计算机科学
COMPUTER SCIENCE

实际应用场景中的大模型高效推理技术综述

刘利龙, 刘国明, 齐保元, 邓雪杉, 薛迪展, 钱胜胜

引用本文

刘利龙, 刘国明, 齐保元, 邓雪杉, 薛迪展, 钱胜胜. 实际应用场景中的大模型高效推理技术综述[J]. 计算机科学, 2026, 53(1): 12-28.

LIU Lilong, LIU Guoming, QI Baoyuan, DENG Xueshan, XUE Dizhan, QIAN Shengsheng. Efficient Inference Techniques of Large Models in Real-world Applications:A Comprehensive Survey [J]. Computer Science, 2026, 53(1): 12-28.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.250300030

实际应用场景中的大模型高效推理技术综述

刘利龙[1]　刘国明[2]　齐保元[3]　邓雪杉[4]　薛迪展[4]　钱胜胜[4]
1 郑州大学河南先进技术研究院　郑州 450003
2 小米汽车科技有限公司集团技术委员会　北京 100085
3 北京小米松果电子有限公司集团技术委员会　北京 100085
4 中国科学院自动化研究所多模态人工智能系统全国重点实验室　北京 100190
(liulilong0401@163.com)

摘　要　近年来,大语言模型(Large Language Models,LLMs)技术迎来了快速发展,其在各行业的应用呈现出蓬勃增长的态势。从自然语言处理到智能推荐,再到信息检索和自动化写作,LLMs正逐渐成为许多领域中不可或缺的工具。然而,随着应用场景的逐渐多样化和需求的不断增加,LLMs推理效率问题日益凸显。在实际应用场景中,快速准确的推理能力对于响应用户请求、处理大规模数据和实时决策至关重要。为了应对这一挑战,学术界展开了广泛的研究和探索,致力于提高LLMs的推理效率。对此,全面调研了实际应用场景中有关LLMs高效推理的文献。首先,介绍了LLMs推理的原理,并分析了在实际应用场景中如何提高LLMs的推理效率。然后,引入了一个针对实际应用场景的分类系统,其主要分为3个层面,分别是算法优化层面、参数优化层面和系统优化层面;并对大模型进行相关研究的总结和归纳。最后,探讨了未来可能的研究方向。

关键词:大语言模型;高效推理;实际应用场景;算法优化;参数优化;系统优化
中图分类号　TP391

Efficient Inference Techniques of Large Models in Real-world Applications:A Comprehensive Survey

LIU Lilong[1],LIU Guoming[2],QI Baoyuan[3],DENG Xueshan[4],XUE Dizhan[4] and QIAN Shengsheng[4]
1 Henan Institute of Advanced Technology,Zhengzhou University,Zhengzhou 450003,China
2 Group Technical Committee,Xiaomi Automobile Technology Co.,Ltd.,Beijing 100085,China
3 Group Technical Committee,Beijing Xiaomi Pinecone Electronics Co.,Ltd.,Beijing 100085,China
4 State Key Laboratory of Multimodal Artificial Intelligence Systems,Institute of Automation,Chinese Academy of Sciences,Beijing 100190,China

Abstract　In recent years,the technologies of LLMs have been rapidly developed,with their applications across various industries experiencing vigorous growth. From natural language processing to intelligent recommendations,and from information retrieval to automated writing,LLMs are becoming indispensable tools in many fields. However,with the diversification of application scenarios and the increase in demands,the efficiency of LLM inference is becoming increasingly prominent. In practical applications,rapid and accurate inference capabilities are crucial for responding to user queries,handling large-scale data,and making real-time decisions. To address this challenge,academia has undertaken extensive research and exploration to enhance the inference efficiency of LLMs. This paper comprehensively surveys the literature on efficient LLM inference in practical application scenarios. Firstly,it introduces the principles of LLMs and analyzes how to improve LLM inference efficiency in practical application scenarios. Secondly,it proposes a taxonomy tailored for real-world applications,which consists of three main levels:algorithm optimization,parameter optimization,and system optimization. This survey summarizes and categorizes related work about LLMs. Finally,it discusses potential future research directions.

Keywords　Large language models,Efficient inference,Practical application scenarios,Algorithm optimization,Parameter optimization,System optimization

到稿日期:2025-03-06　返修日期:2025-07-02
基金项目:国家重点研发计划(2023YFC3310700);国家自然科学基金(62276257)
This work was supported by the National Key Research and Development Program of China(2023YFC3310700) and National Natural Science Foundation of China(62276257).
通信作者:钱胜胜(shengsheng.qian@nlpr.ia.ac.cn)

1 引言

随着人工智能技术的迅猛发展，大型语言模型[1]（LLMs）已逐渐成为推动语言理解、生成和推理任务的核心驱动力。这些模型在自然语言处理（Natural Language Processing，NLP）的各个领域展现出前所未有的能力，能够生成高度连贯的文本，进行复杂的推理，并且在多种语言环境中表现出色。以 ChatGPT[2]，Copilot[3] 和 Llama[4] 等实际应用为例，它们的成功不仅展示了 LLMs 的巨大潜力，也反映出社会各界对智能语言处理技术的日益增长的需求。这些应用的普及推动了人们对更高效、更智能的语言模型的需求。然而，随着模型规模的日益庞大和结构的日趋复杂，LLMs 在实际应用中的推理效率面临着严峻的挑战。本文首先通过图 1 所示的组织结构框架对这些挑战进行深入分析。

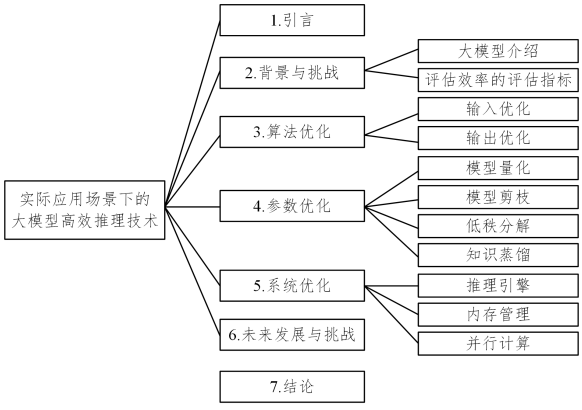


图 1 本文组织结构

Fig. 1 Organization of this paper

在实际应用中，LLMs 面临着一系列问题和挑战，其中最显著的包括输入输出限制、模型参数的复杂性以及自回归解码过程中的局限性。输入输出的限制主要体现在模型对长文本处理的局限性上；而模型参数的复杂性则直接影响了计算资源的消耗，进而导致推理延迟的增加；自回归解码的过程尽管能够生成高质量的文本，但其逐步生成的机制也往往成为推理速度的瓶颈。这些问题的存在，使得在资源有限的环境中实现高效的模型推理变得更加困难。

为了应对上述挑战，本文调研了多种推理优化技术。这些技术主要通过改进模型的输入输出处理方式和解码方法来有效降低计算成本，同时尽量保证模型的性能不受影响。具体而言，优化措施包括通过输入剪枝和输出筛选来减少不必要的计算，从而提高推理效率。参数优化也是解决问题的关键方向之一。例如，模型剪枝和量化技术能够在减少参数量的同时，维持模型的准确性，这为 LLMs 的轻量化部署提供了新的可能性。在系统优化方面，本文探讨了并行计算和 KV 缓存优化等技术的应用，通过在现有硬件条件下最大化资源利用率，来实现更高效的模型推理。

如表 1 所列，本综述与现有文献相比，在研究焦点上有所不同。尽管 LLMs 的基础设计和硬件加速是该领域的重要议题，但这些方面通常需要大量的资源和专业的知识。例如，开发基于 FPGA 的硬件加速器要求精通硬件描述语言、内存优化技术以及高度定制的计算单元。而对于 ASICs，如 Google 的 TPU，虽然其能够提供高度优化的计算能力，但其开发成本极高且开发周期极长[5]。因此，本文不深入探讨这些内容，而是聚焦于资源受限环境下的解决方案，特别是那些在有限资源下实现的模型压缩和推理技术。

表 1 高效推理相关综述

Table 1 Survey on efficient inferences

综述	侧重方面				
	算法	参数	系统	硬件	模型框架
A Survey on Hardware Accelerators for Large Language Models[5]				√	
A Survey on Model Compression for Large Language Models[6]		√			
A Survey of Knowledge Distillation in Deep Learning[7]		√			
A Survey of Knowledge Distillation[8]		√			
A Comprehensive Survey of Compression Algorithms for Language Models[9]		√			
A Comprehensive Survey of Accelerated Generation Techniques in Large Language Models[10]	√				
Model Compression and Efficient Inference for Large Language Models：A Survey[11]	√				√
A Survey on Efficient Inference for Large Language Models[12]	√	√	√	√	√
本文	√	√	√		

综述[6]主要聚焦于 LLMs 的参数层面的设计和实现，其主要提及量化、剪枝、蒸馏等参数层面的技术，而对 LLMs 的算法及系统层面的讨论较少。综述[7]和综述[8]主要侧重于利用知识蒸馏技术进行优化模型的方式。综述[9]主要对语言模型的压缩算法技术进行了详细的综述与分析。综述[10]则主要涉及到了 LLMs 中的加速生成技术分类，分为推测解码、提前退出和非自回归方法。本文不仅涵盖了这些已有文献中的重要内容，还特别评估了在各种实际应用场景中不同模型压缩技术的表现，力求在保持性能的同时减少计算和内存需求。虽然这一权衡在综述[11]和综述[12]中有所提及，但本文的讨论更侧重于实际部署需求场景中的可操作性和灵活性。

本文则聚焦于大模型实际应用中的高效推理技术，尤其关注算法的实用性和灵活性，力求为实际应用提供更具针对性和可操作性的指导。与以往综述相比，本文基于实际应用场景需求，创新性地构建了一个包含算法优化、参数优化和系统优化 3 个层面的分类系统，这种多维度的视角不仅涵盖了大模型高效推理的各个关键环节，而且能更细致地剖析不同类型技术在实际应用中的作用和关联。这有助于读者从整体上把握大模型高效推理技术的全貌，也能方便地找到针对特定实际问题的解决方案，为读者提供更清晰、更有条理的知识框架。例如，当面临一个需要在移动设备上快速部署大模型

文本生成任务的场景时，读者可以迅速定位到算法优化中的输入压缩技术以及系统优化中的内存管理策略等相关内容进行参考。本文也深入探讨了多种具体算法，如详细阐述了输入压缩和输出优化技术如何在保持模型性能的前提下有效降低计算成本并提高推理速度，这些技术可直接应用于实际的文本处理任务，如新闻自动摘要、智能客服对话生成等，为开发者提供了易于理解和实现的解决方案，加快了大模型技术在实际业务中的推理优化方案的落地。然而，本文也存在一定的局限性。本文尽管努力涵盖了多种高效推理技术，但主要聚焦于文本大模型的高效推理技术，对多模态大模型的综述相对较少。多模态大模型在实际应用场景中具有巨大的潜力，如在智能驾驶、医疗影像分析等领域被广泛应用，其推理效率的提升同样关键。因此，对多模态大模型高效推理技术的深入研究和综述将是我们后续工作的重要方向。

2 背景与挑战

在自然语言处理领域，LLMs 以其卓越的语言理解和生成能力，成为推动技术进步的关键力量。然而，该模型在实际应用中面临着前所未有的挑战，这主要源自于计算资源的巨大需求和推理过程中的高内存占用。本综述将深入探讨 LLMs 在实际应用中的背景和面临的挑战，包括庞大参数规模带来的资源消耗，以及推理引擎效率和并行计算能力的优化需求。

2.1 大模型

2.1.1 大语言模型

语言建模作为语言模型(LMs)的核心功能，涉及对单词序列进行建模和预测后续单词的分布。近年来的研究表明，扩展语言模型不仅增强了其语言建模能力，还使其具备处理超越传统自然语言处理任务的新能力。这些扩展的语言模型被称为 LLMs。

主流的 LLMs 采用 Transformer[13] 架构设计。Transformer 架构由多个堆叠的 Transformer 块组成，每个块包括多头自注意力(MHSA)块、前馈网络(FFN)和层归一化(LN)操作。每个块接收前一个块的输出特征作为输入，并通过子模块处理以获取输出特征。在第一个块之前，使用分词器将原始输入句子转换为标记序列，并通过嵌入层将这些标记转换为输入特征。随后，额外的位置嵌入被添加到输入特征中，以编码每个输入标记的序列信息。

