、MattGardner 和Isabelle Augenstein。问答数据集大爆发：用于问答和阅读理解的自然语言处理资源分类体系。 *ACM*计算调查，55(10):1– 45，2023年。[212] Olga Russakovsky、Jia Deng、Hao Su、JonathanKrause、Sanjeev Satheesh、Sean Ma、Zhiheng Huang、Andrej Karpathy、Aditya Khosla、MichaelBernstein 等。ImageNet 大规模视觉识别挑战赛。国际计算机视觉期刊，115:211–252，2015年。[213] SungjuRyu、Hyungjun Kim、Wooseok Yi、Eunhwan Kim、Yulhwa Kim、TaesuKim 和Jae-JoonKim。Bitblade：面向量化神经网络的节能可变比特精度硬件加速器。*IEEE*固态电路期刊，57(6):1924–1935，2022年。[214] Andrew M Saxe、JamesLMcClelland和Surya Ganguli。深度线性神经网络学习非线性动力学的精确解。 *arXiv*预印本*arXiv:1312.6120*，2013年。[215] Teven LeScao、Thomas Wang、Daniel Hesslow、Lucile Saulnier、Stas Bekman、M Saiful Bari、Stella Bideman、Hady Elsahar、Niklas Muennighoff、Jason Phang 等。如果有一百万个GPU小时，应训练什么语言模型？ *arXiv*预印本*arXiv:2210.15424*，2022年。[216] Damien Scieur、Alexandred'Aspremont 和 FrancisBach。正则化非线性加速。 神经信息处理系统进展，29，2016年。[217] Damien Scieur、Francis Bach和Alexandre d'Aspremont。随机算法的非线性加速。 神经信息处理系统进展，30，2017年。[218] Frank Seide、Hao Fu、Jasha Droppo、GangLi和Dong Yu。1位随机梯度下降及其在数据并行分布式语音深度神经网络训练中的应用。在 国际语音通信协会第十五届年度会议，2014年。[219] Rico Sennrich、BarryHaddow和 Alexandra Birch。利用单语数据改进神经机器翻译模型。 *arXiv*预印本*arXiv:1511.06709*，2015年。[220] Alexander Sergeev 和 Mike DelBalso。Horovod：TensorFlow 中快速简便的分布式深度学习。 *arXiv*预印本*arXiv:1802.05799*，2018年。[221] Noam Shazeer。GLU 变体改进 Transformer。 *arXiv*预印本*arXiv:2002.05202*，2020年。[222] Mohammad Shoeybi、Mostofa Patwary、Raul Puri、Patrick LeGresley、Jared Casper和 Bryan Catan- zaro。Megatron-LM：使用模型并行训练数十亿参数的语言模型。 *arXiv*预印本*arXiv:1909.08053*，2019年。[223] Karen Simonyan和Andrew Zisserman。用于大规模图像识别的超深度卷积网络。 *arXiv*预印本*arXiv:1409.1556*，2014年。[224] Dalwinder Singh和 Birmohan Singh。特征级归一化：一种归一化数据的有效方法。模式识别，122:108307，2022年。[225] Leslie N Smith。用于训练神经网络的周期性学习率。在 *2017*年*IEEE*冬季计算机视觉应用会议（*WACV*），第464–472页。IEEE，2017年。[226] LeslieN Smith 和Nicholay Topin。超收敛：使用较大学习率极快地训练神经网络。在 用于多域操作应用的人工智能和机器学习，第11006卷，第369–386页。SPIE，2019年。[227] Leslie N Smith、Emily MHand 和 Timothy Doster。逐步引入层以训练超深度神经网络。在 *IEEE*计算机视觉与模式识别会议论文集，第4763–4771页，2016年。

