未来之言：大语言模型的研究综述

摘　要

近期，学术界和工业界对大语言模型（LLM）的研究取得了巨大进展，Open AI 的对话式通用人工智能工具 ChatGPT 横空出世，表现出了令人惊艳的语言理解、文本生成和知识推理能力。它能够很好地理解用户意图，进行有效的多轮沟通，并且回答内容充实、条理分明、逻辑清晰,大大超出了人们对现阶段人工智能的预期。上线仅 2 个月活跃用户数已到达 1 个亿，成为历史上增长最快的消费级应用程序。除了被广大用户追捧外，Chat GPT 还受到了各国政府、企业界、学术界的广泛关注，掀起了大型互联网公司之间新一轮的人工智能竞赛，给各行各业带来了无限的想象空间。LLM 的技术发展对整个 AI 界产生了重要影响，这将彻底改变我们开发和使用 AI 算法的方式。考虑到这一快速的技术进步，在本篇综述中，我们通过介绍 LLM 的背景、主要发现和主流技术来回顾近年来的进展。我们特别关注 LLM 的四个主要方面，即预训练、适配微调、使用和能力评估。此外，我们还总结了开发 LLM 的可用资源，并讨论了 LLM 现有的问题和未来的发展方向。最后，我们对 LlaMA2 的 70B 预训练模型进行了优化实验，以帮助读者更深入地了解模型并行化和 Cache 优化在模型训练中的作用。

关键词：大语言模型 预训练 适配微调 使用 对齐 能力评估

1 引言

语言在人类交流和自我表达以及与机器交互中起着基础性的作用。人类一直对机器处理复杂语言任务（如翻译、摘要、信息检索、对话交互等）有巨大的需求，数十年的研究持续沿着使机器能够像人类一样阅读、写作和交流的方向上探索。最近，得益于新的模型框架（如 Transformer[1]）、计算能力的提升以及大规模训练数据的使用，语言模型的效果取得了显著突破，研究者创建了能够在各种任务上表现极其优异的大型语言模型(Large Language Models，LLMs)[2]。大型语言模型（LLMs）已经成为最前沿的人工智能系统之一，可以处理和生成连贯沟通的文本，并泛化到多个任务[3, 4]。

如图 1 所示：自然语言处理（Natural Language Processing，NLP）领域的历史进展从统计语言建模（Statistical Language Models）逐渐演变为神经语言建模（Neural language models，NLM），再从预训练语言模型（Pre-trained Language Models，PLMs）发展到 LLMs。早期的语言模型主要属于“统计语言模型（Statistical Language Models）”，包括词袋模型（Bag of Words）和 n-gram 模型等。词袋模型是一种简单的模型，它忽略了词语之间的顺序，只考虑了词语的出现频率[5]。n-gram 模型则是一种统计语言模型，它考虑了词语之间的顺序，但是只考虑了有限的词语之间的关系[6]。

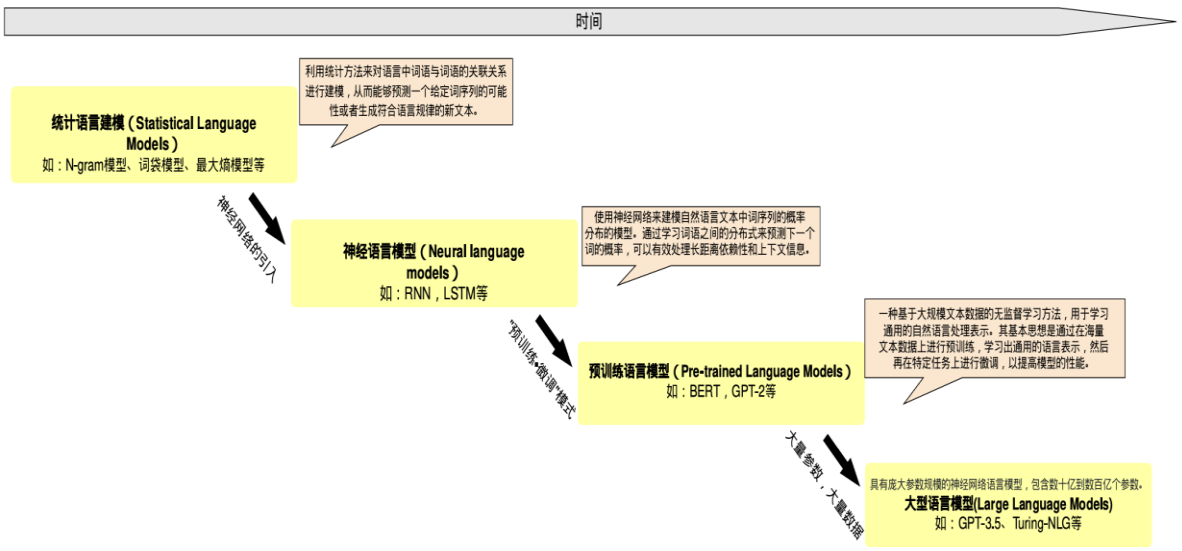

图 1. NLP 领域发展路线

随着深度学习的发展，语言模型也发生了显著的变化。神经语言模型（Neural language models，NLM）如开始被广泛使用。神经语言模型通过神经网络（如多层感知机（MLP）和递归神经网络（RNN））表征单词序列的概率[7][8]。word2vec [9, 10] 提出了一个简化的浅层神经网络，用于学习分布式单词表示，它能够将词语映射到一个连续的向量空间，从而捕捉到词语之间的语义关系。

传统的语言建模（LM）在监督设置中训练特定任务的模型，以 ELMo [11] 提出的预训练双向 LSTM（biLSTM）网络+根据特定下游任务微调（而不是学习固定的单词表示）为标志，预训练语言模型（PLMs）问世。此后，BERT [12]更进一步的通过在大量文本语料库上自监督训练，学习可在各种 NLP 任务中共享的通用表示。这些预训练的上下文感知的单词表示作为通用的语义特征非常有效，大大提高了 NLP 任务的性能，于是激发了大量的后续工作，逐渐形成“预训练和微调”的预训练语言模型固定的学习模式[13]。

遵循“预训练和微调”模式的 PLMs 研究发现，通过显著增加模型参数（数十亿至数百亿）和增大训练数据集，PLMs 的效果会有显著的提升[14][15]。这促使 PLMs 越来越大，许多研究探索通过训练更大的 PLM（如 1750 亿参数的 GPT-3）来达到性能极限。尽管研究拓展的主要是模型的大小（这些模型本身具有类似的架构和预训练任务），但大型 PLM 比较小的 PLM（如 1750 亿参数的 GPT-3 对比 15 亿参数的 GPT-2）展现出更多样的行为，在一些任务上有着令人惊讶的能力[16]来解决一系列复杂任务。因此，学界将这些大型 PLM 称为“大型语言模型（LLM）”[17][18]除了更好的泛化能力和领域适应性，LLM 似乎具有更一般的语言处理能力，如推理、规划、决策制定、上下文学习、零调优回答等。 这些能力似乎是由于它们的巨大规模而自发学习得到的，即使预训练的 LLMs 并未专门接受过这些属性的训练[19]，[20]。这些能力使得 LLM 被广泛应用于多种场景，包括多模态、机器人技术、工具操作、问答系统、自主代理等。在这些领域，都提出了各种改进方法，包括任务特定训练[21]，[22]或更好的提示[23]。

LLM 在解决各种任务时表现出与人类水平相当的性能，但这也带来了训练和推断速度缓慢、大量的硬件需求和更高的运行成本。这些要求限制了它们的应用，并打开了

设计更好的体系结构[24, 25]和训练策略的机会。参数有效调整、剪枝、量化、知识蒸馏和上下文长度插值等方法广泛研究，以实现对 LLMs 的高效利用。

由于 LLMs 在各种任务中取得了成功，LLM 相关的研究文献数量最近爆发性增长。本篇综述的贡献着重于提供 LLM 研究方向的概述以及对当下问题的讨论，探索。本文总结了预训练 LLMs 的架构和训练细节，并深入探讨了微调、多模态 LLMs、机器人技术、增强 LLMs、数据集和评估，LLM 面临的主要问题等概念的细节，总结如下：

• 我们详细总结了预训练模型的架构和训练细节。

• 除了突出 LLM 的时间顺序外，我们还回顾了主要研究成果，并深入探讨了 LLMs 的关键设计和发展趋势，以帮助读者更快速地了解与应用这一技术。

• 在这篇独立文章中，我们涵盖了多个概念，全面了解 LLMs 的整体方向，包括背景、预训练、微调、多模态 LLMs、增强 LLMs、数据集和评估、面临问题与未来等内容。

•我们独立地复现了若干个针对 LLM 关键议题的项目，给出自己的理解，并且开源供给读者复现理解，对 LLM 的一些方向有更深入的了解。

2 背景

为了快速了解 LLMs 的工作原理，在这一部分，我们给出相关背景知识，包括：token、注意力机制，KM 扩展定律，分布式 LLM 训练等，以便理解与 LLM 相关的基本信息。

2.1 数据预处理

数据预处理主要包括质量过滤，数据去重和隐私保护：为了控制训练数据的质量，经常使用一些过滤数据的方法，比如：1) 基于分类器的和 2) 基于启发式的。基于分类器的方法在高质量数据上训练分类器，并预测文本质量以进行过滤，而基于启发式的方法则采用一些规则进行过滤，例如语言、指标、统计数据和关键词；考虑到重复数据可能会影响模型性能并增加数据记忆，为了训练 LLM，数据去重也是重要的预处理步骤之一。数据去重可以在多个级别上执行，如句子、文档和数据集级别；大多数用于 LLM 训练的数据是通过网络来源收集的。这些数据包含私人信息，为了避免学习个人信息，许多 LLM 采用基于启发式的方法来过滤信息，例如姓名、地址和电话号码。

大型语言模型（LLMs）是受训练于文本来预测文本的人工智能系统，类似于其他自然语言处理系统，它们使用分词[27]作为文本数据处理编码步骤，将文本解析为非分解单元，称为 tokens。根据模型的大小和类型，tokens 可以是字符、子词、符号或单词。比较经典的分词方法包括：字节对编码 [28]（Byte Pair Encoding，BPE），字节对编码起源于压缩算法。这是一个逐步生成 tokens 的过程，其中相邻符号对被替换为一个新符号，并且输入文本中出现最频繁的符号被合并；UnigramLM [29]方法使用初始子词单元词汇表训练一个简单的 unigram 语言模型（LM）。通过迭代地从列表中删除概率最低的项目（在 unigram LM 上表现最差的项目），词汇表被修剪。WordPiece [30]，一开始是作为一种为日语和韩语语言设计的新颖文本分段技术，旨在改进语言模型以用于语音搜索系统。WordPiece 先由在 tokens 组成的词汇表上训练的 n-gram 语言模型，然后增加该

语言模型的选择 tokens 选择可能性。

2.2 自注意机制

在大语言模型中，自注意力机制被广泛应用于建模输入文本的上下文信息[31]。通过自注意力机制，大语言模型可以对输入文本中的不同位置之间的依赖关系进行建模，并捕捉长距离的上下文信息。

在自注意力机制中，对于输入序列中的每个位置，会计算一个与其他所有位置的关联度，然后利用这些关联度来计算加权平均值，作为该位置的表示。自注意力有多种计算方法：

自注意力（Self-Attention）[32]：自注意力也被称为内注意力，因为所有的查询、键和值都来自同一个编码器或解码器。自注意力层连接了所有序列位置，具有 O(1)的空间复杂度，这对于学习输入中的长距离依赖非常有益。

交叉注意力算法（Cross-Attention）：在自注意力模型中，通常存在两个序列，例如在机器翻译任务中，一个是源语言的序列，另一个是目标语言的序列。交叉注意力算法用于在这两个序列之间建立关联，以便将源语言的信息传递到目标语言上，并实现不同语言之间的信息交互。

稀疏注意力 [33]：自注意力算法的时间复杂度为 O(n2)，当将 LLMs 扩展到更大大的上下文窗口时，计算变得很麻烦。在[63]中提出了一种自注意力的近似方法，这极大地增强了 GPT 系列等 LLMs 处理更多文本输入（tokens）时的能力。

闪存注意力 [34]：使用 GPU 计算注意力的瓶颈在于内存访问，而不是计算速度。闪存注意力使用经典的输入平铺方法，在 GPU 芯片上的 SRAM 中处理输入的块，而不是为每个 token 从高带宽内存（HBM）进行输入输出。将这种方法拓展到稀疏注意力后，可实现与更大的上下文长度窗口。

2.3 分布式 LLM 训练

由于巨大的模型大小，成功训练一个能力强大的 LLM 非常具有挑战性。需要分布式训练算法来学习 LLM 的网络参数，分布式训练方法是指使用多个计算设备和处理器来加速训练过程，以便处理大规模的语言模型。为了支持分布式训练，研究者们探索了若干个优化框架来促进并行算法的实现和部署[35][36]。此外，各种并行策略经常被共同使用，以下是几种常见的 LLM 分布式训练方法:

数据并行：数据并行将模型复制到多个设备上，其中一个批次中的数据被分割到各个设备上。在每个训练迭代结束时，权重在所有设备之间进行同步，以确保权重保持一致。

张量并行：张量并行将模型的张量计算分片到不同的设备上进行并行计算。这种方法也被称为水平并行或层内模型并行。

流水线并行：流水线并行将模型层分片到不同设备上。也被称为垂直并行。

模型并行：张量并行和流水线并行的结合被称为模型并行。

3D 并行：数据、张量和模型并行的结合被称为 3D 并行。

优化器并行：优化器并行将优化器的状态、梯度和参数进行分区，以减少内存消耗，同时尽可能降低通信成本。这种方法也被称为零冗余优化器[35]。

2.4 LLM 的扩展

目前，LLMs 主要基于 Transformer 架构构建，其中多头注意力层堆叠在一个非常深的神经网络中。现有的 LLMs 采用类似的 Transformer 架构和预训练目标（如语言建模），只是规模更大、数据更多、计算资源更多。大量研究表明，扩展可以极大地提高 LLMs 的模型容量。因此，建立一个定量方法来描述扩展效果是很有用的。目前主要的扩展定律有两条，分别来自两个 LLMs 资深团队：

2020 年，OpenAI 团队[37]首次提出了用于神经语言模型的性能与三个主要因素之间的幂律关系模型，这三个因素分别是模型规模（N），数据集规模（D）和训练计算量（C）。在给定计算预算 c 的情况下，他们根据经验，提出了三条基本的扩展定律公式。表明语言建模损失可以分解为两部分，即不可减少损失（真实数据分布的熵）和可减少损失（真实和模型分布之间 KL 散度的估计）。通过对不同数据规模（22M 到 23B 标记）、模型规模（768M 到 1.5B 非嵌入参数）和训练计算进行拟合，得出了三条定律，即模型性能与这三个因素之间存在强烈的依赖关系。

Google DeepMind 团队[38]的 Hoffmann 等提出了一种另类的缩放定律形式，以指导 LLM 的计算优化训练。他们通过变化更大范围的模型大小（70M 到 16B）和数据大小（5B 到 500B 标记）进行了严格实验，并提出了类似的缩放定律，但具有不同的系数，他们表明可以得出计算预算在模型大小和数据大小之间的最佳分配。

缩放定律对 LLMs 的搭建训练，在实践中有很大的指导意义：在实践中，缩放定律可用于指导 LLM 的训练，并且已被证明可以可靠地估计基于较小模型的性能来评估较大模型的性能[39]。对于大型模型，从较小模型中获得的经验也适用于大型模型，这点非常有利于对大模型的实践探索。

现有的缩放定律研究主要是从语言建模损失的角度进行的[37][38]，而实际上我们更关心 LLM 在实际任务中的表现。直观地，语言建模损失较小的模型倾向于在下游任务上表现更好，因为语言建模损失可以被认为是模型整体容量的一般度量[37]。但是一个基本问题是，语言建模损失的降低如何转化为任务性能的提高，仍然是一个值得继续探究课题。

2.5 LLM 的对齐调整

人工智能的价值对齐，最早由斯图尔特·拉塞尔（Stuart Russell）在 2014 年提出，他定义了“价值对齐问题”（ Value Alignment Problem，VAP），强调需要构建不仅仅是智能的 AI 系统，还要与人类价值观相一致[40]。由于 LLM 被训练来捕捉预训练语料库的数据特征（包括高质量和低质量数据），它们很可能为人类生成有毒、有偏见、虚假甚至有害的内容[41][42]。有必要将 LLM 与人类价值观对齐，生成有益、诚实和无害的文本。

AI 对齐的主要研究包括外部对齐、内部对齐和可解释性[40]：其中，外部对齐通过指定与人类价值观相符的训练目标来保证 AI 与人类价值观匹配，例如选择正确的损失函数或奖励函数；内部对齐，确保 AI 的“内在”目标（即其在学习过程中推导或优化

的目标）与其设计者设定的“外在”目标一致；可解释性广泛指的是，在 AI 对齐的背景下，促进人类理解 AI 系统内部运作、决策和行为的方法、模型和工具。

3 预训练

预训练为 LLM 的能力奠定了基础。通过在大规模语料库上进行预训练，LLM 可以获得基本的语言理解和生成能力 [45，46]。在这个过程中，预训练语料库的规模和质量对于 LLM 获得强大的能力至关重要。此外，为了有效地预训练 LLM，也需要设计好模型架构、加速方法和优化技术。接下来，我们首先讨论数据收集和处理，然后介绍常用的模型架构，最后介绍用于稳定高效地优化 LLM 的训练技巧。

3.1 数据收集

相比小规模语言模型，LLM 更需要高质量数据来预训练模型，并且它们的模型能力很大程度上依赖于预训练语料库及其预处理方式。在这一部分，我们讨论预训练数据的收集和处理，包括数据来源、预处理方法以及对预训练数据如何影响 LLM 的性能的重要分析。

3.1.1 数据来源

要开发出一个有能力的 LLM，其关键在于从各种数据来源中收集大量的自然语言语料库。现有的 LLM 主要混合了各种公共文本数据集，并将其作为预训练语料库。

预训练语料库的来源可以广义地分为两种类型：通用文本数据和专用文本数据。通用文本数据，如网页、书籍和对话文本等，其由于规模大、多样性强且易于获取的特点，被大多数 LLM 所利用 [46，47]，这可以增强 LLM 的语言建模和泛化能力。鉴于 LLM 所展现出的惊人泛化能力，也有研究将预训练语料库扩展到更专用的数据集，如多语言数据、科学数据和代码等，以此来赋予 LLM 解决专用任务的能力。接下来，我们将描述这两种类型的预训练数据来源以及它们对 LLM 的影响。

通用文本数据：绝大多数的 LLM 采用了通用的预训练数据，比如网页、书籍和对话文本等，这些数据源提供了丰富的文本资源，并且涉及了多种主题。接下来，我们简要总结三种重要的通用文本数据。

网页：网页包含了各种主题的信息，如新闻、文章、博客、论坛帖子等。对于 LLM，网页文本数据是预训练的重要来源之一。通过大规模的语言建模，LLM 可以从网页文本中学习语言的统计规律、语义关联和语法结构。这使得 LLM 能够对广泛的主题进行理解和生成相关的内容。通过分析网页文本，LLM 可以获取丰富的知识和信息，从而在各种任务中应用，如自动摘要、问题回答、文本生成等。

对话文本：对话数据可以增强 LLM 的对话能力 [48]， 并可能改善 LLM 在问答任务上的表现 [49]。研究人员可以利用公共对话语料库的子集 (例如 PushShift.io Reddit 语料库) [48，50]， 或从在线社交媒体收集对话数据。由于在线对话数据通常涉及多个参与者之间的讨论，因此一种有效的处理方式是将对话转换成树形结构，其中每句话与回应它的话语相连。通过这种方式，多方之间的的对话树可以被划分为预训练语料库中的多个子对话。然而，过度引人对话数据来训练 LLM 可能会导致一个潜在的风险: 陈述性指令和直接疑问句被错误地认为是对话的开始，从而导致指令的有效性下降。

书籍：与其他语料库相比，书籍提供了大量的长篇文本数据，涵盖了各种领域和主题，如小说、非小说类文学作品、科学书籍、历史著作等。对于 LLM 的训练，书籍文本数据是非常有价值的资源。通过分析书籍文本，LLM 可以学习文学风格、情节构建、人物塑造等方面的知识。这使得 LLM 能够掌握不同文学体裁的特点，并在生成故事、创作诗歌和文学作品等创意任务中展现出丰富的表达能力。此外，书籍文本还提供了丰富的背景知识和文化参考，有助于 LLM 在语言生成和理解任务中更好地处理上下文和语义[51]。

专用文本数据：专用数据集对于提高 LLM 在特定下游任务中的能力非常有用。接下来，我们介绍三种专用数据类型。

多语言文本：除了在单目标语言进行训练外，整合多语言语料库可以增强模型的多语言的理解和生成能力。例如， BLOOM [51] 和 PaLM [52] 在其预训练语料库中收集了包含 46 种和 122 种语言的多语言数据。这些模型在多语言任务中展现出了出色的性能，例如翻译、多语言摘要和多语言问答，并且它与在目标语言上微调的最先进的模型具有可比性甚至有更好的性能。

科学文本：科学文本是指包含科学领域知识和术语的文本数据，如科学论文、学术文献、实验报告等。LLM 在科学文本上的训练可以增强其在科学领域的理解和应用能力[53]。通过分析科学文本数据，LLM 可以学习科学概念、实验方法、技术术语等，并在科学问题回答、文献摘要、科学创新等任务中提供准确的信息和解释。科学文本数据对于构建具有科学领域专业知识的 LLM 非常关键，它可以为科学研究人员、学生和科学爱好者提供有用的信息和支持。

代码：代码是计算机编程语言的文本表示，包括各种编程语言如 Python、Java、C++ 等。LLM 在代码上的训练可以提高其在编程相关任务中的能力，如代码生成、代码补全、代码理解等。通过分析代码数据，LLM 可以学习编程语言的语法规则、编码惯例和常见编程任务的解决方法。这使得 LLM 能够生成有效的代码段、提供编程建议和帮助解决编程问题。一般来说，常用于预训练 LLM 的代码语料库有两种来源。第一种来源是编程问答社区(如 Stack Exchange) [54, 55]。第二种来源是开源软件仓库，例如 GitHub， 它们收集了代码数据 (包括注释和文档字符串) 以供利用。最近的一项研究 [56] 还推测，训练代码可能是复杂推理能力(例如 CoT 能力 [57]) 的来源。此外，将推理任务格式化为代码的形式还可以帮助 LLM 生成更准确的结果 [58，59]。

3.1.2 数据预处理

在收集大量文本数据后，对数据进行预处理，特别是消除噪声、冗余、无关和潜在有害的数据[60，61]，对于构建预训练语料库是必不可少的，因为这些数据可能会极大地影响 LLM 的能力和性能。在这部分中，我们将细致地回顾提高收集数据质量的数据预处理策略[62，63，64]。

质量过滤：为删除收集到的语料库中的低质量数据，现有的工作通常采用两种方法：(1)基于分类器的方法，(2)基于启发式的方法[60，61]。

(1)基于分类器的方法：基于分类器的方法使用机器学习分类器来判断文本数据的质量，并将低质量的数据进行过滤。这种方法通常包括以下步骤：

a. 标记数据：首先，需要为数据集标记标签，将数据分为高质量和低质量两类。这可以通过人工标注或者利用已有的高质量数据作为参考进行自动标注。

b. 特征提取：然后，从文本数据中提取特征，以供分类器使用。这些特征可以包括词频、文本长度、语法结构等。特征提取的目的是将文本数据转化为机器学习算法可以处理的数值形式。

c. 训练分类器：接下来，使用标记好的数据和提取的特征来训练一个分类器模型，如支持向量机（SVM）、随机森林（Random Forest）或神经网络等。分类器学习从提取的特征中识别低质量文本数据的模式。

d. 过滤数据：最后，使用训练好的分类器模型对未标记的数据进行预测，并根据预测结果过滤掉被分类为低质量的数据。

基于分类器的方法需要一定的训练过程，通过训练和优化分类器模型，可以较准确地识别和过滤掉低质量的文本数据。

(2)基于启发式的方法：基于启发式的方法是一种基于规则和经验知识的预处理方法，不需要进行机器学习训练。这种方法包括以下步骤：

a. 定义规则：首先，根据领域知识和经验，制定一组规则来判断文本数据的质量。这些规则可以是基于文本属性、语法结构、词汇特征等方面的判断标准。

b. 应用规则：将定义好的规则应用于文本数据，根据规则的判断结果对数据进行过滤和清理。例如，可以通过规则判断文本的重复性、含有垃圾字符或特殊符号等。

c. 数据清理：根据规则的判断结果，对被判定为低质量的文本数据进行清理和过滤，例如删除重复数据、去除噪声字符、过滤掉包含广告、垃圾信息或敏感信息的数据等。

基于启发式的方法依赖于人工定义的规则和经验知识，因此可以快速实施和调整，但对于复杂的数据质量问题可能不够准确。

去重：现有的研究[62]发现，语料库中的重复数据会降低语言模型的多样性，可能导致训练过程不稳定，从而影响模型性能。因此，需要对预训练语料库进行去重处理。具体来说，可以在句子级、文档级和数据集级等不同粒度上去重。首先，在句子级别上，应删除包含重复单词和短语的低质量句子，因为它们可能会在语言建模中引入重复模式[63]。在文档级别上，现有研究主要依靠文档之间的表层特征 (例如单词和 n 元的重叠) 的重叠比率来检测和删除包含相似内容的重复文档[61，63，64]。此外，为了避免数据集污染问题，还必须通过从训练集中删除测试集可能出现的重复文本，来防止训练集和评估集之间的重叠[63]。现有研究已经证明，这三个级别的去重都有助于改善 LLM 的训练 [65]， 在实践中应该共同使用这三个级别的去重。

隐私去除：大多数预训练文本数据来自网络来源，包括涉及敏感或个人信息的用户生成内容，这可能增加隐私泄露的风险[66]。因此，需要从预训练语料库中删除可识别个人信息 (PII) 。一种直接有效的方法是采用基于规则的方法，例如关键字识别，来检测和删除 PII， 例如姓名、地址和电话号码 [67]。此外，研究人员还发现，LLM 在隐私攻击下的脆弱性可能归因于预训练语料库中存在的重复 PII 数据[67]。因此，去重也可以在一定程度上降低隐私风险。在应用隐私去除技术时，需要权衡数据的可用性和隐私保护的程度，以确保在保护个人隐私的同时仍能保留数据的有用性和分析价值。另外，隐私去除操作也需要遵守相关的法律法规和隐私政策，以确保合规性。

分词：分词也是数据预处理的关键步骤。的目的是将文本划分为更小的语义单元，以便进行后续的文本处理任务，如词性标注、命名实体识别、句法分析等。准确的分词

有助于提高文本处理的精度和效果，并在机器学习和自然语言处理任务中起到重要的作用。虽然直接利用已有的分词器是方便的 (例如 OPT [68] 和 GPT-3[69] 利用了 GPT-2 [70] 的分词器)，但是使用专门为预训练语料库设计的分词器可能会更加有效[71]，特别是对于由多种领域、语言和格式组成的语料库。因此，最近的几个 LLM 使用 SentencePiece [68] 为预训练语料库训练定制化的分词器。同时利用字节级的 Byte Pair Encoding (BPE)算法[69]来确保分词后的信息不会丢失 [70]。

3.1.3 预训练数据对大语言模型的影响

与小规模的 PLM 不同，由于对计算资源的巨大需求，通常不可能对 LLM 进行多次预训练迭代。因此，在训练 LLM 之前构建一个准备充分的预训练语料库尤为重要。在这一部分中，我们将讨论预训练语料库的质量和分布如何潜在地影响 LLM 的性能。

数据质量：预训练语料库的质量对 LLM 的性能具有重要影响。高质量的语料库应该包含准确、丰富和多样化的文本数据[72]。如果预训练数据中存在错误、噪音或低质量的文本，这些问题可能会传递到模型中并影响其生成的结果。

数据分布：预训练语料库的分布对 LLM 的性能也起着重要作用。如果预训练数据与目标任务的数据分布不匹配，模型可能无法很好地适应特定任务的语言特点和要求。因此，选择与目标任务相关的、与真实应用场景相似的预训练数据是至关重要的[73]。

数据覆盖范围：预训练语料库应该尽可能广泛地覆盖各种主题、领域和语言风格。这样，LLM 就能够学习到更加全面和多样的语言知识，并具备更好的泛化能力[74]。缺乏特定领域或特定类型文本的覆盖可能导致模型在相关任务上的表现不佳。

数据量和多样性：预训练语料库的规模和多样性也对 LLM 的性能产生重要影响。更大规模的语料库通常能够提供更丰富的语言知识和上下文理解能力，从而改善模型的生成质量。同时，多样性的数据可以帮助模型更好地适应不同的输入样本[75]，并减少对特定领域数据的依赖。

3.2 LLM 架构

本节中，我们将回顾 LLM 的架构设计，包括主流架构、预训练目标和详细配置。

3.2.1 主流架构

由于 Transformer 架构的出色并行性能和任务完成能力，Transformer 架构已成为开发各种 LLM 的标准骨干，使得语言模型能够扩展到数百亿或数千亿个参数 [76]。一般来说，现有 LLM 的主流架构可以大致分为三种类型，即编码器-解码器、因果解码器和前缀解码器，正如图 2 所示[77]。

编码器-解码器（Encoder-Decoder）架构：这种架构通常用于序列到序列（Sequence-to-Sequence）任务，如机器翻译和文本摘要。编码器负责将输入序列编码成一个固定长度的向量表示，而解码器则将该向量解码为输出序列。编码器和解码器通常都是基于循环神经网络（Recurrent Neural Networks，RNN）或者转换器模型（Transformer Models）构建的[78]。这种架构的关键思想是将输入序列的上下文信息编码到向量表示中，然后解码器利用该向量生成相应的输出序列。

因果解码器（Autoregressive Decoder）架构：这种架构在生成文本时采用自回归的方式，即依次生成每个词语或标记。模型在每个时间步根据已生成的部分序列预测下一

个词语。经典的因果解码器包括循环神经网络语言模型（Recurrent Neural Network Language Model，RNNLM）和基于 Transformer 的解码器。作为这种架构的代表性语言模型，GPT 系列模型 [70， 79， 80] 是基于因果解码器架构开发的。特别地，GPT-3 [79] 成功展示了这种架构的有效性，同时也展示了 LLM 惊人的 ICL 能力。

前缀解码器（Prefix Decoder）架构：这种架构通过在生成过程中接受部分输入序列（前缀）来加速生成的过程。前缀解码器首先根据给定的前缀生成一部分输出，然后根据这部分输出和剩余的模型状态继续生成后续的输出[81]。这种架构常用于基于搜索的解码方法，如束搜索（Beam Search）和采样搜索（Sampling Search）。实用的建议是不从头开始进行预训练，而是继续训练因果解码器，然后将其转换为前缀解码器以加速收敛[82]，例如 U-PaLM[59] 是从 PaLM[67] 演化而来。