Transformer 架构的核心概念是自注意力机制，其在 MHSA 块中得以应用。假设输入特征表示为 $\boldsymbol{X}=[x_1, x_2, \cdots, x_n]$，MHSA 块对其进行线性投影，得到一组查询 $\boldsymbol{Q}$、键 $\boldsymbol{K}$ 和值 $\boldsymbol{V}$：

$$\boldsymbol{Q}_i=\boldsymbol{X}\boldsymbol{W}_{Qi}, \boldsymbol{K}_i=\boldsymbol{X}\boldsymbol{W}_{Ki}, \boldsymbol{V}_i=\boldsymbol{X}\boldsymbol{W}_{Vi} \tag{1}$$

其中，$\boldsymbol{W}_{Qi}$、$\boldsymbol{W}_{Ki}$ 和 $\boldsymbol{W}_{Vi}$ 分别是第 i 个注意力头部的投影矩阵。接下来，对每个 $(\boldsymbol{Q}_i, \boldsymbol{K}_i, \boldsymbol{V}_i)$ 元组应用自注意力操作，得到第 i 个注意力头部的特征 $\boldsymbol{Z}_i$：

$$\boldsymbol{Z}_i=Attention(\boldsymbol{Q}_i, \boldsymbol{K}_i, \boldsymbol{V}_i)=\mathrm{Softmax}\left(\frac{\boldsymbol{Q}_i\boldsymbol{K}_i^{\mathrm{T}}}{\sqrt{d_k}}\right)\boldsymbol{V}_i \tag{2}$$

其中，d_k 是查询(键)的维度。

注意，自注意力操作涉及矩阵乘法，其计算复杂度随输入长度的增加呈二次增长。最终，MHSA 块将所有注意力头部的特征连接起来，并通过线性投影形成其输出 $\boldsymbol{Z}$：

$$\boldsymbol{Z}=Concat(Z_1, Z_2, \cdots, Z_h)\boldsymbol{W}_O \tag{3}$$

其中，$\boldsymbol{W}_O$ 是投影矩阵。自注意力机制使得模型能够识别不同输入部分的重要性(无论它们之间的距离如何)，从而能够捕捉输入句子中的长距离依赖关系和复杂关联。

Transformer 块中的另一个重要模块是 FFN，其通常位于 MHSA 块之后，由两个线性变换层和一个非线性激活函数组成。它接收来自 MHSA 块的输出特征 X，并进行处理：

$$FFN(X)=\boldsymbol{W}_2\sigma(\boldsymbol{W}_1X) \tag{4}$$

其中，$\boldsymbol{W}_1$ 和 $\boldsymbol{W}_2$ 分别表示两个线性层的权重矩阵，$\sigma(\cdot)$ 表示激活函数。

2.1.2 模型推理

主流的 LLMs，即仅有解码器类型的 LLMs，通常采用自回归方法生成输出句子。具体而言，自回归方法逐个生成标记。在每个生成步骤中，LLMs 将整个标记序列作为输入，包括输入标记和先前生成的标记，并生成下一个标记。随着序列长度的增加，生成过程的时间成本迅速增加。为了应对这一挑战，引入了键-值(KV)缓存技术，以加速生成过程。顾名思义，KV 缓存技术涉及在 MHSA 块中存储和重复使用先前的键(K)和值(V)对。由于显著优化了生成延迟，该技术已广泛应用于 LLMs 推理引擎中。

基于上述方法和技术，LLMs 推理过程可以分为两个阶段。

1)预填充阶段：LLMs 计算并存储初始输入标记的 KV 缓存，并生成第一个输出标记。

2)解码阶段：LLMs 使用 KV 缓存逐个生成输出标记，随后使用新生成标记的键(K)和值(V)对更新它。

2.2 评估效率的评估指标

LLMs 推理过程中的主要低效因素包括算法复杂度、参数规模和系统效率。复杂的提示设计增加了计算复杂度，例如自注意力操作的二次增长。庞大的模型增加了计算和内存成本，研究人员通过模型修剪和模型压缩等策略来应对挑战。优化推理引擎和并行计算能力可以显著提高推理速度和效率，促进更高效、可靠的模型部署和应用。

2.2.1 评估指标

在实际应用场景中，评估大模型推理效率的指标包括计算效率和内存效率两个主要方面。

首先是计算效率，这涉及到在训练、微调和执行 LLMs 时所需的处理能力。评估计算效率时，需要考虑到浮点运算的数量、算法的效率以及 GPU 或 TPU 等处理单元的利用率。在实际应用中，关键在于如何在保证输出质量的前提下最小化计算需求，这可以通过优化算法选择和并行计算等手段实现。

其次是内存效率，这涉及到执行推理过程时所需的 RAM 和存储量。尤其是在处理数十亿参数的 LLMs 时，需要大量内存来存储模型权重并处理大规模数据集。为了提升内存效率，可以采用优化数据结构、模型修剪等策略，以有效减少内存占用，提高系统整体的推理速度和稳定性。

在实际应用场景中，综合考虑计算效率和内存效率的

优化策略，可以帮助大模型在资源有限的环境下更高效地运行，从而实现更快速、更可靠的推理过程，符合现实世界中对大模型应用的实际需求。

2.2.2 存在的挑战

在资源受限的场景中部署 LLMs，同时保留它们强大的能力，对从业者和研究者都是一个重大挑战。如表 2 所列，本节进一步分析了 LLM 推理过程中的 3 个低效原因，重点关注以下 3 个关键因素。

表 2 现有挑战

Table 2 Existing challenges

层面	问题	解决方案
算法层面	提示设计复杂性增加，自注意力操作计算复杂度二次增加	创新的提示设计方法，优化自注意力操作以减小计算复杂度
参数层面	模型参数数量庞大，增加计算成本和内存使用成本	修剪模型和精简存储结构，以降低资源需求
系统层面	推理引擎效率和并行计算能力不足，影响大模型部署	优化数据流管理和并行化操作，开发高效的推理框架和软件工具

1）算法层面：在实际应用场景中，模型的表现往往取决于所使用的提示和输出。许多研究提出了创新的提示设计方法，实验证明良好设计的提示能够最大限度地释放语言模型的潜力。然而，这些技术不可避免地导致提示变得更为复杂，这也带来了一定挑战。例如，在解码过程中，自注意力操作的计算复杂度随着输入长度的增加呈二次增长趋势，这显著影响了推理的效率。

2）参数层面：主流的 LLMs 通常包含数十亿甚至数万亿参数。例如，LLaMA-70B[4] 模型包含 700 亿参数，而 GPT-3[14] 模型的参数规模高达 1750 亿。这一庞大的模型规模，在 LLMs 推理过程中显著增加了计算成本、内存访问成本和内存使用成本。为了应对这一挑战，研究人员探索使用模型修剪、精简存储结构等策略，以在保持推理质量的同时降低资源需求。

3）系统层面：推理引擎的推理性能和并行计算能力，对于大规模语言模型的部署至关重要。有效地利用 GPU/TPU 等硬件加速器，优化推理过程中的数据流管理和并行化操作，可以显著提高推理速度和效率。同时，开发高效的推理框架和系统工具，如高效的模型加载策略，也是解决推理效率问题的关键。

面对 LLMs 在实际应用场景中的推理挑战，算法、参数和系统 3 个方面的优化是关键策略，能够帮助突破资源限制，提高推理效率，并实现更加高效和可靠的模型部署和应用。

3 算法优化

在算法优化层面，现有的研究主要致力于对两个关键方向进行高效推理优化。首先是输入压缩，即通过减少模型输入数据的方式来有效削减推理成本。这种方法通常涉及数据降维或选择性输入采样，以在不牺牲模型准确性的前提下提高推理效率。其次是输出优化，即通过优化输出内容的结构，使得模型能够实现更高效的批量处理和并行推理。这些技术通过减少生成延迟，有效提高了整体推理速度，为实际应用中的高效计算提供了重要支持。

3.1 输入优化

精心设计的提示虽能最大化释放语言模型潜力，但也会导致提示复杂化，增加计算成本和内存使用。现有研究主要利用输入压缩和检索增强生成[15]（Retrieval-Augmented Generation，RAG）两种策略来应对这一挑战。输入压缩主要是探索动态减小输入序列长度的机会，从而提高 Transformer 的计算效率。其灵感类似于人类阅读理解，即并非所有单词都会受到同等程度的关注，一些单词会受到更多关注，而其他则可能被忽略。图 2 展示了用户如何通过提供精确的提示词进行输入优化（提示词可能涉及修改或简化），以帮助生成更准确的内容的方法。RAG 则结合了信息检索和生成模型的优势，将通过检索引擎获取的文本作为模型输入的一部分，从而增强了生成模型的信息获取能力，流程如图 2 所示。

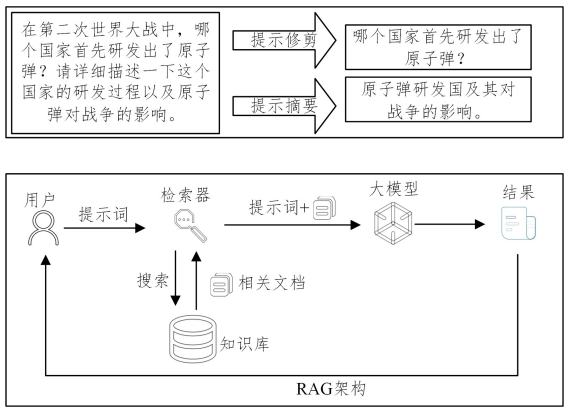


图 2 输入优化

Fig. 2 Input optimization

3.1.1 输入压缩

压缩输入是通过各种方法移除不必要的标记、句子或文档，以精简原始提示并保留其语义信息的核心目标。这些技术包括 DYNAICL[16]，它通过训练语言模型的元控制器动态确定最适合输入的上下文示例数量，从而在计算预算内适应性地压缩输入。另外，选择性上下文提出将标记合并，并根据自信息指标进行单元级提示修剪；而 STDC[17] 则通过迭代地移除导致性能下降最少的短语节点来实现提示的有效压缩。这些方法，如 PCRL[18] 和 RECOMP[19]，利用强化学习和基于句子级别的策略，进一步优化压缩效果。例如，根据预训练编码器编码问题和文档，并据此选择性地移除文档，以减少不必要的信息负担。TPC[20] 提出了一种新颖的通用提示压缩方法，通过使用任务描述符生成与上下文相关的任务描述，并利用上下文感知的句子编码器计算提示中每个句子的相关性，从而生成压缩后的提示。这种方法无需依赖输入问题或手动设计的提示模板，实现了跨任务和领域的通用提示压缩。

在压缩的实施方面，一些技术（如 Prompt Compression[21] 和 Gisting[22]）专注于离线压缩，用于固定前缀提示，允许通过调整软标记来模拟和优化系统提示。相比之下，Auto Compressors[23] 和 ICAE[24] 则采用在线压缩方法。前者训练预训练语言模型，通过无监督学习将输入压缩为摘要向量；后者通过自编码器将上下文压缩为短期记忆槽，从而实现更高效的输入处理。

技术的进步还体现在对标记重要性评分的利用。例如，SpAtten[25]和 LTP[26]通过评分排序和逐层阈值的学习，精确地识别和保留了对模型贡献最大的标记，从而优化了整体压缩效果。而 SMART-TRIM[27]则通过轻量级修剪模块的集成，无需预训练或数据增强，即可实现对特定任务输入和参数的有效修剪，从而在实际应用中提高了模型推理速度和内存效率。而文献[28]提出的动态输入剪枝(Dynamic Input Pruning，DIP)和缓存感知掩码(Cache-Aware Masking)的结合策略，在修剪输入激活时考虑缓存状态，优先保留缓存中的权重，从而提高缓存命中率和模型吞吐量。与传统的输入压缩方法相比，这种方法在降低内存占用的同时，显著提升了模型的推理速度，尤其适用于移动设备端的大语言模型推理任务。

3.1.2 检索增强生成

RAG 技术是一种创新的方法，旨在利用外部知识源来提升 LLMs 的响应质量。相较于传统方法将所有信息堆积在单一过长的提示中，RAG 技术采用精确的方式将相关的检索信息直接整合到原始提示中。这种做法不仅确保了模型获取到必要的信息，还显著减小了提示的复杂度和长度。

举例来说，FLARE[29]通过预测即将出现的句子，主动选择何时以及如何引入必要的外部信息，从而有效提高了推理效率。另一方面，REPLUG[30]将 LLMs 视作一个“黑匣子”，并采用可调整的检索模型来增强其性能。它在 LLMs 输入之前放置已检索到的文档，通过 LLMs 自身来监督和优化检索模型的操作，从而实现更高的信息获取和处理效率。

此外，Self-RAG[31]采用了结合检索和自我反思的方法，进一步提高了 LLMs 的质量和准确性。通过引入反思标记，Self-RAG 使得 LLMs 在推理阶段更具可控性和精准性，从而更好地适应复杂的推断任务和信息需求。