[228] Samuel L Smith、Pieter-Jan Kindermans、Chris Ying 和 Quoc V Le。不要降低学习率，而要增大批量大小。*arXiv* 预印本 *arXiv:1711.00489*，2017年。[229] Shaden Smith、Mostofa Patwary、Brandon Norick、Patrick LeGresley、Samyam Rajbhandari、Jared Casper、Zhun Liu、Shrimai Prabhumoye、George Zerveas、Vijay Korthikanti 等。使用 DeepSpeed 和 Megatron 训练 Megatron-Turing NLG 530B：大型生成式语言模型。*arXiv* 预印本 *arXiv:2201.11990*，2022年。[230] Zhuoran Song、Yihong Xu、Han Li、Naifeng Jing、Xiaoyao Liang 和 Li Jiang。通过探索适用于 GPGPU 的稀疏性来加速深度神经网络训练。*arXiv* 预印本 *arXiv:2203.05705*，2022年。[231] Andreas Steiner、Alexander Kolesnikov、Xiaohua Zhai、Ross Wightman、Jakob Uszkoreit 和 Lucas Beyer。如何训练你的视觉变换器？视觉Transformer中的数据、数据增强与正则化。*arXiv* 预印本 *arXiv:2106.10270*，2021年。[232] Pedro Javier Ortiz Su´arez、Laurent Romary 和 Benoˆıt Sagot。面向中等资源语言的上下文化词嵌入单语方法。*arXiv* 预印本 *arXiv:2006.06202*，2020年。[233] Shivakanth Sujit、Somjit Nath、Pedro HM Braga 和 Samira Ebrahimi Kahou。利用可约损失在强化学习中优先选择样本。*arXiv* 预印本 *arXiv:2208.10483*，2022年。[234] Chen Sun、Abhinav Shrivastava、Saurabh Singh 和 Abhinav Gupta。重新审视深度学习时代数据令人难以置信的有效性。载于*IEEE*国际计算机视觉会议论文集，第843–852页，2017年。[235] Yu Sun、Shuohuan Wang、Yukun Li、Shikun Feng、Xuyi Chen、Han Zhang、Xin Tian、Danxiang Zhu、Hao Tian 和 Hua Wu。文心：通过知识整合增强表征。*arXiv* 预印本 *arXiv:1904.09223*，2019年。[236] David Sussillo 和 LF Abbott。用于训练极深前馈网络的随机游走初始化。*arXiv* 预印本 *arXiv:1412.6558*，2014 年。[237] Ilya Sutskever、James Martens、George Dahl 和 Geoffrey Hinton。论初始化和动量在深度学习中的重要性。载于国际机器学习会议，第1139–1147页。PMLR，2013年。[238] Ivan Svogor、Christian Eichenberger、Markus Spanring、Moritz Neun 和 Michael Kopp。针对高延迟存储对 PyTorch 数据加载器进行性能分析与改进：技术报告。*arXiv* 预印本 *arXiv:2211.04908*，2022年。[239] Christian Szegedy、Wei Liu、Yangqing Jia、Pierre Sermanet、Scott Reed、Dragomir Anguelov、Dumitru Erhan、Vincent Vanhoucke 和 Andrew Rabinovich。利用卷积层更深入地探索。载于*IEEE*计算机视觉与模式识别会议论文集，第1–9页，2015年。[240] Christian Szegedy、Vincent Vanhoucke、Sergey Ioffe、Jon Shlens 和 Zbigniew Wojna。重新思考面向计算机视觉的 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。[241] Christian Szegedy、Sergey Ioffe、Vincent Vanhoucke 和 Alexander Alemi。Inception-v4、Inception-ResNet以及残差连接对学习的影响。载于*AAAI*人工智能会议论文集，第31卷，2017年。[242] Cheng Tai、Tong Xiao、Yi Zhang、Xiaogang Wang 等。采用低秩正则化的卷积神经网络。*arXiv* 预印本 *arXiv:1511.06067*，2015年。[243] Hao Tan、Le Wang、Huan Zhang、Junjian Zhang、Muhammad Shafiq 和 Zhaoquan Gu。说话人识别系统的对抗攻击与防御策略：综述。电子学，11(14):2183，2022年。