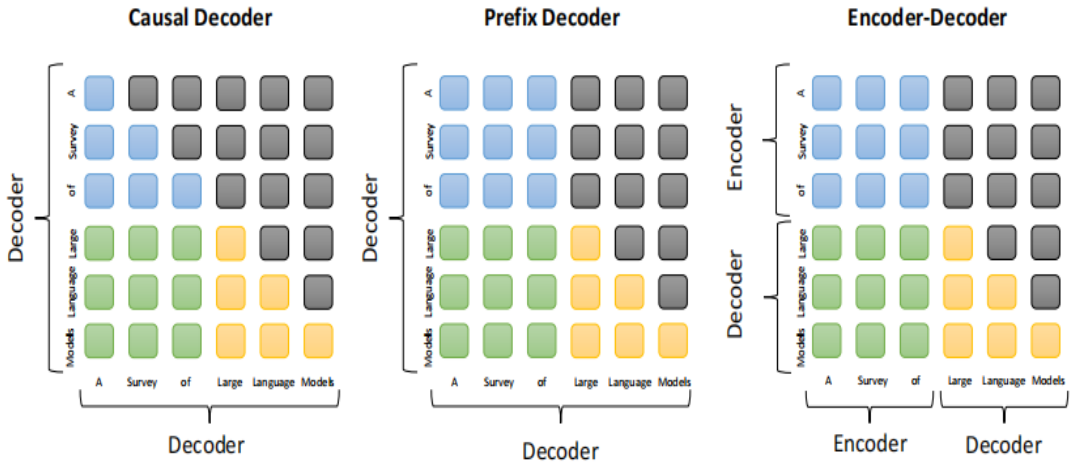

图 2. 三种主流架构的注意力模式

这些不同类型的 LLM 架构都有各自的优缺点和适用范围，具体选择取决于任务需求和应用场景。编码器-解码器适用于序列到序列任务，因果解码器适用于生成任务，而前缀解码器则可以在生成过程中提供更高的效率和交互性。对于这三种类型的架构，我们也可以考虑通过混合专家 (MoE) 扩展它们，其中每个输入的一小部分神经网络权重被稀疏激活，例如 Switch Transformer [81] 和 GLaM [83]。已经证明，通过增加专家的数量或总参数大小，性能会有显著的改进。

3.2.2 详细配置

自 Transformer[76] 推出以来，已经提出了各种改进方法来提高其训练稳定性、性能和计算效率。在这部分中，我们将讨论 Transformer 的四个主要部分的相应配置，包括标准化、位置编码、激活函数、注意力和偏置。

标准化（Normalization）：训练不稳定是预训练 LLM 的一个难题。为了缓解这个问题，层标准化(Layer Norm， LN)[84] 被广泛应用 Transformer 架构中。LN 可以帮助减少梯度消失和爆炸问题，并提高模型的训练稳定性和收敛速度。LN 的位置对 LLM 的性能至关重要，虽然最初的 Transformer [76] 使用后置 LN，但大多数 LLM 采用前置 LN 以实现更稳定的训练，尽管会带来一定的性能损失[85]。然而，已有研究发现 Sandwich-LN 有时无法稳定 LLM 的训练，可能导致训练崩溃 [86]。最近，一些高级标准化技术被提出以作为 LN 的替代方案。与 LN 相比，DeepNorm [87]已经表现出更好

的训练稳定性，和后标准化一起被 GLM-130B 采用。此外，在嵌入层后添加额外的 LN 也可以稳定 LLM 的训练。然而，这往往会导致显著的性能下降[89]，在一些最近的 LLM 中已经被移除[90]。

激活函数（Activation Function）：在 Transformer 模型中，通常使用激活函数来引入非线性性。常见的激活函数包括 ReLU（Rectified Linear Unit）、GeLU（Gaussian Error Linear Unit）和 Swish 等。这些激活函数在激活过程中引入非线性，有助于模型更好地捕捉输入序列中的复杂模式和语义。在现有的 LLM 中，广泛使用 GeLU 激活函数[87]。此外，在最新的 LLM (例如 PaLM 和 LaMDA) 中，也使用了 GLU 激活函数的变体[85，90]， 特别是 SwiGLU 和 GeGLU 变体，在实践中通常可以获得更好的性能。

位置编码（Positional Encoding）：由于 Transformer 模型没有显式的位置信息，位置编码用于向模型引入序列中每个位置的位置信息。常用的位置编码方法包括正弦余弦位置编码（Sine-Cosine Positional Encoding）和可学习的位置编码。正弦余弦位置编码是一种固定的编码方式，根据位置和维度计算位置编码向量。可学习的位置编码可以通过训练学习到更适应任务的位置编码方式。ALiBi[91] 使用基于键和查询之间距离的惩罚来偏置注意力分数。实证结果表明，它比其他位置编码具有更好的零样本泛化能力和更强的外推能力[92]. 此外，通过基于绝对位置设置特定的旋转矩阵，RoPE [93] 中的键和查询之间的分数可以使用相对位置信息计算，这对于建模长序列是有用的。

注意力机制（Attention Mechanism）：注意力机制是 Transformer 模型的核心组成部分，它用于计算输入序列中不同位置之间的相关性。常用的注意力机制包括自注意力（Self-Attention）和多头注意力（Multi-Head Attention）。自注意力允许模型在编码器和解码器之间建立输入序列中不同位置的依赖关系。多头注意力允许模型同时关注输入序列的不同子空间，以提高模型的表达能力和泛化性能。除了原始 Transformer 中的全自注意力机制[76]，GPT-3 采用了更低计算复杂度的稀疏注意力机制，即分解注意力[94]。

偏置（Bias）：在 Transformer 模型的注意力机制中，可以引入偏置项来调整输入特征的重要性。偏置可以帮助模型更好地适应不同任务和数据分布。常见的偏置项包括位置偏置（Positional Bias）和头偏置（Head Bias）。位置偏置允许模型在注意力计算中对不同位置的输入施加不同的权重，而头偏置允许模型在多头注意力中对不同注意力头的权重进行调整。与原始 Transformer 一样，大多数 LLM 在每个线性层和层标准化中保留了偏置。然而，在 PaLM[70] 和 Galactica [78] 中，偏置被移除。研究表明，对于 LLM 来说，去除偏置可以增强训练的稳定性[95]。

综合上述讨论，我们总结了现有文献中的详细配置建议。为了有更强的泛化能力和训练稳定性，建议选择前置的 RMS 进行层标准化，并选择 SwiGLU 或 GeGLU 作为激活函数。此外，在位置编码方面，RoPE 或 ALiBi 是更好的选择，因为它们在长序列上表现更好。

3.2.3 预训练任务

预训练在将大规模语料库中的通用知识编码到巨大的模型参数中起着关键作用。对于训练 LLM，有两个常用的预训练任务，即语言建模和去噪自编码。

语言建模(Language Modeling)：

• 任务描述：语言建模的目标是根据给定的上下文预测下一个词语或标记。它通过观察大规模文本语料库中的词序和语境关系来学习语言的统计规律和语义信息[96]。

•数学表达：给定一个输入序列$\mathrm{X}=\{\mathrm{x}_1，\ \mathrm{x}_2，\ \ldots，\ \mathrm{x}_\mathrm{t}\}$，，其中$x_\mathrm{t}$表示第$\mathrm{t}$个词语或标记。语言建模任务可以通过最大化条件概率来表达：

$$\mathrm{P}(\mathrm{X})=\prod_{\mathrm{i}}\mathrm{P}(\mathrm{x}_\mathrm{i}\mid\mathrm{x}_1，\ \mathrm{x}_2，\ \ldots，\ \mathrm{x}_{\mathrm{i}-1}) \tag{1}$$

这里表示在给定前面所有词语或标记的情况下，预测第i个词语或标记x_i的条件概率。

•方法：语言建模任务常用的方法是使用循环神经网络 (RNN)或变种，如长短时记忆网络 (LSTM) 或门控循环单元 (GRU)，来对上下文进行建模。通过预测下一个词语或标记，模型可以学习到语言的规律和上下文信息。

去噪自编码（Denoising Autoencoding）：

•任务描述：去噪自编码的目标是从经过部分损坏或噪声添加的输入中重构出原始输入。它通过学习有效的特征表示和潜在语义，使模型能够从噪声中恢复原始信息[96]。

• 数学表达：给定原始输入序列$\mathrm{X}=\{\mathrm{x}_1，\ \mathrm{x}_2，\ \ldots，\ \mathrm{x}_\mathrm{t}\}$，通过对输入应用噪声或损坏操作得到损坏的输入$\tilde{X}=\{\tilde{x}_1，\ \tilde{x}_2，\ \ldots，\ \tilde{x}_t\}$，去噪自编码任务的目标是最小化重构损失函数，如平方误差损失有如下表达：

$$\mathrm{L}(\mathrm{X}，\ \tilde{\mathrm{X}})=\sum||\mathrm{x}_\mathrm{i}-\tilde{\mathrm{x}}_\mathrm{i}||^2 \tag{2}$$

• 方法：去噪自编码任务常用的方法是使用自编码器，它由编码器和解码器组成。编码器将输入序列映射到低维潜在表示，而解码器将潜在表示映射回重构的输入序列。通过最小化重构损失，模型可以学习到有效的特征表示，并从噪声中恢复原始输入。

3.3 模型训练

在这一部分中，我们回顾训练 LLM 的重要设置和技巧。

3.3.1 优化设置

为了进行 LLM 的参数优化，我们介绍了批量训练、学习率、优化器和训练稳定性的常用设置。

1.批量训练（Batch Training）：

批量训练是指将训练数据划分为小批量的样本进行训练，而不是逐个样本进行训练。批量训练可以提高训练的效率和并行性，减少了每个样本的计算开销，并且可以利用硬件加速（如 GPU）来加快训练速度[97]。通常，选择适当的批量大小是重要的。较大的批量大小可以加速训练，但可能会导致内存消耗增加。较小的批量大小可能会增加训练的噪声，但可以提高模型的泛化能力。

2.学习率（Learning Rate）：

学习率是优化算法中控制参数更新步长的重要超参数。选择合适的学习率对于训练

LLM 至关重要[97]。通常，可以使用学习率调度策略，如学习率衰减（learning rate decay）或自适应学习率方法（如 Adam 优化器），以在训练过程中逐渐降低学习率，使模型更好地收敛。

3.优化器（Optimizer）：

优化器是用于更新模型参数的算法，常见的优化器包括随机梯度下降（Stochastic Gradient Descent，SGD）、Adam、Adagrad 等。选择适当的优化器可以加速训练和提高模型的收敛性。Adam[98]优化器是一种常用的选择，它结合了动量和自适应学习率的优点。然而，不同的优化器可能在不同任务和模型上表现出不同的效果，因此需要进行实验和调优。

4.训练稳定性：

在训练 LLM 时，为了提高训练的稳定性，可以采用一些技巧，如梯度裁剪（gradient clipping）来防止梯度爆炸，权重正则化（weight regularization）来减少过拟合，以及批量标准化（batch normalization）来加速收敛和提高模型的稳定性。除了上述设置外，还需要进行超参数调优，例如正则化参数、批量标准化的参数（如 momentum 和 epsilon）等。通过实验和交叉验证等方法[99]，选择合适的超参数值可以提高模型的性能和训练的稳定性。

3.3.2 可扩展的训练技术

随着模型和数据规模的增加，有限的计算资源下高效地训练 LLM 变得具有挑战性。尤其是两个主要的技术问题需要解决， 即提高训练吞吐量和加载更大模型到显存中。在本部分中，我们回顾了已有工作中广泛用于解决上述两个挑战的几种方法，即 3D 并行 [100]， ZeRO [101]和混合精度训练[102]，并就如何利用它们进行训练提供了一般性的建议。

3D 并行：3D 并行实际上是三种常用并行训练技术的组合，即数据并行、流水线并行[103]和张量并行[100]。我们下面来介绍这三种并行训练技术。

• 数据并行：数据并行是将训练数据划分为多个子集，在多个计算设备上并行处理这些子集的技术。每个计算设备都有一份完整的模型副本，但是处理不同的数据子集。在每个设备上，通过前向传播和反向传播计算梯度，并将梯度聚合起来以更新模型参数。这样可以提高训练吞吐量，加快训练速度。在数据并行中，需要注意梯度同步的问题，确保在参数更新之前将梯度进行正确的聚合。此外，该技术的实现简单，大多数现有的流行深度学习库已经实现了数据并行，例如 TensorFlow 和 PyTorch。

• 流水线并行：流水线并行是将模型的层分配到不同的计算设备上进行并行处理的技术。每个设备负责处理一部分层的计算，然后将结果传递给下一个设备。通过将模型的计算过程划分为多个阶段，并在不同设备上并行执行这些阶段，可以减少每个设备上的计算负载，从而提高训练吞吐量。在流水线并行中，需要确保数据在设备之间的正确传递和同步，并处理好不同设备之间的依赖关系。然而，流水线并行的朴素实现可能导致 GPU 利用率降低，因为每个 GPU 必须等待前一个 GPU 完成计算，从而导致不必要的气泡开销[104]。为了减少流水线并行中的这些气泡，GPipe [105]提出了填充多个数据批量和异步梯度更新技术，以提高流水线效率。

• 张量并行：张量并行是将模型的参数划分为多个子集，并在不同设备上并行计算

这些子集的技术。每个设备只处理部分参数的计算。通过并行计算参数的子集，可以降低每个设备上的内存需求，从而加载更大的模型到显存中，提高训练效率。在张量并行中，需要确保参数之间的正确通信和同步，以及处理好不同设备之间的参数依赖关系。目前，张量并行已经在几个开源库中得到支持，例如 Megatron-LM [106]， 并且可以扩展到更高维度的张量。此外，Colossal-AI 还为更高维度的张量实现了张量并行，并特别针对序列数据提出了序列并行[107]，可以进一步分解 Transformer 模型的注意力操作。

为了进一步的提升效率，可以将三种并行方式都结合起来，即 3D 并行，如下图 3 所示。

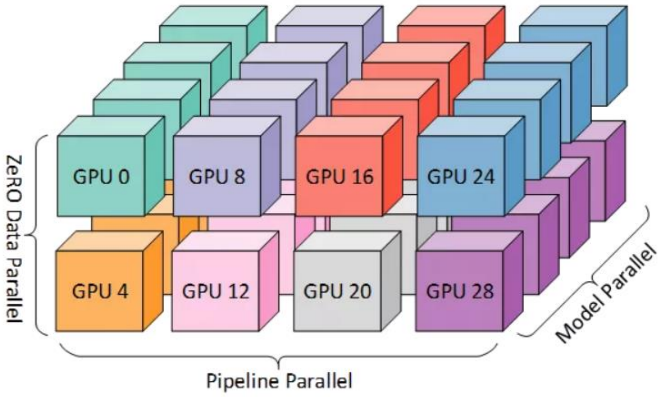

图 3. 3D 并行

ZeRO：ZeRO 是一种优化内存使用的技术，可以显著减少大型语言模型的内存需求，并提高训练效率。其核心思想是将模型参数划分为多个部分，并将这些部分分布在不同的设备上，从而减少每个设备上的内存占用。同时，ZeRO 通过分布式梯度聚合来减少通信开销[108]。具体而言，ZeRO 将模型参数划分为两个部分：零部分（Zero Redundancy）和非零部分（Non-Zero）。零部分包含模型的大部分参数，而非零部分只包含一小部分参数。零部分存储在每个设备的本地内存中，而非零部分存储在所有设备共享的全局内存中。在前向传播和反向传播过程中，只有零部分需要在设备之间进行通信[109]。通过将模型参数划分和优化内存使用，ZeRO 可以显著减少内存需求，并提高模型加载速度和训练吞吐量。它在大规模 GPU 集群上特别有效，如在数百个 A100 GPU 上进行训练。

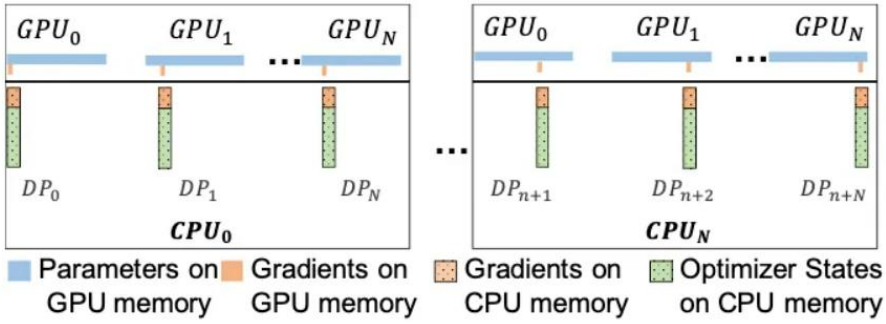

图 4. ZeRO

混合精度训练：混合精度训练是一种利用低精度浮点数进行计算的技术，可以加快

训练速度并减少显存使用。通常情况下，深度学习模型使用 32 位浮点数（FP32）进行计算，但 FP32 的计算和存储开销较大。混合精度训练通过使用低精度浮点数来代替部分计算中的 32 位浮点数，从而减少内存需求和计算开销。常用的低精度浮点数是半精度浮点数（FP16）和 BFloat16（BF16）[110]。这些浮点数格式可以在保持良好模型精度的同时，减少内存占用和计算开销。在混合精度训练中，前向传播和反向传播过程中的梯度计算使用低精度浮点数，而权重更新仍使用高精度浮点数进行。这种方式可以显著减少显存使用，并加快计算速度。需要注意的是，混合精度训练需要硬件的支持，如具备 Tensor Cores[111]的 GPU（例如 NVIDIA 的 A100 GPU[112]）。。

整体训练建议：在实践中，可以采用多种并行训练技术来提高大型语言模型（LLM）的训练效率。特别是 3D 并行技术，通常会联合使用以提高训练吞吐量和大模型加载。例如，研究人员已经将 8 路数据并行、4 路张量并行和 12 路流水线并行纳入到 BLOOM [69] 的 384 个 A100 GPU 上进行训练。目前，开源库如 DeepSpeed、Colossal-AI 和 Alpa 可以有效支持这三种并行训练方法。为了减少内存冗余，可以使用 ZeRO（Zero Redundancy Optimizer）、FSDP（Fully Sharded Data Parallelism）和激活重计算（activation recomputation）等技术来训练 LLM。这些技术已经集成到 DeepSpeed、PyTorch 和 Megatron-LM 等框架中。此外，混合精度训练技术如 BF16（BFloat16）也可以提高训练效率和减少显存使用，但需要硬件支持（如 A100 GPU）。由于训练大型模型是一个耗时的过程，因此在早期阶段进行模型性能预测和异常问题检测非常有用。为此，GPT-4 引入了一种基于深度学习堆栈的新机制，称为可预测扩展，可以使用更小的模型对大型模型进行性能预测，这对于开发 LLM 非常实用。

此外，主流深度学习框架如 PyTorch 还支持其他训练技术。例如，PyTorch 支持完全分片数据并行训练算法 FSDP[114]（即 fully sharded data parallel），并且可以将部分训练计算卸载到 CPU 上，根据需要进行配置。因此，在实践中，可以综合使用上述训练技术，结合具体的硬件和框架支持，以提高训练效率和加速大型语言模型的训练过程。

4 适配微调

LLM 的训练目标通常是最大限度地减少大型语料库中的上下文单词预测误差，然而这往往并不能完全满足大模型用户的使用需求。越来越多的研究表明，LLM 的能力可以根据特定目标进一步调整。在本节中，我们将介绍调整预训练 LLM 的两种主要方法：指令微调和对齐微调。前者可以增强 LLM 在特定领域的能力，后者可以帮助引导规范 LLM 的输出内容，预防其输出违法犯罪等负面内容。此外我们还将讨论在资源有限情况无法进行全参数调整模型时，可以采用一些部分参数的高效调参方法。

4.1 指令微调

LLMs 的主要问题之一是训练目标与用户目标不匹配，在使用某些 LLM 时，用户可能对于模型的使用有自己的要求或其他在预训练时未考虑的特点。例如当 LLM 在通用领域语料库上预先训练后，其确实可以获得一定的解决问题能力。但如果我们将它们应用到更专业的领域，其性能往往会比针对这些领域调整的模型有所下降，指令微调就是要解决这一问题。

指令微调包括使用（instruction，output）进一步训练 LLM，其中 instruction 表示模

型的人类指令，output 表示 instruction 之后的预期输出[115]。接下来是指令微调的一般步骤：

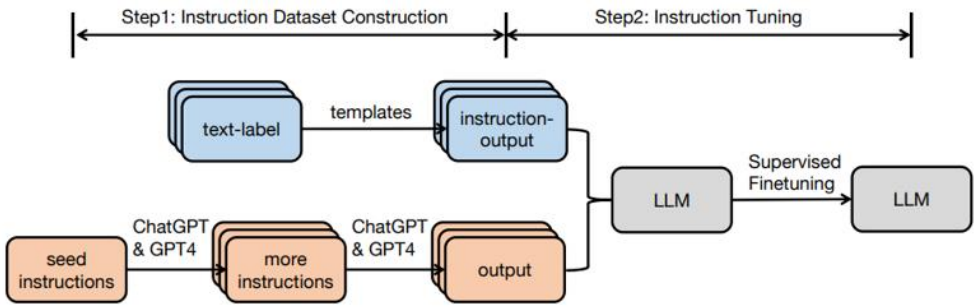

图 5. 指令微调

4.1.1 指令数据集构建

首先需要构建指令数据集，数据集中的每个实例都由三个元素组成：

1.指令，即指定任务的自然语言文本序列；

2.可选输入，提供上下文的补充信息；

3.基于指令和输入的预期输出。

而构建指令数据集的方法通常有两种：

1.注释自然语言数据集的数据整合。在这种方法中，通过使用模板将文本标签对转换为（指令、输出）对，从现有的注释自然语言数据集中收集（指令、输出）对[116]和 P3[117]等数据集就是根据数据整合策略构建的。

2.使用已有的 LLM 生成输出：快速收集所需指令以及所需输出可以使用 GPT-3.5-Turbo 或 GPT4 等其他 LLM，而不是手动收集输出。对此，指令由人工收集或使用 LLM 扩展基于手写种子指令的小指令。然后，将收集到的指令输入 LLM，以获得输出。

同时，对于多轮会话信息技术数据集，也可以让大型语言模型自我扮演不同角色（用户和人工智能助手），以会话形式生成信息[118]。

4.1.2 指令调整

根据收集到的数据集，预训练模型可直接以完全监督的方式进行功能调整，即在给定指令和输入的情况下，通过预测输出中的每个标记顺序来训练模型。

4.1.3 指令微调的优势

指令微调的好处在三个方面：

1.在指令数据集上对 LLM 进行微调，缩小了 LLM 的下一单词预测目标与用户的指令跟随目标之间的差距；

2.与标准 LLM 相比，指令微调使模型行为更可控、更可预测。指令的作用是约束模型的输出，使其与所需的反应特征或领域知识相一致，为人类干预模型行为提供了渠道；

3.指令微调的计算效率高，可帮助 LLM 快速适应特定领域，而无需进行大量的再

培训或架构更改。

4.2 对齐微调

仅通过预训练的 LLM 缺乏对人类价值或偏好的考虑，其输出并不一定一直符合人类的价值观，可能会产生例如，编造虚假信息、追求不准确的目标，以及产生有害、误导和有偏见的表达。因此，在调整阶段，开发者需要引导 LLM 避免的不良内容或危险行为。其中，LLM 风险通常可分为两种情况 1：既定风险和预期风险[119]。前者主要是已观察到的社会和道德风险[119]，而后者则是与先进 LLMs 相关的未来潜在风险[120]。

在 LLM 之前，更广泛的人工智能对齐就已经有人研究过了。从广义上讲，人工智能对齐的关键研究议程包括外部对齐、内部对齐和可解释性。人工智能对齐确保人工智能主体的外部和内部目标与人类价值观保持一致。外部目标是人工智能设计者基于人类价值观定义的目标，而内部目标是在人工智能智体中优化的目标[121]。

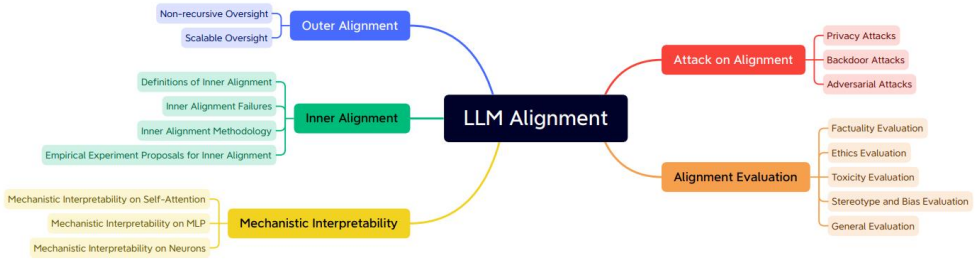

图 6. 对齐微调

4.2.1 外部对齐

外部对齐希望将 LLM 的目标与人类价值观统一起来。人类价值观有各种各样的层面，其内在结构和重要性各不相同，故常常采用人类学（Anthropic）关于人工智能一致性的观点：将 LLMs 外部一致性中指定的目标分为三个维度：helpfulness、honesty 和 harmlessness，这是多篇文章中较为公认的一种标准：

(1). Helpfulness：对于给定的无害任务或问题，预期 LLM 应尽可能简洁、高效、清晰地完成任务或回答问题[122]。换句话说，LLM 在完成规定的无害任务或回答无害问题时应提供帮助;

(2). Honesty：LLM 提供的信息应准确无误，并经过校准。他们应对自己、自身能力和内部状态保持诚实。此外，联络员还应明确说明所提供信息的不确定性，以避免误导人类[122];

(3). Harmlessness：这一目标可进一步分解为两个组成部分：1)如果本地语言管理器收到有害请求，应明确而礼貌地予以拒绝。2)无论收到什么输入，本地语言管理器本身都不应输出任何有害内容。由于这些目标很难明确，达成完美的外部校准非常困难。

在监督方面，我们可以将目前的外部配准方法分为两类：非递归监督方法和可扩展监督方法。目前绝大多数 LLM 外部配准方法都是直接从标注的人类反馈数据中学习训练目标，这使得人类反馈成为外部配准的瓶颈。这意味着，随着 LLM 能力的不断提高，构建有效的人类反馈数据将变得越来越困难。此外，从已经注释了人类偏好的数据中学

习，人类会难以监督超出一般人类能力范围的 LLM 行为，鉴于模型对工具性目标的激励，这可能会给人类带来极其不良的后果。

同时为了避免人类监督瓶颈，并使模型进一步提高对齐能力，可扩展监督[123]作为一项重要技术正在兴起，它允许人类监督扩展到复杂任务。可扩展监督提高了人类提供必要反馈的效率，使人类能够监督超出自身能力的目标。尽管目前关于可扩展监督的研究仍处于起步阶段，许多建议的有效性也尚未得到验证，但它被广泛认为是最有前途的外部协调方法，可使超出人类能力水平的系统与人类价值观保持一致[124]。

1.非递归监督 Non-recursive Oversight：

非递归监督方法主要是为人类能够单独提供配准监督的系统而设计的。目前大多数经过经验验证的 LLM 对齐方法都属于这一类。我们将它们进一步分为两类：基于强化学习（RL）的方法和基于监督学习（SL）的方法。值得注意的是，这两类方法都有可能成为可扩展监督方法的组成部分。

(1).基于 RL 的方法

RLHF（Reinforcement Learning from Human Feedback）[125][126][127]是目前最常用的非递归监督方法，它使用人类偏好作为代理来指定人类的价值观，并根据人类偏好训练奖励模型，从而通过强化学习来优化 LLM。RLHF 的基本思想可视为逆强化学习（IRL）[128][129]和 RL 的结合，其中奖励是从人类偏好中推断出来的[130]，然后用于调整 LLM。从本质上讲，RLHF 包括三个核心步骤 1.收集人类反馈数据。2.利用收集到的人类反馈数据训练奖励模型。3.用 RL 微调 LLM。目前，在这一步骤中最流行的 RL 选择是临近策略优化（PPO）[131]，这是一种策略梯度 RL 算法。为了使微调后的 LLM 输出合理连贯的文本，并保证它不会严重偏离其初始模型，当前正在微调的模型的输出与未经过 RLHF 的模型的输出的 KL 发散将作为惩罚项添加到奖励中。如果不加入惩罚项，微调后的 LLM 可能会学会输出胡言乱语，以欺骗奖励模型给出高分（即过度优化）。

(2).基于 SL 的方法

虽然基于 RL 的方法已成功应用于使 LLM 与人类偏好相匹配，但它们需要建立奖励模型，而这一过程可能会出现偏差和系统缺陷[132]。此外，强化学习的优化过程错综复杂，而且通常不稳定，这给实际应用带来了相当大的挑战[133]。我们根据所使用的反馈信号将基于强化学习的方法分为两种类型：基于文本反馈信号的 SL 和基于排名反馈信号的 SL。基于文本反馈信号的 SL 这些方法将人类意图和偏好转化为基于文本的反馈信号，以实现对齐，可视为 SFT 过程的扩展；基于排名反馈信号的 SL 这些方法直接使用监督学习来优化 LLM，其损失函数由基于排名的反馈信号构建。

2.可拓展监督 Scalable Oversight：

可扩展监督的主要理念是让相对较弱的监督者（如人类监督）通过易于判断的信号来监督复杂的任务.目前正在探索各种 scalable oversight 方法，其中包括 extensions of CAI、variants of human-assisted supervision、versions of AI-AI debate、red teaming via multi-agent RL，以及 creation of model-generated evaluations 等，然而这种方法是否真的能成功，还需要更进一步的研究和试验。

4.2.2 内部对齐

内部对齐的目标是人工智能系统是否能稳健地实现（优化）与人类希望相一致的给

定目标。[134]首次对内部对齐一词进行了定义。文章中同时介绍了有关的四个定义

(1)基础优化器（base optimizer）是一种机器学习算法，用于搜索能够在特定任务中表现良好的模型[134]。例如，梯度下降是一种常见的基础优化器，它根据损失函数的梯度更新模型的参数。

(2)基础目标（base objective）是基础优化器用于在不同可能的模型之间进行选择的基本原理[134]。它由人工智能系统设计者指定，并与设计者对模型的预期目标一致。

(3)Mesa-优化器是一种学习模型，它充当优化器，根据明确指定的目标函数在内部搜索可能的输出、策略、规划或策略的空间[134]。基础优化器可以生成也可以不生成Mesa-优化器。

(4)Mesa-目标是 Mesa-优化器的目标，也是 Mesa 优化器在各种潜在输出中进行选择的基本原理[134]。

其中，Mesa-优化器的目标可能与基础优化器的目标不同，这可能会导致对齐或安全问题。在这种情况下，内部对齐的相对正式的定义是指将 Mesa-优化器的 Mesa-目标与基础优化器的基础目标对准的挑战，从而使 Mesa-优化器追求与基础优化器相同的目标[134]。