3.2 输出优化

在大规模语言模型的实际应用中，输出优化是提高推理效率和结果质量的关键环节。输出优化涵盖了多个方面，包括早期退出、采样优化、级联推理以及推测解码(见图 3)。这些技术不仅可以显著减少推理时间，还能有效改善模型生成的输出连贯性和多样性，从而更好地适应不同的应用场景需求。

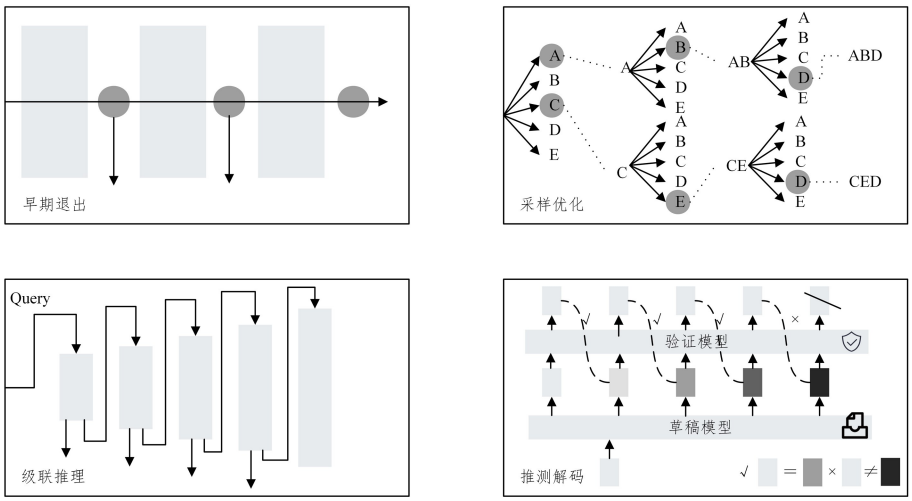


图 3 输出优化

Fig. 3 Output optimization

3.2.1 早期退出

早期退出通过跳过特定的计算过程来实现，如图 3 中“早期退出”部分所示。其核心理念是，对于简单的输入样本，通常只需较少的计算即可实现准确预测。早期，研究者们常常根据自己的标准来探索这一策略：DeeBERT[32]使用熵作为停止标准；RightTool[33]以预测的 softmax 分数为依据；PABEE[34]则在内部分类器的中间预测连续不变时停止推理。PCEE-BERT[35]提出了混合早期退出标准，结合确信分数和耐心计数器，当中间层的确信度达到足够数量时，将提前结束。SkipBERT[36]通过跳过浅层计算加速推理，使用早期退出标准来避免更深层的计算。Short-Cutting Transformer[37]提出了基于线性变换的方法，将中间表示直接转换为最终表示，绕过了中间的 Transformer 计算。它与 CALM[38]采用相同的早期退出策略，即当最高和次高概率之间的差异大于 CALM 的置信阈值时停止。MuE[39]将动态早期退出策略扩展到多模态 LLMs，但在同时包含编码器和解码器的多模态架构中，编码器和解码器之间的退出决策面临独特挑战。MuE 根据层次输入的相似性提出了退出标准，这一想法是受到饱和观察的启发。LGViT[40]为通用 ViTs 提出了一个早退框架，具有多种退出头，如局部感知和全局聚合头，以平衡效率和准确性。

3.2.2 采样优化

采样算法对 LLMs 的生成质量有着至关重要的影响。传统的贪婪采样方法倾向于选择具有最高概率的标记。为了更高效地生成接近最优的序列，现代研究引入了并行采样技术，如束搜索。束搜索[41]在每一步迭代中维护一个固定数量的高分数序列(称为束宽)，从而显著提高了解码效率。

此外，还有多种随机采样技术被提出，如 top-k[42]，top-p[43]以及温度控制，旨在通过引入随机性来增加生成结果的多样性。然而，这些方法在实际应用中面临着挑战，如大规模词汇表(数万个词汇)带来的内存压力增加，以及冗余的键值(KV)缓存导致的采样效率问题。

为了解决这些问题，一些方法如 LightSeq[44] 提出了有效的分层实现。它将词汇表划分为多个组，每组内使用少量的 GPU 指令来检索候选项，然后对这些候选项重新排名，以获取 top-k 的标记。这种分层实现有效地降低了内存压力和计算成本，提高了采样的效率和生成质量，如图 3 中“采样优化”部分所示。

3.2.3 级联推理

级联推理是一种利用多种不同规模的语言模型来处理各种复杂度推理请求的技术策略。与直接使用单一大型模型相比，CascadeBERT[45] 内部集成了多个深度不同的分类器，这些分类器按照级联的方式组织在一起，并根据问题的难度自动选择最合适的模型进行推理，如图 3 中“级联推理”部分所示。Tabi[46] 则专注于优化非生成型语言模型的服务，采用类似的方法将小型模型和大型模型整合，以应对具有不同置信度的查询。Frugal-GPT[47] 则通过学习方法动态地分配查询给不同的语言模型 API，以在性能和成本之间找到最佳平衡。另外，一项并行工作着眼于优化模型的重用和查询结果的缓存，以探索最小化推理成本的最佳方法。Mixture-of-Thought[48] 将级联思想应用于语言模型推理任务中，旨在通过随机抽取 Chain-of-Thought[49] 和 Program-of-Thought[50] 的线索来生成答案，从而节省成本。总体而言，级联推理为提高推理效率开辟了新的路径，但如何精确设计调度机制以避免影响模型质量仍然是一个具有挑战性的课题。

相比之下，混合专家模型[51]（Mixture-of-Experts，MoE）是另一种处理复杂任务的方法。MoE 模型由多个专家模型组成，每个专家模型负责处理数据的不同子集或方面。在推理阶段，MoE 模型会动态选择最适合当前输入的专家模型来生成结果。MoE 侧重于在一个模型内部通过专家的协作来提高整体性能；而级联推理更注重通过组合不同规模和类型的模型来应对不同类型的推理任务，以达到优化性能和成本的目的。

3.2.4 推测解码

在提升 LLMs 解码效率的研究中，推测解码技术[52] 通过引入推测执行的思想，突破了传统顺序执行的限制，实现了解码过程的并行化，从而显著提升了推理速度。该技术的实现主要依赖两个关键步骤：草稿构建与草稿验证。草稿构建利用小型模型快速生成预测，而草稿验证则确保这些预测与 LLMs 的标准输出保持一致性，如图 3 中“级联推理”部分所示。

在草稿模型的设计与优化方面，研究人员提出了多种方法来提高性能和准确性。DistillSpec[53] 通过从目标 LLMs 中蒸馏出一个较小的草稿模型，既减小了模型大小，又保证了预测的质量。SSD[54] 自动识别目标 LLMs 中的子模型作为草稿模型，省去了单独训练的步骤。OSD[55] 动态调整草稿模型的输出分布，以匹配用户查询的分布，并通过监控被拒绝的草稿标记来优化模型。PaSS[56] 使用目标 LLMs 本身作为草稿模型，整合可训练的前瞻标记，以同时生成后续标记。REST[57] 引入了基于检索的推测解码方法，使用非参数检索数据存储作为草稿模型，增加了解码的灵活性。SpecInfer[58] 通过集合增强调优技术，将草稿模型的输出分布与目标 LLMs 的输出分布对齐。Lookahead[59] 解码在并行生成目标 LLMs 的 n-gram 时，协助生成草稿标记，提高了解码的效率。Medusa[60] 通过特别调整 LLMs 的头部，优化了模型结构以生成后续草稿标记。Eagle[61] 采用轻量级的 Transformer 层作为自回归头，整合了目标 LLMs 的上下文特征来生成草稿标记。

草稿构建策略的创新也是提高解码效率的关键。Spectr[62] 提出生成多个草稿标记序列，并采用 k-顺序草稿选择技术进行验证，利用推测采样确保输出分布的等效性。SpecInfer 采用了类似的方法，但进一步将草稿标记序列合并成“标记树”，并引入了树注意力机制进行验证，这种方法被称为“标记树验证器”。阶段推测解码和级联推测草稿（CS Drafting）将推测解码集成到标记生成过程中，以加速草稿的构建。这些创新策略不仅提高了草稿构建的效率，还确保了解码过程的准确性和可靠性。

推测解码技术的不断创新和优化，显著提高了 LLMs 解码的效率。随着技术的进一步发展，推测解码有望在未来的 LLMs 应用中发挥更大的作用，提高实用性和响应速度，为用户带来更加流畅和高效的体验。

表 3 对上述优化技术进行了总结分析。可以看出，这些方法虽然在提升推理效率和降低计算开销方面具有显著优势，但也面临着准确性损失、计算复杂度增加以及对外部系统依赖等挑战。因此，未来的算法优化工作需要在不同技术之间进行权衡，以实现更高效、更精准的模型推理。同时，硬件性能的提升和新算法的出现，可能会为应对这些挑战提供新的机遇。未来的研究不仅需要关注单一优化技术的效果，还应关注技术间的协同作用，以推动算法优化的持续进步。

表 3 优化技术的对比

Table 3 Comparison of optimization technologies

优化技术	优势	挑战与限制
输入压缩	提高计算效率；降低内存使用；保留重要语义	损失准确性；模型调整难
动态压缩	动态调整上下文灵活；适应计算预算	需要大量训练；参数调节复杂
选择性上下文	根据信息指标剪枝；降低计算开销	可能忽略重要信息；需要精确的选择机制
检索增强生成	增强信息获取能力；减少输入复杂度	依赖外部检索系统；有信息检索延迟
早期退出	提高推理速度；适应输入复杂度	停止标准难以统一；可能导致性能波动
采样优化	增强生成多样性；提高解码效率	增加内存消耗；影响输出一致性
级联推理	动态选择合适的模型；降低计算负担	需要精确的调度机制；影响模型质量
推测解码	并行化解码过程；提高解码效率	依赖草稿模型质量；可能增加复杂度
混合专家模型	针对任务选择模型；优化性能和成本	计算复杂度高；需要大量训练

4 参数优化

大语言模型凭借其卓越的能力，为自然语言处理领域带来了革命性的进展。然而，庞大的参数量导致大语言模型在推理部署时面临巨大的计算和存储压力[63]。本章着眼于实际应用场景的限制，针对大模型参数优化方面的各种策略和

方法，主要从模型量化、模型剪枝、低秩分解、知识蒸馏4个方面进行总结和讨论。

4.1 模型量化

模型量化是一种重要的模型压缩方法，其核心思想是将神经网络模型运行时涉及的权重和激活值等数据从常规的高精度表示（FP32和FP16）转换为低精度表示（如INT8和INT4等），从而显著减少模型的存储需求、计算成本以及访存开销，并尽可能地保持模型的性能[64]。常规的神经网络模型采用FP32进行存储计算；相较于FP32，使用FP16量化权重和激活值可以减少1/2的内存使用量，使用INT8则可以减少3/4的内存使用量[65]。在实际的应用场景中，通过量化大语言模型的权重和激活值，可以减少GPU占用内存大小和位宽需求，降低数据的存储负担。同时，整型计算单元相比于浮点计算单元能耗更小，量化策略还可以有效降低计算能耗，提高大模型的计算并行性和推理速度。图4展示了模型量化的过程。首先通过聚类将高精度浮点数权重转化为低精度表示。然后通过量化和编码模型的权重，减少计算资源的消耗，同时尽量保持模型性能。

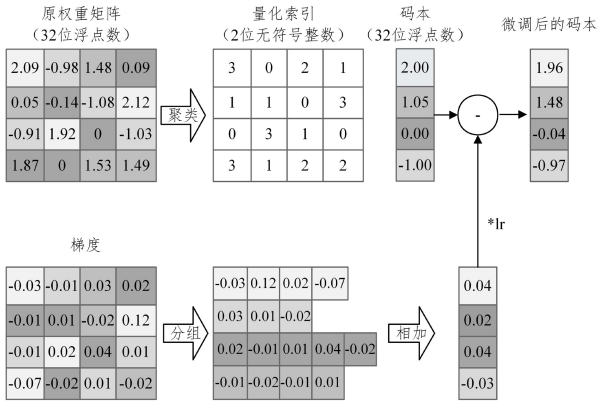


图4 模型量化

Fig. 4 Model quantization

4.1.1 量化策略概述

大语言模型经过量化压缩后，可以得到更加轻量级的模型，但是部分量化方法可能导致模型性能大幅度下降，难以取得令人满意的压缩效果。由于大语言模型的独特性，针对大语言模型的方法还面临着一些额外的挑战。