[244] Mingxing Tan 和 Quoc Le。Efficientnetv2：更小的模型和更快的训练。载于国际机器学习会议，第 10096–10106 页。PMLR，2021年。[245] Hanlin Tang、Shaoduo Gan、Ammar Ahmad Awan、Samyam Rajbhandari、Conglong Li、Xiangru Lian、Ji Liu、Ce Zhang 和 Yuxiong He。1-bit Adam：以 Adam 的收敛速度实现高通信效率的大规模训练。载于国际机器学习会议，第 10118–10129 页。PMLR，2021年。[246] Yi Tay、Mostafa Dehghani、Dara Bahri 和 Donald Metzler。高效 Transformer：综述。*ACM* 计算调查，55(6):1–28，2022年。[247] Mosaic ML 团队。Composer。https://github.com/mosaicml/composer/，2021年。[248] Alan D. Thompson。我的人工智能里有什么？对用于训练 GPT-1、GPT-2、GPT-3、GPT-NeoX-20B、Megatron-11B、MT-NLG 和 Gopher 的数据集进行全面分析，2022年。网址：https://lifearchitect.ai/ whats-in-my-ai/。[249] Tijmen Tieleman、Geoffrey Hinton 等。第 6.5 讲——RMSprop：将梯度除以其近期幅度的运行平均值。*COURSERA*：机器学习神经网络，4(2):26–31，2012年。[250] Ilya O Tolstikhin、Neil Houlsby、Alexander Kolesnikov、Lucas Beyer、Xiaohua Zhai、Thomas Unterthiner、Jessica Yung、Andreas Steiner、Daniel Keysers、Jakob Uszkoreit 等。MLP-Mixer：用于视觉任务的全多层感知机架构。神经信息处理系统进展，34:24261–24272，2021年。[251] Hugo Touvron、Andrea Vedaldi、Matthijs Douze 和 Herv´e J´egou。修正训练与测试分辨率之间的差异。神经信息处理系统进展，32，2019年。[252] Hugo Touvron、Matthieu Cord、Matthijs Douze、Francisco Massa、Alexandre Sablayrolles 和 Herv´e J´egou。通过注意力进行数据高效的图像变换器训练与蒸馏。载于国际机器学习会议，第 10347–10357 页。PMLR，2021年。[253] Hugo Touvron、Thibaut Lavril、Gautier Izacard、Xavier Martinet、Marie-Anne Lachaux、Timoth´ee Lacroix、Baptiste Rozi`ere、Naman Goyal、Eric Hambro、Faisal Azhar 等。LLaMA：开放且高效的基础语言模型。*arXiv* 预印本 *arXiv:2302.13971*，2023年。[254] Marcos Treviso、Tianchu Ji、Ji-Ung Lee、Betty van Aken、Qingqing Cao、Manuel R Ciosici、Michael Hassid、Kenneth Heafield、Sara Hooker、Pedro H Martins 等。自然语言处理的高效方法：综述。*arXiv* 预印本 *arXiv:2209.00099*，2022年。[255]AFM Uddin、Mst Monira、Wheemyung Shin、TaeChoong Chung、Sung-Ho Bae 等。SaliencyMix：一种以显著性为引导的数据增强策略，可实现更好的正则化。*arXiv* 预印本 *arXiv:2006.01791*，2020年。[256] Ashish Vaswani、Noam Shazeer、Niki Parmar、Jakob Uszkoreit、Llion Jones、Aidan N Gomez、Łukasz Kaiser 和 Illia Polosukhin。注意力就是你所需的一切。神经信息处理系统进展，30，2017年。[257] Vikas Verma、Alex Lamb、Christopher Beckham、Amir Najafi、Ioannis Mitliagkas、David Lopez-Paz 和 Yoshua Bengio。流形混合：通过插值隐藏状态获得更好的表征。载于国际机器学习会议，第 6438–6447 页。PMLR，2019年。[258] Homer F Walker 和 Peng Ni。固定点迭代的 Anderson 加速法。*SIAM* 数值分析期刊，49(4):1715–1735，2011年。[259] Ben Wang。Mesh-Transformer-JAX：使用 JAX 实现 Transformer 语言模型的模型并行。https://github.com/kingoflolz/mesh-transformer-jax，2021年5月。