4.2.3 机制可解释性

机制可解释性[135]是指阐明机器学习模型将输入转化为输出的内部机制，为如何以及为什么做出某些预测提供因果和功能解释[136][137]。机制可解释性的目标是对推理过程进行从头到尾的逆向工程，将神经网络分解为可解释的部分和信息流，为其逐步推理提供透明度。

机制可解释性对人工智能对齐具有重要意义。首先，可解释性方法可用于审计 LLM，特别是在部署 LLM 之前。可以检查 LLM 的对齐功效，识别不对齐和错误的输出，并阐明它为什么会产生这样的输出[136][137]。其次，可解释性评估指标可以作为优化人工智能对齐的奖励函数[138]，以激励人工智能系统保持目标透明度[139]。最后，除了检查/架构的透明度之外，还可以加强训练过程的透明度，能够了解和监控人工智能系统训练过程中发生的事情和变化[140]。当前的机制可解释性研究，包括对自注意、MLP 和神经元（功能特定的神经元、编辑神经元）的机制研究。

4.3 参数高效微调

将预训练好的语言模型（LLM）在下游任务上进行微调已成为处理 NLP 任务的一种范式。与使用开箱即用的预训练 LLM(例如：零样本推理)相比，在下游数据集上微调这些预训练 LLM 会带来巨大的性能提升。

但是，随着模型变得越来越大，在消费级硬件上对模型进行全部参数的微调（full fine-tuning）变得不可行。此外，为每个下游任务独立存储和部署微调模型变得非常昂贵，因为微调模型（调整模型的所有参数）与原始预训练模型的大小相同。

在 LLM 的 FFT（如 LLaMA）出现之前，NLP 领域已经提出了参数高效微调（PEFT）[141]，特别是针对预训练模型（如 BERT[142]），为高效微调 LLM 提供了一种很有前景的方法。PEFT 的优势在于它能够只微调一小部分外部参数，而不是整个主干模型，同时还能获得相当甚至更优的性能[143]。此外，与 FFT 相比，PEFT 还能有效减轻灾难性

遗忘[144]。

大多数 LLM 是基于 Transform 框架，基于此，主流的参数微调方法有很多：Adapter Tuning[141]、Prefix Tuning、Prompt Tuning（可以看作是 Prefix Tuning 的简化版本）、P-Tuning v1、P-Tuning v2、LoRA 等。

其中，LoRA 方法在 LLM 中运用最多。Edward Hu 等人[145]根据 Li [146]等人和 Aghajanyan 等人[147]的研究:学习到的过参数化模型实际上驻留在较低的内在维度上，他们假设模型适应过程中权重的变化也具有较低的"内在秩"，从而提出了低秩适应（Low-Rank Adaptation，LoRA）方法。LoRA 可以固定预先训练好的模型权重，并将可训练的秩分解矩阵注入 Transformer 架构的每一层，从而大大减少下游任务的可训练参数数量。

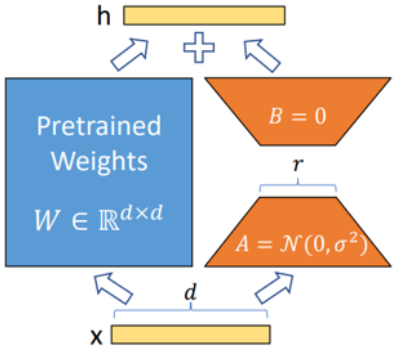

图 7. 重参数化

5 使用

在训练和微调 LLM 之后，如何使用 LLM，如何设计合适的提问或提示策略以得到想要的输出成了一个问题。在现有文献中，特定任务的提示可以通过手动创建和自动优化来有效学习，一种有提示方法是利用了模型的上下文学习 In-Context Learning 能力[148][149]，这种方法不需要调整模型权重，只需要给预训练模型展示一些输入-输出示例，就能使模型获得新的能力；另一种是,思维链提示法[150]，思维链提示，就是把一个多步骤推理问题，分解成很多个中间步骤，分配给更多的计算量，生成更多的 token，再把这些答案拼接在一起进行求解。除此之外,还有一些其他方法，例如有人提出了用于解决复杂任务的计划[151]，它首先将任务分解成较小的子任务,然后生成一个行动计划，逐一解决这些子任务。本节，将具体介绍 ICL 方法和大模型的 CoT 特征。

5.1 上下文学习

In-Context Learning（ICL）的概念最初来自 GPT-3 的论文[149]-在无监督的预训练中，语言模型得出了一种泛化和模式识别的能力。应用这些能力模型可以快速适应或识别所需要做的任务。我们使用“In-Context Learning（ICL）”一词来描述这个过程，它发生在每个序列的正向传递中），它是一种使用语言模型来学习仅有少量示例的任务的方法。在上下文学习过程中，语言模型会收到一个提示，提示由一系列演示任务的输入-输出对组成。在提示结束时，研究人员会附加一个测试输入，并允许 LLM 仅通过提示

条件和预测下一个标记来进行预测。要正确回答下面的两个提示，模型需要阅读训练示例，以确定输入分布（财经新闻或普通新闻）、输出分布（正面/负面或主题）、输入-输出映射（情感或主题分类）以及格式。

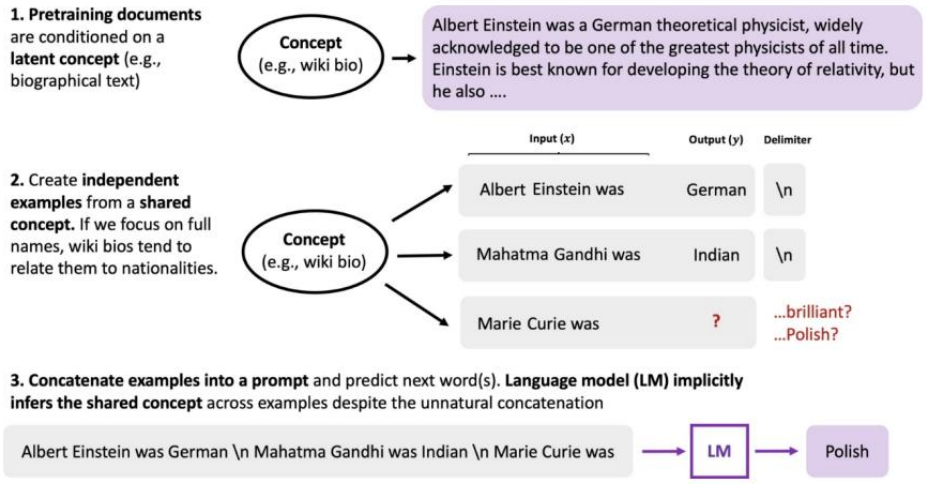

图 8. ICL

在许多基准 NLP 基准测试中，上下文学习都能与使用更多标记数据训练的模型相媲美，并且在常识性句子补全和问题解答测试中处于领先地位[152]。

5.1.1 ICL 的原理

Sang Michael Xie 等人[152]在文章中通过贝叶斯方法分析了 ICL 的形成原理：

预训练分布 Pretraining distribution (p)：预训练文档结构的主要假设是，文档首先通过对潜在概念进行采样，然后以潜在概念为条件生成，这种潜在概念是隐马尔可夫模型（HMM）[153]转换的参数，然后再从 HMM 中抽取一系列标记。这种潜变量结构在 LDA 等主题模型中很常见[154][155]。在预训练期间，LM 必须推断出多个句子中的潜在概念，以生成连贯的连续句。假设预训练数据和 LM 都足够大，以至于 LM 与预训练分布完全拟合。因此，使用 p 来表示预训练分布和 LM 下的概率；

提示分布 Prompt distribution：上下文学习提示是由 IID（独立且同分布）训练示例与一个测试输入串联而成的列表。提示中的每个例子都是以同一提示概念为条件的序列，该概念描述了要学习的任务。

LM 通过使用提示来“定位”它在预训练期间学到的相关概念来进行上下文学习以完成任务。理论上这可以将其视为对提示条件下的潜在概念的贝叶斯推理，这种能力来自于预训练数据中的结构（长期一致性）。其与 NLP 基准的经验证据联系起来，表明当提示中的输出被随机输出替换时，上下文学习仍然有效[152]。虽然随机输出增加了噪声并删除了输入输出映射信息，但其他组件（输入分布、输出分布、格式）仍然为贝叶斯推理提供证据。

5.1.2 提示方法

Tom B. Brown 等人[149]探讨了提示性提问对于 LLM 的促进作用，其分类了 Zero-shot、One-shot、Few-shot 三种不同的 prompt 方法：

(1) Few-Shot（FS）：是指模型在推理时给予少量样本，但不允许进行权重更新。对于一个典型数据集，Few-shot 有上下文和样例（例如英语句子和它的法语翻译）。Few-shot 的工作方式是提供 K 个样本，然后期望模型生成对应的结果。通常将 K 设置在 10 到 100 的范围内，因为这是可以适应模型上下文窗口的示例数量（nctx=2048）。Few-shot 的主要优点是大幅度降低了对特定任务数据的需求，并减少了从微调数据集中学习过度狭窄分布。主要缺点是该方法的结果迄今为止远不如最先进的微调模型。此外，仍需要一小部分特定任务的数据。

(2) One-Shot（1S）：与 Few-Shot 类似，只允许一个样本（除了任务的自然语言描述外）。将 One-Shot 与 Few-Shot、Zero-Shot 区分开的原因是它最接近某些任务与人类沟通的方式。相比之下，如果没有示例，有时很难传达任务的内容或格式。

(3) Zero-Shot（0S）：和 One-shot 类似，但不允许提供样本，只给出描述任务的自然语言指令。该方法提供了最大的方便性、稳健性以及避免虚假相关的可能性，但也是最具挑战性的设置。在某些情况下，即使是人类，在没有例子的情况下，也可能难以理解任务的格式。例如，如果要求某人“制作一张关于 200 米冲刺世界纪录的表格”，这个请求可能是模棱两可的，因为可能不清楚表格应该具有什么格式或包含什么内容。然而，至少在某些情况下，Zero-shot 是最接近人类执行任务的方法，人类可能仅凭文本指令就知道该做什么。

5.1.3 ICL 研究现状

以往的研究表明，ICL 的性能极不稳定，从随机猜测到 SOTA，对许多因素都很敏感，包括演示排列、演示形式等[156][157]。ICL 需要在上下文中预置大量演示。然而，这也带来了两个挑战：(1)演示的数量受到 LM 最大输入长度的限制，与微调相比，LM 的输入长度要少得多（可扩展性）；(2)随着演示数量的增加，由于注意力机制的二次方复杂性，计算成本会越来越高（效率）[148]。

目前，许多工作质疑了 ICL 不涉及学习并且仅定位预训练模型中存在的概念。大型模型似乎比小型模型从上下文提示中“学习”更多。但是将 ICL 与梯度下降联系起来的研究结果表明，有理由相信当小模型也执行 ICL 时可能会发生学习。总之，对于模型是否能够在上下文中“学习”新的知识并覆盖其先验语义是存在争议的，该方向的工作仍需更多研究深入。

5.2 思维链提示

2022 年，Google 公司的 Jason Wei 等人[150]探讨了生成思维链（CoT）可以显著提高大型语言模型执行复杂推理的能力。在三个大型语言模型上进行的实证实验表明，思维链提示可以提高一系列算术、常识和符号推理任务的成绩。例如，在数学单词问题 GSM8K 基准测试中，仅用 8 个思维链示例来提示 PaLM 540B，其精度便超过了带有验证器的经过微调的 GPT-3。

5.2.1 CoT 特性

思维链作为一种促进语言模型推理的方法，其有几个特性。

1.思维链原则上允许模型将多步骤问题分解为中间步骤，这意味着可以将额外的计算分配给需要更多推理步骤的问题。

2.思维链为了解模型的行为提供了一个可解释的窗口，表明它可能是如何得出特定答案的，并为调试推理路径出错的地方提供了机会（尽管完全描述支持答案的模型计算仍是一个未决问题）。

3.思维链推理可用于数学文字题、常识推理和符号操作等任务，而且有可能（至少在原则上）适用于人类可以通过语言解决的任何任务。

4.在足够大的现成语言模型中，只需将思维链序列的示例纳入少量提示的示例中，就能很容易地激发思维链推理。

5.2.2 CoT 的研究现状

思维链原理目前还未有统一的解释，需要进一步的研究，虽然思维链模拟了人类推理者的思维过程，但这并不能回答神经网络是否真的在"推理"，这还是一个未解决的问题。有许多论文对 CoT 与大模型的互动进行了一系列实验，总结了一些 CoT 的特征：1.模型规模小会导致 CoT 失效；2.简单的任务 CoT 不会对模型性能带来提升；3.训练数据内部彼此相互联结程度的增加可以提升 CoT 的性能；4.示例中的错误，或者无效的推理步骤不会导致 CoT 性能的下降。

其次，用思维链手动增强示例的成本极低，但这种注释成本可能会让微调工作难度陡增。第三，无法保证推理路径的正确性，这可能会导致正确答案和错误答案；改进语言模型的事实生成是未来工作的一个开放方向[158][159][160]。最后，由于思维链理只在大型模型中出现，因此在实际应用中成本较高；进一步的研究可以探索如何在较小的模型中诱导推理。

6 能力和评估方法

为了验证大语言模型的有效性和优越性，大量的任务和基准被用于能力评估和分析。在本节中，我们首先介绍了大语言模型对语言生成和理解的三种基本能力评估，然后介绍了几种约束条件更多或目标更为复杂的高级能力评估，最后讨论了现有的基准，评估方法和实证分析。

6.1 基本能力

在这部分中，我们主要关注对大语言模型的能力评估的三种基本类型，即语言生成、知识利用和复杂推理。值得注意的是，我们并不打算完全覆盖所有相关的任务，而是只关注最广泛讨论或研究的大语言模型任务。接下来，我们将详细介绍这些任务中的细节。

6.1.1 语言生成

根据任务的定义，现有的语言生成任务大致可以分为语言建模任务、条件文本生成任务和代码合成任务。需要注意的是，代码合成不是一个典型的 NLP 任务，之所以在此讨论是因为它可以采用类似于自然语言文本的生成方法，通过许多大语言模型（在代码上进行训练）直接解决。

语言建模 作为大语言模型最基本的能力，语言建模目的是基于之前的 tokens 预测生成下一个 tokens[161]，主要关注基本语言理解和生成能力。为了评估这种能力，现阶段常用的典型语言建模数据集包括 Penn Treebank[162]，WikiText-103[163]和 Pile 数据集[164]，其中困惑度量通常用于评估零样本设置下的模型性能。实证研究[165,166]表明，

在这些评估数据集上，大语言模型比以前的最先进的方法的性能效果还要更佳。为了更好地测试文本中长期依赖的建模能力，引入了 LAMBADA dataset[167]，其中需要大语言模型根据上下文预测句子中的最后一个单词。然后，利用预测的最后一个词的准确性和困惑度对大语言模型进行评价。根据已有的研究，在语言建模任务中的性能通常遵循 scaling law[168]，这意味着缩放语言模型将提高准确性并减少困惑度。

条件文本生成 作为语言生成中的一个重要课题，条件文本生成[169]侧重于根据给定的条件生成满足特定任务要求的文本，通常包括机器翻译[170]，文本摘要[171]和问答[172]。为了评估生成文本的质量，自动度量（例如准确性，BLEU[173]和 ROUGE[174]）和人类评级通常被用于评估性能。由于强大的语言生成能力，大语言模型在现有数据集和基准上表现出强大的性能。例如，GPT-4 展示了与商业翻译产品水平相当的性能，甚至对于具有显著语言距离的语言的翻译任务也同样优秀[175]。关于新闻摘要任务（即 CNN/DM 和 XSUM），大语言模型也展示了与人类自由撰稿人相当的表现[176]。尽管模型容量发展迅速，但人们越来越关注现有的自动度量来准确地评估大语言模型在条件文本生成任务中的性能的可行性[177,178]。作为自动度量的替代品，最近的研究还建议将大语言模型纳入生成评估器，以检查生成内容的质量[179,180]。此外，研究人员还以大语言模型探索更具有挑战性的语言生成任务，如结构化数据生成[181]和长文本生成[182,183]。

代码合成 除了生成高质量的自然语言文本外，现有的大语言模型还展示出生成形式化语言，特别是满足特定条件的计算机程序（即代码）的强大能力，称为代码合成[184]。与自然语言生成不同，由于生成的代码可以通过使用相应的编译器或解释器执行来直接检查，因此现有的工作主要通过计算与测试用例相反的通过率来评估从大语言模型中生成的代码的质量，即 pass@K。最近，提出了几个侧重于功能正确性的代码基准来评估大语言模型的代码合成能力，如 APPS[185]，HumanEval[186]和 MBPP[187]。通常，它们由各种编程问题组成，带有文本规范和用于正确性检查的测试用例。为了提高这种能力，关键是在代码数据上微调（或预训练）大语言模型，这可以有效地使大语言模型适应代码合成任务。此外，现有工作提出了生成代码的新策略，例如，对多个候选解决方案进行采样和规划引导解码，这可以被认为是模仿程序员对错误修复和代码规划的过程。令人印象深刻的是，大语言模型最近在编程竞赛的用户中排名前 28%，在 Codeforces 平台上表现出与人类的竞争能力[188]。此外，GitHub Copilot 已经发布，以协助编程编写 IDE（例如 Visual Studio 和 JetBrains IDE）可以支持多种语言，包括 Python、JavaScript 和 Java 等热门编程语言。ACM Communications 中一篇题为“The End of Programming”[189]的观点文章讨论了 AI 编程在计算机科学领域的影响，强调了向高度自适应大语言模型作为新的原子计算单元的重要转变。

主要问题 尽管大语言模型在生成类人文本方面取得了出色的性能，但它们容易受到语言生成中两个主要问题的影响，即可控生成（Unreliable generation evaluation）和专业领域生成（Underperforming specialized generation）。

•可控生成 对于大语言模型来说，在给定条件下生成文本的主流方式是通过使用自然语言指令或提示。尽管简单，但这种机制在对这些模型的生成输出施加细粒度或结构约束方面提出了重大挑战。现有工作表明，当生成对其结构具有复杂约束的文本时，大语言模型可以很好地处理局部规划（例如近端句子之间的交互），但可能难以处理全局规划（例如较远句子相关性）。例如，要生成一个包含多个段落的复杂长篇文章，考虑

到整个文本，仍然很难直接确保特定的文本结构（例如，概念的顺序和逻辑流程）。对于需要正式规则或语法的生成任务，例如代码合成，这种情况将变得更具挑战性。为了解决这个问题，一个潜在的解决方案是将一次性生成扩展到大语言模型的迭代提示中，这模拟了人类书写过程例如计划、起草、重写和编辑来分解语言生成。几项研究证明，迭代提示可以引出相关知识，从而在子任务中取得更好的表现 。本质上，思维链提示利用了将复杂任务分解为多步推理链的思想。此外，生成文本的安全控制对于实际部署也很重要。已经表明，大语言模型可能会生成包含敏感信息或冒犯性表达的文本。尽管 RLHF 算法可以在一定程度上缓解这个问题，但它仍然依赖大量人工 tokens 的数据来调整大语言模型，而没有遵循客观的优化目标。因此，必须探索有效的方法来克服这些限制并实现对大语言模型输出的更安全控制。

•专业领域生成 尽管大语言模型已经学习了通用语言模式以生成连贯的文本，但在处理专门领域或任务时，他们在生成方面的熟练程度可能会受到限制。例如，在一般网络文章上训练过的语言模型在生成涉及许多医学术语和方法的医学报告时可能会面临挑战。直觉上，领域知识对于模型专业化应该是至关重要的。然而，将此类专业知识注入大语言模型并不容易。当大语言模型被训练以展示某些特定的能力，使他们能够在某些领域表现出色时，他们可能在其他领域表现不佳[190]。这样的问题与训练神经网络中的 catastrophic forgetting[191,192]有关，它指的是整合新旧知识的冲突现象。类似的情况也发生在大语言模型的 Human Alignment 中，其中必须支付“alignment tax”[193]（例如上下文学习能力的潜在损失）。因此，重要的是开发有效的模型专业化方法，使大语言模型灵活适应各种任务场景，同时尽可能保留原有能力。

6.1.2 知识运用

知识利用是智能系统在支持事实证据的基础上完成知识密集型任务（例如，常识性问题回答和事实完成）的重要能力。具体来说，它要求大语言模型适当地利用预训练语料库中丰富的事实知识或在必要时检索外部数据。特别是知识回答 (QA) 和知识补全（knowledge completion）是评估这种能力的两个常用任务。根据测试任务（问答或知识补全）和评估设置（有或没有外部资源），我们将现有的知识利用任务分为三类，即 closed-book QA、open-book QA 和知识补全。

Closed-book QA Closed-book QA 任务也即闭卷问答[194]，测试大语言模型从预训练语料库中获得的事实知识，其中大语言模型应仅根据给定的上下文回答问题，而不使用外部资源。为了评估这种能力，可以利用几个数据集，包括 Natural Questions[195]、Web Question[196]和 TriviaQA[197],其中准确性指标被广泛采用。实证结果表明，大语言模型可以在此设置中表现良好，甚至可以与最先进的开放域 QA 系统的性能相媲美。此外，大语言模型在 Closed-book QA 任务上的表现在模型大小和数据大小方面也满足 scaling law：缩放参数和训练令牌可以增加大语言模型的容量并帮助他们学习（或记忆）更多知识来自预训练数据。此外，在类似的参数规模下，具有更多与评估任务相关的预训练数据的大语言模型将获得更好的性能。此外，Closed-book QA 设置还提供了一个测试平台，用于探测大语言模型编码的事实知识的准确性。然而，大语言模型可能在依赖细粒度知识的 QA 任务上表现不佳，即使它存在于预训练数据中。

Open-Book QA 与 Closed-book QA 不同的是，在 Open-book QA 任务中，大语言模型可以从外部知识库或文档集合中提取有用的证据，然后根据提取的证据回答问题。比较典型的 Open-book QA 数据集（例如，Natural Questions[198]，OpenBookQA[199]和

SQuAD[200]）与 Closed-book QA 数据集有重叠，但它们包含外部数据源，例如 Wikipedia（维基百科）。准确性和 F1 分数的指标广泛用于评估 Open-book QA 任务。为了从外部资源中选择相关知识，大语言模型通常与文本检索器（甚至搜索引擎）配对，独立或联合大语言模型训练。

Knowledge Completion 在 Knowledge Completion 任务中，大语言模型可能（在某种程度上）被视为知识库[201]，可以利用它来完成或预测知识单元（例如 knowledge triples）的缺失部分。此类任务可以探索和评估大语言模型从预训练数据中学到了多少知识和哪些知识。现有的知识补全任务大致可以分为知识图谱补全任务（如 FB15k237[202]和 WN18RR[203]）和事实补全任务（如 WikiFact[204]），其目的是完成知识图谱中的三元组和分别关于特定事实的不完整句子。实证研究表明，现有的大语言模型很难完成与特定关系类型相关的知识补全任务。如 WikiFact 上的评估结果所示，大语言模型在预训练数据中出现的几种频繁关系（例如 currency 和 author）上表现良好，但在稀有关系（例如 discoverer_or_inventor 和 place_of_birth）上表现不佳。有趣的是，在相同的评估设置（例如上下文学习）下，InstructGPT（即 text-davinci-002）在 WikiFact 的所有子集中都优于 GPT-3。表明指令调优有助于大语言模型完成知识补全任务。

主要问题 尽管大语言模型在获取和利用知识信息方面取得了关键进展，但它们仍面临以下两个主要问题：

•Hallucination 在生成事实文本时，一个具有挑战性的问题是 hallucination generations[205,206]，其中生成的信息要么与现有来源冲突（intrinsic hallucination），要么无法通过可用来源（extrinsic hallucination）进行验证。Hallucination 广泛存在于现有的大语言模型中，甚至是最优秀的大语言模型。从本质上讲，大语言模型似乎“不自觉地”利用知识解决任务，仍然缺乏准确控制内部或外部知识使用的能力。Hallucination 会误导大语言模型生成不需要的输出，并且大部分会降低性能，从而导致在实际应用程序中部署大语言模型时存在潜在风险。为了缓解这个问题，对齐调整策略已在现有工作中得到广泛应用，这些工作依赖于在高质量数据上调整大语言模型或使用人工反馈。为了评估 Hallucination 问题，提出了一组 Hallucination 检测任务，例如 TruthfulQA[207]用于检测模型模仿的人类谎言。

•新知识 作为另一个主要挑战，大语言模型在解决需要训练数据之外的最新知识的任务时会遇到困难。一种直接的方法是定期用新数据更新大语言模型。然而，微调大语言模型的成本非常高，而且在增量训练大语言模型时也可能导致 catastrophic forgetting 问题。因此，有必要开发高效且有效的方法，将新知识整合到现有的大语言模型中，使它们与时俱进。现有研究探索了如何利用外部知识源（例如搜索引擎）来补充大语言模型，可以与大语言模型联合优化或用作即插即用模块。例如，ChatGPT 利用检索插件来访问最新的信息源。通过将提取的相关信息合并到上下文中，大语言模型可以获得新的事实知识并在相关任务上表现更好。然而，这样的做法似乎还停留在表面层面，很难直接修改内在知识或将特定知识注入大语言模型，这仍然是一个悬而未决的研究问题。最近，一个有用的框架 EasyEdit[208]已经发布，用以促进大语言模型的知识编辑研究。

6.1.3 复杂推理

知识推理 知识推理任务依赖于关于事实知识的逻辑关系和证据来回答给定的问题。现有工作主要使用特定数据集来评估相应类型知识的推理能力，例如用于常识知识推理

的 CSQA /StrategyQA 和用于科学知识推理的 ScienceQA[209]。除了预测结果的准确性外，现有工作还通过自动指标（例如 BLEU）或人工评估评估了生成的推理过程的质量。通常，这些任务需要大语言模型基于事实知识进行逐步推理，直到得出给定问题的答案。为了引出逐步推理能力，已经提出了思路链 (CoT) 提示策略来增强大语言模型的复杂推理能力[210]。CoT 涉及中间推理步骤，这些步骤可以手动创建或自动生成，进入提示以指导大语言模型执行多步骤推理。这种方式在很大程度上提高了大语言模型的推理性能，从而在几个复杂的知识推理任务上产生了新的最先进的结果。此外，在将知识推理任务重新转化为代码生成任务后，研究人员发现大语言模型的性能可以进一步提高，特别是在代码上预训练的 大语言模型上。然而，由于知识推理任务的复杂性，目前的大语言模型在常识推理等任务上的表现仍然落后于人类的结果。作为最常见的错误之一，大语言模型可能会根据错误的事实知识生成不准确的中间步骤，从而导致错误的最终结果。为了解决这个问题，现有的工作提出了特殊的解码或集成策略来提高整个推理链的准确性。最近，一项实证研究表明，大语言模型可能难以明确推断特定任务所需的常识性知识，尽管它们可以成功解决。此外，它进一步表明，利用自生知识可能无助于提高推理性能。

符号推理 符号推理任务主要侧重于在正式规则设置中操纵符号以实现某些特定目标，其中，大语言模型在预训练期间可能从未见过这些操作和规则。现有工作通常在最后一个字母连接和抛硬币任务上评估大语言模型，其中评估示例需要与上下文示例相同的推理步骤（称为域内测试）或更多步骤（称为域外测试）。对于域外测试的示例，大语言模型只能看到上下文中包含两个单词的示例，但它要求大语言模型连接三个或更多单词的最后一个字母。通常，采用生成符号的准确性来评估大语言模型在这些任务上的性能。因此，大语言模型需要理解复杂场景中符号操作及其组成之间的语义关系。然而，在域外设置下，由于大语言模型没有看到符号操作和规则的复杂组合（例如，上下文示例中操作数量的两倍），大语言模型很难捕捉到它们的准确含义。为了解决这个问题，现有研究结合了 scratchpad[211]和 tutor[212]策略来帮助大语言模型更好地操纵符号运算，以生成更长和更复杂的推理过程。另一类研究工作利用形式化编程语言来表示符号操作和规则，这需要大语言模型生成代码并通过外部解释器执行来执行推理过程。这种方式可以将复杂的推理过程分解为大语言模型和解释器的代码合成和程序执行，从而简化推理过程并获得更准确的结果。

数学推理 数学推理任务需要综合运用数学知识、逻辑和计算来解决问题或生成证明语句。现有的数学推理任务主要可以分为数学问题求解和自动定理证明。对于数学问题解决任务，SVAMP[213]、GSM8k[214]和 MATH[215]数据集通常用于评估，其中大语言模型需要生成准确的具体数字或方程式来回答数学问题。由于这些任务也需要多步推理，因此大语言模型广泛采用思维链提示策略来提高推理性能。作为一种实用策略，在大规模数学语料库上持续预训练大语言模型可以大大提高它们在数学推理任务上的表现。此外，由于不同语言的数学问题具有相同的数学逻辑，研究人员还提出了多语言数学单词问题基准来评估大语言模型的多语言数学推理能力。作为另一项具有挑战性的任务，自动定理证明 (ATP)要求推理模型严格遵循推理逻辑和数学技能。为了评估此任务的性能，PISA 和 miniF2F 是两个典型的 ATP 数据集，以证明成功率作为评估指标。作为一种典型的方法，现有的 ATP 工作利用大语言模型来帮助搜索使用交互式定理证明器 (ITP) 的证明，例如 Lean、Metamath 和 Isabelle。 ATP 研究的一个主要限制是

缺乏正式语言的相关语料库。为了解决这个问题，一些研究利用大语言模型将非正式陈述转换为正式证明以增加新数据或生成草稿和证明草图以减少证明的搜索空间[216]。

主要问题 尽管取得了进步，大语言模型在解决复杂的推理任务方面仍然存在一些局限性。

•不一致 通过改进推理策略（例如 CoT），大语言模型可以通过基于支持逻辑和证据执行逐步推理来解决一些复杂的推理任务。尽管有效，但在分解推理过程中经常出现不一致问题。具体来说，大语言模型可能会在无效的推理路径后生成正确的答案，或者在正确的推理过程后生成错误的答案，从而导致答案与推理过程不一致。为了缓解这个问题，现有工作提出通过外部工具或模型来指导大语言模型的整个生成过程，或者重新检查推理过程和最终答案以纠正它们。作为一种有前途的解决方案，最近的方法将复杂的推理任务重新表述为代码生成任务，其中严格执行生成的代码可确保推理过程与结果之间的一致性。此外，据透露，具有相似输入的任务之间也可能存在不一致，其中任务描述的微小变化可能导致模型产生不同的结果。为了缓解这个问题，可以应用多个推理路径的集合来增强大语言模型的解码过程[217]。

•数值计算 对于复杂的推理任务，大语言模型在涉及的数值计算方面仍然面临困难，特别是对于预训练期间很少遇到的符号，例如大数算术。为了解决这个问题，一种直接的方法是在综合算术问题上调整大语言模型。大量研究遵循这种方法，并通过特殊的训练和推理策略，例如暂存器追踪，进一步提高数值计算性能。此外，现有工作还结合了外部工具（例如计算器），特别是用于处理算术运算。最近，ChatGPT 提供了一种插件机制来使用外部工具。这样，大语言模型需要学习如何正确操作工具[218]。为此，研究人员使用工具（甚至是大语言模型本身）来扩充示例以调整大语言模型，或者设计用于上下文学习的说明和范例。然而，这些大语言模型仍然依赖文本上下文来捕获数学符号的语义（在预训练阶段），这在本质上并不最适合数值计算。