1）当语言模型的参数规模超过10亿时，语言模型对于参数的依赖性更高，需要更高的量化精度才能保持模型性能。

2）LLMs的激活模式与中小模型不同。在数十亿参数以上的模型中会出现极端的异常特征值，这些离群的异常特征会严重影响量化精度。

大语言模型在规模和复杂性上远超中小语言模型，传统方法可能无法满足其特定需求。因此，很多适用于中小语言模型的量化方法，难以直接迁移到大语言模型的量化压缩中使用。因此，下文将针对大模型的量化策略展开，并根据使用场景将量化方法分为两种主要的类型：训练后量化（Post-Training Quantization，PTQ）和量化感知训练（Quantization-Aware Training，QAT）。1）训练后量化方法在模型训练完成后对其参数进行量化，主流方法是利用少量校准数据优化权值和激活值的量化参数，适用于可访问数据量受限的场景。2）量化感知训练方法中将需要量化的节点插入网络中，并在训练过程中将所有神经网络参数和量化参数一起优化，通过共同训练的过程使得模型更加适应量化造成的误差。4.1.2节与4.1.3节将更加详细地介绍训练后量化和量化感知训练两种方法在大模型量化领域的发展和创新。

4.1.2 训练后量化

与CNN模型或BERT等较小的Transformer模型相比，针对大语言模型的激活更难量化。LLMs的参数扩大到数十亿以上时，将出现较大的量化误差和精度下降。GPTQ[66]，LUT-GEMM[67]，AWQ[68]和SqueezeLLM[69]等方法只涉及权重的量化，ZeroQuant-v2[70]，Gpt3.int8()[71]，SmoothQuant[72]和Rptq[73]等方法同时关注权重和激活的量化。

LUT-GEMM和Gpt3.int8()是较早应用于大语言模型的权重量化技术。LUT-GEMM提出了一种新的利用查找表（lookup Table，LUT）的去量化方法，并采用二进制编码量化（BCQ）的非均匀量化方法。Gpt3.int8()使用8位量化实现大语言模型中的矩阵乘法，在保持模型性能精度的同时，将推理过程中的GPU显存使用量减半；进一步引入混合精度分解，在FP16中保留异常值，并使用INT8进行其他非异常值的激活。Dettmers等通过分析推理扩展法则，研究了大语言模型中模型规模和比特精度之间的权衡，实验结果表明，4位精度在实现模型比特总数和零样本精度之间的平衡方面是最优的。为了解决4位量化下模型精度下降明显的问题，GPTQ在传统算法OBQ的基础上提出了一种基于二阶信息的层级量化技术，将权重的位宽减少到3位或4位，同时与未压缩版本相比保持了最小的精度损失。

AWQ提出了不同权重在大语言模型的性能中并非同等重要的主张，将通道精度分离技术引入模型量化中，通过网格搜索选择重参数化系数，有效地减小了重构误差。SqueezeLLM将离群值存储在全精度稀疏矩阵中，并对剩余权重进行非均匀量化，根据量化灵敏度确定非均匀量化值，实现了3位的无损压缩。ZeroQuant[74]引入了权重的分组量化策略和激活的标记量化方法。在此方法的基础上，ZeroQuant-v2提出了低秩补偿（LoRC）技术，采用低秩矩阵来减轻量化的不准确性。SmoothQuant引入比例因子，扩大了权重通道的数据范围，缩小了相应激活通道的数据范围。

APQ-ViT[75]提出了Matthew-effect Preserving Quantization（MPQ）方法，更好地保留了激活函数的幂律分布特性，并采用了Blockwise Bottom-elimination Calibration（BBC）方案优化校准指标。

ResQ[76]则是一种新的混合精度量化方法，利用主成分分析（PCA）找出激活方差最高的低秩子空间（通常是隐藏维度的1/8），并将这些系数保持在高精度（如8位），其余部分则量化为4位。ResQ在每个子空间内应用随机旋转来抑制异常值，从而降低量化误差。ResQ无需基于梯度的优化，因此计算需求更低。此外，一些研究还专注于如何在不同硬件平台上实现高效的模型部署。例如，LSAQ[77]可根据边缘设备资源动态调整量化策略。LSAQ通过评估各层的重要性，并为不同重要性的层分配不同精度，降低了存储需求并保持了模型性能；它还设计了安全内存策略以应对推理过程中的

内存占用，确保模型在资源受限环境中高效运行，为大语言模型在边缘设备上的部署提供了新方法。

4.1.3 量化感知训练

与训练后量化不同，量化感知训练在模型训练过程中考虑量化的影响，通过引入量化操作模拟推理时的量化误差，使得模型在训练时适应量化策略，从而提高模型在量化后的性能，避免出现严重的精度损失。量化感知训练需要更多的训练资源，往往需要对模型进行全参数再训练，但是再训练大模型的成本过高，需要寻求更加高效的优化策略或改进方法，以降低成本。

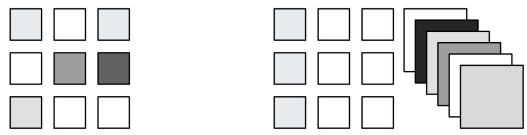
为了减少对数据的需求，LLM-QAT[78]创新性地引入了一种无数据的方法来生成训练数据。同时，LLM-QAT 通过量化权重、激活值和键值缓存提高了模型的吞吐量，展示了将模型量化至 4 位的可行性。为了减少计算量，许多方法采用参数有效调谐（PEFT）策略来加速 QAT。PEQA[79]方法中首先将连接层的参数矩阵量化为低位整数，然后针对特定的下游任务进行微调。QLoRA[80]首先将 LLM 的权重量化为 4 位，然后通过 LoRA[81]方法在 BF16 格式下对每个 4 位权重矩阵进行微调，从而使得量化后的大模型能够在单个 GPU 上进行微调。Norm Tweaking[82]方法中仅对 LayerNorm 层进行训练，冻结其他权重；利用知识蒸馏将量化模型的输出分布与 FP16 模型的输出分布进行匹配，生成校准数据集来减少量化模型对特定数据集的依赖。

Q-ViT[83]提出了信息矫正模块（IRM），用于最大化量化注意力模块中的信息熵；并通过分布引导蒸馏（DGD）方案，在优化过程中消除量化模型和全精度模型之间的分布差异。

4.2 模型剪枝

大模型在大多数时间都存在一定的冗余的、可去除的参数。剪枝的目的就是识别并去除模型中不重要的多余组件，从而减少神经网络模型的参数量和计算量，降低模型推理中的内存成本，并提高计算速度。从优化的角度看，可以将模型剪枝定义为一个约束优化问题：在特定的约束条件下，寻找最优的模型子结构和相应的权重，最优化剪枝模型的性能。在实际应用场景中，针对大模型的剪枝优化问题，需要考虑到端侧设备的计算能力和存储空间。为了适应这些实际应用中的限制条件，通常需要在优化问题中设置资源限制，并求解带有约束的剪枝优化问题。

基于现有的研究结果可以发现，尽管剪枝在小规模、中等规模的语言模型中效果良好，但由于针对大语言模型的剪枝后微调成本高，因此难以有效优化剪枝后模型的性能。针对中等规模语言模型的剪枝方法大多采用剪枝后微调的方法来提高压缩后模型的性能。由于参数量巨大，针对大语言模型的全微调方法成本过高。为了解决这个问题，某些修剪方法选择结合参数高效调优技术来降低微调成本。另外的方法放弃了微调，转而致力于优化修剪过程来保留模型性能。通常，根据权重单元的粒度和结构化，可以将剪枝方法分为两种类型：非结构化剪枝和结构化剪枝（见图 5）。非结构化剪枝通过删除特定参数减小模型复杂度，而结构化剪枝则通过删除神经元、通道或层等结构组件来简化模型。



（a）非结构化剪枝　　（b）结构化剪枝

图 5 模型剪枝

Fig. 5 Model pruning

4.2.1 非结构化剪枝

非结构化剪枝的稀疏模式的剪枝方法采用的权重单元粒度为权重粒度，粒度最细，如图 5（a）所示。与结构化剪枝方法相比，这类剪枝方法剪枝算法简单，随机对独立的权重或者神经元链接进行剪枝，被修剪的权重值归零。SparseGPT[84]和 Wanda[85]是 LLMs 非结构化剪接方法的两种先驱。SparseGPT 是一种针对大规模语言模型的稀疏化技术，旨在通过引入稀疏性来减少模型的参数数量。SparseGPT 在剪枝过程中使用了精心设计的剪枝和优化策略，在保持模型性能的同时，显著减少了计算资源的消耗。LoRAPrune[86]将参数高效调整（PEFT）方法与剪枝相结合，以提高下游任务的性能。Wanda 基于模型在特定任务或数据上的表现动态调整剪枝策略，使得模型在不同任务和数据集上保持较好的泛化性。

非结构化剪枝方法通常能获得较高的压缩率，并大幅度减少模型权重的存储需求。但是，非结构化剪枝得到的模型的稀疏性不具有结构化规律，导致在推理过程中引起不规则的访存和计算。这样的压缩率难以转化为推理加速效果，并且剪枝后模型的精度是不可控的。非结构化修剪通常需要在剪枝后对 LLMs 进行大量的重新训练，以恢复模型的准确性，这对于训练复杂度极高的 LLMs 来说训练成本高昂。因此，常规的非结构化剪枝方法并不能用于大模型的低成本部署。

在非结构化剪枝方法的基础上，最近的研究工作开始探索如何更有效地利用剪枝后的模型激活信息来优化剪枝过程。例如，NEURONAL 方法[87]提出了基于零阶自适应神经元对齐的剪枝策略，利用密集预训练模型的激活信息来生成稀疏模型，并最大化稀疏模型与密集模型之间的神经元对齐，从而在不重新训练的情况下实现高性能的模型压缩。通过自适应选择块状和行状稀疏比例，NEURONAL 在多个大规模语言模型家族和语言任务中表现出优越的性能-运行时权衡。这表明，结合模型激活信息和自适应调整剪枝策略，可能是未来非结构化剪枝方法的一个重要研究方向。

4.2.2 结构化剪枝

结构化剪枝采用的权重单元粒度较粗，包括通道、形状、滤波器和层级等粒度，如图 5（b）所示。大部分结构化剪枝算法在 channel 或者 layer 上进行剪枝，保留了原始卷积结构，得到的模型的稀疏性具有结构性规律。结构化剪枝方法虽然相比于非结构化剪枝方法压缩率更低，但是通常无需专用的硬件就能得到实际的硬件加速效果。

LLM-Pruner[88]是针对 LLMs 的结构化修剪方法的早期代表性方法。其核心思想是采用了一种依赖性检测算法在修剪过程中动态去除非关键耦合结构。具体而言，LLM-Pruner 利用泰勒展开式将权重的重要性表述为损失的变化，通过使

用 Hessian 矩阵的对角线近似 Fisher 信息矩阵的一阶导数，来评估各个权重的重要性。随后，通过求和或其他方法汇总一组权重的重要性，以确定该组权重的重要性，并根据预定义的修剪比例去除重要性较低的权重组。调优过程中，LLM-Pruner 采用了如 LoRA 等参数高效调优技术，利用少量数据对修剪后的模型进行快速微调。MorphNet[89] 通过自动化地调整神经网络的结构，利用稀疏化和膨胀策略，根据反向传播过程中计算得到的梯度信息，逐层分析和调整神经网络的宽度和深度。通过减少计算资源利用效率较低的神经元数量，并在必要时增加关键层的神经元数量，MorphNet 显著降低了计算成本和内存占用。NetAdapt[90] 通过逐层剪枝和迭代优化，在资源限制下调整深度神经网络的结构。它评估每层的计算成本和性能影响，根据特定的资源约束条件（如计算时间和内存使用）进行优化。通过每次迭代后的性能评估和反馈机制，NetAdapt 逐步改进剪枝策略，显著减少计算资源和内存占用，同时尽量保持或提高模型性能，使其适用于资源受限的设备和应用场景。LLM Surgeon[91] 将基于块对角因子化近似的方法扩展到大语言模型，并创新性地考虑了权重之间的相关性，在多轮修剪之间更新权重和曲率估计。

DCT-ViT[92] 方法通过引入离散余弦变换，将输入图像的空间信息转换到频域，识别并删除影响较小的高频信息，从而在保持模型性能的同时显著减少计算成本。模型采用分阶段架构设计，仅在训练的后期进行高频信息剪枝，以避免过早剪枝导致准确率下降。X-Pruner[93] 提出了一种可解释性感知的剪枝框架，通过可解释性感知掩码评估每个单元对预测类的贡献，学习每层的剪枝阈值和剪枝率正则化项，并通过类智能正则器自适应地调整剪枝速率。