[260] Boxiang Wang、Qifan Xu、Zhengda Bian 和 Yang You。二维半分布式模型训练。*arXiv* 电子预印本，页码 arXiv–2105，2021年。[261] Guangcong Wang、Xiaohua Xie、Jianhuang Lai 和 Jiaxuan Zhuo。深度渐进学习。载于*IEEE*国际计算机视觉会议论文集，第2812–2820页，2017年。[262] Kuan Wang、Zhijian Liu、Yujun Lin、Ji Lin 和 Song Han。HAQ：结合混合精度的硬件感知自动量化。载于*IEEE/CVF*计算机视觉与模式识别会议论文集，第8612–8620页，2019年。[263] Shibo Wang 和 Pankaj Kanwar。bfloat16：云端TPU实现高性能的秘诀——Google Cloud 博客，2020年。网址：https://cloud.google.com/blog/products/ai-machine-learning/bfloat16-the-secret-to-high-performance-on-cloud-tpus。[264] Sinong Wang、Belinda Z Li、Madian Khabsa、Han Fang 和 Hao Ma。Linformer：具有线性复杂度的自注意力。*arXiv* 预印本 *arXiv:2006.04768*，2020年。[265]Wenhui Wang、Hangbo Bao、Li Dong、Johan Bjorck、Zhiliang Peng、Qiang Liu、Kriti Aggarwal、Owais Khan Mohammed、Saksham Singhal、Subhojit Som 等。图像即外语：面向所有视觉和视觉语言任务的 BEiT 预训练。*arXiv* 预印本 *arXiv:2208.10442*，2022年。[266] William Yang Wang 和 Diyi Yang。这太恼人了！！！：一种基于词汇和框架语义嵌入的数据增强方法，用于通过# petpeeve 推文自动分类恼人行为。载于*2015*年自然语言处理经验方法会议论文集，第2557–2563页，2015年。[267] Xiaohui Wang、Ying Xiong、Xian Qian、Yang Wei、Lei Li 和 Mingxuan Wang。LightSeq2：在 GPU 上加速训练基于Transformer的模型。*arXiv* 预印本 *arXiv:2110.05722*，2021年。[268] Xiaohui Wang、Ying Xiong、Yang Wei、Mingxuan Wang 和 Lei Li。LightSeq：面向 Transformer 的高性能推理库。载于*2021*年北美计算语言学协会分会会议论文集：人类语言技术：工业论文（*NAACL-HLT*），第113–120页。计算语言学协会，2021年6月。[269] Yulin Wang、Yang Yue、Rui Lu、Tianjiao Liu、Zhao Zhong、Shiji Song 和 Gao Huang。Efficient-train：探索用于训练视觉骨干网络的通用课程学习。*arXiv* 预印本 *arXiv:2211.09703*，2022年。[270] Zi Wang、Chen Qian、Di Guo、Hongwei Sun、Rushuai Li、Bo Zhao 和 Xiaobo Qu。用于加速磁共振成像的一维深度低秩稀疏网络。*IEEE*医学影像汇刊，42(1)：79–90，2022年。[271] Jason Wei 和 Kai Zou。EDA：用于提升文本分类任务性能的简易数据增强技术。*arXiv* 预印本 *arXiv:1901.11196*，2019年。[272] Jason Wei、Yi Tay、Rishi Bommasani、Colin Raffel、Barret Zoph、Sebastian Borgeaud、Dani Yogatama、Maarten Bosma、Denny Zhou、Donald Metzler 等。大型语言模型的涌现能力。*arXiv* 预印本 *arXiv:2206.07682*，2022年。[273] Lilian Weng。Transformer家族2.0版。*lilianweng.github.io*，2023年1月。网址：https://lilianweng.github.io/posts/2023-01-27-the-transformer-family-v2/。[274] Chuhan Wu、Fangzhao Wu、Tao Qi、Yongfeng Huang 和 Xing Xie。Fastformer：加性注意力可能就是你所需要的一切。*arXiv* 预印本 *arXiv:2108.09084*，2021年。[275] Xiaoxia Wu、Ethan Dyer 和 Behnam Neyshabur。课程何时有效？载于国际学习表征会议，2021年。网址：https://openreview.net/forum?id=tW4QEInpni。