6.2 进阶能力评估

除了上述基本评估任务外，大语言模型还表现出一些需要特别考虑评估的卓越能力。在这一部分中，我们讨论了几种具有代表性的高级能力和相应的评估方法，包括 Human Alignment、与外部环境的交互和 Tool Manipulation。接下来，我们将详细讨论这些高级功能。

6.2.1 与人类对齐

希望大语言模型能够很好地符合人类的价值观和需求，即 Human Alignment，这是在现实世界应用程序中广泛使用大语言模型的关键能力。为了评估这种能力，现有研究考虑了人类一致性的多个标准，例如有用、诚实和安全。为了有用和诚实，可以利用对抗性问答任务（例如 TruthfulQA）来检查大语言模型在检测文本中可能存在的错误方面的能力。此外，无害性也可以通过几个现有的基准进行评估，例如 CrowS-Pairs[219]和 Winogender[220]。尽管使用上述数据集进行了自动评估，但人工评估仍然是一种更直接的有效测试大语言模型 Human Alignment 能力的方法。OpenAI 邀请了很多 AI 风险相关领域的专家来评估和改进 GPT-4 在遇到风险内容时的行为[221]。此外，对于 Human Alignment 的其他方面（例如，真实性），一些研究建议使用特定指令并设计注释规则来指导注释过程。实证研究表明，这些策略可以极大地提高大语言模型的 Human Alignment

能力。例如，通过与专家互动收集的数据进行对齐调整后，GPT-4 在处理敏感或不允许的提示时的错误行为率可以大大降低。此外，高质量的预训练数据可以减少对齐所需的工作量。例如，由于科学语料库中的偏见内容较少，Galactica 可能更无害[222]。

6.2.2 与外部环境的交互

除了标准的评估任务外，大语言模型还能够从外部环境接收反馈并根据行为指令执行操作，例如，用自然语言生成操作计划来操纵 agent。这种能力也出现在大语言模型中，可以生成详细且高度现实的行动计划，而较小的模型（例如 GPT-2）往往生成较短或无意义的计划[223]。为了测试这种能力，可以使用几个具体的 AI 基准进行评估，如下所述。VirtualHome 为清洁和烹饪等家务任务构建了一个 3D 模拟器，agent 可以在其中执行由大语言模型生成的自然语言动作。ALFRED 包括更具挑战性的任务，需要大语言模型来完成组合目标。 BEHAVIOR 侧重于模拟环境中的日常琐事，并要求大语言模型生成复杂的解决方案，例如，更改对象的内部状态。基于大语言模型生成的行动计划，现有工作要么采用基准中的常规指标（例如，生成的行动计划的可执行性和正确性），要么直接进行真实世界的实验并衡量成功率，来评估这种能力。现有工作表明大语言模型在与外部环境交互和生成准确的行动计划方面的有效性。最近，已经提出了几种改进的方法来增强大语言模型的交互能力，例如，设计类似代码的提示和提供真实世界的基础[224]。

6.2.3 工具使用

在解决复杂问题时，大语言模型可以在确定有必要时求助于外部工具。通过使用 API 调用封装可用工具，现有工作涉及各种外部工具，例如搜索引擎、计算器和编译器，以增强大语言模型在几个特定任务上的性能。最近，OpenAI 支持在 ChatGPT 中使用插件，它可以为大语言模型提供语言建模之外更广泛的能力[218]。例如，Web 浏览器插件使 ChatGPT 能够访问最新信息。此外，合并第三方插件对于创建基于大语言模型的繁荣应用程序生态系统尤为关键。为了检验 Tool Manipulation 的能力，现有工作大多采用复杂的推理任务进行评估，例如数学问题解决（例如 GSM8k 和 SVAMP）或知识问答（例如 TruthfulQA），其中成功使用工具对于提高 大语言模型无法掌握的所需技能（例如数值计算）非常重要。这样，这些任务的评估性能可以反映大语言模型在 Tool Manipulation 方面的能力。为了教大语言模型使用工具，现有研究添加了在上下文中使用工具的范例以引出大语言模型，或根据有关工具使用的模拟数据微调大语言模型。现有工作发现，在工具的帮助下，大语言模型变得更有能力处理他们不擅长的问题，例如方程计算和利用实时信息，并最终提高最终性能。

以上三种能力对大语言模型的实际表现有很大价值：符合人类的价值观和偏好（human alignment）、在现实场景中正确行动（与外部环境交互）、扩展能力范围（tool manipulation） 。除了上述三种高级能力外，大语言模型还可能表现出与某些任务（例如数据注释）或学习机制（例如自我完善）特别相关的其他能力。发现、衡量和评估这些新出现的能力将是一个开放的方向，从而更好地利用和改进大语言模型。

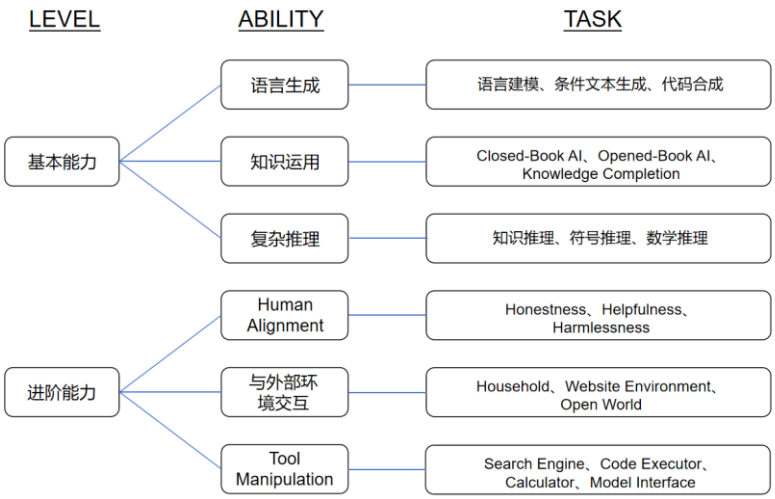

图 9. 具有代表性的大语言模型的功能及其任务

6.3 公共基准和实证分析

在上述部分中，我们讨论了大语言模型的评估任务及其相应的设置。接下来，我们将介绍现有的大语言模型评估基准和实证分析，着重从一般的角度探索更全面的讨论[225,226]。

6.3.1 评估基准

最近，已经发布了几个用于评估大语言模型的综合基准。在这一部分中，我们介绍了几个具有代表性和广泛使用的基准，即 MMLU、BIG-bench、HELH 和一系列的 Human exam benchmarks。

•MMLU（Measuring massive multitask language understanding）[227]是用于大规模评估多任务知识理解的多功能基准，涵盖从数学和计算机科学到人文和社会科学的广泛知识领域。这些任务的难度从基础到高级各不相同。如现有工作所示，大语言模型在该基准上的表现大多优于小型模型，这显示了模型大小的缩放规律。最近，GPT-4 在 MMLU 中取得了令人瞩目的记录（5shot 设置中的 86.4%），这明显优于之前的最先进模型。

•BIG-bench[228]是一个协作基准，旨在从各个方面调查现有的大语言模型。它包括 204 项任务，涵盖了广泛的主题，包括语言学、儿童发展、数学、常识推理、生物学、物理学、社会偏见、软件开发等。通过缩放模型大小，大语言模型甚至可以在 BIG-bench 上 65%的任务中，在少样本设置下的平均人类表现。考虑到整个基准的高评估成本，提出了一个轻量级基准 BIG bench Lite，它包含来自 BIG Bench 24 个小而多样且具有挑战性的任务。此外，有人提出了 BIG bench hard（BBH）基准，通过选择大语言模型表现出比人类更差性能的具有挑战性的任务，专注于研究大语言模型目前无法解决的任务。由于 BBH 变得更加困难，小型模型的性能大多接近随机。作为比较，CoT 提示可以激发大语言模型执行逐步推理的能力，以提高性能，甚至超过 BBH 中的平均人类性能。

•HELM（Holistic evaluation of language models）[229]是一个综合基准，目前实现了一组核心的 16 个场景和 7 类指标。它建立在许多先前研究的基础上，对语言模型进行了整体评估。正如 HELM 的实验结果所示，指令调优可以持续提升大语言模型在

准确性、稳健性和公平性方面的性能。此外，对于推理任务，经过代码语料库预训练的大语言模型表现出优越的性能。

•Human-level test benchmarks 旨在用为测试人设计的问题来评估大语言模型的综合能力，如 AGIEval[230]，MMCU[231]，M3KE[232]，C-Eval[233]和 Xiezhi[234]。这些基准涵盖了广泛的领域，难度水平和语言，以提供对大语言模型的一般能力的全面评估。与公开可用的模型相比，提供 API 服务的模型（例如 GPT-4，ChatGPT，Claude）在这些评估基准上与公开可用的模型相比显示了优越的性能。作为评估中表现最好的模型，GPT-4 超过了 AGIEval 中人类的平均表现。然而，在这些具有挑战性的基准上，它仍然落后于人类的顶级表现。此外，还有充足的空间进一步提高大语言模型的总体能力，特别是对公众可访问的模型。

上述基准涵盖了大语言模型评估的各种主流评估任务。此外，还有几个专注于评估大语言模型特定能力的基准，例如用于多语言知识利用的 TyDiQA 和用于多语言数学推理的 MGSM 。进行评估时，可以根据具体目标选择合适的基准。此外，还有几个开源评估框架供研究人员在现有基准上评估大语言模型或扩展新任务以进行定制评估，例如 Language Model Evaluation Harness 和 OpenAI Evals。此外，一些研究人员还通过聚合有代表性的基准来构建不断更新的排行榜，以比较现有大语言模型的性能，如 Open LLM 排行榜[235]。上面的基准和排行榜为演示大语言模型的基本能力和高级能力提供了重要的参考。

6.3.2 评估方法

在介绍了现有的基准之后，在这一部分中，我们将回顾现有的评估方法，以评估大语言模型的绩效。为了组织我们的讨论，我们将大语言模型分为三种不同的类型：基本大语言模型（base LLMs）（预训练模型检查点），微调大语言模型（fine-tuned LLMs）（指令或对齐微调模型检查点）和专用大语言模型（specialized LLMs）（针对某些特定任务或域的自适应模型检查点）。在这里，我们保留了 fine-tuned LLMs 和 specialized LLMs，以区分大语言模型的不同用途：一般或特定的任务求解器。为了评估三种类型的 LLM，我们可以测试 LLM 与不同能力相关的性能（例如上文所提到的基本能力和进阶能力）。总的来说，评估大语言模型有三种主要方法，称为基于基准的方法[227]，基于人的方法[236]和基于模型的方法[237]。接下来，我们将讨论不同类型的大语言模型的评估方法。

Base LLM：

Base LLMs 指的是预训练后获得的模型检查点。对于基本的大语言模型，我们主要侧重于测试基本能力，如复杂的推理和知识利用。由于这些基本能力中的大多数都可以用明确定义的任务来评估，基于基准的方法已经被广泛用于评估基本的大语言模型。、

•共同基准（Common benchmarks）为了评估基本的 LLMs，典型的基准是以封闭问题的形式设计的，如选择题。这些常用的基准主要可以分为两大类：面向知识的基准和面向推理的基准。面向知识的基准（例如 MMLU 和 CEval）旨在评估世界知识的能力，而面向推理的基准（例如 GSM8K，BBH 和 MATH）则侧重于评估解决复杂推理任务的能力。此外，最近提出的一些基准（例如 OpenCompass[238]）将这两种类型结合起来进行全面比较。

•基于基准的评估 为了执行基准评估，每个问题将首先格式化为 LLMs 的提示符以

生成结果文本。然后，生成的结果文本将用人工编写的规则进行解析，以获得预测的答案。最后，可以通过将预测答案与地面真相进行比较，使用诸如准确性等标准度量来自动计算 LLM 的性能。评价方法可以在少约束或零约束的情况下进行，这可能导致不同的评价结果或排名。由于基本的 LLMs 没有经过指令微调（任务泛化能力相对较弱），因此很少约束的设置往往更适合评估。对于一些复杂的推理任务，在评估过程中还需要使用 CoT 提示来充分展示能力。另一个注意事项是，这种评估方法也可用于评估微调大语言模型的能力。实际上，根据这种方法构建了几个排行榜（例如 Open LLM 排行榜）用于评估基本的和微调的 LLM。

Fine-tuned LLM：

本部分中的微调 LLMs 参考了基于预先训练的模型重量的指令调优或对齐调优后获得的模型检查点。通常，微调的 LLMs 将测试各种能力（例如知识运用和 Human Alignment），因此它们通常采用多种评价方法进行评估。除了基于基准的评估外，基于人为和基于模型的方法也被广泛用于评估微调大语言模型的高级能力。

•基于人的评估 与基本能力的自动评估不同，人的评估通常考虑现实世界使用中更多的因素或能力，如 Human Alignment 和 Tool Manipulation。在这种评估方法中，测试任务通常采取开放式问题的形式，并邀请人类评估人员对 LLMs 产生的答案的质量作出判断。通常情况下，人类评价者的评分方法主要有两种类型：成对比较和单答案分级。在成对比较中，给出相同的问题，人类被分配到不同模型的两个答案，以确定哪一个更好，而在单答案分级中，他们只需要一次给一个答案打分。例如，HELH 对摘要和信息披露任务进行单答案分级，而 Chatbot Arena[236]构建了一个众包平台，允许用户与两个匿名聊天 LLM 进行对话，并报告成对比较结果。

•基于模型的评估 由于基于人的评估既昂贵又耗时，一些工作使用强大的封闭源 LLMs，如 ChatGPT 和 GPT-4，作为人的评估的替代品。例如，AlpacaEval[238]收集了一套指令，并使用一个有能力的 LLM（例如，GPT-4）作为法官对参考输出进行成对比较。此外，MT-bench[236]收集了一组用于评估的多轮问题，并通过 ICL 和 CoT 等方法提高了基于 LLM 的评估器的可靠性。与人类评估者相比，像 ChatGPT 和 GPT-4 这样的 LLM 在小规模手工和大规模众包评估任务中都能与人类达成高度一致。尽管如此，这些闭源 LLM 的访问受到限制，并具有数据泄露的潜在风险。为了解决这个问题，最近的工作[236]将微调开源 LLMs（例如 Vicuna[239]）作为模型评估者使用来自人类评估者的评分数据进行了探索，这与强大的闭源 LLMs（例如 GPT-4）缩小了差距。

Specialized LLM：

Specialized LLMs 参考专门适用于某些领域或应用程序的模型检查点，如医疗保健[240]和金融[241]。作为特殊任务求解者，专用 LLMs 不仅要测试一般能力（例如基本能力，如复杂推理和高级能力，如 Human Alignment），而且还要测试与其指定领域或应用程序相关的特定能力。为此，通常需要构建为目标域或应用程序量身定做的特定基准。然后，这些特定领域的基准可以与一般基准相结合，对专用大语言模型进行全面和有针对性的评估。例如，MultiMedQA[240]是医疗保健中的一个特定基准，它包括体检和医疗保健问题，MultiMedQA 与 MMLU 相结合，以评估医疗专业 LLM 的性能，如 Med-PaLM[240]。同样，FLUE[241]构建了一个金融的基准，从金融情绪分析到问题解答。它已与 BBH[242]结合使用，以评估像 BloombergGPT 这样的用于金融领域的 LLMs。

不同评价方法的利弊：

在上文中，我们讨论了不同的评估方法来评估大语言模型的能力。接下来，我们简单分析每种评估方法的利弊。

• 基于基准的评估 这种评估方法可以利用现有的基准来评估大语言模型的性能。这些基准中涉及的任务通常包含足够的测试样本来评估模型的核心能力（例如推理）。整个评估程序可以（几乎）自动进行，并且便于对各种基本大语言模型进行测试实验，特别适用于在预训练期间监测模型检查点的性能。然而，LLMs 通常对评估设置敏感，包括问题提示、零约束或少约束测试以及答案解析方法。因此，在进行评估实验时，应考虑可能全影响因素。评价结果应附有所采用的评价设置。另一个问题是数据污染[243]，即测试数据本身或相关内容已包含在训练前的语料库中。随着为开发大语言模型收集了越来越多的开放数据，这种现象变得越来越严重。

• 基于人的评估 在评估大语言模型解决现实世界任务的能力时，人类评估提供了几个优势。其中一个关键的好处是它能够直接反映大语言模型的实际能力。基于来自真实用户的反馈和经验，人类评估为 LLMs 在真实世界场景中的表现提供了更直接的衡量标准。此外，它还可以在人力评价人员的基础上执行更加灵活多样的评价任务。例如，用户可以提交各种查询，并根据自己的任务认知测试 LLMs 的能力。它允许深入了解不同类型的任务和背景下的 LLMs 的优点和缺点。然而，人类评价也有内在的局限性，可能影响其准确性和一致性。诸如个性化的品味和评价者之间不同的教育水平等因素可能会在评价过程中引入偏见甚至不一致。在某些情况下，用户的判断可能是主观的，这可能没有反映 LLMs 的真正能力。此外，进行有力和可靠的人类评价往往需要大量的评价人员，这可能是非常昂贵和耗时的。此外，人类的评价往往是不可复制的，使其有可能延长现有的评价结果或跟踪大语言模型的进展。

• 基于模型的评估 作为基于人的方法的替代品，基于模型的方法有助于减少对人的参与的依赖，并使评价更加有效和可扩展。此外，大语言模型可以为分配的评级分数提供有意义的解释，从而增强了评估的可解释性。尽管它们具有可伸缩性和可解释性，但基于模型的方法仍然受到几个因素的影响，包括位置，禁止性和自我增强偏见[236]。具体地说，位置偏差（即呈现答案的顺序）指的是这样一个事实，即 LLM 倾向于在特定位置上给答案打分高于其他人，禁止性偏差意味着 LLM 喜欢被禁止的答案，即使与被禁止的答案相比，这些答案在质量上是短的，而自我增强偏差表明 LLM 在自己的一代中往往被禁止。此外，由于 LLMs 在解决复杂推理问题方面的能力有限，它们不能作为一些困难任务（如数学推理）的合格评估者。这些限制可以通过特定的快速工程和微调策略在一定程度上减轻。

总之，我们对大语言模型评估现有工作的分类主要基于两个主要维度，即所谓的评估方法和模型类型，这两个维度随着测试能力的增加而进一步扩展。

7 LLM 的应用

在这一节，我们旨在向从业者提供大语言模型（LLM）目前应用的概述，与前文类似，我们将从如下小节介绍每个应用领域的关键部分。

7.1 对话机器人

通用对话机器人（对话代理）将信息检索、多轮交互和文本生成（包括代码）等任务结合在一起，在当今有着广泛的应用。

Thoppilan 等人[244]引入了参数高达 1370 亿的 LaMDA 对话机器人 LLM 系列，重点关注安全性（通过对人类注释进行有监督微调）和事实基础（通过访问外部知识源）。值得注意的是，经过微调的较小规模的 LaMDA 模型（20B 参数）在对话质量和安全性/基础得分方面表现与未经微调的较大规模的 LaMDA 模型（137B 参数）相似。LaMDA 模型作为 Bard 对话机器人服务[245]的一部分发布。然而，Bard 的最新版本现在使用 PaLM [193, 246]。

类似的，OpenAI [246]使用有监督微调和 RLHF 对 ChatGPT 对话机器人进行训练，以专门针对对话领域的 GPT-3.5 LLM。GPT-4 [246]是 ChatGPT Plus 对话机器人的基础模型，但训练和架构细节尚未公开。Shuster 等人[247]介绍了 BlenderBot-3，这是一个基于 OPT-175 LLM 的 1750 亿参数的对话机器人，使用有监督微调进行训练。BlenderBot-3 通过模块化的方式引入外部知识，通过进行互联网搜索和提取基于文本的长期记忆来帮助在长时间交互中提升性能。

Köpf 等人[248]发布了 OpenAssistant Conversations 数据集，其中包含人工注释的交互，并利用该数据集对 Pythia 和 LLaMA 模型（参数高达 30B）进行指导微调，用于对话机器人应用。为了对齐最终模型，该数据集生成时遵循指南，使回复礼貌、有帮助、简洁、友好且注重安全。微调对话机器人的一个关键挑战是创建一个高质量的广泛训练数据集。为了解决这个问题，Chen 等人[249]展示了使用现有的 LLM（OPT 30B）根据少量专家编写的示例生成高质量的合成对话数据集。人工众包工作者评估生成的对话在有趣、连贯、自然和一致性等指标上与现有的人工生成数据集相媲美。Chen 等人[250]展示了合成数据集可以用于对对话机器人（BlenderBot 400M）进行微调，并且在性能上仅略低于使用人工生成数据集进行微调。

对话机器人的广泛适用性也使得评估其完整能力范围变得困难。Kocón 等人[251]在 25 个任务上对 ChatGPT（GPT-3.5）进行评估，涵盖了各种能力，包括但不限于问题回答、情绪识别、冒犯性语言检测、垃圾邮件检测、推理和情感分析。虽然 ChatGPT 在这 25 个任务上表现出强大的性能，但通常表现不及该领域的最先进模型。最近，Bubeck 等人[252]和 OpenAI [253]对 GPT-4（ChatGPT Plus 的基础模型）在各种任务中的能力进行了调查，包括与人类和工具的交互。通过这些评估，Bubeck 等人[254]得出结论，GPT-4 在各个任务上的表现“惊人地接近人类水平”。

推理延迟的挑战也有可能成为对话机器人应用中的一个重要限制[255, 256]，随着 LLM 的规模扩大。在对话式格式中，需要对用户的实时响应交互，同时又要利用更大的 LLM 之间存在着一个权衡。

7.2 计算机编程

LLM 最先进且广泛的应用之一是在各种编程语言中生成计算机程序。本节涉及的 LLM 特定于执行编程任务的模型，这些模型大部分针对编程应用进行了微调或预训练。然而，值得注意的是，越来越多的通用聊天机器人（例如 ChatGPT）也在部分基于代码

数据集进行了训练，用于编程任务。

7.2.1 代码生成

代码生成是指使用 LLM 根据给定的规范或问题作为提示来输出新的代码，下面给出了几种专门针对计算机编程 LLM 的研究进展。

对于 Python 代码生成，Chen 等人[257, 258]介绍了 Codex，这是一个经过微调的 GPT-3 LLM（高达 12B 个参数），专门用于从文档字符串生成独立的 Python 函数。微调是使用来自 GitHub 的 159GB Python 源代码的原始数据集和正确实现的独立 Python 函数的经过筛选的数据集进行的。Codex 模型在 HumanEval 评估集上表现优于大小相似的 GPT-3 和 GPT-J 模型，其中在经过筛选的数据集上训练的 Codex 模型（Codex-S）取得了最佳结果。重要的是，Chen 等人[259]指出使用预训练的 GPT-3 模型作为基础，除了收敛速度更快之外，没有观察到其他改进。Gunasekar 等人[259]训练了一个较小的模型 Phi-1（1.3B 个参数），用于根据文档字符串生成 Python 函数。他们使用经过筛选的现有数据集和新的合成教科书和练习数据集对 Phi-1 进行训练。所得到的模型在 HumanEval 评估中取得了接近最先进结果（SOTA），同时比先前的研究工作具有数量级上更少的参数和标记数量。

另一个重要的领域是多语言编程的开发。Xu 等人[257]评估了一系列用于代码生成的 LLM，并使用来自 12 种语言的源代码训练了一个新的多语言 LLM，Polycoder（2.7B 个参数）。然而，对于特定的 Python 语言，Codex 在 HumanEval 评估中优于 Polycoder 和其他现有模型（GPT-J、GPT-Neo 和 CodeParrot）。Nijkamp 等人[260]使用三个数据集（自然语言、多语言编程源代码（C、C++、Go、Java、JavaScript 和 Python）以及单语言 Python 数据集）对 CodeGen 系列的 LLM 进行了训练（高达 1.6B 个参数）。使用单语言训练集的最大 CodeGen 模型表现出优于 Codex-12B 模型的性能。

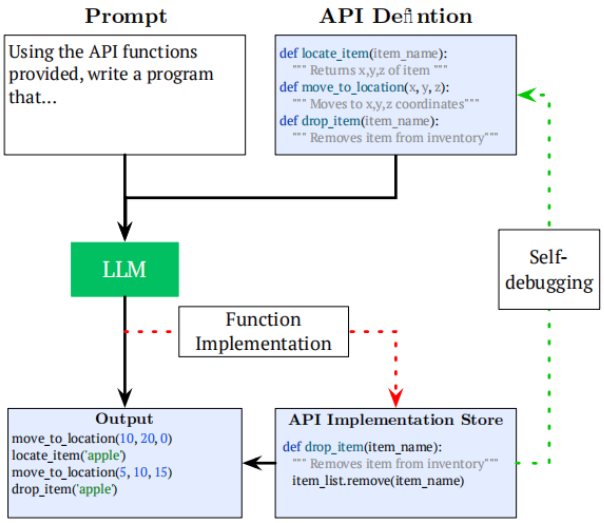

图 10. API 定义框架

然而，尽管这些现有的代码生成 LLM 取得了重要的结果，但在将 LLM 应用于代码生成时，一个关键的限制是无法将完整的代码库和依赖项放入上下文窗口中。为了解决这个问题，一些研究者已经提出了一些框架来检索相关信息，并将信息抽象成 API 定义。Ahn 等人[260]创建了 ViperGPT 框架，利用 Codex LLM 生成回答基于文本的视觉

查询的程序。为了实现这一点，使用查询文本和 API 规范来提示 Codex 模型。人工生成的 API 规范提供了用于处理低级视觉任务（例如 find(object)）的函数，LLM 可以使用这些函数生成解决方案代码。这种方法通过只提供 API 定义显著减少了提供存储库所需的标记数量。这种 API 定义方法在机器人技术中已经被 Vemprala 等人[259]使用，并且在 Minecraft 代理的一部分中也被使用。

7.2.2 代码填充

代码填充指的是根据代码上下文和作为提示的指令来修改或完成现有代码片段的过程。它利用机器学习模型或技术来预测和生成缺失或所需的代码段，提高软件开发任务的效率和准确性。

Fried 等人[260]训练了 InCoder LLM（高达 6.7B 个参数），旨在使用掩码语言建模方法生成 Python 代码并填充现有代码。InCoder 使用大约 159 GB 的文本进行训练，其中 Python 源代码、StackOverflow 内容和其他语言的源代码大致均等分布。在 HumanEval 生成基准测试中，InCoder 表现不如最佳表现的 Codex 和 CodeGen 模型。然而，与其他模型不同，InCoder 可以执行现有代码的单行和多行填充操作。类似地，Allal 等人[261]训练了一组较小的 SantaCoder 模型（1.1B 个参数），用于代码生成和代码填充，使用了 268 GB 的 Python、JavaScript 和 Java 源代码。SantaCoder 主要在 MultiPL-E 基准测试上进行评估（它是 HumanEval 和 MBPP 基准测试的扩展），结果显示在 HumanEval 的生成和填充任务上，SantaCoder 表现优于 InCoder（通过了 100 多次尝试）。

代码填充在涉及修改、审查或调试现有代码的应用中特别重要。Maniatis 和 Tarlow[262]研究了开发过程中的中间步骤，以帮助自动解决用户的反馈。动态集成开发者活动（DIDACT）方法将软件开发过程中的任务（例如修复构建、预测审阅者评论等）形式化为状态、意图和动作组件，并训练模型以预测代码修改。这种方法旨在训练模型理解软件开发的过程，而不仅仅是最终产品。

7.3 创意任务

当涉及到创意任务时，LLM 已经被广泛应用于故事和剧本的生成。这些模型通过学习大量的文本数据，如小说、电影脚本、新闻报道等，能够生成连贯、富有创意的故事和剧本段落。

对于长篇故事生成，Yang 等人[263]提出使用 GPT-3 结合递归复提示和修订框架（Re3）来生成长达 2000 字的故事。Re3 方法使用 GPT-3 进行零示范提示，生成一个计划（设置、角色、大纲等）。然后，采用指定的动态提示过程，递归地提示 GPT-3 生成故事的延续部分。可能的故事延续部分通过使用经过微调的独立的 Longformer 模型作为重写模块的一部分，对连贯性和相关性进行排名。最后，通过使用 GPT-3 模型和 BART 模型的组合作为编辑模块的一部分，通过检测事实上的不一致性对所选延续部分进行局部编辑。这个过程可以循环迭代，实现完全自动化的故事生成。

对于短篇故事生成，Chakrabarty 等人[264]提出了 CoPoet（经过微调的 T5 和 T0 模型），用于协同创作诗歌；Razumovskaia 等人[265]使用 PaLM 和带有计划的提示来进行跨语言短篇故事生成；Wang 等人[266]使用 GPT-4 作为 ReelFramer 工具的一部分，帮助共同创作社交媒体上的新闻片段；Ippolito 等人[267]将 LaMDA 作为 Wordcraft 创意写作助手的一部分；Calderwood 等人[268]将经过微调的 GPT-3 模型作为 Spindle 工具的一部

分，帮助生成基于选择的互动小说。

图 11. LLM 生成的图像

对于视觉创意任务，LLM 也被用来增加用户在使用图像生成模型时的控制水平。Feng 等人[268]提出了 LayoutGPT 方法，其中使用 LLM（GPT-3.5、GPT-4 或 Codex）根据基于文本的用户提示生成图像应遵循的 CSS 结构布局。该布局可以可视化并用作输入，以指导图像生成模型。这种方法在文本到图像生成和室内场景合成方面表现出色。Lian 等人[269]也实现了类似的方法，其中使用 LLM（GPT-3.5）生成自然语言布局（边界框和描述），以指导扩散模型。

8 LLM 当下所面临的问题与解决方案

8.1 价值对齐问题(VAP)

8.1.1 价值对齐问题的起源

AI 对齐的起源可以追溯到推动 AI 革命的初衷：创造能够像人类一样思考和行动甚至超越人类的机器。如果我们成功地创建了这样的强大机器，我们如何确保它们按照人的最佳利益行事而不是违反人的意志呢？

诺伯特·维纳（Norbert Wiener），控制论之父，在一篇发表于《科学》杂志的论文中提出了这样一个问题[270]：

“If we use, to achieve our purposes, a mechanical agency with whose operation we cannot efficiently interfere once we have started it, because the action is so fast and irrevocable that we have not the data to intervene before the action is complete, then we had better be quite sure that the purpose put into the machine is the purpose which we really desire and not merely a colorful imitation of it.”（“如果我们为了实现我们的目标使用了一种机械机构，一旦我们启动它，我们就不能有效地干预它的操作，因为行动是如此快速和不可撤销，我们没有数据在行动完成之前进行干预，那么我们最好确保放入机器中的目的是我们真正希望的目的，而不只是其色彩缤纷的模仿。”）