2SSP[94] 是一种新型结构化剪枝方法，结合了宽度剪枝和深度剪枝的优势。它先通过宽度剪枝去除 FFN 中的神经元，再用深度剪枝去除注意力子模块，有效减少了模型参数和计算量，同时保持网络连通性。实验表明，2SSP 在语言建模和下游任务中的性能优于现有剪枝方法，剪枝时间更短，其稀疏率平衡机制还能确保不同稀疏率下的性能良好。Mamba-Shedder[95] 则是一种针对选择结构化状态空间模型（SSMs）的结构化压缩方法，特别是 Mamba 及其混合模型。Mamba-Shedder 研究这些模型在不同粒度下对选定组件移除的敏感性，旨在减小模型大小和计算开销，同时保持准确性。Multi Pruner[96] 则是通过多维度、迭代的细粒度剪枝策略（包括残差块、多层感知机通道和注意力头），在保持模型结构平衡的同时，实现了更高的压缩率和零样本任务性能。

量化和剪枝是两种常用的神经网络模型压缩技术，旨在减少模型的存储需求、计算成本，并提高推理速度。量化通过将神经网络中的权重和激活值从高精度（如 FP32）转换为低精度（如 INT8），在不显著损失模型性能的情况下减少内存占用和计算负担。而剪枝则是通过去除神经网络中不重要的参数或结构，减小模型的复杂度，从而降低计算资源需求和内存占用。为了便于理解，表 4 对量化和剪枝进行了详细对比，可以看出量化和剪枝各有优缺点。量化方法通常更加高效，尤其是在数据受限或需要快速压缩时，但可能面临较大的精度损失。而剪枝则可以通过去除冗余参数来减轻计算负担，但对于大规模模型而言可能需要更多的训练资源。

表 4 量化和剪枝的对比

Table 4 Comparison of quantization and pruning

方法	描述	优点	缺点
训练后量化（PTQ）	训练完成后进行量化；不需重新训练	无需再训练；适用数据受限场景	可能导致较大量化误差；精度受限
量化感知训练（QAT）	在训练过程中模拟量化误差；优化量化影响	优化量化误差；精度较高	需要大量训练；成本高
非结构化剪枝	删除特定权重或神经元链接；减小模型复杂度	高压缩率；减少存储需求	精度难控制；通常需再训练；访存不规则
结构化剪枝	删除结构组件；保留原始结构	适合硬件加速；提高推理速度	压缩效果低；结构复杂性增加

4.3 低秩分解

低秩分解（Low-Rank Decomposition，LRF）属于结构优化方法的一种，其通过细化模型的体系结构实现模型效率和性能之间的平衡。低秩分解旨在用两个低秩矩阵 $\boldsymbol{B}^{m\times r}$ 和 $\boldsymbol{C}^{r\times n}$ 来近似一个矩阵 $\boldsymbol{A}^{m\times n}$，如式(5)所示：

$$\boldsymbol{A}^{m\times n}\approx\boldsymbol{B}^{m\times r}\times\boldsymbol{C}^{r\times n} \tag{5}$$

低秩分解可以减少需要加载的参数数量，从而加速解码速度。截断奇异值分解是一种经典的低秩分解方法。除此之外，还有一些常见的分解方法。CP 分解[97] 的基本思想是将一个高阶张量分解成多个低阶张量的外积。Tucker 分解[98] 是一种更通用的张量分解方法，也称为高阶主成分分析（Higher Order PCA），它将张量表示为一个核心张量和一组因子矩阵的乘积。如图 6 所示，LoRA 通过将大规模预训练模型中的权重矩阵进行低秩分解，只学习两个小矩阵来适应新任务，从而减少参数更新的数量并提高计算效率。

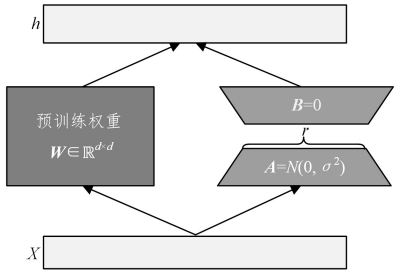


图 6 LoRA 低秩分解

Fig. 6 LoRA low-rank decomposition

LPLR[99] 和 ZeroQuant-V2[70] 使用量化方法压缩权重矩阵后，通过低秩补偿（LoRC）来恢复模型性能。LoRD[100] 将 LLMs 的密集权重矩阵分解为两个较小的密集矩阵的乘积，利用高度优化的浮点密集矩阵乘法内核，并保持模型的完全可微分性能。SVD-LLM[101] 基于变换后的权重矩阵的奇异值与压缩损失之间的关系，设计了一种截断感知的数据白化技术，以识别移除后造成最小损失的奇异值。

FLORA[102] 基于神经架构搜索（NAS）的端到端自动化框架，通过低秩感知候选过滤策略有效识别并消除表现不佳的候选项；除此之外，还设计了特定的低秩训练范式，采用权重继承构建超网络，并实现低秩模块间的梯度共享。FLORA 方法在 DeiT-B 和 Swin-B 模型上节省了 55% 和 42% 的 FLOPs 使用量。

4.4 知识蒸馏

知识蒸馏(Knowledge Distillation，KD)是一种重要且成熟的模型压缩技术，其核心思想是使用一个参数量较大的教师模型作为指导，来提炼较小的学生模型。在蒸馏的过程中，教师模型的知识被转移到学生模型上。通常来说，一个知识蒸馏系统由以下 3 部分组成：知识(knowledge)、蒸馏算法(distillation algorithm)和教师-学生架构(teacher-student architecture)。

其实，在大语言模型涌现之前，知识蒸馏的想法和技术就已经被提出。2006 年，Bucilua 等[103]最先提出将大模型的知识迁移到小模型。2015 年，Hinton 等[104]正式提出知识蒸馏的概念。早期的知识蒸馏研究重点在于将复杂冗长的神经网络中的知识转移到更高效轻量的架构中，从而在资源受限环境下实现深度学习模型的部署。大语言模型的知识蒸馏方法与较小模型有明显不同。相比于小规模语言模型，大语言模型通常具有更加优秀的性能，但是大语言模型对用户开放的权限有限，这使得之前很多传统的知识蒸馏方法和思路都不能直接在大语言模型上使用。

根据学生模型能够从教师模型得到的知识类型，知识蒸馏方法可以分为两种主要类型：黑盒知识蒸馏和白盒知识蒸馏。由于大部分 LLMs 的闭源特性所施加的限制，针对大语言模型的蒸馏方法研究中，基于黑盒的方法的研究要远比白盒方法丰富。

图 7 展示了白盒蒸馏和黑盒蒸馏的两种知识蒸馏方法。白盒蒸馏中，教师模型的结构和内部激活函数对学生模型是可见的，学生通过学习教师模型的特征和激活函数输出，能够更精确地模仿教师模型的行为。在黑盒蒸馏中，学生模型只通过教师模型的最终输出进行学习，教师模型的内部细节对学生不可见。两者的主要区别在于，白盒蒸馏允许学生模型获得更多的细节信息，而黑盒蒸馏则依赖于教师模型的最终结果来指导学习。

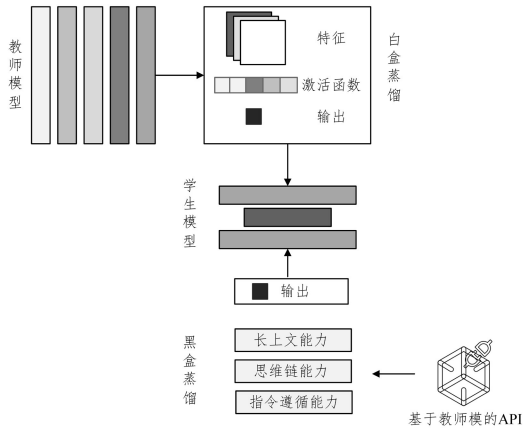


图 7 知识蒸馏

Fig. 7 Knowledge distillation

4.4.1 黑盒知识蒸馏

黑箱知识蒸馏方法中，学生模型无法访问教师模型的内部结构和参数，因此仅使用教师模型输出的最终结果来进行学习。尽管如此，利用大语言模型处理复杂问题的出色能力，还是可以设计出多种黑盒条件下的方法来训练学生模型学习到教师模型各个方面的能力。大语言模型通常具有很强的通用性和泛化能力。因此，针对大语言模型的蒸馏任务，我们不仅期待学生模型学习到固定数据集里的知识，还希望其能够继承大语言模型的各种突出性能，比如上下文学习能力、涌现能力和特定领域的任务处理能力等。

1)上下文学习。上下文学习(In-context Learning，ICL)是大型模型能力的一种重要表现[105]，指的是大型模型在不更新模型参数的情况下，根据部分输入和对应响应示例，为新的输入生成正确输出的能力。上下文学习的蒸馏旨在为较小的模型注入大模型的上下文跟踪理解功能，将大语言模型处理复杂指令、对话历史等复杂问题的能力转移到学生模型中。In-context Learning Distillation[106]方法介绍了两种上下文学习范式：元上下文微调(Meta-ICT)和多任务上下文微调(Multitask-ICT)。Multitask-ICT 的性能优于 Meta-ICT，但需要更多计算。Meta-ICT 通过在元学习任务上进行上下文学习的微调训练来适应目标任务；Multitask-ICT 则直接在目标任务上进行微调训练来学习任务目标。在 Meta-ICT 中，学生模型在大量的元任务上进行训练，学习快速适应新任务的能力，从而得以在少量样本帮助下迅速适应目标任务。而在 Multitask-ICT 中，学生模型直接使用目标任务中的示例学习目标任务本身，并捕捉同一任务中数据点之间的关系。文献[107]则在知识蒸馏中对教师模型采用特定的提示策略，促使教师模型输出更具连贯性和逻辑性的推理步骤，这实际上也是在强化学生模型对上下文的理解和学习能力，使其能够更好地把握问题的来龙去脉，进而生成更符合逻辑和语境的输出。这进一步印证了在知识蒸馏过程中对上下文学习能力进行蒸馏和传承的可行性和重要性。

2)特定领域任务处理。大语言模型中的部分模型是针对某个领域定制训练得到的，在特定领域有着强大的性能。LawGPT[108]建立在 OpenLLAMA 的基础上，通过构建结合了现实世界的法律法规和相关的司法解释资料的数据集来对学生模型进行训练。此外，作者利用 ChatGPT API 辅助数据集的构建，对现有数据的基础集生成补充数据。DISCO[109]使用 LLM 获得反事实数据，并使用特定任务的教师模型对生成扰动进行过滤和提炼，从而生成高质量的对抗性样本数据。但是，Jiang 等[110]指出了上述方法的一个隐患：这样的训练可能导致学生模型缺乏整合反馈的能力，只是基于教师模型的行为进行模仿。

4.4.2 白盒知识蒸馏

白盒知识蒸馏指的是可以访问教师模型的结构和参数的蒸馏方法。这种方法能够有效地利用教师模型的参数特征和输出逻辑来增强学生模型的性能。MiniLLM[111]提出采用标准白盒方法，但将标准方法中的正向 Kullback-Leibler 散度替换为逆向 KLD，防止学生模型高估教师分布的低概率区域，促使学生专注于教师分布的主要模态，生成更准确的响应。相似地，GKD[112]利用反向 KLD 和 Jensen-Shannon 散度(JSD)来帮助学生模型学习重要信息，并使用 onpolicy KD 来缓解训练和评估之间的分布不匹配问题。TED[113]提出了一种任务感知的分层 KD 方法，在教师和学生模型的每一层之后添加过滤器，并训练学生过滤器的输出特征与相应的教师过滤器对齐，从而达到学习目的。MiniMoE[114]通过使用混合

专家(MoE)模型,有效提高了学生模型的性能。

DearKD[115]提出了一种两阶段的学习框架。在第一阶段,通过多头卷积注意力机制捕获归纳偏置,利用 CNN 的中间层和输出层的特征信息指导 Transformer 的初期学习;在第二阶段,则让 Transformer 自主学习归纳偏差,充分利用其灵活性和表达能力。Multimodal Distillation[116]训练单模态(RGB 帧)的学生模型通过最小化预测概率分布与教师模型预测概率分布之间的 KL 散度,促使学生模型学习到多模态(RGB、光流、音频等)教师模型的性能。

5 系统优化

在深入探讨大模型高效推理技术的过程中,系统优化的关键作用逐渐凸显。随着推理效率的不断优化,一系列创新技术应运而生,它们不仅极大地提高了硬件资源的利用效率,还显著增强了 LLMs 的推理速度和内存管理能力。

5.1 推理引擎框架

在探索高效推理技术的实际应用场景时,系统优化扮演着至关重要的角色,特别是在系统级推理效率的优化方面,出现了一系列的创新技术。例如,FlexGen[117]推理引擎通过采用线性规划的搜索方法,有效地协调了 GPU、CPU 和磁盘等硬件资源,实现了在资源受限的 GPU 上对 LLMs 的高效推理。通过量化权重和注意力缓存,FlexGen 显著提高了特定模型的推理速度,同时减少了内存的使用。