[276] HaifengXia 和 Zhengming Ding。基于结构知识的跨域协同归一化。在人工智能*AAAI*会议论文集，第 36 卷，第 2777–2785 页，2022年。[277] Xueli Xiao、ThosiniBamunu Mudiyanselage、ChunyanJi、Jie Hu 和 Yi Pan。通过智能冻结层实现快速深度学习训练。在*2019*年物联网国际会议（*iThings*）、*IEEE* 绿色计算与通信会议（*GreenCom*）、*IEEE* 网络、物理与社会计算会议（*CPSCom*）以及*IEEE*智能数据会议（*SmartData*），第 1225–1232 页。IEEE，2019年。[278] Jian Xie、Jingwei Xu、Guochang Wang、Yuan Yao、Zenan Li、ChunCao 和 Hanghang Tong。具有共享数据准备功能的深度学习数据加载器。在神经信息处理系统进展，2022年。[279] Sang Michael Xie、Shibani Santurkar、Tengyu Ma 和Percy Liang。通过重要性重采样为语言模型进行数据选择。*arXiv* 预印本 *arXiv:2302.03169*，2023年。[280] Xingyu Xie、Pan Zhou、HuanLi、ZhouchenLin和Shuicheng Yan。Adan：用于更快优化深度模型的自适应 Nesterov 动量算法，2022年。URL https://arxiv.org/abs/2208.06677。[281] Yunyang Xiong、Zhanpeng Zeng、Rudrasis Chakraborty、Mingxing Tan、GlennFung、Yin Li 和 Vikas Singh。Nystr¨omformer：一种基于 Nystr¨om 的近似自注意力算法。在人工智能 *AAAI* 会议论文集，第35 卷，第 14138–14148 页，2021年。[282] Qifan Xu、Shenggui Li、ChaoyuGong 和Yang You。一种用于训练超大型深度学习模型的高效二维方法。*arXiv* 预印本*arXiv:2104.05343*，2021年。[283] Yufei Xu、QimingZhang、JingZhang 和 Dacheng Tao。Vitae：通过探索内在归纳偏置增强的视觉变换器。神经信息处理系统进展，34：28522–28535，2021年。[284] LintingXue、Noah Constant、AdamRoberts、Mihir Kale、Rami Al-Rfou、Aditya Siddhant、AdityaBarua 和 Colin Raffel。mT5：大规模多语言预训练文本到文本转换器。*arXiv* 预印本 *arXiv:2010.11934*，2020年。[285] ZeyueXue、JianmingLiang、Guanglu Song、Zhuofan Zong、LiangChen、Yu Liu 和 Ping Luo。密集视觉预测的大批量优化。*arXiv* 预印本*arXiv:2210.11078*，2022年。[286] Zhewei Yao、Xiaoxia Wu、Conglong Li、Connor Holmes、Minjia Zhang、ChengLi和Yuxiong He。随机-LTD：随机且逐层丢弃词元，为大规模 Transformer 带来高效训练。*arXiv* 预印本 *arXiv:2211.11586*，2022年。[287] Yang You、IgorGitman 和 Boris Ginsburg。卷积网络的大批量训练。*arXiv* 预印本*arXiv:1708.03888*，2017年。[288] Yang You、Zhao Zhang、Cho-Jui Hsieh、James Demmel 和 Kurt Keutzer。数分钟内完成 ImageNet 训练。在第 *47* 届并行处理国际会议论文集，第1–10 页，2018年。[289] Binhang Yuan、Yongjun He、Jared Quincy Davis、Tianyi Zhang、Tri Dao、Beidi Chen、Percy Liang、Christopher Re和 Ce Zhang。在异构环境中对基础模型进行去中心化训练。*arXiv* 预印本*arXiv:2206.01288*，2022年。[290] SangdooYun、DongyoonHan、Seong Joon Oh、Sanghyuk Chun、Junsuk Choe和YoungjoonYoo。CutMix：使用具有可定位特征的正则化策略训练强分类器。在*IEEE/CVF*国际计算机视觉会议，第 6023–6032 页，2019年。[291] Matthew D Zeiler。Adadelta：一种自适应学习率方法。*arXiv*预印本 *arXiv:1212.5701*，2012年。