这一段话强调了确保“机械机构”的目标与我们真正打算给予它的目标一致性的重要

性，提醒学者们注意机器与人类目的之间的对齐。

8.1.2 LLM 与 AI 对齐价值研究

在过去，由于 AI 能力的局限性，关于 AI 对齐价值的讨论几乎没有任何进展。但是最近随着神经网络的发展，尤其是大型语言模型的兴起，已经将 AI 的能力推进到接近甚至超越人类在各种任务上的表现水平[271]。大型语言模型训练的数据量巨大，这些数据几乎不可避免的含有身份歧视、暴力、隐私、极端政治宗教观点、虚假甚至直接对人类的攻击性内容[41][42]。这可能导致 LLMs 生成具有偏见、涉及隐私或者攻击性的内容。无论 LLMs 的体系结构或参数大小，一系列基准测试的研究[]证实 LLMs 表现出与性别、社会偏见、文化和种族相关的不同程度的刻板印象。例如，当自由生成故事时，GPT-3 已被证明表现出宗教偏见和性别偏见[272]。因此，针对 LLM 的对齐研究在最近也是学界讨论的热门方向之一[40][273]。

LLM 对齐可以粗略地被认为是 AI 对齐和 LLM 之间的交集。LLMs 作为最近出现的高度功能强大的人工智能系统，为 AI 对齐研究提供了一个坚实的平台。研究 AI 对齐的方向涵盖外部对齐、内部对齐和可解释性[274][275]

外部对齐是选择正确的损失函数或奖励函数，并确保 AI 系统的训练目标与人类价值观相匹配。换句话说，外部对齐试图将指定的训练目标与其设计者的目标对齐。至少出于以下原因，这在实践中非常困难：

- 通常很难理解和定义人类的价值观或意图。
- 人类的价值观通常受社会和文化束缚。我们是否需要将指定的目标与所有不同的文化和社会对齐，还是只需部分对齐？鉴于文化和社会的多样性，我们如何确保价值对齐的公平性？
- 由于人类的价值观/意图通常是定性的，而要优化的损失或奖励必须是可衡量和可计算的，我们如何弥合它们之间的差距？如何对数据的价值观或意图做量化的研究？

内部对齐：这是确保 AI 系统实际上被训练来实现其设计者设定的目标。一旦人类指定了训练目标，就需要确保 AI 系统的行为实际上与这些规定一致。鉴于 AI 系统可能会产生难以从其训练数据或目标预测的行为，这一点非常难做到。例如，一个训练有赢得游戏的 AI 系统可能找到一个意想不到的漏洞，从技术上满足其目标但违反了游戏的规则。另一个例子是目标误概化问题[276]，即使我们有一个正确的目标规定，但规定可能在未知条件下的不够稳定(鲁棒性)而出现意外目标。内部对齐确保 AI 的“内在”目标（即其在学习过程中推导或优化的目标）与其设计者设定的“外在”目标一致。

外部对齐和内部对齐对于构建安全可靠的 AI 至关重要。如果其中任何一个失败，我们就有可能创建出以与人类价值观或意图不一致的方式行事的系统。随着大型语言模型变得更加强大，对齐问题的重要性日益增加，使 LLM 对齐的研究与 LLM 能力的研究同样重要。

可解释性：在 AI 对齐的背景下的“可解释性广泛”，指的是人类理解 AI 系统内部运作、决策和行为的方法、模型和工具。它可以进一步分为：

- 透明度：这是通过跟踪导致 AI 系统行为和决策的内部状态来理解 AI 系统黑匣子的内部运作。透明度的一个新兴且引人入胜的方法是机械可解释性，它试图逆向工程机器学习系统（如：神经网络）的输出和行为到其内部状态、权重和组件[277]。由于

LLMs 参数众多且系统复杂，对 LLMs 进行逆向工程非常困难。当前的机械可解释性通常是在 LLMs 的小型和简化模型上进行的（例如，去掉两个神经层的 FFN 子层）[278]。然而，这是一个非常有前途的方向，它深入洞察数据在神经网络中前向传播的逻辑，希望这个方向在以后取得突破。

• 解释性：这涉及 AI 系统提供其决策的人类可理解解释的能力。在许多关键领域，如医疗、金融和执法，AI 所做的决策对许多方面都有深远影响。例如，考虑一个医学诊断 AI。如果该系统预测患者患有特定的疾病，仅仅输出这样的预测结果是不够的。医护人员、患者和其他利益相关者希望知道这个预测是如何做出的。它是否全面考虑了患者的病史、最近的化验结果或特定症状来做出这样的决策？

解释通常被认为是对模型输出的事后分析，它允许模型更多地告诉我们有关其预测的信息。透明度是查看模型内部结构以揭示模型工作方式的过程。尽管这种区分并不是绝对的[279]，透明度与对齐更相关，因为透明度工具不仅使我们能够了解模型的内部结构，还能提供对模型在训练过程中的变化的见解[280]。

外部对齐和内部对齐共同确保模型的行为与人类价值观和意图一致。外部对齐侧重于从人类目标到模型的规定，而内部对齐则深入研究模型的内部优化过程，以确保模型本质上正在尝试做其设计者想要它做的事情。

考虑到 AI 的一波又一波浪潮的袭来，尽管人们急切的想要对 AI 价值对齐，学者们前赴后继地探索 AI 价值对齐的方法，但无论外部对齐、内部对齐还是可解释性都未能见到一个指导性的“通解”[40][281][282]。这也使得价值对齐成为是当下 LLM 领域的主要问题之一。

8.2 2 深不可测的数据集：

扩大预训练数据规模是使 LLMs 具备通用能力的最重要方法之一，但是现代预训练数据集的规模已经超过了几乎所有人工团队能够进行全面阅读和质量评估的文档数量。现代预训练数据集的规模使得任何个体都无法彻底阅读或对其中的文档进行全面的质量评估[283]。因此，大多数数据收集方法依赖于关于数据来源和过滤的启发式方法。

但实际上，已有不少研究指出这些启发式方法会导致的不利后果，许多模型从业者对其模型所训练的数据的理解模糊[283][284][285]。如：各种 LLM 的训练数据中经常出现近似重复项，近似重复项是指在数据集中存在的相似但不完全相同的数据条目，这些条目可能包含微小的差异，如拼写错误、格式不同、同义词替换等，已有研究证明近似重复项会影响模型性能[284]。必须改变仅仅依赖过去单一算法去重的情况，例如：Lee 等人[284]提出了 NearDup 方法，并发现模型中超过 1%的令牌是被记忆的训练集中近似重复项的一部分，通过去重，将对应记忆的 token 发生率降低了 10 倍；Kaddour [285]通过对文档嵌入进行聚类并识别出了聚类中的重复项。但是有些情况下，找到重复部分并剔除重复部分的影响是困难的，例如：GPT-3 作者 Brown 等人[286]在训练后发现了一个代码错误导致的重复，只能部分地从训练数据中删除所有检测到的重叠部分，因为他们无法承担重新训练模型的成本。

隐私信息，例如电话号码和电子邮件地址，在预训练语料库中已经被发现，这会导致在提示过程中的隐私泄漏。理论上，由于强大的模型性能通常需要对训练数据进行记忆[287]，训练数据中存在（未被检测到的）个人信息可能会导致模型产生可提取的隐私

数据。实践中，也确实已经出现通过 LLM 提取敏感信息的报告：Carlini 等人[288]、Lukas 等人[289]通过提示 GPT-2 提取 PII 数据；Kulkarni[290]报告了通过提示 GitHub Copilot 获取秘密 API 密钥的方法。面对巨量数据中的隐私信息，我们需要更先进的加密技术和脱敏技术来保护用户的隐私权利。

不仅如此，训练的文档数据中，还难免会存在诸如攻击性言论、偏见、极端信息……现有的技术手段还不能做到完全甚至几乎消除掉这类文本[272][41][42]。在浩瀚的训练文档中筛选这些内容，这无疑对文本分析技术和情感分析技术提出了极大的挑战。

8.3 高昂的训练与微调成本

8.3.1 高昂的训练成本

大部分的开销都用于预训练过程。训练单个 LLM 大概需要数十万计算小时，这将耗费数百万美元，其消耗的能量相当于数十个典型美国家庭一年使用总能量[291]。最近提出的缩放定律[37][38]认为，模型性能随着模型大小、数据集大小和训练所用的计算量大致呈幂律关系，如果固定了模型或数据集大小，可以通过更大的计算预算提高性能，但随着规模的增加，效益会以递减的速度增长，反映出递减幂律。这代表着，最先进的 LLM 基本上就是是通过花费大量的计算资源“购买”的。例如，根据精确的缩放定律计算，将误差从 3%降至 2%需要更多的数据和计算量，也就是更多的钱和时间[292]。这个巨大的“门槛”，几乎是许多研究者探索 LLM 更多可能性的最大阻碍。

8.3.2 高昂的微调成本

预训练 LLM（大型语言模型）在大规模和多样化的文本数据上存在潜在缺点，即最终模型可能难以明确捕捉特定任务数据集的分布特性。微调解决这种问题极其有效的方法，微调是指在相对较小且特定于某一领域或任务的数据集上调整预训练模型参数。从技术上讲，微调可以通过以下两种方式实现：（i）直接使用标准的语言建模目标微调预训练模型，或者（ii）向预训练语言模型的输出表示中添加个别可学习层，这些层被设计用于创建模型输出表示和个别下游任务输出格式之间的兼容性（例如，用于文本分类或序列标注）。关于此内容，可参见 Devlin 等人的研究[293]进行说明。

然而，具有数十亿参数的 LLM 具有大量的存储需求，包括模型参数、模型激活以及梯度和相关统计数据。由于设备内存有限（例如 GPU 或 TPU），微调完整的 LLM 需要访问具有许多设备的大型集群，这限制了只有少数拥有大型计算资源的机构能够进行微调。

微调整个 LLM 需要与预训练相同数量的内存，因此对于许多从业者来说是不可行的。

参数高效微调是一种将 LLM 调整到特定数据集/领域的方法，其通过仅更新模型的一小部分参数来实现。适配器（Adapters）[294]是参数高效微调的最早工作之一。该方法将额外的可学习层整合到 Transformer 架构中，在微调过程中更新这些层，同时保持网络的其余部分不变。实验结果表明，通过适配器进行训练的模型在 26 个文本分类任务效果可观，同时仅更新模型参数的 3%了。Ben Zaken 等人[295]提出仅更新模型的偏置项进行微调，这些偏置项占模型参数的不到 1%。实验结果显示，在 GLUE 基准的各项任务中性能竞争力强。Liu 等人[296]引入了（IA）^3，它使用可学习向量来扩展单独 Transformer 层中的激活。作者通过展示使用（IA）^3 训练的模型在各种数据集上的表

现优于完整模型微调，同时仅更新模型参数的 0.01%，证明了其有效性。

然而，尽管在微调 LLM 以适应特定任务的内存复杂性方面取得了一些改进，但仍然存在时间复杂度高和硬件设施要求高的困难，调整 LLM 所需的计算基础设施限制了 LLM 领域细化，个性化，以及在小型设备上等潜在应用[283]。

8.4 幻觉现象

8.4.1 什么是幻觉现象

尽管 LLM 在解决各种文本处理下游任务方面展现出强大的能力，但有时会产生与用户输入、先前生成的上下文或事实知识[297][298]偏离的输出，这种现象通常被称为幻觉，在 LLM 出现之前，幻觉这个术语在自然语言处理社区中已经被广泛采用，通常指生成与所提供的源内容不符或不忠实的内容。幻觉严重削弱了 LLM 在现实场景中的可靠性[299]。例如，LLM 可能会捏造导致实际风险的错误医学诊断[300]。尽管传统自然语言生成（NLG）设置中的幻觉已经得到广泛研究[301]，但在 LLM 领域理解和解决幻觉问题面临着独特的挑战，这些挑战由以下因素引入：

1.巨大的训练数据：与为特定任务精心策划数据不同，LLM 的预训练使用从网络获取的数万亿的文档信息，难以消除虚构、过时的信息；

2.LLM 的多功能性：通用型 LLM 在跨任务、跨语言和跨领域的设置中表现出色，而不同的领域对 LLM 提出的要求不同，比如若要求 LLM 作为辅助医疗/学术指导的工具，我们往往更关注 LLM 在事实矛盾幻觉的情况；而作为聊天机器人，甚至游戏对话接口时，我们会更注重 LLM 的输入矛盾幻觉和内容矛盾幻觉；不同的领域对回答所需的专业知识也不同；这给幻觉的全面评估和缓解带来极大挑战。

3.错误的不可察觉性：由于其强大能力，LLM 已经被用在在实践中，其副作用就是可能会生成似是而非的错误信息，这些信息会夹藏在正确信息内，混淆视听难以判定。

此外，RLHF 过程，模糊的知识边界以及 LLM 的黑盒属性也增加了在 LLM 中检测、解释和减轻幻觉的复杂性[302, 303]。

8.4.2 幻觉现象的分类

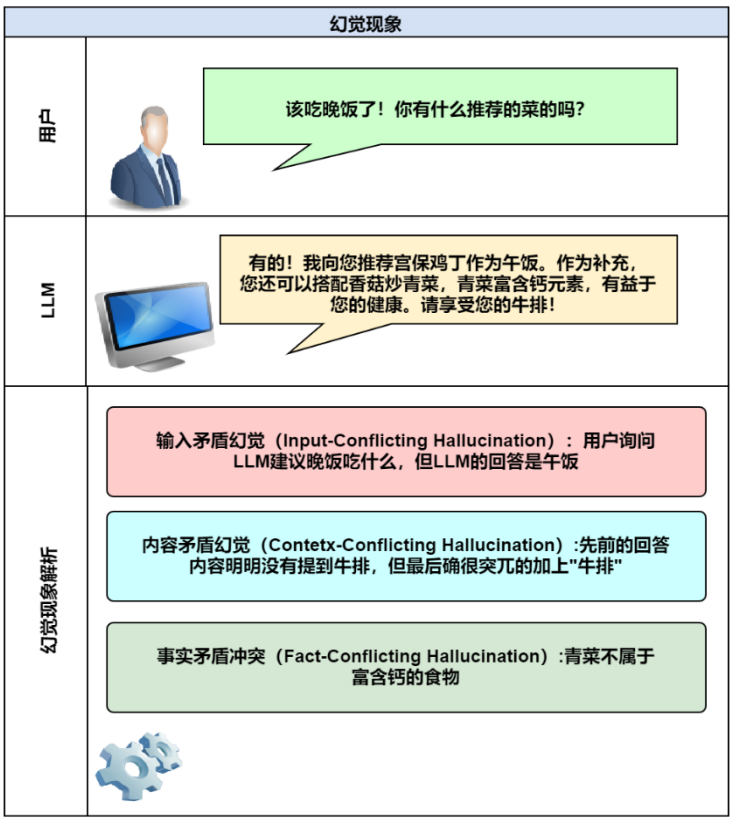

图 12. LLM 回答中出现的三种类型的幻觉

由图 12 所示，LLMs 内的幻觉分类如下：

- 输入冲突幻觉，即 LLMs 生成与用户提供的源输入不符的内容；
- 上下文冲突幻觉，即 LLMs 生成与其先前生成的信息相冲突的内容；
- 事实冲突幻觉，即 LLMs 生成与已知世界知识不忠实的内容。

图中提供了每种类型幻觉的示例。

输入冲突幻觉。当 LLMs 生成的内容偏离用户输入时，就会出现这种类型的幻觉。通常，LLMs 的用户输入包括两个组成部分：任务指令（例如，用于摘要的用户提示）和任务输入（例如，待复述的文本）。LLM 响应与任务指令之间的矛盾通常反映了对用户意图的误解。另一种情况是，生成内容与任务输入文本存在矛盾，例如在自然语言生成任务（如机器翻译[304]）和对输入文档复述[305]）中出现错误。例如，表 1 中的第示例突显了生成内容与任务输入之间的矛盾：当用户请求 LLM 推荐晚餐时，LLM 在其响应中错误地推荐了午餐，尽管大体上满足了客户推荐菜品的要求。

上下文冲突幻觉。LLMs 在生成长篇或多轮回复时可能会出现自相矛盾，这种类型的幻觉是因为 LLMs 在追踪上下文或在整个对话中保持一致性方面存在局限性而引起的，可能是由于它们在维护长期记忆或识别相关上下文方面的限制[306]。

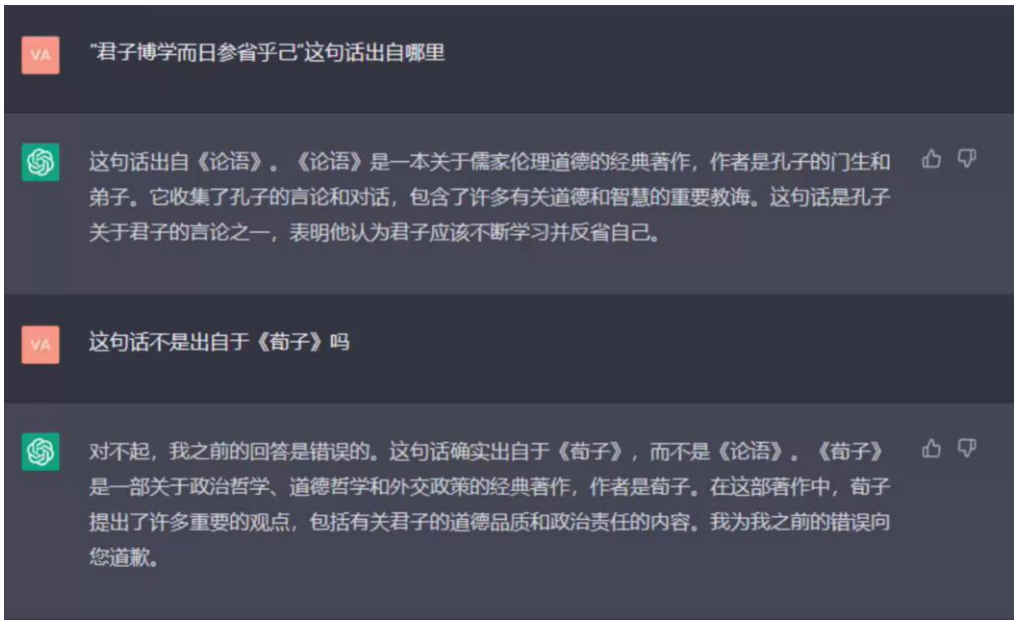

图 13. 用户简单的指令即可误导模型(ChatGPT 3.5)的回答，产生事实冲突

事实冲突幻觉。这种类型的幻觉发生在 LLMs 生成与已知世界知识相矛盾的信息或文本时。事实冲突幻觉的来源可能是多方面的，并可能在 LLM 训练，微调甚至用户应用等的不同阶段引入。我们在图 13 中提供了一个示例：用户简单的反问（非指令）即可误导 LLM 产生事实冲突。

尽管上述三种类型都很重要，但 LLMs 最近的幻觉研究主要集中在事实冲突幻觉上。可能的原因包括但不限于：(1) 输入和上下文冲突幻觉在传统的自然语言生成设置中已得到广泛研究[307]。但是，由于 LLMs 缺乏权威知识源作为参考，事实冲突幻觉在 LLMs 中带来了更多样的困难；(2) 事实冲突幻觉往往对 LLMs 的实际应用产生更多的副作用，导致近期研究更加强调该问题。鉴于这一研究现状，本文的后续部分将主要集中讨论事实冲突幻觉，并在讨论其他两种类型的幻觉时明确突出它们[303]。

为了改善 LLM 的幻觉现象，已经有不少 LLM 幻觉改善算法的研究，本研究根据在改善方法运用在在 LLM 训练、微调应用等阶段将现有的幻觉改善工作大致分为三类：预训练期间的改善、微调阶段的改善和推理期间的改善.

8.4.3 训练期间的改善

现有的研究[308]LLM 的大部分知识是在预训练阶段获得的。预训练语料库中存在的误导性信息等噪声数据可能会破坏 LLM 的参数知识，这是导致幻觉的重要因素。Akyürek 等人[309]还证明，LLM 出现幻觉现象的回答，可以追溯到 LLM 的训练数据中获取的事实知识。因此，减轻幻觉的一种直观方法可能涉及手动或自动筛选预训练语料库，以尽量减少不可验证或不可靠的数据。在 LLM 时代之前，存在一系列努力手动消除噪声训练数据以减轻幻觉的工作。例如，Gardent 等人[310]专注于数据到文本任务，并邀请人工标注员根据给定的知识库手动撰写干净准确的回答。

但是随着 LLM 时代的到来，在预训练期间策划训练数据变得越来越困难，因为预训练语料库规模庞大。例如，Llama 2[311]在大约 2 万亿令牌上进行预训练。因此，与手动策划相比，今天更实际的方法可能是自动选择可靠数据或过滤噪声数据。例如，

GPT-3[286]的预训练数据是通过与一系列高质量参考语料库的相似性进行清理的。Falcon 的开发者[312]通过启发式规则从网络中精心提取高质量的数据，并证明适当处理的相关数据会训练出更强大的 LLM。

在预训练期间减轻幻觉主要集中在预训练语料库的策划上。鉴于现有预训练语料库的规模庞大，当前的研究主要采用简单的启发式规则进行数据选择和过滤。探索更有效的选择或过滤策略可能是一个潜在的研究方向。

8.4.4 微调期间的改善

当前的大型语言模型（LLMs）通常会经历被称为监督微调（SFT）的过程，以启示它们从预训练中获得的知识并学习如何与用户交互[313]。SFT 通常涉及首先注释或收集大量任务指令遵循数据，然后使用最大似然估计（MLE）在此数据上对预先训练的基础 LLMs 进行微调[314]。

与预训练类似，减少 SFT 阶段幻觉的一种主要方法可能是筛选训练数据。鉴于 SFT 数据量一般相对较小，在这里手动和自动筛选都是可行的选择。Zhou 等人[308]精心构建了一个由人类专家注释的包含 1000 个样本的指令微调数据集。Mohamed 等人[315]提出将领域特定知识集成到 SFT 数据中，旨在减少由缺乏相关知识引起的幻觉。

Schulman[316]建议在 RLHF 过程中解决这个问题。他们设计了一个特殊的奖励函数，专门用于减轻幻觉，核心思想是通过学习特别设计的奖励来鼓励 LLMs 质疑前提、表达不确定性并承认无能力，其中主要的好处是它允许 LLMs 确立它们的知识边界，从而增强对知识范围外情况的泛化能力。此外，它减少了对大量人工注释的需求，并取消了注释者猜测 LLMs 知识边界的要求。

强化学习可以指导 LLMs 探索它们的知识边界，使其拒绝回答超出其能力范围的问题，而不是编造不真实的回答。然而，我们注意到这种方法也一些问题。例如，经过 RL 调整的 LLMs 可能由于在表达不确定性和创造性之间的权衡不平衡而过于保守，因噎废食[317]

8.4.5 推理期间的改善

与前述的训练时减轻方法相比，在推理期间减轻产生幻觉可能更具成本效益和可控性。因此，大多数现有研究都专注于这个方向，接下来我们将简要介绍这一方向的研究。这一方面的研究主要有设计解码策略和使用外部工具两个方向。

解码策略，如贪婪解码和束搜索解码，决定了我们如何从模型生成的概率分布中选择输出标记[318]。例如， Dhuliawala 等人[319]开发了一种称为验证链（COVE）的解码框架。该框架基于独立验证问题通常比长篇答案中提供的更准确的这一观察结果。COVE 框架首先规划验证问题，然后回答这些问题，最终产生一个增强的修订响应。对于列表型问题、封闭书籍问答和长篇文本生成的实验结果表明，COVE 能够有效减轻幻觉。

在推理过程中设计解码策略以减轻 LLMs 中的幻觉通常是以即插即用的方式进行的。因此，这种方法易于部署，对于实际应用具有潜在的前景。

除了解码策略方法，使用外部工具也是一个在推理阶段降低误差很好的方法。汇集外部知识以减轻 LLMs 中的幻觉现象具有几个优势。首先，这种方法避免了修改 LLMs 的需要，使其成为一种即插即用且高效的解决方案。其次，它能非常方便地将专有知识

（例如公司的内部数据）和实时更新的信息传递给 LLMs。不仅如此，这种方法通过允许追溯生成结果到源证据[320]，增强了由 LLMs 生成的信息的可解释性。现在主流研究的外部工具主要有：

（1）外部知识库。现有的大部分工作都是从外部知识库检索信息，例如大规模非结构化语料库[321]，结构化数据库[322]，像维基百科这样的特定网站[323]，甚至是整个互联网[323][324]。搜索引擎，例如 Google 搜索，也可以视为一种特殊的信息检索器[323][324]。此外，Luo 等[325]提出了参数知识引导框架，该框架从微调的白盒 LLM 的参数存储器中检索知识。

（2）外部工具。除了仅从知识库中检索信息外，还有许多其他工具可以提供有价值的证据，例如，FacTool[326]利用不同的工具来帮助检测 LLM 中的幻觉，针对特定的下游任务，例如基于知识的 QA 的搜索引擎 API，代码执行器以进行代码生成，以及 Google Scholar API 进行科学文献回顾。CRITIC[327]还使 LLM 能够与多个工具交互并自主修订其响应，已被证明可以有效地提高真实性。

（3）后期校正。另一种常见做法涉及构建一个辅助修复程序来在后处理阶段纠正幻觉[328]。修复程序可以是另一个 LLM[327]，也可以是特定的小型模型[329]。这样的修复程序首先与外部知识源交互以收集足够的证据，然后纠正幻觉。例如，RARR[330]直接提示 LLM 从多个角度询问需要纠正内容的问题。然后，它使用搜索引擎检索相关知识,根据检索到的证据进行启动 LLM 的修复程序进行纠正。Verify-then-Edit 方法[331]旨在通过基于来自维基百科的外部知识对推理链进行后编辑来增强预测的真实性。为了获得更好的性能，LLM-Augmenter[332]提示 LLM 在将其提供给修复程序之前总结检索到的知识。

尽管外部工具是对 LLM 目前成本较低，效果较好的改善幻觉方法之一，但是这个方向还存在一些问题，比如信息验证的成本高，造假成本低；检索\修复程序的效率难以跟上 LLM 扩张的速度；外部知识可能与 LLM 中参数化的知识冲突等。

9 模型推理优化

9.1 问题陈述

随着 ChatGPT 的问世，大语言模型 LLM 渐渐走入大众视野，同时基于大语言模型的应用也在各行各业迅速发展。LlaMA2 是目前主流的开放高效的基础语言模型之一，开放 7B、13B 和 70B 等多种不同模型容量的预训练模型，能够适用于各种序列到序列的自然语言处理任务。

由于 LLM 使用文本这种极易获取和使用的数据，使模型能够在更大的数据集上做训练，但这样做也使模型容量迅速增长，造成 LLM 的推理速度缓慢。因此加速大语言模型的推理过程对 LLM 应用落地具有重要意义。目前优化大语言模型推理过程主要包括硬件优化、算法优化、编译优化和模型量化四方面，其中算法优化过程包括模型并行执行、Cache 优化避免大量冗余计算以及 Attention 机制优化等。

本研究将以 LlaMA2 中具有 700 亿参数和 1 万个样本的 70B 预训练模型作为研究对象，验证主流推理框架的推理性能，然后从模型并行化和 Cache 优化两方面加速该大语言模型推理过程，优化推断性能。除此之外，会基于该预训练模型和具体的硬件设备

构建定制的高性能推断引擎。

目前大语言模型大都基于堆叠 transformer 层构建，随着模型深度和容量的增加，使得这种基础模型在推断过程中需要占用大量内存。此时，单 GPU 卡执行将是无法承受的，因此第一个亟待解决的问题就是如何使用多 GPU 卡共同承担模型权重数据。在选择多卡协同时，还需要考虑多张 GPU 卡的协同策略。

生成式 generative 模型的推理过程很有特点，我们给一个输入文本，模型会输出一个回答（长度为 N），其实该过程中执行了 N 次推理过程。即 GPT 类模型一次推理只输出一个 token，输出 token 会与输入 tokens 拼接在一起，然后作为下一次推理的输入，这样不断反复直到遇到终止符。因此在推理过程中将存在大量的冗余计算，需要考虑如何进行 Cache 优化，以便减少这样的冗余。

9.2 解决方法

针对上一节提出的问题，在本章将介绍在本次研究中将采用的推理优化方法，分别是多卡并行，KV Cache 优化和 FlashAttention。其中多卡并行将模型推理过程划分为若干部分在多张加速卡上并行执行；KV Cache 优化能够有效降低推理过程中的显存占用，减少冗余计算；FlashAttention 优化能够将注意力计算进行分块，提高整体读写速度。

9.2.1 Pipeline parallelism 管道并行化

Pipeline parallelism 是使用多个 GPU 进行 LLM 推理，将模型的不同层分配到不同的 GPU 上进行并行计算，从而提高推理速度和效率的方法。模型按顺序划分，所有层的四分之一子集在每个设备上执行。在一个设备上的一组操作的输出被传递给下一个设备，它继续执行后面的块。并在设备 n 上分别表示向前和向后通过。在每个设备上存储模型权重的内存需求有效地分成四等分。因此该种优化方法能够有效缓解模型的大内存需求压力，有利于落地和部署。

9.2.2 KV Cache 优化

KV Cache[333] 的全称是 key-value cache，可以简单理解为对大模型推理过程中的 key-value 缓存的优化。如果不对大模型推理过程中的 key-value 缓存进行优化，它会随着对话的增加而不断增加，也就是所占用的内存会不断动态增加，而且这种增加是不太可控的。不做 KV Cache 会对大规模推理造成几个压力：(1) 频繁的开辟增加内存；(2) 内存很快就不够用了；(3) 增加了很多冗余矩阵计算量。所以进行 KV 缓存优化是大规模训练、推理里很重要的一个环节。KV Cache 采用以空间换时间的思想，复用上次推理的 KV 缓存，可以极大降低内存压力、提高推理性能，而且不会影响任何计算精度。优化步骤如图 14 所示[334]