Orca[118]和 S^3[119]通过迭代级调度和预见序列长度的方法,优化了 GPU 的利用率,显著提高了吞吐量。然而,这些技术也面临着内存碎片化的问题。vLLM[120]通过 PagedAttention 技术解决了这一问题,允许在非连续的内存空间中存储连续的键和值,从而提高了内存效率和推理速度。

在追求高效率的同时,用户体验和服务质量(QoE)同样不容忽视。Andes[121]通过定义基于 LLMs 的文本流服务的 QoE,并提出一种 QoE 感知的服务系统,优化了用户体验。DeepSpeed[122]推理技术通过多 GPU 推理方法,增强了密集和稀疏 Transformer 模型的效率,并提供了一种混合推理技术,利用 CPU 和 NVMe 内存,进一步提高了推理速度。

Flash-Decoding[123]及其改进版本 FlashDecoding++[124]通过分解键/值并行计算注意力,然后组合生成最终输出,显著提高了长上下文推理的速度。FlashDecoding++通过一系列优化技术,在 Nvidia 和 AMD GPU 上显著加快了推理速度,显示出在相同吞吐量下更明显的速度优势。这些技术的发展不仅推动了高效推理技术的进步,也为实际应用场景中的系统优化提供了有力的支持。以上推理引擎的主要特点如表 5 所列。

表 5 推理引擎

Table 5 Inference engines

推理引擎框架	主要特点
FlexGen	资源协调;量化权重
Orca	GPU 优化;迭代调度
S^3	吞吐量提高;预见序列
vLLM	内存效率;Paged Attention
Andes	用户体验;QoE 感知
DeepSpeed	多 GPU 推理;混合推理
Flash-Decoding	键值并行;长上下文

5.2 内存管理

在高效推理技术的实际应用中,内存管理是一个至关重要的环节,它直接影响到 LLMs 服务的效率和性能。在处理长上下文时,KV 缓存的存储对内存的需求尤为显著。传统方法通常基于预设的最大请求长度进行 KV 缓存的存储空间分配,但这种方法在请求提前终止时会导致内存资源的浪费。为了解决这一问题,S^3提出了预测生成长度上限的方法,以减少预分配空间的浪费。即便如此,当遇到存储碎片化问题时,静态分配方式仍显得力不从心。

vLLM 技术通过分页存储 KV 缓存,有效地解决了内存碎片化的问题。它将一个大的内存空间划分为多个物理块,动态地将 KV 缓存映射到这些物理块中,从而显著减少了存储碎片化,提高了 LLMs 服务的吞吐量。LightLLM[125]在此基础上进一步细化,采用基于 token 级别的内存管理机制,减少了与不规则边界相关的资源浪费,确保了 KV 缓存的充分利用。CAKE[126]则进一步优化 KV 缓存的分配,通过分析模型各层的注意力动态,提出了一种“蛋糕切割”问题的解决方案,以更精细的方式管理 KV 缓存。

然而,分页存储方式也带来了新的挑战。它导致注意力操作符中的内存访问变得不规则,这要求我们重新考虑 KV 缓存虚拟地址空间与物理地址空间之间的映射关系。为了提高注意力操作符的效率,必须设计出便于连续内存访问的加载模式。例如,vLLM 的 PagedAttention 将 K 缓存的头部大小维度结构化为连续向量;FlashInfer[127]则根据设计的内存访问方案,优化了 KV 缓存的数据布局。

针对当前的长文本场景,SCOPE[128]更关注长文本生成任务中预填充和解码阶段的 KV 缓存优化,通过分别处理这两个阶段的缓存,来减少长输出任务中的缓存偏差。MEDA[129]则专注于多模态长文本场景下,如何根据跨模态注意力熵来进行动态的 KV 缓存分配。

尽管如此,高效的内存管理仍然是 LLMs 服务中的一个主要挑战。随着对长序列推理需求的增加,KV 缓存的内存占用成为优化的主要目标。在增量解码过程中,KV 缓存的内存动态变化,传统方法的连续内存块预分配方案在多种情况下会导致内存浪费。vLLM 的分页注意力、SpecInfer 的树注意力以及 LightLLM 的基于 token 级别的内存管理机制,都是为减少内存使用而提出的创新方法。然而,这些方法的碎片化内存管理机制可能会增加推理延迟,尤其是在提高批处理大小时,细粒度的内存管理方法可能只带来有限的吞吐量优势。因此,在 LLMs 推理中,如何在减少内存使用与其他算法创新和系统级优化之间找到平衡,是一个需要持续探索的挑战。

5.3 并行计算

在探索 LLMs 的高效并行计算策略时,首先聚焦于模型并行技术。图 8 展示了数据通过多个 GPU 并行计算进行分配和处理,以加速神经网络的训练过程。这些技术最初为分布式训练深度神经网络特别是基于 Transformer 的模型而设计。张量模型并行[130](TP)通过将模型层分割并部署在不同设备上,显著降低了推理延迟。例如,TP 在具有高速 NVLink 连接的多 GPU 环境中被广泛应用。PaLM[131]推理技术

进一步扩展了 TP，采用 2D 张量并行，声称在超过 256 个设备的大集群中具有更低的理论通信复杂性。对于多查询注意力，PaLM 还涉及到数据并行与混合张量分区策略。管道模型并行[132](PP)将模型层顺序排列在多个设备上，虽然增加了吞吐量，但并未从根本上减少处理单个输入所需的时间。而序列并行[133](SP)则通过将长序列的处理分布到多个 GPU 上，分担了计算和存储负载。

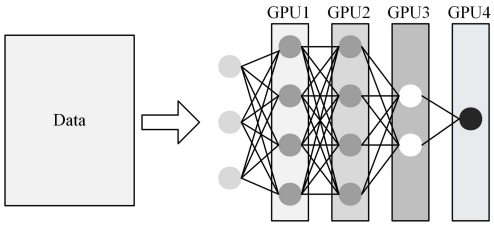


图 8 并行计算

Fig. 8 Parallel computing

为了实现最佳性能和资源利用，自动并行化技术如 Alpa[134]，FlexFlow[135]和 Galvatron[136]等被广泛研究。这些技术通过调整成本模型，在适应自回归推理 Transformer 模型的可预测运行时，使得动态规划和整数线性规划等算法易应用于 LLMs 服务，如 AlpaServe[137]，FlexFlow-Serve[138]和 SpotServe[139]，从而确定最有效的并行策略，无需手动干预。同时，还有方法支持使用更大但速度较慢的存储器，如 CPU DRAM，来保存模型参数和 KV 缓存，以补充有限的设备存储器，如 GPU DRAM。

分散推理作为一种结合了模型和数据并行的方法，允许多个分散的节点协作处理数据和推断输出。这种方法在硬件资源分布广泛的场景中尤其有用。例如，Petals[140]项目利用互联网上的普通 GPU，为 BLOOM 176B 模型提供了服务。尽管分散推理利用了被忽视的消费级 GPU 来运行 LLMs，但它也面临着设备异构性、有限的计算和内存容量、低带宽网络、容错性和隐私保护等方面的实际挑战。这些挑战需要通过创新的解决方案来克服，以实现更广泛的并行计算应用。

6 未来发展与挑战

随着大模型技术的迅猛发展，其在自然语言处理、智能推荐等多个领域的应用日益广泛。高效推理技术作为大模型实际应用的核心，正受到越来越多的关注。本文综述了当前大模型高效推理技术的主要进展，并对未来发展的方向和可能遇到的挑战进行了深入探讨。

未来的算法优化将深化现有技术，并探索新的推理方法。例如，通过引入更先进的神经网络架构和元学习机制，提高模型对长距离依赖的捕捉能力，降低计算复杂度。未来的研究还可能会设计出更具弹性的架构，或探索模型的动态适应性，使其能够根据输入数据的复杂度自动调整计算资源分配，从而在保证推理速度的同时提高模型性能。

参数优化是提高大模型推理效率的另一关键方向。在数亿级别参数量的大模型的压缩任务中，量化方法面临的精度下降的问题尤为突出。未来研究需要探索更优化的量化策略，以最大限度地保持模型的推理性能。剪枝技术通过去除冗余的神经元或连接来减少计算负担，未来可针对自适应的动态剪枝方法进一步探索，使模型结构能根据实际需求实时调整，从而在不同场景中实现性能与效率的最佳平衡。资源受限环境中的大模型部署对高效推理技术提出了更高要求，需要在降低计算和存储开销的同时，确保模型的实时性和稳定性，并在多样化的应用场景中保持较高的适应能力。

系统优化在大模型推理中的发展也面临诸多挑战。随着模型规模和复杂度的增加，平衡内存使用、计算效率与推理延迟的同时，实现更高效的内存管理和推理速度优化，将变得更加艰难。此外，自动化和分散推理技术的应用也带来了设备异构性、容错性等一系列问题。未来的研究需要在这些方面持续探索，以推动实际应用场景中各种优化方法的进一步改进和扩展。

1)降低模型压缩成本。考虑到实际应用场景中计算资源的限制，如何在有限的硬件条件下有效压缩模型，同时维持模型性能，是一个亟待解决的重要问题。未来研究需要在多个方面取得突破，包括开发更高效的压缩算法，探索新的权重量化方法，以及优化模型架构以适应资源受限的环境。在确保模型精度和推理速度的前提下，显著降低计算和存储成本，不仅能推动大规模模型在更广泛领域的应用，也将有助于人工智能技术的普及和落地。

2)多模态压缩技术的发展。跨领域融合为大模型的高效推理技术开辟了新的应用前景。然而，目前各种模型压缩方法在跨领域或跨任务迁移时效果仍不理想。未来的发展可能会进一步推进跨模态、多模态的模型压缩技术，提高压缩方法对不同数据类型的适应性。例如，结合自然语言处理与计算机视觉技术，开发适应文本数据、图像数据甚至视频数据等多种模态数据的压缩方法。这将可能为智能机器人、自动驾驶等复杂的前沿领域拓展创新的应用场景。

3)模型高效推理技术的社会安全与伦理考量。随着模型高效推理技术的发展，我们也需要关注其对环境和社会的影响，包括模型训练和推理过程中的能源消耗问题，以及符合伦理标准的应用考量，避免加剧社会不平等和歧视问题。推理过程中的安全性问题同样不容忽视，例如数据信息泄露和对抗性干扰和攻击等。如何在压缩过程中保护数据隐私，提高模型的安全性，并应对潜在的偏见，将是未来研究的重要课题。在大模型的应用中，后门攻击(Backdoor Attack)是一种严重的安全威胁。例如文献[141]中的方法，攻击者可以在模型训练阶段嵌入恶意触发器，当模型在实际应用中遇到特定触发条件时，攻击者将执行预定的恶意操作。对于大语言模型及其衍生的智能代理(如自动化助手)，这类攻击的危害尤为显著。攻击者能够在模型的训练数据中注入看似无害的触发器，待模型部署后，通过特定输入或环境变化激活后门，使得智能代理执行有害的行为，如删除文件，执行恶意代码，或进行不正当的购买行为等。

总之，针对大模型的实际应用场景，未来的高效推理方法的发展将聚焦于智能化、多样化与安全性的综合平衡，通过技术创新和跨领域合作，推动大模型在广泛应用中的进一步优化。持续的研究和创新，是推动大模型技术发展和跨越障碍的关键。通过不断的技术革新和优化，我们期待大模型技术

能够在更广泛的实际应用场景中发挥更加重要的作用，为社会带来更深远的影响。

结束语 本文综述了 LLMs 在实际应用场景中的高效推理技术。LLMs 以其在语言理解、生成和推理任务中的卓越表现，成为各行业的关键技术。面对模型规模庞大带来的效率挑战，学术界和工业界的研究者们通过算法优化、参数优化和系统优化 3 个层面，开展了一系列提高推理效率的工作。算法优化通过改进输入输出和解码方法减少了计算成本；参数优化利用剪枝和量化技术缩减模型规模，推动了模型的轻量化部署；系统优化则借助并行计算和内存管理技术，提高了资源利用率和模型运行速度。展望未来，我们预见 LLMs 推理技术将在硬件加速、自动化模型优化和神经网络架构创新等方面取得新进展。同时，我们指出了实现这些技术可能面临的安全性、可解释性以及环境与社会影响等挑战。最终，我们强调了持续研究和创新在推动 LLMs 实际应用中的关键作用。本综述旨在为研究人员和工程师提供全面视角，帮助他们把握 LLMs 推理效率挑战，促进技术进步和创新，期望未来的研究能进一步提高 LLMs 在资源受限环境中的实用性和可访问性，从而为社会创造更大的价值。

参 考 文 献

[1] CHANG Y P, WANG X, WANG J D, et al. A survey on evaluation of large language models[J]. arXiv: 2307.03109, 2023.