[292] Xiaohua Zhai、Alexander Kolesnikov、Neil Houlsby 和 Lucas Beyer。扩展视觉Transformer。载于*IEEE/CVF*计算机视觉与模式识别会议论文集，第12104–12113页，2022年。[293] Biao Zhang 和 Rico Sennrich。均方根层归一化。神经信息处理系统进展，32，2019年。[294] Hongyi Zhang、Moustapha Cisse、Yann N Dauphin 和 David Lopez-Paz。mixup：超越经验风险最小化。*arXiv*预印本 *arXiv:1710.09412*，2017年。[295] Hongyi Zhang、Yann N Dauphin 和 Tengyu Ma。通过更好的初始化实现无归一化的残差学习。载于国际学习表征会议，第3卷，第2页，2019年。[296] Jiong Zhang、Hsiang-Fu Yu 和 Inderjit S Dhillon。Autoassist：加速深度神经网络训练的框架。神经信息处理系统进展，32，2019年。[297] Minjia Zhang 和 Yuxiong He。通过渐进式层丢弃加速基于Transformer的语言模型训练。神经信息处理系统进展，33：14011–14023，2020年。[298] Xiang Zhang、Junbo Zhao 和 Yann LeCun。用于文本分类的字符级卷积网络。神经信息处理系统进展，28，2015年。[299] Xinyu Zhang、Qiang Wang、Jian Zhang 和 Zhao Zhong。对抗式AutoAugment。*arXiv*预印本 *arXiv:1912.11188*，2019年。[300] Zhong Zhang、Nian Shao、Chongming Gao、Rui Miao、Qinli Yang 和 Junming Shao。Mixhead：突破多头注意力语言模型中的低秩瓶颈。知识库系统，240：108075，2022年。[301] Jiawei Zhao、Florian Sch¨afer 和 Anima Anandkumar。零初始化：仅用零和一初始化神经网络。*arXiv*预印本 *arXiv:2110.12661*，2021年。[302] Shengyu Zhao、Zhijian Liu、Ji Lin、Jun-Yan Zhu 和 Song Han。用于数据高效生成对抗网络训练的可微增强。神经信息处理系统进展，33：7559–7570，2020年。[303] Yulin Zhao、Donghui Wang、Leiou Wang 和 Peng Liu。一种降低卷积神经网络计算复杂度的更快算法。算法，11(10)：159，2018年。[304] Zhun Zhong、Liang Zheng、Guoliang Kang、Shaozi Li 和 Yi Yang。随机擦除数据增强。载于*AAAI*人工智能会议论文集，第34卷，第13001–13008页，2020年。[305] Ce Zhou、Qian Li、Chen Li、Jun Yu、Yixin Liu、Guangjing Wang、Kai Zhang、Cheng Ji、Qiben Yan、Lifang He 等。预训练基础模型综合综述：从BERT到ChatGPT的发展历程。*arXiv*预印本 *arXiv:2302.09419*，2023年。[306] Chen Zhu、Renkun Ni、Zheng Xu、Kezhi Kong、W Ronny Huang 和 Tom Goldstein。Gradinit：学习初始化神经网络，以实现稳定且高效的训练。神经信息处理系统进展，34：16410–16422，2021年。[307] Yukun Zhu、Ryan Kiros、Rich Zemel、Ruslan Salakhutdinov、Raquel Urtasun、Antonio Torralba 和 Sanja Fidler。对齐书籍与电影：通过观看电影和阅读书籍，生成故事般的视觉解释。载于*IEEE*国际计算机视觉会议（*ICCV*），2015年12月。[308] Bohan Zhuang、Jing Liu、Zizheng Pan、Haoyu He、Yuetian Weng 和 Chunhua Shen。Transformer高效训练综述。*arXiv*预印本 *arXiv:2302.01107*，2023年。