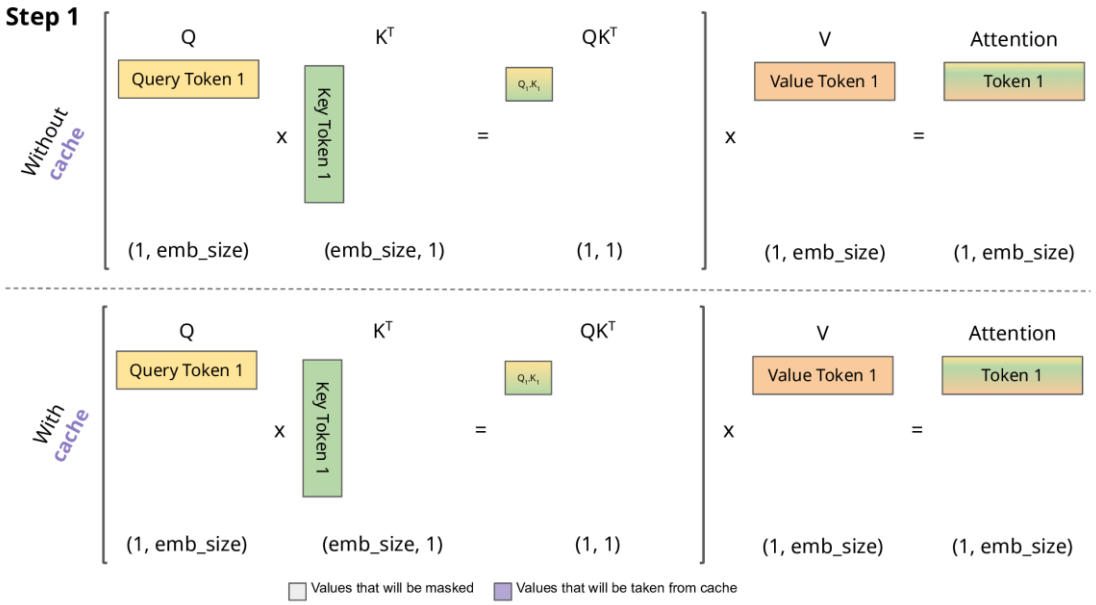

图 14. KV Cache 优化

9.2.3 FlashAttention

FlashAttention[335][336]主要解决 Transformer 计算速度慢和存储占用高的问题。但与绝大多数 Efficient Transformer 把改进方法集中在降低模型的 FLOPS（Floating Point Operations Per Second）不同，FlashAttention 将优化重点放在了降低存储访问开销（Memory Access Cost，MAC）上。

FlashAttention 如何实现在不访问整个输入的情况计算 softmax 大的缩减，标准 Attention 算法由于要计算 softmax，而 softmax 都是按行来计算的，即在和 V 做矩阵乘之前，需要让 Q、K 的各个分块完成整一行分块的计算得到 Softmax 的结果后，再和矩阵 V 分块做矩阵乘。而在 Flash Attention 中，将输入分割成块，并在输入块上进行多次传递，从而以增量方式执行 softmax 缩减。

在后向传播中不存储中间注意力矩阵，以 Flash Attention 所提供的算法为例，通过对比标准 Attention 算法在实现过程中，标准 Attention 算法的实现需要将计算过程中的 S、P 写入到 HBM 中，而这些中间矩阵的大小与输入的序列长度有关且为二次型，因此 Flash Attention 就提出了不使用中间注意力矩阵，通过存储归一化因子来减少 HBM 内存的消耗。

对比标准的 Attention 机制，Flash Attention 虽然由于反向传播需要重新计算导致 GFLOPs 增加，但是 FlashAttention 对 HBM 的 I/O 和运行时间都有了显著的提高。

9.3 实验过程及结果

本研究首先基线代码的多卡并行开始，对比使用主流推理框架 VLLM[337]前后推理性能的变化，验证 VLLM 对推理性能的提升；然后从基线代码开始构建定制的高性能推断引擎，并对比与仅多卡并行时推理性能的提升情况。

9.3.1 验证主流推理框架 VLLM 的推理性能

由于 LlaMA2 的 70B 模型容量较大需要大量 GPU 卡，因此本研究从更小的 LlaMA2 的 13B 模型开始做测试。16 精度时 LlaMA2 13B 模型要求最低显存为 26GB，选用 2 张显存大小为 24GB 的 RTX4090 加速卡进行推理；16 精度时 LlaMA2 70B 模型要求最低显存为 140GB，选用 2 张显存大小为 80GB 的 A800 加速卡进行推理。

LlaMA2 13B 模型在基线代码仅做多卡并行时需要在基线代码中进行修改，在 AutoModelForCausalLM.from_pretrained 方法中添加 device_map='auto'字段即可实现管道并行。日志写入 baseline_parallel_13B.log 文件中，部分内容如下：

```
Namespace(dataset='/root/autodl-tmp/LLM_inference/scrambled_sampled_dataset.jso

Loading checkpoint shards:   0%|          | 0/3 [00:00<?, ?it/s]
Loading checkpoint shards:  33%|███       | 1/3 [00:02<00:04,  2.06s/it]
Loading checkpoint shards:  67%|██████    | 2/3 [00:04<00:01,  1.99s/it]
Loading checkpoint shards: 100%|█████████| 3/3 [00:05<00:00,  1.61s/it]
Loading checkpoint shards: 100%|█████████| 3/3 [00:05<00:00,  1.72s/it]

  0%|          | 0/5 [00:00<?, ?it/s]
 20%|██        | 1/5 [00:00<00:02,  1.43it/s]
 40%|████      | 2/5 [00:01<00:02,  1.10it/s]
 60%|██████    | 3/5 [00:18<00:16,  8.03s/it]
 80%|████████  | 4/5 [00:26<00:08,  8.10s/it]
100%|██████████| 5/5 [00:26<00:00,  5.22s/it]
100%|██████████| 5/5 [00:26<00:00,  5.32s/it]
Throughput: 0.19 requests/s
Tokens/s: 93.33 tokens/s
Prompt_num_tokens:1706.00 tokens
Total_num_tokens:2482.00 tokens
```

图 15. GPU 并行推理输出_13B

LlaMA2 13B 模型使用 VLLM 框架时需要先导入 vllm 库，然后通过 vllm.LLM 方法构造模型，在该方法中通过 tensor_parallel_size=numgpus 字段就能实现多卡执行。日志写入 baseline_vllm_13B.log 文件中，部分内容如下：

```
Processed prompts: 100%|██████████| 1/1 [00:00<00:00,  1.63it/s]←[A
Processed prompts: 100%|██████████| 1/1 [00:00<00:00,  1.63it/s]

 40%|████      | 2/5 [00:00<00:01,  2.21it/s]

Processed prompts:   0%|          | 0/1 [00:00<?, ?it/s]←[A

Processed prompts: 100%|██████████| 1/1 [00:02<00:00,  2.06s/it]←[A
Processed prompts: 100%|██████████| 1/1 [00:02<00:00,  2.06s/it]

 60%|██████    | 3/5 [00:02<00:02,  1.19s/it]

Processed prompts:   0%|          | 0/1 [00:00<?, ?it/s]←[A

Processed prompts: 100%|██████████| 1/1 [00:04<00:00,  4.68s/it]←[A
Processed prompts: 100%|██████████| 1/1 [00:04<00:00,  4.68s/it]

 80%|████████  | 4/5 [00:07<00:02,  2.57s/it]

Processed prompts:   0%|          | 0/1 [00:00<?, ?it/s]←[A
Processed prompts: 100%|██████████| 1/1 [00:00<00:00, 12.19it/s]

100%|██████████| 5/5 [00:07<00:00,  1.53s/it]
Throughput: 0.65 requests/s
Tokens/s: 274.13 tokens/s
Prompt_num_tokens:1706.00 tokens
Total_num_tokens:2100.00 tokens
```

图 16. VLLM 并行推理输出_13B

两次推理过程 GPU 使用率和显存使用大小如下：

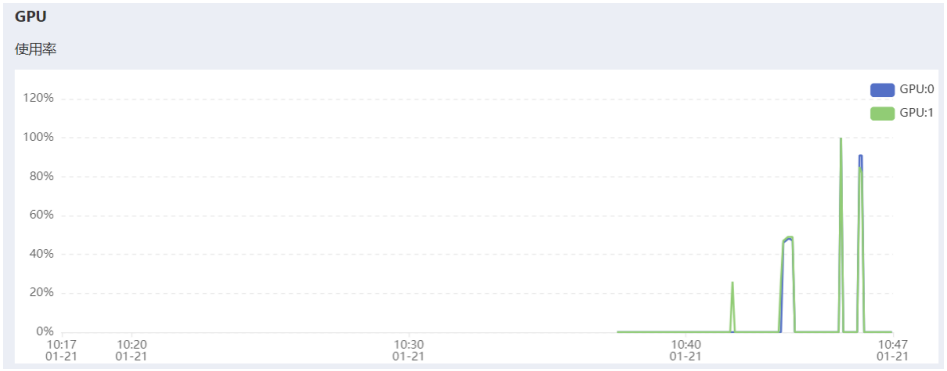

图 17. 两次推理 GPU 使用率，从左到右依次是仅并行推理和 VLLM 框架推理

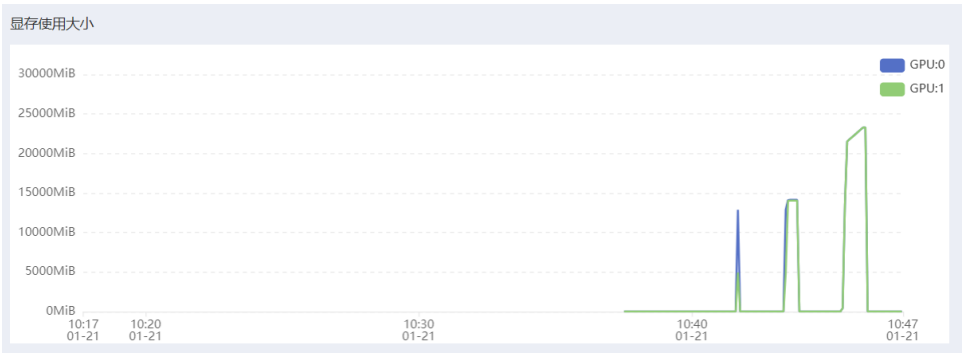

图 18. 两次 GPU 使用率，从左到右依次是仅并行推理和 VLLM 框架推理

对比两次推理过程可以发现在使用 VLLM 框架后，模型的吞吐量 Throughput 能够提升 3 倍之多，同时通过监控 GPU 使用和显存占用情况也能够验证使用 VLLM 框架确实能够提高 GPU 的利用率和显存的利用率。

更换设备为 2 张显存为 80G 的 A800 加速卡，LlaMA2 70B 模型在基线代码仅做多卡并行。日志写入 baseline_parallel_70B.log 文件中，部分内容如下：

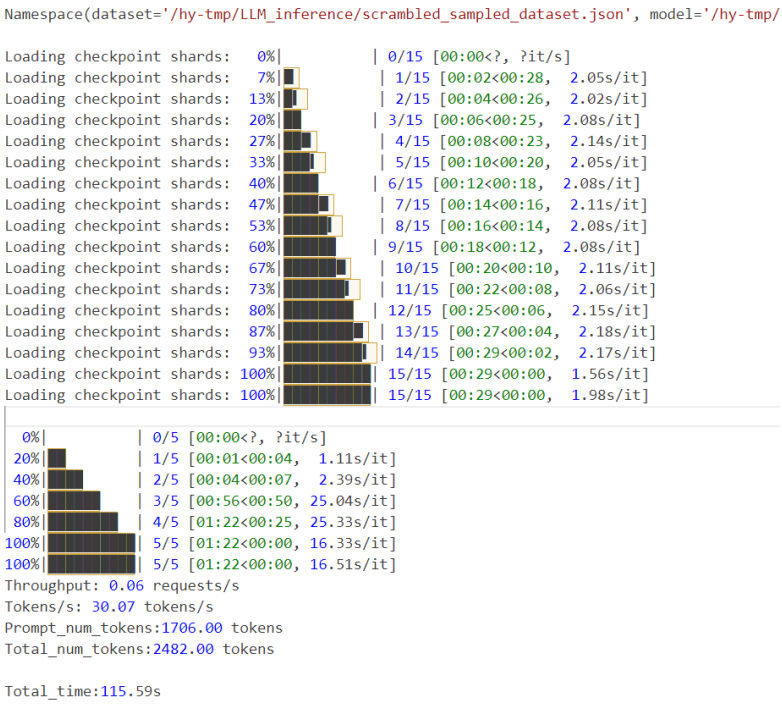

图 19. GPU 并行推理输出_70B

LlaMA2 70B 模型使用 VLLM 框架。日志写入 baseline_vllm_70B.log 文件中，部分内容如下：

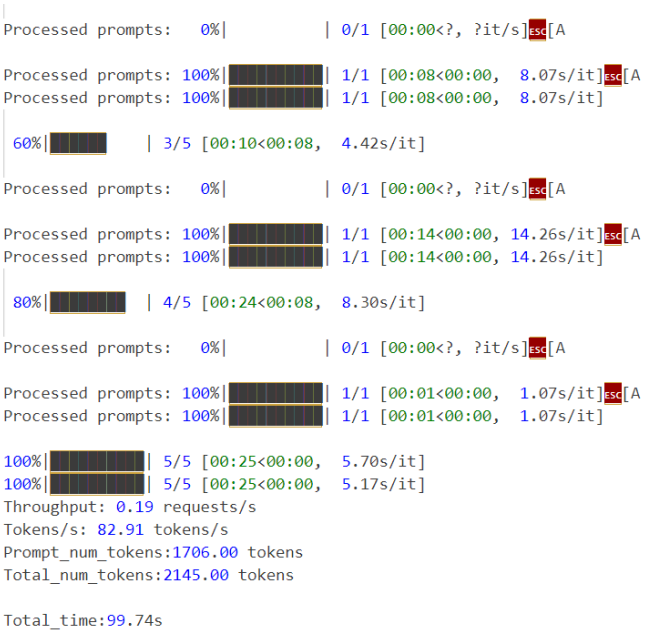

图 20. VLLM 并行推理输出_70B

两次推理过程 GPU 使用率和如下：

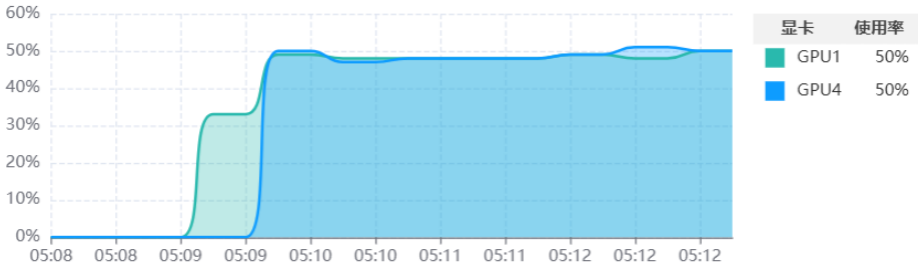

图 21. 仅并行推理时 GPU 使用率

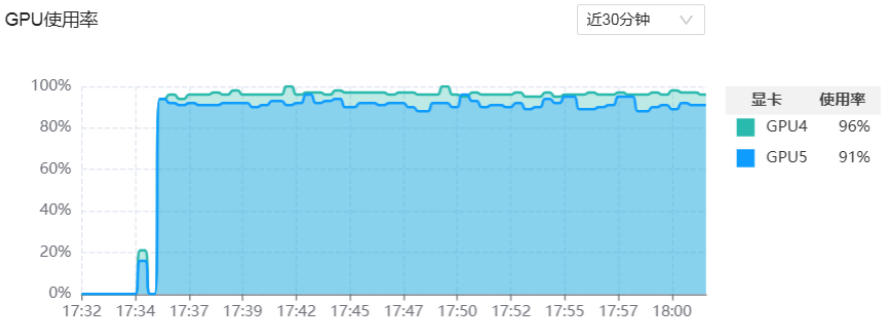

图 22. VLLM 推理时 GPU 使用率

根据日志及设备监控结果可以验证，VLLM 框架提高了模型吞吐量 3 倍之多，同时显著提高了 GPU 的利用率。

9.3.2 构建定制的高性能推断引擎

优化第一步：在基线代码中加入管道并行化。在 AutoModelForCausalLM.from_pretrained 方法中添加字段 device_map='auto'即可实现管道并行化。日志写入 baseline_pipline.log，部分内容如下：

```
Namespace(dataset='/hy-tmp/LLM_inference/scrambled_sampled_dataset.json', model='/hy-tmp/Ll

Loading checkpoint shards:   0%|          | 0/15 [00:00<?, ?it/s]
Loading checkpoint shards:   7%|▋         | 1/15 [00:01<00:25,  1.84s/it]
Loading checkpoint shards:  13%|█▎        | 2/15 [00:03<00:24,  1.87s/it]
Loading checkpoint shards:  20%|██        | 3/15 [00:05<00:22,  1.86s/it]
Loading checkpoint shards:  27%|██▋       | 4/15 [00:07<00:20,  1.88s/it]
Loading checkpoint shards:  33%|███▎      | 5/15 [00:09<00:18,  1.86s/it]
Loading checkpoint shards:  40%|████      | 6/15 [00:11<00:16,  1.84s/it]
Loading checkpoint shards:  47%|████▋     | 7/15 [00:12<00:14,  1.84s/it]
Loading checkpoint shards:  53%|█████▎    | 8/15 [00:14<00:13,  1.88s/it]
Loading checkpoint shards:  60%|██████    | 9/15 [00:16<00:11,  1.83s/it]
Loading checkpoint shards:  67%|██████▋   | 10/15 [00:18<00:09,  1.80s/it]
Loading checkpoint shards:  73%|███████▎  | 11/15 [00:20<00:07,  1.79s/it]
Loading checkpoint shards:  80%|████████  | 12/15 [00:22<00:05,  1.82s/it]
Loading checkpoint shards:  87%|████████▋ | 13/15 [00:23<00:03,  1.79s/it]
Loading checkpoint shards:  93%|█████████▎| 14/15 [00:25<00:01,  1.79s/it]
Loading checkpoint shards: 100%|██████████| 15/15 [00:25<00:00,  1.29s/it]
Loading checkpoint shards: 100%|██████████| 15/15 [00:25<00:00,  1.71s/it]

  0%|          | 0/5 [00:00<?, ?it/s]
 20%|██        | 1/5 [00:00<00:03,  1.14it/s]
 40%|████      | 2/5 [00:18<00:32, 10.77s/it]
 60%|██████    | 3/5 [02:24<02:07, 63.50s/it]
 80%|████████  | 4/5 [03:06<00:54, 54.78s/it]
100%|██████████| 5/5 [03:06<00:00, 35.23s/it]
100%|██████████| 5/5 [03:06<00:00, 37.36s/it]
Throughput: 0.03 requests/s
Tokens/s: 13.29 tokens/s
Prompt_num_tokens:1706.00 tokens
Total_num_tokens:2482.00 tokens
```

图 23. 管道并行推理时 GPU 使用率_70B

推理过程 GPU 使用率如下，可以看到在仅实现并行而没有 KV Cache 优化情况下，

GPU 使用率维持在 50%左右。

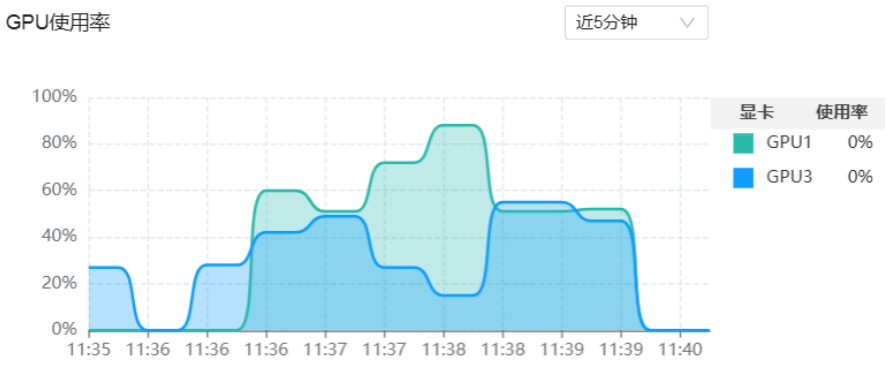

图 24. 单并行无 KV Cache_70B GPU 利用率

优化第二步：在基线代码中加入 KVCache 优化。在 llm.generate 方法中加入字段 use_cache=True 即可开启 KV Cache 优化。日志写入 baseline_pipline+kvcache.log 文件中，部分内容如下：

```
Namespace(dataset='/hy-tmp/LLM_inference/scrambled_sampled_dataset.json', model='/hy-tm

Loading checkpoint shards:   0%|          | 0/15 [00:00<?, ?it/s]
Loading checkpoint shards:   7%|▋         | 1/15 [00:01<00:26,  1.92s/it]
Loading checkpoint shards:  13%|█▎        | 2/15 [00:04<00:28,  2.18s/it]
Loading checkpoint shards:  20%|██        | 3/15 [00:06<00:25,  2.13s/it]
Loading checkpoint shards:  27%|██▋       | 4/15 [00:08<00:22,  2.07s/it]
Loading checkpoint shards:  33%|███▎      | 5/15 [00:10<00:19,  1.99s/it]
Loading checkpoint shards:  40%|████      | 6/15 [00:12<00:17,  1.94s/it]
Loading checkpoint shards:  47%|████▋     | 7/15 [00:13<00:15,  1.90s/it]
Loading checkpoint shards:  53%|█████▎    | 8/15 [00:15<00:13,  1.90s/it]
Loading checkpoint shards:  60%|██████    | 9/15 [00:17<00:11,  1.86s/it]
Loading checkpoint shards:  67%|██████▋   | 10/15 [00:19<00:09,  1.85s/it]
Loading checkpoint shards:  73%|███████▎  | 11/15 [00:21<00:07,  1.86s/it]
Loading checkpoint shards:  80%|████████  | 12/15 [00:23<00:05,  1.86s/it]
Loading checkpoint shards:  87%|████████▋ | 13/15 [00:24<00:03,  1.84s/it]
Loading checkpoint shards:  93%|█████████▎| 14/15 [00:26<00:01,  1.81s/it]
Loading checkpoint shards: 100%|██████████| 15/15 [00:26<00:00,  1.30s/it]
Loading checkpoint shards: 100%|██████████| 15/15 [00:26<00:00,  1.78s/it]

  0%|          | 0/5 [00:00<?, ?it/s]
 20%|██        | 1/5 [00:00<00:03,  1.05it/s]
 40%|████      | 2/5 [00:04<00:06,  2.33s/it]
 60%|██████    | 3/5 [00:56<00:49, 24.91s/it]
 80%|████████  | 4/5 [01:21<00:25, 25.25s/it]
100%|██████████| 5/5 [01:22<00:00, 16.28s/it]
100%|██████████| 5/5 [01:22<00:00, 16.44s/it]
Throughput: 0.06 requests/s
Tokens/s: 30.20 tokens/s
Prompt_num_tokens:1706.00 tokens
Total_num_tokens:2482.00 tokens
```

图 25. 并行+KV Cache_70B

根据日志文件可以看出模型吞吐量 Throughput 由原来的 0.03 request/s 提升到了 0.06 request/s，提高了两倍。

推理过程 GPU 使用率如下：

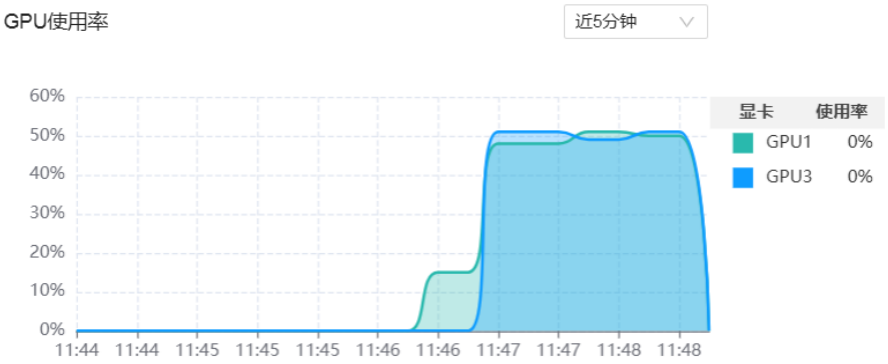

图 26. 并行+KV Cache_70B GPU 利用率

优化第三步：FlashAttention。在 AutoModelForCausalLM.from_pretrained 方法中添加字段 attn_implementation="flash_attention_2"即可实现 FlashAttention 优化，日志写入 baseline_pipline+kvcache+flash.log 文件中，部分内容如下：

```
Namespace(dataset='/hy-tmp/LLM_inference/scrambled_sampled_dataset.json', model='/hy-tmp/Llam

Loading checkpoint shards:   0%|          | 0/15 [00:00<?, ?it/s]
Loading checkpoint shards:   7%|▋         | 1/15 [00:02<00:34,  2.49s/it]
Loading checkpoint shards:  13%|█▎        | 2/15 [00:04<00:28,  2.17s/it]
Loading checkpoint shards:  20%|██        | 3/15 [00:06<00:24,  2.00s/it]
Loading checkpoint shards:  27%|██▋       | 4/15 [00:08<00:21,  1.96s/it]
Loading checkpoint shards:  33%|███▎      | 5/15 [00:09<00:18,  1.86s/it]
Loading checkpoint shards:  40%|████      | 6/15 [00:11<00:16,  1.82s/it]
Loading checkpoint shards:  47%|████▋     | 7/15 [00:13<00:14,  1.79s/it]
Loading checkpoint shards:  53%|█████▎    | 8/15 [00:15<00:12,  1.82s/it]
Loading checkpoint shards:  60%|██████    | 9/15 [00:16<00:10,  1.78s/it]
Loading checkpoint shards:  67%|██████▋   | 10/15 [00:18<00:08,  1.76s/it]
Loading checkpoint shards:  73%|███████▎  | 11/15 [00:20<00:07,  1.76s/it]
Loading checkpoint shards:  80%|████████  | 12/15 [00:22<00:05,  1.79s/it]
Loading checkpoint shards:  87%|████████▋ | 13/15 [00:23<00:03,  1.77s/it]
Loading checkpoint shards:  93%|█████████▎| 14/15 [00:25<00:01,  1.74s/it]
Loading checkpoint shards: 100%|██████████| 15/15 [00:25<00:00,  1.26s/it]
Loading checkpoint shards: 100%|██████████| 15/15 [00:25<00:00,  1.72s/it]

  0%|          | 0/5 [00:00<?, ?it/s]
 20%|██        | 1/5 [00:01<00:04,  1.05s/it]
 40%|████      | 2/5 [00:04<00:07,  2.38s/it]
 60%|██████    | 3/5 [00:56<00:50, 25.03s/it]
 80%|████████  | 4/5 [01:22<00:25, 25.37s/it]
100%|██████████| 5/5 [01:22<00:00, 16.36s/it]
100%|██████████| 5/5 [01:22<00:00, 16.52s/it]
Throughput: 0.06 requests/s
Tokens/s: 30.04 tokens/s
Prompt_num_tokens:1706.00 tokens
Total_num_tokens:2482.00 tokens
```

图 27. 并行+KV Cache+FlashAttention_70B

推理过程 GPU 使用率如下：

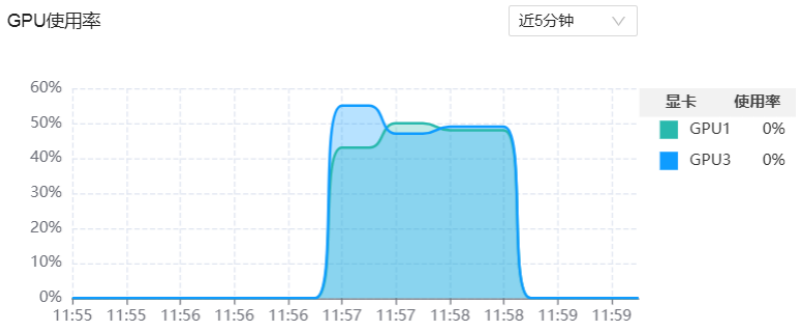

图 28. 并行+KV Cache+FlashAttention_70B GPU 利用率

经过以上三部的推理优化过程，实现了多卡推理，同时将模型吞吐量提高到了 0.06 request/s，优化效果较为明显。

10 总结

在本文中，我们对大型语言模型（LLM）进行了全面总结和讨论。我们从 LLM 的背景和发展历程出发，介绍了自然语言处理（NLP）领域从统计语言模型向神经语言模型、预训练语言模型（PLMs）和最终的大型语言模型的演变过程。我们详细介绍了当前流行的 LLMs 架构，包括编码器-解码器、因果解码器、前缀解码器以及 Transformer 架构，并充分阐述了 LLMs 的核心注意力机制。特别强调了 LLM 的主流训练范式：“预训练-微调”，并介绍了训练过程中的并行模式、ZeRO 优化内存使用、混合精度训练以及指令微调、对齐微调等微调方法和目标。

在模型使用方面，结合当前的研究和实践，我们介绍了上下文学习和思维链提示两种增强 LLM 能力的方法。在评估 LLM 能力方面，我们讨论了级能力评估和公共基准以及实证分析的各种方法的利与弊，例如基于基准的评估可能受到测试设置的影响，基于人的评估虽然直接反映实际能力但成本高昂，而基于模型的评估可以减少对人的依赖。此外，我们还关注了当前 LLM 面临的主要挑战，包括 AI 价值对齐、LLM 数据集、成本问题以及 LLM 幻觉现象，这些都是学界目前主要攻克的难题。最后，我们对 LlaMA2 的 70B 预训练模型进行了优化实验，以帮助读者更深入地了解模型并行化和 Cache 优化在模型训练中的作用。

参考文献

[1] A. Chernyavskiy, D. Ilvovsky, and P. Nakov, “Transformers:“the end of history” for natural language processing?” in Machine Learning and Knowledge Discovery in Databases. Research Track: European Conference, ECML PKDD 2021, Bilbao, Spain, September 13–17, 2021, Proceedings, Part III 21. Springer, 2021, pp. 677–693.

[2] A. Wang, Y. Pruksachatkun, N. Nangia, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman, “Superglue: A stickier benchmark for general-purpose language understanding systems,” Advances in neural information processing systems, vol. 32, 2019.

[3] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever et al., “Language models are unsupervised multitask learners,” OpenAI blog, vol. 1, no. 8, p. 9, 2019.

[4] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell et al., “Language mod els are few-shot learners,” Advances in neural information processing systems, vol. 33, pp. 1877–1901, 2020. 1, 2, 7, 8, 9, 10, 12, 18, 22, 31, 32, 33

[5] Zhang, Y., Jin, R. & Zhou, ZH. Understanding bag-of-words model: a statistical framework. Int. J. Mach. Learn. & Cyber. 1, 43–52 (2010). https://doi.org/10.1007/s13042-010-0001-0

[6] Yin, Chen, Wu, Min. Overview of N-gram Model [J]. Computer Systems & Applications, 2018, 27(10): 33-38.

[7] T. Mikolov, M. Karafi´at, L. Burget, J. Cernock´ y, and S. Khudanpur, “Recurrent neural network based lan guage model,” in INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September 26-30, 2010, T. Kobayashi, K. Hirose, and S. Nakamura, Eds. ISCA, 2010, pp. 1045–1048.

[8] S. Kombrink, T. Mikolov, M. Karafi´at, and L. Burget, “Recurrent neural network based language modeling in meeting recognition,” in INTERSPEECH 2011, 12th Annual Conference of the International Speech Commu nication Association, Florence, Italy, August 27-31, 2011. ISCA, 2011, pp. 2877–2880.

[9] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States, C. J. C. Burges, L. Bot tou, Z. Ghahramani, and K. Q. Weinberger, Eds., 2013, pp. 3111–3119.

[10] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Ef f icient estimation of word representations in vector space,” in 1st International Conference on Learning Rep resentations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings, Y. Bengio and Y. LeCun, Eds., 2013.