[2] OpenAI. 2023. Introducing ChatGPT[EB/OL]. https://openai.com/blog/chatgpt.

[3] Microsoft. Announcing microsoft copilot, your everyday aicompanion[EB/OL]. [2023-12-04]. https://blogs.microsoft.com/blog/2023/09/21/announcing-microsoft-copilot-your-everyday-ai-companion/.

[4] TOUVRON H, LAVRIL T, IZACARD G, et al. Llama: Open and efficient foundation language models [J]. arXiv: 2302.13971, 2023.

[5] KACHRIS C. A survey on hardware accelerators for large language models[J]. arXiv: 2401.09890, 2024.

[6] ZHU X, LI J, LIU Y, et al. A survey on model compression for large language models[J]. arXiv: 2308.07633, 2023.

[7] SHAO R R, LIU Y, ZHANG W, et al. A Survey of Knowledge Distillation in Deep Learning [J]. Journal of Computer Science, 2022, 45(8): 1638-1673.

[8] HUANG Z H, YANG S Z, LIN W, et al. A Survey of Knowledge Distillation [J]. Journal of Computer Science, 2022, 45(3): 624-653.

[9] PARK S, CHOI J, LEE S, et al. A comprehensive survey of compression algorithms for language models[J]. arXiv: 2401.15347, 2024.

[10] KHOSHNOODI M, JAIN V, GAO M, et al. A comprehensive survey of accelerated generation techniques in large language models[J]. arXiv: 2405.13019, 2024.

[11] WANG W, CHEN W, LUO Y, et al. Model compression and efficient inference for large language models: A survey[J]. arXiv: 2402.09748, 2024.

[12] ZHOU Z, NING X, HONG K, et al. A survey on efficient inference for large language models[J]. arXiv: 2404.14294, 2024.

[13] VASWANI A, SHAZEER N, PARMAR N, et al. Attention is all you need[C]// Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17). 2017: 6000-6010.

[14] BROWN T, MANN B, RYDER N, et al. Language models are few-shot learners[J]. Advances in Neural Information Processing Systems, 2020, 33: 1877-1901.

[15] GAO Y, XIONG Y, GAO X, et al. Retrieval-augmented generation for large language models: A survey [J]. arXiv: 2312.10997, 2023.

[16] ZHOU W, JIANG Y E, COTTERELL R, et al. Efficient prompting via dynamic in-context learning [J]. arXiv: 2305.11170, 2023.

[17] YIN F. VIG J, LABAN P, et al. Did you read the instructions? rethinking the effectiveness of task defi nitions in instruction learning[J]. arXiv: 2306.01150, 2023.

[18] JUNG H, KIM K J. Discrete prompt compression with reinforcement learning[J]. IEEE Access, 2024, 12: 72578-72587.

[19] XU F, SHI W, CHOI E. Recomp: Improving retrieval-augmented lms with compression and selective augmentation [J]. arXiv: 2310.04408, 2023.

[20] LISKAVETS B, ROY S, USHAKOV M, et al. Task-agnostic Prompt Compression with Context-aware Sentence Embedding and Reward-guided Task Descriptor [J]. arXiv: 2502.13374, 2025.

[21] WINGATE D, SHOEYBI M, SORENSEN T. Prompt compression and contrastive conditioning for controllability and toxicity reduction in language models[J]. arXiv: 2210.03162, 2022.

[22] MU J, LI X, GOODMAN N. Learning to compress prompts with gist tokens[J]. Advances in Neural Information Processing Systems, 2023, 36: 19327-19352.

[23] CHEVALIER A, WETTIG A, AJITH A, et al. Adapting language models to compress contexts [J]. arXiv: 2305.14788, 2023.

[24] GE T, HU J, WANG L, et al. In-context autoencoder for context compression in a large language model[J]. arXiv: 2307.06945, 2023.

[25] WANG H, ZHANG Z, HAN S. Spatten: Efficient sparse attention architecture with cascade token and head pruning[C]// 2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA). IEEE, 2021: 97-110.

[26] KIM S, SHEN S, THORSLEY D, et al. Learned token pruning for transformers[C]// Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2022: 784-794.

[27] WANG Z, CHEN J, ZHOU W, et al. Smarttrim: Adaptive tokens and attention pruning for efficient vision-language models [J]. arXiv: 2305.15033, 2023.

[28] FEDERICI M, BELLI D, VAN BAALEN M, et al. Efficient llm

inference using dynamic input pruning and cache-aware masking[J]. arXiv:2412.01380,2024.

[29] JIANG Z,XU F F,GAO L,et al. Active retrieval augmented-generation[C]// Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. 2023:7969-7992.

[30] SHI W,MIN S,YASUNAGA M,et al. Replug:Retrieval-augmented black-box language models[J]. arXiv: 2301. 12652, 2023.

[31] ASAI A,WU Z,WANG Y,et al. Self-rag:Learning to retrieve, generate,and critique through self-reflection[J]. arXiv: 2310. 11511,2024.

[32] XIN J,TANG R,LEE J,et al. DeeBERT:Dynamic early exiting for accelerating BERT inference[J]. arXiv:2004.12993,2020.

[33] SCHWARTZ R,STANOVSKY G,SWAYAMDIPTA S,et al. The right tool for the job:Matching model and instance complexities[J]. arXiv:2004.07453,2020.

[34] ZHOU W,XU C,GE T,et al. Bert loses patience:Fast and robust inference with early exit[J]. Advances in Neural Information Processing Systems,2020,33:18330-18341.

[35] ZHANG Z,ZHU W,ZHANG J,et al. PCEE-BERT:Accelerating BERT inference via patient and confident early exiting [C]// Findings of the Association for Computational Linguistics:NAACL 2022. 2022:327-338.

[36] WANG J,CHEN K,CHEN G,et al. Skipbert:Efficient inference with shallow layer skipping[C]// Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics(Volume 1:Long Papers). 2022:7287-7301.

[37] DIN A Y,KARIDI T,CHOSHEN L,et al. Jump to conclusions:Short-cutting transformers with linear transformations[J]. arXiv:2303.09435,2023.

[38] SCHUSTER T,FISCH A,GUPTA J,et al. Confident adaptive language modeling[J]. Advances in Neural Information Processing Systems,2022,35:17456-17472.

[39] TANG S,WANG Y,KONG Z,et al. You need multiple exiting:Dynamic early exiting for accelerating unifiedvision language model[C]// Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2023:10781-10791.

[40] XU G,HAO J,SHEN L,et al. Lgvit:Dynamic early exiting for accelerating vision transformer[C]// Proceedings of the 31st ACM International Conference on Multimedia. 2023:9103-9114.

[41] MEISTER C,VIEIRA T,COTTERELL R. Best-first beam search[J]. Transactions of the Association for Computational Linguistics,2020,8:795-809.

[42] FAN A,LEWIS M,DAUPHIN Y. Hierarchical neural story generation[J]. arXiv:1805.04833,2018.

[43] HOLTZMAN A,BUYS J,DU L,et al. The curious case of neural text degeneration[J]. arXiv:1904.09751,2019.

[44] WANG X,XIONG Y,WEI Y,et al. LightSeq:A high performance inference library for transformers[J]. arXiv: 2010. 13887,2020.

[45] LI L,LIN Y,CHEN D,et al. Cascadebert:Accelerating inference of pre-trained language models via calibrated complete models cascade[J]. arXiv:2012.14682,2020.

[46] WANG Y,CHEN K,TAN H,et al. Tabi:An efficient multi-level inference system for large language models[C]// Proceedings of the Eighteenth European Conference on Computer Systems. 2023:233-248.

[47] CHEN L,ZAHARIA M,ZOU J. Frugalgpt:How to use large language models while reducing cost and improving performance [J]. arXiv:2305.05176,2023.

[48] YUE M,ZHAO J,ZHANG M,et al. Large language model cascades with mixture of thoughts representations for cost-efficient reasoning[J]. arXiv:2310.03094,2023.

[49] WEI J,WANG X,SCHUURMANS D,et al. Chain-of-thought prompting elicits reasoning in large language models[J]. Advances in neural information processing systems, 2022, 35: 24824-24837.

[50] CHEN W,MA X,WANG X,et al. Program of thoughts prompting:Disentangling computation from reasoning for numerical reasoning tasks[J]. arXiv:2211.12588,2022.

[51] SHAZEER N,MIRHOSEINI A,MAZIARZ K,et al. Outrageously large neural networks:The sparsely-gated mixture-of-experts layer[J]. arXiv:1701.06538, 2017.

[52] XIA H,YANG Z,DONG Q,et al. Unlocking efficiency in large language model inference:A comprehensive survey of speculative decoding[J]. arXiv:2401.07851,2024.

[53] ZHOU Y,LYU K,RAWAT A S,et al. Distillspec:Improving speculative decoding via knowledge distillation[J]. arXiv:2310. 08461,2023.

[54] ZHANG J,WANG J,LI H,et al. Draft & verify:Lossless large language model acceleration via self-speculative decoding[J]. arXiv:2309.08168,2023.

[55] LIU X,HU L,BAILIS P,et al. Online speculative decoding[J]. arXiv:2310.07177,2023.

[56] MONEA G,JOULIN A,GRAVE E. Pass:Parallel speculative sampling[J]. arXiv:2311.13581,2023.

[57] HE Z,ZHONG Z,CAI T,et al. Rest:Retrieval-based speculative decoding[J]. arXiv:2311.08252,2023.

[58] MIAO X,OLIARO G,ZHANG Z,et al. Specinfer:Accelerating generative large language model serving with tree-based speculative inference and verification[J]. arXiv:2305.09781,2023.

[59] FU Y,BAILIS P,STOICA I,et al. Break the sequential dependency of llm inference using lookahead decoding[J]. arXiv: 2402.02057,2024.

[60] CAI T,LI Y,GENG Z,et al. Medusa:Simple llm inference acceleration framework with multiple decoding heads[J]. arXiv: 2401.10774,2024.

[61] LI Y,ZHANG C,ZHANG H. Eagle:Lossless acceleration of llm decoding by feature extrapolation[EB/OL]. [2023-12-08]. https://sites.google.com/view/eagle-llm.

[62] SUN Z,SURESH A T,RO J H,et al. Spectr:Fast speculative decoding via optimal transport[J]. Advances in Neural Information Processing Systems,2023,36:30222-30242.

[63] LI S,CHEN J,SHEN Y,et al. Explanations from large language

models make small reasoners better[J]. arXiv: 2210. 06726, 2022.

[64] YANG G, LO D, MULLINS R, et al. Dynamic stashing quantization for efficient transformer training[J]. arXiv: 2303. 05295, 2023.

[65] CHENG Y, WANG D, ZHOU P, et al. A survey of model compression and acceleration for deep neural networks[J]. arXiv: 1710. 09282, 2017.

[66] FRANTAR E, ASHKBOOS S, HOEFLER T, et al. GPTQ: Accurate quantization for generative pre-trained transformers [C]// The Eleventh International Conference on Learning Representations. 2022.

[67] PARK G, PARK B, KIM M, et al. Lut-gemm: Quantized matrix multiplication based on luts for efficient inference in large-scale generative language models[J]. arXiv: 2206. 09557, 2022.

[68] LIN J, TANG J, TANG H, et al. AWQ: Activation-aware Weight Quantization for On-Device LLM Compression and Acceleration[J]. Proceedings of Machine Learning and Systems, 2024, 6: 87-100.

[69] KIM S, HOOPER C, GHOLAMI A, et al. Squeezellm: Dense-and-sparse quantization[J]. arXiv: 2306. 07629, 2023.

[70] YAO Z, WU X, LI C, et al. Zeroquant-v2: Exploring post-training quantization in llms from comprehensive study to low rank compensation[J]. arXiv: 2303. 08302, 2023.

[71] DETTMERS T, LEWIS M, BELKADA Y, et al. Gpt3. int8(): 8-bit matrix multiplication for transformers at scale[J]. Advances in Neural Information Processing Systems, 2022, 35: 30318-30332.

[72] XIAO G, LIN J, SEZNEC M, et al. Smoothquant: Accurate and efficient post-training quantization for large language models [C]// International Conference on Machine Learning. PMLR, 2023: 38087-38099.

[73] YUAN Z, NIU L, LIU J, et al. Rptq: Reorder-based post-training quantization for large language models[J]. arXiv: 2304. 01089, 2023.

[74] YAO Z, YAZDANI AMINABADI R, ZHANG M, et al. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers[J]. Advances in Neural Information Processing Systems, 2022, 35: 27168-27183.