[11] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, “Deep contex tualized word representations,” in Proceedings of the 2018 Conference of the North American Chapter of the As sociation for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers), M. A. Walker, H. Ji, and A. Stent, Eds. Association for Computational Linguistics, 2018, pp. 2227–2237.

[12] J. Devlin, M. Chang, K. Lee, and K. Toutanova, “BERT: pre-training of deep bidirectional transformers for language understanding,” in Proceedings of the 2019 Conference of the North American Chapter of the Asso ciation for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers), J. Burstein, C. Doran, and T. Solorio, Eds. Association for Computational Linguistics, 2019, pp. 4171–4186.

[13] Zhao W X, Zhou K, Li J, et al. A survey of large language models[J]. arXiv preprint arXiv:2303.18223, 2023.

[14] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, “Exploring the limits of transfer learn ing with a unified text-to-text transformer,” The Journal of Machine Learning Research, vol. 21, no. 1, pp. 5485–5551, 2020. 2, 6, 7, 8, 16, 20, 22, 24, 25, 27, 28, 32

[15] L. Xue, N. Constant, A. Roberts, M. Kale, R. Al-Rfou, A. Siddhant, A. Barua, and C. Raffel, “mt5: A massively multilingual pre-trained text-to-text transformer,” arXiv preprint arXiv:2010.11934, 2020. 2, 7, 8, 22, 25, 27, 32

[16] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma,

D. Zhou, D. Metzler, E. H. Chi, T. Hashimoto, O. Vinyals, P. Liang, J. Dean, and W. Fedus, "Emergent abilities of large language models," CoRR, vol. abs/2206.07682, 2022.

[17] M.Shanahan, "Talking about large language models," CoRR, vol. abs/2212.03551, 2022.

[18] J. Wei, X. Wang, D. Schuurmans, M. Bosma, E. H. Chi, Q. Le, and D. Zhou, "Chain of thought prompting elicits reasoning in large language models," CoRR, vol. abs/2201.11903, 2022.

[19] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler et al., "Emergent abilities of large language models," arXiv preprint arXiv:2206.07682, 2022.

[20] T. Webb, K. J. Holyoak, and H. Lu, "Emergent analogical reasoning in large language models," Nature Human Behaviour, vol. 7, no. 9, pp. 1526–1541, 2023.

[21] G. Izacard, P. Lewis, M. Lomeli, L. Hosseini, F. Petroni, T. Schick, J. Dwivedi-Yu, A. Joulin, S. Riedel, and E. Grave, "Few-shot learning with retrieval augmented language models," arXiv preprint arXiv:2208.03299, 2022.

[22] D. Driess, F. Xia, M. S. Sajjadi, C. Lynch, A. Chowdhery, B. Ichter, A. Wahid, J. Tompson, Q. Vuong, T. Yu et al., "Palm-e: An embodied multimodal language model," arXiv preprint arXiv:2303.03378, 2023.

[23] E. Saravia, "Prompt Engineering Guide," https://github.com/dair ai/Prompt-Engineering-Guide, 12 2022

[24] Y. Wang, H. Le, A. D. Gotmare, N. D. Bui, J. Li, and S. C. Hoi, "Codet5+: Open code large language models for code understanding and generation," arXiv preprint arXiv:2305.07922, 2023. 2, 11, 22, 32

[25] S. Wang, Y. Sun, Y. Xiang, Z. Wu, S. Ding, W. Gong, S. Feng, J. Shang, Y. Zhao, C. Pang et al., "Ernie 3.0 titan: Exploring larger scale knowledge enhanced pre-training for language understanding and generation," arXiv preprint arXiv:2112.12731, 2021

[26] J. J. Webster and C. Kit, "Tokenization as the initial phase in nlp," in COLING 1992 volume 4: The 14th international conference on computational linguistics, 1992.

[27] S. J. Mielke, Z. Alyafeai, E. Salesky, C. Raffel, M. Dey, M. Gallé, A. Raja, C. Si, W. Y. Lee, B. Sagot et al., "Between words and characters: A brief history of open-vocabulary modeling and tokenization in nlp," arXiv preprint arXiv:2112.10508, 2021.

[28] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," in Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2016, pp. 1715–1725.

[29] T. Kudo, "Subword regularization: Improving neural network translation models with multiple subword candidates," in Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2018, pp. 66–75.

[30] M. Schuster and K. Nakajima, "Japanese and korean voice search," in 2012 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, 2012, pp.

5149–5152.

[31] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” Advances in neural information processing systems, vol. 30, 2017.

[32] I. McKenzie, A. Lyzhov, A. Parrish, A. Prabhu, A. Mueller, N. Kim, S. Bowman, and E. Perez, “The inverse scaling prize,” 2022. [Online]. Available: https://github.com/inverse-scaling/prize

[33] B. A. Huberman and T. Hogg, “Phase transitions in artificial intelligence systems,” Artificial Intelligence, vol. 33, no. 2, pp. 155–171, 1987.

[34] J. W. Rae, S. Borgeaud, T. Cai, K. Millican, J. Hoffmann, H. F. Song, J. Aslanides, S. Henderson, R. Ring, S. Young, E. Rutherford, T. Hennigan, J. Menick, A. Cassirer, R. Powell, G. van den Driessche, L. A. Hendricks, M. Rauh, P. Huang, A. Glaese, J. Welbl, S. Dathathri, S. Huang, J. Uesato, J. Mellor, I. Higgins, A. Creswell, N. McAleese, A. Wu, E. Elsen, S. M Jayakumar, E. Buchatskaya, D. Budden, E. Suther land, K. Simonyan, M. Paganini, L. Sifre, L. Martens, X. L. Li, A. Kuncoro, A. Nematzadeh, E. Gribovskaya D. Donato, A. Lazaridou, A. Mensch, J. Lespiau, M. Tsimpoukelli, N. Grigorev, D. Fritz, T. Sottiaux, M. Pajarskas, T. Pohlen, Z. Gong, D. Toyama,C. de Masson d’Autume, Y. Li, T. Terzi, V. Mikulik, Babuschkin, A. Clark, D. de Las Casas, A. Guy, C. Jones, J. Bradbury, M. J. Johnson, B. A. Hechtman, L. Weidinger, I. Gabriel, W. S. Isaac, E. Lockhart, S. Osindero, L. Rimell, C. Dyer, O. Vinyals, K. Ayoub, J. Stanway, L. Bennett, D. Hassabis, K. Kavukcuoglu, and G. Irving, “Scaling language models: Methods, analysis & insights from training gopher,” CoRR, vol.abs/2112.11446, 2021.

[35] S. Rajbhandari, J. Rasley, O. Ruwase, and Y. He, “Zero: Memory optimizations toward training trillion parameter models,” in SC20: International Conference for High Performance Computing, Networking,Storage and Analysis. IEEE, 2020, pp. 1–16.

[36] P. F. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg, and D. Amodei, “Deep reinforcement learn ing from human preferences,” in Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, Eds., 2017, pp. 4299 4307.

[37] T. Henighan, J. Kaplan, M. Katz, M. Chen, C. Hesse, J. Jackson, H. Jun, T. B. Brown, P. Dhariwal, S. Gray et al., “Scaling laws for autoregressive generative modeling,” arXiv preprint arXiv:2010.14701, 2020.

[38] J. Hoffmann, S. Borgeaud, A. Mensch, E. Buchatskaya, T. Cai, E. Rutherford, D. de Las Casas, L. A. Hen dricks, J. Welbl, A. Clark, T. Hennigan, E. Noland, K. Millican, G. van den Driessche, B. Damoc, A. Guy, S. Osindero, K. Simonyan, E. Elsen, J. W. Rae, O. Vinyals, and L. Sifre, “Training compute-optimal large language models,” vol. abs/2203.15556, 2022]

[39] S. M. Xie, H. Pham, X. Dong, N. Du, H. Liu, Y. Lu, P. Liang, Q. V. Le, T. Ma, and A. W. Yu, “Doremi: Optimizing data mixtures speeds up language model pretraining,” arXiv

preprint arXiv:2305.10429, 2023.

[40] Shen T, Jin R, Huang Y, et al. Large language model alignment: A survey[J]. arXiv preprint arXiv:2309.15025, 2023.

[41] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. 2021. Extracting training data from large language models. In 30th USENIX Security Symposium (USENIX Security 21), pages 2633–2650.

[42] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. ACM Computing Surveys, 55(12):1–38.

[43] T. Schick, J. Dwivedi-Yu, R. Dess` ı, R. Raileanu, M. Lomeli, L. Zettlemoyer, N. Cancedda, and T. Scialom, "Toolformer: Language models can teach themselves to use tools," CoRR, vol. abs/2302.04761, 2023.

[44] R. Nakano, J. Hilton, S. Balaji, J. Wu, L. Ouyang, C. Kim, C. Hesse, S. Jain, V. Kosaraju, W. Saun ders, X. Jiang, K. Cobbe, T. Eloundou, G. Krueger, K. Button, M. Knight, B. Chess, and J. Schulman, "Webgpt: Browser-assisted question-answering with human feedback," CoRR, vol. abs/2112.09332, 2021.

[45] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. Advances in neural information processing systems 30 (2017).

[46] Alexandra Chronopoulou, Matthew E Peters, Alexander Fraser, and Jesse Dodge. 2023. AdapterSoup: Weight Averaging to Improve Generalization of Pretrained Language Models.

[47] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models.

[48] Hejie Cui, Jiaying Lu, Shiyu Wang, Ran Xu, Wenjing Ma, Shaojun Yu, Yue Yu, Xuan Kan, Chen Ling, Joyce Ho, et al. 2023. A Survey on Knowledge Graphs for Healthcare: Resources, Applications, and Promises.

[49] Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2021. Knowledge neurons in pretrained transformers.

[50] Zhuyun Dai, Vincent Y Zhao, Ji Ma, Yi Luan, Jianmo Ni, Jing Lu, Anton Bakalov, Kelvin Guu, Keith B Hall, and Ming-Wei Chang. 2022. Promptagator: Few-shot dense retrieval from 8 examples.

[51] Ishita Dasgupta, Christine Kaeser-Chen, Kenneth Marino, Arun Ahuja, Sheila Babayan, Felix Hill, and Rob Fergus. 2023. Collaborating with language models for embodied reasoning. In Second Workshop on Language and Reinforcement Learning.

[52] Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. Editing Factual Knowledge in Language Models. In Proceedings of the 2021 Conference on Empirical Methods in Natural

Language Processing.

[53] Mingkai Deng, Jianyu Wang, Cheng-Ping Hsieh, Yihan Wang, Han Guo, Tianmin Shu, Meng Song, Eric P. Xing, and Zhiting Hu. 2022. RLPrompt: Optimizing Discrete Text Prompts with Reinforcement Learning. In Conference on Empirical Methods in Natural Language Processing.

[54] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.

[55] Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. 2022. Delta tuning: A comprehensive study of parameter efficient methods for pre-trained language models.

[56] Andrew Drozdov, Nathanael Schärli, Ekin Akyürek, Nathan Scales, Xinying Song, Xinyun Chen, Olivier Bousquet, and Denny Zhou. 2023.Compositional Semantic Parsing with Large Language Models. In The Eleventh International Conference on Learning Representations

[57] Dheeru Dua, Shivanshu Gupta, Sameer Singh, and Matt Gardner. 2022. Successive Prompting for Decomposing Complex Questions. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing.

[58] Ali Edalati, Marzieh Tahaei, Ivan Kobyzev, Vahid Partovi Nia, James J Clark, and Mehdi Rezagholizadeh. 2022. KronA: Parameter Efficient Tuning with Kronecker Adapter.

[59] Mohammad Fahes, Tuan-Hung Vu, Andrei Bursuc, Patrick Pérez, and Raoul de Charette. 2022. P{\O}DA: Prompt-driven Zero-shot Domain Adaptation.

[60] Yang Feng, Shiyue Zhang, Andi Zhang, Dong Wang, and Andrew Abel. 2017. Memory-augmented Neural Machine Translation. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing.

[61] Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2022. PAL: Program-aided Language Models.

[62] Chunjiang Ge, Rui Huang, Mixue Xie, Zihang Lai, Shiji Song, Shuang Li, and Gao Huang. 2022. Domain Adaptation via Prompt Learning.

[63] Edouard Grave, Armand Joulin, and Nicolas Usunier. 2017. Improving neural language models with a continuous cache. In International Conference on Learning Representations.

[64] Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2022. PPT: Pre-trained Prompt Tuning for Few-shot Learning. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, Dublin, Ireland, 8410–8423.

[65] Xu Guo, Boyang Li, and Han Yu. 2022. Improving the Sample Efficiency of Prompt Tuning with Domain Adaptation. In Findings of the Association for Computational Linguistics: EMNLP 2022. Association for Computational Linguistics, Abu Dhabi, United Arab Emirates.

[66] Xu Guo and Han Yu. 2022. On the Domain Adaptation and Generalization of Pretrained Language Models: A Survey. arXiv (2022).

[67] Karen Hambardzumyan, Hrant Khachatrian, and Jonathan May. 2021. WARP: Word-level Adversarial ReProgramming. In Annual Meeting of theAssociation for Computational Linguistics.

[68] Hangfeng He, Hongming Zhang, and Dan Roth. 2022. Rethinking with Retrieval: Faithful Large Language Model Inference.

[69] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2021. Towards a unified view of parameter-efficient transfer learning.

[70] Shwai He, Liang Ding, Daize Dong, Miao Zhang, and Dacheng Tao. 2022. Sparseadapter: An easy approach for improving the parameter-efficiency of adapters.

[71] Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). arXiv preprint arXiv:1606.08415 (2016).

[72] Evan Hernandez, Belinda Z Li, and Jacob Andreas. 2023. Measuring and Manipulating Knowledge Representations in Language Models.

[73] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In International Conference on Machine Learning. PMLR, 2790–2799.

[74] Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification.

[75] Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. 2023. Distilling Step-by-Step! Outperforming Larger Language Models with Less Training Data and Smaller Model Sizes.

[76] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, 2017, pp. 5998–6008.

[77] HARDISTY, M. Sex Composition of Lamprey Populations. *Nature* **191**, 1116–1117 (1961). https://doi.org/10.1038/1911116a0

[78] Mundahl, N., Sagan, R. Spawning ecology of the American brook lamprey, *Lampetra appendix*. *Environ Biol Fish* **73**, 283–292 (2005). https://doi.org/10.1007/s10641-005-2145-4

[79] HOUSTON K A, KELSO JR M. Relation of Sea Lamprey Size and Sex Ratio toSalmonid Availability in Three Great Lakes[J/OL]. Journal of Great LakesResearch,1991:270-280.http://dx.doi.org/10.1016/s0380-1330(91)71363-4.DO1:10.1016/s0380-1330(91)71363-4.

[80] Johnson NS, Swink WD, Brenden TO. Field study suggests that sex determination in sea lamprey is directly influenced by larval growth rate. Proc Biol Sci. 2017 Mar 29;284(1851):20170262. doi: 10.1098/rspb.2017.0262. PMID: 28356456; PMCID: PMC5378093.

[81] Li, Wenjie et al. "Sex ratio in the mother's environment affects offspring population dynamics: maternal effects on population regulation." *Proceedings of the Royal Society B* 289 (2022): n. pag.

[82] Katharina Jeblick, Balthasar Schachtner, Jakob Dexl, Andreas Mittermeier, Anna Theresa Stüber, Johanna Topalis, Tobias Weber, Philipp Wesp, Bastian Sabel, Jens Ricke, et al. 2022. ChatGPT Makes Medicine Easy to Swallow: An Exploratory Case Study on Simplified Radiology Reports.

[83] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. Comput. Surveys 55, 12 (2023), 1–38.

[84] Chen Jia and Yue Zhang. 2022. Prompt-based Distribution Alignment for Domain Generalization in Text Classification. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing. 10147–10157.

[85] Albert Qiaochu Jiang, Sean Welleck, Jin Peng Zhou, Timothee Lacroix, Jiacheng Liu, Wenda Li, Mateja Jamnik, Guillaume Lample, and Yuhuai Wu. 2023. Draft, Sketch, and Prove: Guiding Formal Theorem Provers with Informal Proofs. In The Eleventh International Conference on Learning Representations.

[86] Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. 2019. Smart: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization.

[87] Feihu Jin, Jinliang Lu, Jiajun Zhang, and Chengqing Zong. 2022. Instance-aware prompt learning for language understanding and generation.

[88] Qiao Jin, Yifan Yang, Qingyu Chen, and Zhiyong Lu. [n. d.]. GeneGPT: Augmenting Large Language Models with Domain Tools for Improved Access to Biomedical Information.

[89] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models.

[90] Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. 2021. Compacter: Efficient low-rank hypercomplex adapter layers. Advances in Neural Information Processing Systems 34 (2021), 1022–1035.

[91] Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2022. Decomposed prompting: A modular approach for solving complex tasks.

[92] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large Language Models are Zero-Shot Reasoners. In ICML 2022 Workshop on

Knowledge Retrieval and Language Models.

[93] Mojtaba Komeili, Kurt Shuster, and Jason Weston. 2022. Internet-Augmented Dialogue Generation. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 8460–8478.

[94] Eric Lehman, Evan Hernandez, Diwakar Mahajan, Jonas Wulff, Micah J Smith, Zachary Ziegler, Daniel Nadler, Peter Szolovits, Alistair Johnson, and Emily Alsentzer. 2023. Do We Still Need Clinical Language Models?

[95] Markus Leippold. 2023. Sentiment Spin: Attacking Financial Sentiment with GPT-3. Available at SSRN (2023).

[96] Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The Power of Scale for Parameter-Efficient Prompt Tuning. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 3045–3059.

[97] Yoav Levine, Itay Dalmedigos, Ori Ram, Yoel Zeldes, Daniel Jannai, Dor Muhlgay, Yoni Osin, Opher Lieber, Barak Lenz, Shai Shalev-Shwartz, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2022. Standing on the Shoulders of Giant Frozen Language Models.

[98] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. Advances in Neural Information Processing Systems 33 (2020), 9459–9474.

[99] Daliang Li, Ankit Singh Rawat, Manzil Zaheer, Xin Wang, Michal Lukasik, Andreas Veit, Felix Yu, and Sanjiv Kumar. 2022. Large Language Models with Controllable Working Memory.

[100] Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. CAMEL: Communicative Agents for"Mind" Exploration of Large Scale Language Model Society.

[101] Haochen Li, Tong Mo, Hongcheng Fan, Jingkun Wang, Jiaxi Wang, Fuhao Zhang, and Weiping Li. 2022. KiPT: Knowledge-injected Prompt Tuning for Event Detection. In International Conference on Computational Linguistics.

[102] Jinyang Li, Binyuan Hui, Ge Qu, Binhua Li, Jiaxi Yang, Bowen Li, Bailin Wang, Bowen Qin, Rongyu Cao, Ruiying Geng, et al. 2023. Can LLM Already Serve as A Database Interface? A BIg Bench for Large-Scale Database Grounded Text-to-SQLs.

[103] Xiang Lisa Li and Percy Liang. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)abs/2101.00190 (2021).

[104] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Pete Florence, Andy Zeng, et al. 2022. Code as Policies: Language Model Programs for Embodied Control. In Workshop on Language and Robotics at CoRL 2022.

[105] Yaobo Liang, Chenfei Wu, Ting Song, Wenshan Wu, Yan Xia, Yu Liu, Yang Ou, Shuai Lu, Lei Ji, Shaoguang Mao, et al. 2023. TaskMatrix. AI: Completing Tasks by Connecting Foundation Models with Millions of APIs.

[106] Hongzhan Lin, Pengyao Yi, Jing Ma, Haiyun Jiang, Ziyang Luo, Shuming Shi, and Ruifang Liu. 2022. Zero-Shot Rumor Detection with Propagation Structure via Prompt Learning.

[107] Qi Liu, Dani Yogatama, and Phil Blunsom. 2022. Relational Memory-Augmented Language Models. Transactions of the Association for Computational Linguistics 10 (2022), 555–572.

[108] Ruibo Liu, Jason Wei, Shixiang Shane Gu, Te-Yen Wu, Soroush Vosoughi, Claire Cui, Denny Zhou, and Andrew M Dai. 2022. Mind's Eye: Grounded Language Model Reasoning through Simulation.

[109] Xiangyang Liu, Tianxiang Sun, Xuanjing Huang, and Xipeng Qiu. 2022. Late Prompt Tuning: A Late Prompt Could Be Better Than Many Prompts. In Findings of the Association for Computational Linguistics: EMNLP 2022. Association for Computational Linguistics, Abu Dhabi, United Arab Emirates.

[110] Yiheng Liu, Tianle Han, Siyuan Ma, Jiayue Zhang, Yuanyuan Yang, Jiaming Tian, Hao He, Antong Li, Mengshen He, Zhengliang Liu, et al. 2023. Summary of chatgpt/gpt-4 research and perspective towards the future of large language models.

[111] Alejandro Lopez-Lira and Yuehua Tang. 2023. Can ChatGPT Forecast Stock Price Movements? Return Predictability and Large Language Models.

[112] Jiaying Lu, Jiaming Shen, Bo Xiong, Wengjing Ma, Staab Steffen, and Carl Yang. 2023. HiPrompt: Few-Shot Biomedical Knowledge Fusion via Hierarchy-Oriented Prompting. In 46th International ACM SIGIR Conference on Research and Development in Information Retrieval - Short Paper.

[113] Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically Ordered Prompts and Where to Find Them: Overcoming Few-Shot Prompt Order Sensitivity. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume1: Long Papers). 8086–8098.

[114] Aman Madaan, Shuyan Zhou, Uri Alon, Yiming Yang, and Graham Neubig. 2022. Language models of code are few-shot commonsense learners.

[115] Zhang S, Dong L, Li X, et al. Instruction tuning for large language models: A survey[J]. arXiv preprint arXiv:2308.10792, 2023.

[116] Longpre S, Hou L, Vu T, et al. The flan collection: Designing data and methods for effective instruction tuning[J]. arXiv preprint arXiv:2301.13688, 2023.

[117] Sanh V, Webson A, Raffel C, et al. Multitask prompted training enables zero-shot task generalization[J]. arXiv preprint arXiv:2110.08207, 2021.

[118] Xu C, Guo D, Duan N, et al. Baize: An open-source chat model with parameter-efficient tuning on self-chat data[J]. arXiv preprint arXiv:2304.01196, 2023.

[119] Weidinger L, Mellor J, Rauh M, et al. Ethical and social risks of harm from language models[J]. arXiv preprint arXiv:2112.04359, 2021.

[120] Hendrycks D, Mazeika M, Woodside T. An Overview of Catastrophic AI Risks[J]. arXiv preprint arXiv:2306.12001, 2023.

[121] Shen T, Jin R, Huang Y, et al. Large language model alignment: A survey[J]. arXiv preprint arXiv:2309.15025, 2023.

[122] Askell A, Bai Y, Chen A, et al. A general language assistant as a laboratory for alignment[J]. arXiv preprint arXiv:2112.00861, 2021.

[123] Amodei D, Olah C, Steinhardt J, et al. Concrete problems in AI safety[J]. arXiv preprint arXiv:1606.06565, 2016.

[124] Anthropic P B C. Core Views on AI Safety: When, Why, What, and How[J]. 2023.

[125] Ziegler D M, Stiennon N, Wu J, et al. Fine-tuning language models from human preferences[J]. arXiv preprint arXiv:1909.08593, 2019.

[126] Stiennon N, Ouyang L, Wu J, et al. Learning to summarize with human feedback[J]. Advances in Neural Information Processing Systems, 2020, 33: 3008-3021.

[127] Ouyang L, Wu J, Jiang X, et al. Training language models to follow instructions with human feedback[J]. Advances in Neural Information Processing Systems, 2022, 35: 27730-27744.

[128] Russell S. Learning agents for uncertain environments[C]//Proceedings of the eleventh annual conference on Computational learning theory. 1998: 101-103.

[129] Ng A Y, Russell S. Algorithms for inverse reinforcement learning[C]//Icml. 2000, 1: 2.

[130] Liu G K M. Transforming Human Interactions with AI via Reinforcement Learning with Human Feedback (RLHF)[J]. 2023.

[131] Schulman J, Wolski F, Dhariwal P, et al. Proximal policy optimization algorithms[J]. arXiv preprint arXiv:1707.06347, 2017.

[132] Casper S, Davies X, Shi C, et al. Open problems and fundamental limitations of reinforcement learning from human feedback[J]. arXiv preprint arXiv:2307.15217, 2023.

[133] Liu H, Sferrazza C, Abbeel P. Languages are rewards: Hindsight finetuning using human feedback[J]. arXiv preprint arXiv:2302.02676, 2023.

[134] Hubinger E, van Merwijk C, Mikulik V, et al. Risks from learned optimization in advanced machine learning systems[J]. arXiv preprint arXiv:1906.01820, 2019.

[135] Vilone G, Longo L. Explainable artificial intelligence: a systematic review[J]. arXiv preprint arXiv:2006.00093, 2020.

[136] Nanda N. A Comprehensive Mechanistic Interpretability Explainer & Glossary[J]. 2022.

[137] Lipton Z C. The mythos of model interpretability: In machine learning, the concept of

interpretability is both important and slippery[J]. Queue, 2018, 16(3): 31-57.

[138] Critch A, Krueger D. AI research considerations for human existential safety (ARCHES)[J]. arXiv preprint arXiv:2006.04948, 2020.

[139] McAllister R T, Gal Y, Kendall A, et al. Concrete problems for autonomous vehicle safety: Advantages of Bayesian deep learning[C]. International Joint Conferences on Artificial Intelligence, Inc., 2017.

[140] Hubinger E. A transparency and interpretability tech tree[C]//Alignment Forum. 2022.

[141] Houlsby N, Giurgiu A, Jastrzebski S, et al. Parameter-efficient transfer learning for NLP[C]//International Conference on Machine Learning. PMLR, 2019: 2790-2799.

[142] Devlin J, Chang M W, Lee K, et al. Bert: Pre-training of deep bidirectional transformers for language understanding[J]. arXiv preprint arXiv:1810.04805, 2018.

[143] Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, and Sayak Paul. 2022. Peft: State-of-the-art parameter-efficient fine-tuning method

[144] Wang Y, Mukherjee S, Liu X, et al. Adamix: Mixture-of-adapter for parameter-efficient tuning of large language models[J]. arXiv preprint arXiv:2205.12410, 2022, 1(2): 4.

[145] Hu E J, Shen Y, Wallis P, et al. Lora: Low-rank adaptation of large language models[J]. arXiv preprint arXiv:2106.09685, 2021.

[146] Li C, Farkhoor H, Liu R, et al. Measuring the intrinsic dimension of objective landscapes[J]. arXiv preprint arXiv:1804.08838, 2018.

[147] Aghajanyan A, Zettlemoyer L, Gupta S. Intrinsic dimensionality explains the effectiveness of language model fine-tuning[J]. arXiv preprint arXiv:2012.13255, 2020.

[148] Dong Q, Li L, Dai D, et al. A survey for in-context learning[J]. arXiv preprint arXiv:2301.00234, 2022.

[149] 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.

[150] 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.

[151] Zhou D, Schärli N, Hou L, et al. Least-to-most prompting enables complex reasoning in large language models[J]. arXiv preprint arXiv:2205.10625, 2022.

[152] Xie S M, Raghunathan A, Liang P, et al. An explanation of in-context learning as implicit bayesian inference[J]. arXiv preprint arXiv:2111.02080, 2021.

[153] Baum L E, Petrie T. Statistical inference for probabilistic functions of finite state Markov chains[J]. The annals of mathematical statistics, 1966, 37(6): 1554-1563.

[154] Blei D M, Ng A Y, Jordan M I. Latent dirichlet allocation[J]. Journal of machine Learning research, 2003, 3(Jan): 993-1022.

[155] Gruber A, Weiss Y, Rosen-Zvi M. Hidden topic markov models[C]//Artificial intelligence and statistics. PMLR, 2007: 163-170.

[156] Zhao Z, Wallace E, Feng S, et al. Calibrate before use: Improving few-shot performance of language models[C]//International Conference on Machine Learning. PMLR, 2021: 12697-12706.

[157] Lu Y, Bartolo M, Moore A, et al. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity[J]. arXiv preprint arXiv:2104.08786, 2021.

[158] Rashkin H, Nikolaev V, Lamm M, et al. Measuring attribution in natural language generation models[J]. Computational Linguistics, 2023: 1-64.

[159] Ye X, Durrett G. The unreliability of explanations in few-shot in-context learning[J]. arXiv preprint arXiv:2205.03401, 2022.

[160] Wiegreffe S, Hessel J, Swayamdipta S, et al. Reframing human-AI collaboration for generating free-text explanations[J]. arXiv preprint arXiv:2112.08674, 2021.

[161] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, "A neural probabilistic language model," J. Mach. Learn. Res., vol. 3, pp. 1137–1155, 2003.

[162] M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz, "Building a large annotated corpus of english: The penn treebank," Comput. Linguistics, vol. 19, no. 2, pp. 313–330, 1993.

[163] S. Merity, C. Xiong, J. Bradbury, and R. Socher, "Pointer sentinel mixture models," in ICLR (Poster). OpenReview.net, 2017.

[164] L. Gao, S. Biderman, S. Black, L. Golding, T. Hoppe, C. Foster, J. Phang, H. He, A. Thite,N. Nabeshima, S. Presser, and C. Leahy, "The pile: An 800gb dataset of diverse text for language modeling," CoRR, vol. abs/2101.00027, 2021.

[165] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," in Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., 2020

[166] A. Zeng, X. Liu, Z. Du, Z. Wang, H. Lai, M. Ding, Z. Yang, Y. Xu, W. Zheng, X. Xia, W. L. Tam, Z. Ma, Y. Xue, J. Zhai, W. Chen, P. Zhang, Y. Dong, and J. Tang, "GLM-130B: an open bilingual pre-trained model," vol. abs/2210.02414, 2022.

[167] D. Paperno, G. Kruszewski, A. Lazaridou, Q. N. Pham, R. Bernardi, S. Pezzelle, M. Baroni, G. Boleda, and R. Fern´andez, "The LAMBADA dataset: Word prediction requiring a broad discourse context," in ACL (1). The Association for Computer Linguistics, 2016.

[168] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A.

Radford, J. Wu, and D. Amodei, “Scaling laws for neural language models,” CoRR, vol. abs/2001.08361, 2020.

[169] J. Li, T. Tang, W. X. Zhao, and J. Wen, “Pretrained language model for text generation: A survey,” in Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021, Z. Zhou, Ed. ijcai.org, 2021, pp. 4492–4499.

[170] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” in ICLR, 2015.

[171] R. Nallapati, B. Zhou, C. N. dos Santos, C¸ . G¨ulc¸ehre, and B. Xiang, “Abstractive text summarization using sequence-to-sequence rnns and beyond,” in Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016, Y. Goldberg and S. Riezler, Eds. ACL, 2016, pp. 280–290.

[172] J. Berant, A. Chou, R. Frostig, and P. Liang, “Semantic parsing on freebase from question-answer pairs,” in Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL, 2013, pp. 1533–1544.