[75] LIU Z, OGUZ B, ZHAO C, et al. Llm-qat: Data-free quantization aware training for large language models[C]// Findings of the Association for Computational Linguistics: ACL 2024. 2024: 467-484.

[76] SAXENA U, SHARIFY S, ROY K, et al. ResQ: Mixed-Precision Quantization of Large Language Models with Low-Rank Residuals[J]. arXiv: 2412. 14363, 2024.

[77] ZENG B, JI B, LIU X, et al. LSAQ: Layer-Specific Adaptive Quantization for Large Language Model Deployment[J]. arXiv: 2412. 18135, 2024.

[78] LIU S, LIU Z, HUANG X, et al. Llm-fp4: 4-bit floating-point quantized transformers[J]. arXiv: 2310. 16836, 2023.

[79] KIM J, LEE J H, KIM S, et al. Memory-efficient fine-tuning of compressed large language models via sub-4-bit integer quantization[J]. arXiv: 2305. 14152, 2024.

[80] DETTMERS T, PAGNONI A, HOLTZMAN A, et al. Qlora: Efficient finetuning of quantized LLMs[J]. arXiv: 2305. 14314, 2024.

[81] HU E J, SHEN Y, WALLIS P, et al. Lora: Low-rank adaptation of large language models[J]. ICLR, 2022, 1(2): 3.

[82] LI L, LI Q, ZHANG B, et al. Norm tweaking: High-performance low-bit quantization of large language models[C]// Proceedings of the AAAI Conference on Artificial Intelligence. 2024: 18536-18544.

[83] LI Y, XU S, ZHANG B, et al. Q-vit: Accurate and fully quantized low-bit vision transformer[J]. Advances in neural information processing systems, 2022, 35: 34451-34463.

[84] FRANTAR E, ALISTARH D. Sparsegpt: Massive language models can be accurately pruned in one-shot[C]// International Conference on Machine Learning. PMLR, 2023: 10323-10337.

[85] SUN M, LIU Z, BAIR A, et al. A simple and effective pruning approach for large language models[J]. arXiv: 2306. 11695, 2023.

[86] ZHANG M, CHEN H, SHEN C, et al. Loraprune: Pruning meets low-rank parameter-efficient fine-tuning[J]. arXiv: 2305. 18403, 2023.

[87] CUNEGATTI E, CUSTODE L L, IACCA G. Zeroth-Order Adaptive Neuron Alignment Based Pruning without Re-Training [J]. arXiv: 2411. 07066, 2024.

[88] MA X, FANG G, WANG X. Llm-pruner: On the structural pruning of large language models[J]. Advances in neural information processing systems, 2023, 36: 21702-21720.

[89] GORDON A, EBAN E, NACHUM O, et al. Morphnet: Fast & simple resource-constrained structure learning ofdeep networks [C]// Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018: 1586-1595.

[90] YANG T J, HOWARD A, CHEN B, et al. Netadapt: Platform-aware neural network adaptation for mobile applications[C]// Proceedings of the European Conference on Computer Vision (ECCV). 2018: 285-300.

[91] VAN DER OUDERAA T F A, NAGEL M, VAN BAALEN M, et al. The Surgeon[J]. arXiv: 2312. 17244, 2023.

[92] LEE J, KIM H. DCT-ViT: High-Frequency Pruned Vision Transformer with Discrete Cosine Transform[J]. IEEE Access, 2024, 12: 80386-80396.

[93] YU L, XIANG W. X-pruner: explainable pruning for vision transformers[C]// Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2023: 24355-24363.

[94] SANDRI F, CUNEGATTI E, IACCA G. 2SSP: A Two-Stage Framework for Structured Pruning of LLMs[J]. arXiv: 2501. 17771, 2025.

[95] MUÑOZ J P, YUAN J, JAIN N. Mamba-Shedder: Post-Transformer Compression for Efficient Selective Structured State Space Models[J]. arXiv: 2501. 17088, 2025.

[96] MUÑOZ J P, YUAN J, JAIN N. Multipruner: Balanced struc-

ture removal in foundation models[J]. arXiv:2501.09949,2025.

[97] SORBER L,VAN BAREL M,DE LATHAUWER L. Optimization-based algorithms for tensor decompositions:Canonical polyadic decomposition,decomposition in rank-(L_r,L_r,1) terms, and a new generalization[J]. SIAM Journal on Optimization, 2013,23(2):695-720.

[98] MØRUP M,HANSEN L K,ARNFRED S M. Algorithms for sparse nonnegative Tucker decompositions[J]. Neural computation,2008,20(8):2112-2131.

[99] SAHA R,SRIVASTAVA V,PILANCI M. Matrix compression via randomized low rank and low precision factorization[J]. arXiv:2310.11028,2023.

[100]KAUSHAL A,VAIDHYA T,RISH I. Lord:Low rank decomposition of monolingual code llms for one-shot compression[J]. arXiv:2309.14021,2023.

[101]WANG X,ZHENG Y,WAN Z,et al. Svd-llm:Truncation-aware singular value decomposition for large language model compression[J]. arXiv:2403.07378,2024.

[102]CHANG C C,SUNG Y Y,YU S,et al. FLORA:Fine-grained Low-Rank Architecture Search for Vision Transformer[C]// Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. 2024:2482-2491.

[103]BUCILUA C,CARUANA R,NICULESCU-MIZIL A. Model compression[C]// Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2006:535-541.

[104]HINTON G,VINYALS O,DEAN J. Distilling the knowledge in a neural network[J]. arXiv:1503.02531,2015.

[105]DONG Q,LI L,DAI D,et al. A survey on in-context learning [J]. arXiv:2301.00234,2022.

[106]HUANG Y,CHEN Y,YU Z,et al. In-context learning distillation:Transferring few-shot learning ability of pre-trained language models[J]. arXiv:2212.10670,2022.

[107]GOYAL V,KHAN M,TIRUPATI A,et al. Enhancing Knowledge Distillation for LLMs with Response-Priming Prompting [J]. arXiv:2412.17846,2024.

[108]ZHOU Z,SHI J X,SONG P X,et al. LawGPT:A Chinese Legal Knowledge-Enhanced Large Language Model[J]. arXiv:2406.04614,2024.

[109]CHEN Z,GAO Q,BOSSELUT A,et al. DISCO:Distilling counterfactuals with large language models[J]. arXiv:2212.10534, 2022.

[110]JIANG Y,CHAN C,CHEN M,et al. Lion:Adversarial distillation of proprietary large language models[J]. arXiv:2305.12870,2023.

[111]GU Y,DONG L,WEI F,et al. Knowledge distillation of large language models[J]. arXiv:2306.08543,2023.

[112]AGARWAL R,VIEILLARD N,STANCZYK P,et al. Gkd:Generalized knowledge distillation for auto-regressive sequence models[J]. arXiv:2306.13649,2023.

[113]LIANG C,ZUO S,ZHANG Q,et al. Less is more:Task-aware layer-wise distillation for language model compression[C]// International Conference on Machine Learning. PMLR, 2023: 20852-20867.

[114]ZHANG C,YANG Y,LIU J,et al. Lifting the curse of capacity gap in distilling language models[J]. arXiv:2305.12129,2023.

[115]CHEN X,CAO Q,ZHONG Y,et al. Dearkd:data-efficient early knowledge distillation for vision transformers[C]// Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022:12052-12062.

[116]RADEVSKI G,GRUJICIC D,BLASCHKO M,et al. Multimodal distillation for egocentric action recognition[C]// Proceedings of the IEEE/CVF International Conference on Computer Vision. 2023:5213-5224.

[117]SHENG Y,ZHENG L,YUAN B,et al. Flexgen:High-throughput generative inference of large language models with a single gpu[C]// International Conference on Machine Learning. PMLR,2023:31094-31116.

[118]YU G I,JEONG J S,KIM G W,et al. Orca:A distributed serving system for {Transformer-Based} generative models[C]// 16th USENIX Symposium on Operating Systems Design and Implementation(OSDI 22). 2022:521-538.

[119]JIN Y,WU C F,BROOKS D,et al. $ S^ 3 $:Increasing GPU Utilization during Generative Inference for Higher Throughput [J]. Advances in Neural Information Processing Systems,2023, 36:18015-18027.

[120]KWON W,LI Z,ZHUANG S,et al. Efficient memory management for large language model serving with pagedattention [C]// Proceedings of the 29th Symposium on Operating Systems Principles. 2023:611-626.

[121]LIU J,CHUNG J W,WU Z,et al. Andes:Defining and enhancing quality-of-experience in llm-based text streaming services [J]. arXiv:2404.16283,2024.

[122]AMINABADI R Y,RAJBHANDARI S,AWAN A A,et al. Deepspeed-inference:enabling efficient inference of transformer models at unprecedented scale[C]// SC22:International Conference for High Performance Computing, Networking, Storage and Analysis. IEEE,2022:1-15.

[123]DAO T,HAZIZA D,MASSA F,et al. Flash-decoding for long-context inference. [EB/OL]. [2023-10-13]. https://pytorch.org/blog/flash-decoding/.

[124]HONG K,DAI G,XU J,et al. Flashdecoding++:Faster large language model inference on gpus[J]. arXiv:2311.01282,2023.

[125]GONG R,BAI S,WU S,et al. Past-future scheduler for llm serving under sla guarantees[C]// Proceedings of the30th ACM International Conference on Architectural Support for Programming Languages and Operating Systems. 2025:798-813.

[126]QIN Z,CAO Y,LIN M,et al. CAKE:Cascading and adaptive KV cache eviction with layer preferences[J]. arXiv:2503.12491,2025.

[127]YE Z,CHEN L,LAI R,et al. Flashinfer:Efficient and customizable attention engine for llm inference serving[J]. arXiv:2501.01005,2025.

[128]WU J,WANG Z,ZHANG L,et al. SCOPE:Optimizing Key-

Value Cache Compression in Long-context Generation[J]. arXiv:2412.13649,2024.
[129]WAN Z,SHEN H,WANG X,et al. Meda:Dynamic kv cache allocation for efficient multimodal long-context inference[J]. arXiv:2502.17599,2025.
[130]TRAN B,LI J,MADRY A. Spectral signatures in backdoor attacks[C]// Proceedings of the 32nd International Confernece on Journal of Machine Learning Research. 2018.
[131]CHOWDHERY A,NARANG S,DEVLIN J,et al. Palm:Scaling language modeling with pathways[J]. Journal of Machine Learning Research,2023,24(240):1-113.
[132]HUANG Y,CHENG Y,BAPNA A,et al. Gpipe:Efficient training of giant neural networks using pipeline parallelism[C]// Proceedings of the 33rd International Conference on Neural Information Processing Systems. 2019:103-112.
[133]LI S,XUE F,BARANWAL C,et al. Sequence parallelism:Long sequence training from system perspective[J]. arXiv: 2105.13120,2021.
[134]ZHENG L,LI Z,ZHANG H,et al. Alpa:Automating inter-and {Intra-Operator} parallelism for distributed deep learning[C]// 16th USENIX Symposium on Operating Systems Design and Implementation(OSDI 22). 2022:559-578.
[135]JIA Z,ZAHARIA M,AIKEN A. Beyond data and model parallelism for deep neural networks[J]. Proceedings of Machine Learning and Systems,2019,1:1-13.
[136]MIAO X,WANG Y,JIANG Y,et al. Galvatron:Efficient transformer training over multiple gpus using automatic parallelism [J]. arXiv:2211.13878,2022.
[137]LI Z,ZHENG L,ZHONG Y,et al. {AlpaServe}:Statistical multiplexing with model parallelism for deep learning serving[C]// 17th USENIX Symposium on Operating Systems Design and Implementation(OSDI 23). 2023:663-679.
[138]LU W,YAN G,LI J,et al. Flexflow:A flexible dataflow accelerator architecture for convolutional neural networks[C]// 2017 IEEE International Symposium on High Performance Computer Architecture(HPCA). IEEE,2017:553-564.
[139]MIAO X,SHI C,DUAN J,et al. Spotserve:Serving generative large language models on preemptible instances[C]// Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems,Volume 2. 2024:1112-1127.
[140]BORZUNOV A,BARANCHUK D,DETTMERS T,et al. Petals:Collaborative inference and fine-tuning of large models[J]. arXiv:2209.01188,2022.
[141]WANG Y,XUE D,ZHANG S,et al. Badagent:Inserting and activating backdoor attacks in llm agents[J]. arXiv:2406.03007,2024.

LIU Lilong, born in 2000, postgraduate. His main research interests include artificial intelligence and natural language processing.

QIAN Shengsheng, born in 1991, Ph.D, professor, is a member of CCF (No. 77702M). His main research interests include data mining and multimedia content analysis.

(责任编辑:柯颖)