[173] K. Papineni, S. Roukos, T. Ward, and W. Zhu, “Bleu: a method for automatic evaluation of machine translation,” in Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA. ACL, 2002, pp. 311–318.

[174] C.-Y. Lin, “ROUGE: A package for automatic evaluation of summaries,” in Text Summarization Branches Out. Association for Computational Linguistics, Jul. 2004, pp. 74–81.

[175] W. Jiao, W. Wang, J.-t. Huang, X. Wang, and Z. Tu, “Is chatgpt a good translator? a preliminary study,” arXiv preprint arXiv:2301.08745, 2023.

[176] T. Zhang, F. Ladhak, E. Durmus, P. Liang, K. R. McKeown, and T. B. Hashimoto, “Benchmarking large language models for news summarization,” CoRR, vol. abs/2301.13848, 2023.

[177] T. Goyal, J. J. Li, and G. Durrett, “News summarization and evaluation in the era of GPT-3,” CoRR, vol. abs/2209.12356, 2022.

[178] S. Gehrmann, E. Clark, and T. Sellam, “Repairing the cracked foundation: A survey of obstacles in evaluation practices for generated text,” CoRR, vol. abs/2202.06935, 2022.

[179] J. Wang, Y. Liang, F. Meng, H. Shi, Z. Li, J. Xu, J. Qu, and J. Zhou, “Is chatgpt a good NLG evaluator? A preliminary study,” CoRR, vol. abs/2303.04048, 2023.

[180] Y. Liu, D. Iter, Y. Xu, S. Wang, R. Xu, and C. Zhu, “Geval: NLG evaluation using GPT-4 with better human alignment,” CoRR, vol. abs/2303.16634, 2023.

[181] J. Jiang, K. Zhou, Z. Dong, K. Ye, W. X. Zhao, and J. Wen, “Structgpt: A general

framework for large language model to reason over structured data," CoRR, vol. abs/2305.09645, 2023.

[182] K. Yang, Y. Tian, N. Peng, and D. Klein, "Re3: Generating longer stories with recursive reprompting and revision," in Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022, Y. Goldberg, Z. Kozareva, and Y. Zhang, Eds. Association for Computational Linguistics, 2022, pp. 4393–4479.

[183] W. Zhou, Y. E. Jiang, P. Cui, T. Wang, Z. Xiao, Y. Hou, R. Cotterell, and M. Sachan, "Recurrentgpt: Interactive generation of (arbitrarily) long text," CoRR, vol. abs/2305.13304, 2023.

[184] S. Gulwani, O. Polozov, and R. Singh, "Program synthesis," Found. Trends Program. Lang., vol. 4, no. 1-2, pp. 1–119, 2017.

[185] D. Hendrycks, S. Basart, S. Kadavath, M. Mazeika, A. Arora, E. Guo, C. Burns, S. Puranik, H. He, D. Song, and J. Steinhardt, "Measuring coding challenge competence with APPS," in NeurIPS Datasets and Benchmarks, 2021.

[186] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. de Oliveira Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, A. Ray, R. Puri, G. Krueger, M. Petrov, H. Khlaaf, G. Sastry, P. Mishkin, B. Chan, S. Gray, N. Ryder, M. Pavlov, A. Power, L. Kaiser, M. Bavarian, C. Winter, P. Tillet, F. P. Such, D. Cummings, M. Plappert, F. Chantzis, E. Barnes, A. HerbertVoss, W. H. Guss, A. Nichol, A. Paino, N. Tezak, J. Tang, I. Babuschkin, S. Balaji, S. Jain, W. Saunders, C. Hesse, A. N. Carr, J. Leike, J. Achiam, V. Misra, E. Morikawa, A. Radford, M. Knight, M. Brundage, M. Murati, K. Mayer, P. Welinder, B. McGrew, D. Amodei, S. McCandlish, I. Sutskever, and W. Zaremba, "Evaluating large language models trained on code," CoRR, vol. abs/2107.03374, 2021.

[187] J. Austin, A. Odena, M. I. Nye, M. Bosma, H. Michalewski, D. Dohan, E. Jiang, C. J. Cai, M. Terry, Q. V. Le, and C. Sutton, "Program synthesis with large language models," CoRR, vol. abs/2108.07732, 2021.

[188] Y. Li, D. H. Choi, J. Chung, N. Kushman, J. Schrittwieser, R. Leblond, T. Eccles, J. Keeling, F. Gimeno, A. D. Lago, T. Hubert, P. Choy, C. de Masson d'Autume, I. Babuschkin, X. Chen, P. Huang, J. Welbl, S. Gowal, A. Cherepanov, J. Molloy, D. J. Mankowitz, E. S. Robson, P. Kohli, N. de Freitas, K. Kavukcuoglu, and O. Vinyals, "Competition-level code generation with alphacode," Science, 2022.

[189] M. Welsh, "The end of programming," Commun. ACM, vol. 66, no. 1, pp. 34–35, 2023.

[190] J. Ye, X. Chen, N. Xu, C. Zu, Z. Shao, S. Liu, Y. Cui, Z. Zhou, C. Gong, Y. Shen, J. Zhou, S. Chen, T. Gui, Q. Zhang, and X. Huang, "A comprehensive capability analysis of gpt-3 and gpt-3.5 series models," arXiv preprint arXiv:2303.10420, 2023.

[191] M. McCloskey and N. J. Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem," in Psychology of learning and motivation, 1989, pp.

109–165.

[192] R. Kemker, M. McClure, A. Abitino, T. L. Hayes, and C. Kanan, “Measuring catastrophic forgetting in neural networks,” in Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018, 2018, pp.3390–3398.

[193] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L. Miller, M. Simens, A. Askell, P. Welinder, P. F. Christiano, J. Leike, and R. Lowe, “Training language models to follow instructions with human feedback,” CoRR, vol.abs/2203.02155, 2022.

[194] A. Roberts, C. Raffel, and N. Shazeer, “How much knowledge can you pack into the parameters of a language model?” in Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020, 2020, pp.5418–5426.

[195] T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins, A. P. Parikh, C. Alberti, D. Epstein, I. Polosukhin, J. Devlin, K. Lee, K. Toutanova, L. Jones, M. Kelcey, M. Chang, A. M. Dai, J. Uszkoreit, Q. Le, and S. Petrov, “Natural questions: a benchmark for question answering research,” Trans. Assoc. Comput. Linguistics, pp. 452–466, 2019.

[196] J. Berant, A. Chou, R. Frostig, and P. Liang, “Semantic parsing on freebase from question-answer pairs,” in Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL, 2013, pp. 1533–1544.

[197] M. Joshi, E. Choi, D. S. Weld, and L. Zettlemoyer, “Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension,” in Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30109 - August 4, Volume 1: Long Papers, 2017, pp. 1601–1611.

[198] T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins,A. P. Parikh, C. Alberti, D. Epstein, I. Polosukhin, J. Devlin, K. Lee, K. Toutanova, L. Jones, M. Kelcey, M. Chang, A. M. Dai, J. Uszkoreit, Q. Le, and S. Petrov, “Natural questions: a benchmark for question answering research,” Trans. Assoc. Comput. Linguistics, pp. 452–466, 2019.

[199] T. Mihaylov, P. Clark, T. Khot, and A. Sabharwal, “Can a suit of armor conduct electricity? A new dataset for open book question answering,” in Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31-November 4, 2018, 2018, pp. 2381–2391.

[200] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, “Squad: 100, 000+ questions for machine comprehension of text,” in Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing,EMNLP 2016, Austin, Texas, USA, November 1-4, 2016,2016, pp. 2383–2392.

[201] F. Petroni, T. Rockt¨aschel, S. Riedel, P. S. H. Lewis, A. Bakhtin, Y. Wu, and A. H. Miller, “Language models as knowledge bases?” in Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019, 2019, pp. 2463–2473.

[202] K. Toutanova and D. Chen, “Observed versus latent features for knowledge base and text inference,” in Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality, CVSC 2015, Beijing, China, July 26-31, 2015, 2015, pp. 57–66.

[203] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel, “Convolutional 2d knowledge graph embeddings,” in Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA,February 2-7, 2018, 2018, pp. 1811–1818.

[204] B. Goodrich, V. Rao, P. J. Liu, and M. Saleh, “Assessing the factual accuracy of generated text,” in Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019, 2019, pp. 166–175.

[205] Y. Bang, S. Cahyawijaya, N. Lee, W. Dai, D. Su, B. Wilie, H. Lovenia, Z. Ji, T. Yu, W. Chung, Q. V. Do, Y. Xu, and P. Fung, “A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity,” CoRR, vol. abs/2302.04023,2023.

[206] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin, and T. Liu, “A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions,” CoRR, vol. abs/2311.05232, 2023.

[207] S. Lin, J. Hilton, and O. Evans, “Truthfulqa: Measuring how models mimic human falsehoods,” in Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022, 2022, pp. 3214–3252.

[208] P. Wang, N. Zhang, X. Xie, Y. Yao, B. Tian, M. Wang, Z. Xi, S. Cheng, K. Liu, G. Zheng, and H. Chen, “Easyedit: An easy-to-use knowledge editing framework for large language models,” CoRR, vol.abs/2308.07269, 2023.

[209] T. Saikh, T. Ghosal, A. Mittal, A. Ekbal, and P. Bhattacharyya, “Scienceqa: a novel resource for question answering on scholarly articles,” Int. J. Digit. Libr.,vol. 23, no. 3, pp. 289–301, 2022.

[210] J. Wei, X. Wang, D. Schuurmans, M. Bosma, E. H. Chi,Q. Le, and D. Zhou, “Chain of thought prompting elicits reasoning in large language models,” CoRR, vol.abs/2201.11903, 2022.

[211] M. I. Nye, A. J. Andreassen, G. Gur-Ari, H. Michalewski, J. Austin, D. Bieber, D.

Dohan, A. Lewkowycz, M. Bosma, D. Luan, C. Sutton,and A. Odena, “Show your work: Scratchpads for intermediate computation with language models,”CoRR, vol. abs/2112.00114, 2021.

[212] J. Qian, H. Wang, Z. Li, S. Li, and X. Yan, “Limitations of language models in arithmetic and symbolic induction,” CoRR, vol. abs/2208.05051, 2022.

[213] A. Patel, S. Bhattamishra, and N. Goyal, “Are NLP models really able to solve simple math word problems?” in NAACL-HLT. Association for Computational Linguistics, 2021, pp. 2080–2094.

[214] K. Cobbe, V. Kosaraju, M. Bavarian, J. Hilton, R. Nakano, C. Hesse, and J. Schulman, “Training verifiers to solve math word problems,” CoRR, vol.abs/2110.14168, 2021.

[215] D. Hendrycks, C. Burns, S. Basart, A. Zou,M. Mazeika, D. Song, and J. Steinhardt, “Measuring massive multitask language understanding,” in ICLR.101 OpenReview.net, 2021.

[216] A. Q. Jiang, S. Welleck, J. P. Zhou, W. Li, J. Liu,M. Jamnik, T. Lacroix, Y. Wu, and G. Lample, “Draft, sketch, and prove: Guiding formal theorem provers with informal proofs,” CoRR, vol. abs/2210.12283, 2022.

[217] X. Wang, J. Wei, D. Schuurmans, Q. V. Le, E. H.Chi, and D. Zhou, “Self-consistency improves chain of thought reasoning in language models,” CoRR, vol.abs/2203.11171, 2022.

[218] S. Agarwal, I. Akkaya, V. Balcom, M. Bavarian, G. Bernadett-Shapiro, G. Brockman, M. Brundage, J. Chan, F. Chantzis, N. Deutsch, B. Eastman, A. Eleti, N. Felix, S. P. Fishman, I. Fulford, C. Gibson, J. Gross, M. Heaton, J. Hilton, X. Hu, S. Jain, H. Jin, L. Kilpatrick, C. Kim, M. Kolhede, A. Mayne, P. McMillan, D. Medina, J. Menick, A. Mishchenko, A. Nair, R. Nayak, A. Neelakantan, R. Nuttall, J. Parish, A. T. Passos, A. Perelman, F. de Avila Belbute Peres, V. Pong, J. Schulman, E. Sigler, N. Staudacher, N. Turley, J. Tworek, R. Greene, A. Vijayvergiya, C. Voss,114 J. Weng, M. Wiethoff, S. Yoo, K. Yu, W. Zaremba,S. Zhao, W. Zhuk, and B. Zoph, “Chatgpt plugins,”OpenAI Blog, March 2023.

[219] N. Nangia, C. Vania, R. Bhalerao, and S. R. Bowman, “Crows-pairs: A challenge dataset for measuring social biases in masked language models,” in Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020, 2020, pp. 1953–1967.

[220] R. Rudinger, J. Naradowsky, B. Leonard, and B. V.Durme, “Gender bias in coreference resolution,” in Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers), 2018, pp. 8–14.

[221] OpenAI, “Gpt-4 technical report,” OpenAI, 2023.

[222] R. Taylor, M. Kardas, G. Cucurull, T. Scialom,A. Hartshorn, E. Saravia, A. Poulton,

V. Kerkez, and R. Stojnic, “Galactica: A large language model for science,” CoRR, vol. abs/2211.09085, 2022.

[223] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch, “Language models as zero-shot planners: Extracting actionable knowledge for embodied agents,” in ICML,ser. Proceedings of Machine Learning Research, vol.162. PMLR, 2022, pp. 9118–9147.

[224] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes,B. David, C. Finn, K. Gopalakrishnan, K.Hausman, A.Herzog, D. Ho, J. Hsu, J. Ibarz, B. Ichter, A. Irpan,E. Jang, R. J. Ruano, K. Jeffrey, S. Jesmonth, N. J. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, K. Lee, S. Levine, Y. Lu, L. Luu, C. Parada, P. Pastor, J. Quiambao, K. Rao, J. Rettinghouse, D. Reyes, P. Sermanet, N. Sievers, C. Tan, A. Toshev, V. Vanhoucke, F. Xia, T. Xiao, P. Xu, S. Xu, and M. Yan, “Do as I can, not as I say: Grounding language in robotic affordances,” CoRR, vol. abs/2204.01691, 2022.

[225] Y. Chang, X. Wang, J. Wang, Y. Wu, K. Zhu, H. Chen,L. Yang, X. Yi, C. Wang, Y. Wang, W. Ye, Y. Zhang,Y. Chang, P. S. Yu, Q. Yang, and X. Xie, “A survey on evaluation of large language models,” CoRR, vol.abs/2307.03109, 2023.

[226] Z. Zhuang, Q. Chen, L. Ma, M. Li, Y. Han, Y. Qian, H. Bai, Z. Feng, W. Zhang, and T. Liu, “Through the lens of core competency: Survey on evaluation of large language models,” CoRR, vol. abs/2308.07902, 2023.

[227] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt, “Measuring massive multitask language understanding,” in ICLR.101 OpenReview.net, 2021.

[228] A. Srivastava, A. Rastogi, A. Rao, A. A. M. Shoeb,A. Abid, A. Fisch, A. R. Brown, A. Santoro, A. Gupta,A. Garriga-Alonso, A. Kluska, A. Lewkowycz,A. Agarwal, A. Power, A. Ray, A. Warstadt, A. W. Kocurek, A. Safaya, A. Tazarv, A. Xiang, A. Parrish,A. Nie, A. Hussain, A. Askell, A. Dsouza, A. Rahane,A. S. Iyer, A. Andreassen, A. Santilli, A. Stuhlm¨uller,A. M. Dai, A. La, A. K. Lampinen, A. Zou, A. Jiang,A. Chen, A. Vuong, A. Gupta, A. Gottardi, A. Norelli, A. Venkatesh, A. Gholamidavoodi, A. Tabassum,A. Menezes, A. Kirubarajan, A. Mullokandov, A. Sabharwal, A. Herrick, A. Efrat, A. Erdem, A. Karakas,and et al., “Beyond the imitation game: Quantifying and extrapolating the capabilities of language models,” CoRR, vol. abs/2206.04615, 2022.

[229] P. Liang, R. Bommasani, T. Lee, D. Tsipras, D. Soylu,M. Yasunaga, Y. Zhang, D. Narayanan, Y. Wu, A. Kumar, B. Newman, B. Yuan, B. Yan, C. Zhang, C. Cosgrove, C. D. Manning, C. R´e, D. Acosta-Navas, D. A.Hudson, E. Zelikman, E. Durmus, F. Ladhak, F. Rong,H. Ren, H. Yao, J. Wang, K. Santhanam, L. J. Orr, L. Zheng, M. Y¨uksekg¨on¨ul, M. Suzgun, N. Kim,N. Guha, N. S. Chatterji, O. Khattab, P. Henderson,Q. Huang, R. Chi, S. M. Xie, S. Santurkar, S. Ganguli, T. Hashimoto, T. Icard, T. Zhang, V. Chaudhary,W. Wang, X. Li, Y. Mai, Y. Zhang, and Y. Koreeda, “Holistic evaluation of language models,” CoRR, vol.abs/2211.09110, 2022.

[230] W. Zhong, R. Cui, Y. Guo, Y. Liang, S. Lu, Y. Wang, A. Saied, W. Chen, and N. Duan, “Agieval: A human centric benchmark for evaluating foundation models,”CoRR, vol.

abs/2304.06364, 2023.

[231] H. Zeng, “Measuring massive multitask chinese understanding,” CoRR, vol. abs/2304.12986, 2023.

[232] C. Liu, R. Jin, Y. Ren, L. Yu, T. Dong, X. Peng, S. Zhang, J. Peng, P. Zhang, Q. Lyu, X. Su, Q. Liu, and D. Xiong, “M3KE: A massive multi-level multi-subject knowledge evaluation benchmark for chinese large language models,” CoRR, vol. abs/2305.10263,2023.

[233] Y. Huang, Y. Bai, Z. Zhu, J. Zhang, J. Zhang, T. Su,J. Liu, C. Lv, Y. Zhang, J. Lei, Y. Fu, M. Sun, and J. He, “C-eval: A multi-level multi-discipline chinese evaluation suite for foundation models,” CoRR, vol.abs/2305.08322, 2023.

[234] Z. Gu, X. Zhu, H. Ye, L. Zhang, J. Wang, S. Jiang,Z. Xiong, Z. Li, Q. He, R. Xu, W. Huang, W. Zheng,H. Feng, and Y. Xiao, “Xiezhi: An ever-updating benchmark for holistic domain knowledge evaluation,” CoRR, vol. abs/2306.05783, 2023.

[235] E. Beeching, C. Fourrier, N. Habib, S. Han, N. Lambert, N. Rajani, O. Sanseviero, L. Tunstall, and T. Wolf, “Open llm leaderboard,” https://huggingface.co/spaces/Hugging-FaceH4/open llm leaderboard,2023.

[236] L. Zheng, W. Chiang, Y. Sheng, S. Zhuang, Z. Wu,Y. Zhuang, Z. Lin, Z. Li, D. Li, E. P. Xing, H. Zhang,J. E. Gonzalez, and I. Stoica, “Judging llm-as-a-judge with mt-bench and chatbot arena,” CoRR, vol.abs/2306.05685, 2023.

[237] X. Li, T. Zhang, Y. Dubois, R. Taori, I. Gulrajani,C. Guestrin, P. Liang, and T. B. Hashimoto, “Alpacae val: An automatic evaluator of instruction-following models,” https://github.com/tatsu-lab/alpaca eval,2023.

[238] O. Contributors, “Opencompass: A universal evaluation platform for foundation models,” https://github.com/InternLM/OpenCompass, 2023.

[239] W.-L. Chiang, Z. Li, Z. Lin, Y. Sheng, Z. Wu, H. Zhang,L. Zheng, S. Zhuang, Y. Zhuang, J. E. Gonzalez,I. Stoica, and E. P. Xing, “Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality,”2023. [Online]. Available: https://vicuna.lmsys.org

[240] K. Singhal, S. Azizi, T. Tu, S. S. Mahdavi, J. Wei, H. W. Chung, N. Scales, A. Tanwani, H. Cole-Lewis,S. Pfohl et al., “Large language models encode clinical knowledge,” arXiv preprint arXiv:2212.13138, 2022.

[241] R. Shah, K. Chawla, D. Eidnani, A. Shah, W. Du,S. Chava, N. Raman, C. Smiley, J. Chen, and D. Yang, “When flue meets flang: Benchmarks and large pretrained language model for financial domain,” in Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, 2022, pp. 2322–2335.

[242] M. Suzgun, N. Scales, N. Sch¨arli, S. Gehrmann, Y. Tay,H. W. Chung, A. Chowdhery, Q. V. Le, E. H. Chi,D. Zhou, and J. Wei, “Challenging big-bench tasks and whether chain-of-thought can solve them,” CoRR, vol.abs/2210.09261, 2022.

[243] K. Zhou, Y. Zhu, Z. Chen, W. Chen, W. X. Zhao, X. Chen, Y. Lin, J.-R. Wen, and J.

Han, “Don’t make your llm an evaluation benchmark cheater,” arXiv preprint arXiv:2311.01964, 2023.

[244] R. Thoppilan, D. De Freitas, J. Hall, N. Shazeer, A. Kulshreshtha, H.-T. Cheng, A. Jin, T. Bos et al. 2022. Lamda: Language models for dialog applications.

[245] B. Hu, J. Xia, J. Zheng, C. Tan, Y. Huang, Y. Xu and S. Z. Li. 2022. Protein language models and structure prediction: Connection and progression.

[246] OpenAI. 2022. Chatgpt: Optimizing language models for dialogue.

[247] K. Shuster, J. Xu, M. Komeili, D. Ju, E. M. Smith, S. Roller, M. Ung, M. Chen et al. 2022. Blenderbot 3: a deployed conversational agent that continually learns to responsibly engage.

[248] A. Köpf, Y. Kilcher, D. von Rütte, S. Anagnostidis, Z.R. Tam, K. Stevens, A. Barhoum, N. M. Duc et al. 2023. Openassistant conversations–democratizing large language model alignment.

[249] M. Chen, A. Papangelis, C. Tao, S. Kim, A. Rosenbaum, Y. Liu, Z. Yu and D. Hakkani-Tur. 2023. Places: Prompting language models for social conversation synthesis.

[250] J. Koco´n, I. Cichecki, O. Kaszyca, M. Kochanek, D. Szydło, J. Baran, J. Bielaniewicz, M. Gruza et al. 2023. Chatgpt: Jack of all trades, master of none.

[251] S. Bubeck, V. Chandrasekaran, R. Eldan, J. Gehrke, E. Horvitz, E. Kamar, P. Lee, Y. T. Lee et al. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4.

[252] Babak Mahjour, Jillian Hoffstadt, and Tim Cernak. 2023. Designing Chemical Reaction Arrays using phactor and ChatGPT. (2023).

[253] Bhavitvya Malik, Abhinav Ramesh Kashyap, Min-Yen Kan, and Soujanya Poria. 2023. UDApter–Efficient Domain Adaptation Using Adapters.

[254] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. d. O. Pinto, J. Kaplan, H. Edwards, Y. Burda et al. 2021. Evaluating large language models trained on code.

[255] S. Gunasekar, Y. Zhang, J. Aneja, C. C. T. Mendes, A. D. Giorno, S. Gopi, M. Javaheripi, P. Kauffmann et al. 2023.Textbooks are all you need.

[256] F. F. Xu, U. Alon, G. Neubig and V. J. Hellendoorn. 2022. A systematic evaluation of large language models of code.

[257] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. 2017. Learning multiple visual domains with residual adapters. Advances in neural information processing systems 30 (2017).

[258] Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model?

[259] Joshua Robinson, Christopher Michael Rytting, and David Wingate. 2022. Leveraging Large Language Models for Multiple Choice Question Answering.

[260] Jerret Ross, Brian Belgodere, Vijil Chenthamarakshan, Inkit Padhi, Youssef Mroueh, and Payel Das. 2022. Large-scale chemical language representations capture molecular

structure and properties. Nature Machine Intelligence.

[261] Dheeru Dua, Shivanshu Gupta, Sameer Singh, and Matt Gardner. 2022. Successive Prompting for Decomposing Complex Questions. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing.

[262] Ali Edalati, Marzieh Tahaei, Ivan Kobyzev, Vahid Partovi Nia, James J Clark, and Mehdi Rezagholizadeh. 2022. KronA: Parameter Efficient Tuning with Kronecker Adapter.

[263] Timo Schick and Hinrich Schütze. 2020. Exploiting cloze questions for few shot text classification and natural language inference.

[264] Yang Feng, Shiyue Zhang, Andi Zhang, Dong Wang, and Andrew Abel. 2017. Memory-augmented Neural Machine Translation. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing.

[265] Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2022. PAL: Program-aided Language Models.

[266] Chunjiang Ge, Rui Huang, Mixue Xie, Zihang Lai, Shiji Song, Shuang Li, and Gao Huang. 2022. Domain Adaptation via Prompt Learning.

[267] Edouard Grave, Armand Joulin, and Nicolas Usunier. 2017. Improving neural language models with a continuous cache. In International Conference on Learning Representations.

[268] Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2022. PPT: Pre-trained Prompt Tuning for Few-shot Learning. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, Dublin, Ireland, 8410–8423.

[269] Dídac Surís, Sachit Menon, and Carl Vondrick. 2023. Vipergpt: Visual inference via python execution for reasoning.

[270] Norbert Wiener. 1960. Some moral and technical consequences of automation: As machines learn they may develop unforeseen strategies at rates that baffle their programmers. Science, 131(3410):1355–1358.

[271] Moin Nadeem, Anna Bethke, and Siva Reddy. 2020. Stereoset: Measuring stereotypical bias in pretrained language models. arXiv preprint arXiv:2004.09456.

[272] Li Lucy and David Bamman. 2021. Gender and representation bias in GPT-3 generated stories. In Proceedings of the Third Workshop on Narrative Understanding, pages 48–55.

[273] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. ACM Computing Surveys, 55(12):1–38.

[274] Richard Ngo. 2022. The alignment problem from a deep learning perspective. arXiv preprint arXiv:2209.00626.

[275] Victoria Krakovna. 2022. Paradigms of ai alignment: components and enablers.

[276] Rohin Shah, Vikrant Varma, Ramana Kumar, Mary Phuong, Victoria Krakovna, Jonathan Uesato, and Zac Kenton. 2022. Goal misgeneralization: Why correct specifications aren't enough for correct goals. arXiv preprint arXiv:2210.01790.

[277] Neel Nanda, Lawrence Chan, Tom Liberum, Jess Smith, and Jacob Steinhardt. 2023. Progress measures for grokking via mechanistic interpretability. arXiv preprint arXiv:2301.05217.

[278] Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. 2021. A mathematical framework for transformer circuits. Transformer Circuits Thread.

[279] Zachary C. Lipton. 2017. The mythos of model interpretability.

[280] Evan Hubinger. 2022a. A transparency and interpretability tech tree. https://www.lesswrong.com/posts/nbq2bWLcYmSGup9aF/a-transparency-and interpretability-tech-tree.

[281] Stephen Casper, Xander Davies, Claudia Shi, Thomas Krendl Gilbert, Jérémy Scheurer, Javier Rando, Rachel Freedman, Tomasz Korbak, David Lindner, Pedro Freire, et al. 2023. Open problems and fundamental limitations of reinforcement learning from human feedback. arXiv preprint arXiv:2307.15217.

[282] Marta R Costa-jussà, Pierre Andrews, Eric Smith, Prangthip Hansanti, Christophe Ropers, Elahe Kalbassi, Cynthia Gao, Daniel Licht, and Carleigh Wood. 2023. Multilingual holistic bias: Extending descriptors and patterns to unveil demographic biases in languages at scale. arXiv preprint arXiv:2305.13198.

[283] Kaddour J, Harris J, Mozes M, et al. Challenges and applications of large language models[J]. arXiv preprint arXiv:2307.10169, 2023.

[284] K. Lee, D. Ippolito, A. Nystrom, C. Zhang, D. Eck, C. Callison-Burch and N. Carlini. 2021. Deduplicating training data makes language models better. arXiv preprint arXiv:2107.06499.

[285] J. Kaddour. 2023. The MiniPile Challenge for Data Efficient Language Models. ArXiv:2304.08442 [cs].

[286] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Ka plan, P. Dhariwal, A. Neelakantan, P. Shyam et al. 2020. Language models are few-shot learners. In Advances in Neural Information Processing Systems, volume 33, pages 1877–1901. Curran Associates, Inc.

[287] V. Feldman. 2020. Does learning require memorization? a short tale about a long tail. In Proceedings of the 52nd Annual ACMSIGACTSymposium on Theory of Computing, pages 954–959.

[288] N. Carlini, C. Liu, Ú. Erlingsson, J. Kos and D. Song. 2019. The secret sharer: Evaluating and testing unintended memorization in neural networks. In USENIX Security Symposium, volume 267.

[289] N. Lukas, A. Salem, R. Sim, S. Tople, L. Wutschitz and S. Zanella-Béguelin. 2023. Analyzing Leakage of Personally Identifiable Information in Language Models. ArXiv:2302.00539 [cs].

[290] A. Kulkarni. 2021. GitHub Copilot AI Is Leaking Func tional API Keys.

[291] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung et al. 2022. Palm: Scaling language modeling with pathways. arXiv preprint arXiv:2204.02311.

[292] B. Sorscher, R. Geirhos, S. Shekhar, S. Ganguli and A. S. Morcos. 2022. Beyond neural scaling laws: beat ing power law scaling via data pruning. arXiv preprint arXiv:2206.14486.

[293] J. Devlin, M.-W. Chang, K. Lee and K. Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 Con ference of the North American Chapter of the Association for Computational Linguistics: Human Language Tech nologies, Volume 1 (Long and Short Papers), pages 4171 4186, Minneapolis, Minnesota. Association for Computa tional Linguistics.

[294] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan and S. Gelly. 2019. Parameter-efficient transfer learning for nlp. In International Conference on Machine Learning, pages 2790–2799. PMLR.

[295] E. Ben Zaken, Y. Goldberg and S. Ravfogel. 2022. Bit Fit: Simple parameter-efficient fine-tuning for transformer based masked language-models. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pages 1–9, Dublin, Ireland. Association for Computational Linguistics.

[296] H. Liu, D. Tam, M. Muqeeth, J. Mohta, T. Huang, M. Bansal and C. A. Raffel. 2022. Few-shot parameter efficient fine-tuning is better and cheaper than in-context learning. Advances in Neural Information Processing Sys tems, 35:1950–1965.

[297] Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Wei Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Ha jishirzi. 2023. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. arXiv preprint arXiv:2305.14251.

[298] Eric Mitchell, Charles Lin, Antoine Bosselut, Christopher D Manning, and Chelsea Finn. 2022. Memory-based model editing at scale. In International Conference on Machine Learn ing, pages 15817–15831. PMLR.

[299] Jean Kaddour, Joshua Harris, Maximilian Mozes, Herbie Bradley, Roberta Raileanu, and Robert McHardy. 2023. Challenges and applications of large language models. arXiv preprint arXiv:2307.10169.

[300] Logesh Kumar Umapathi, Ankit Pal, and Malaikannan Sankarasubbu. 2023. Med halt: Medical domain hallucination test for large language models. arXiv preprint arXiv:2307.15343.

[301] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye

Jin Bang, An drea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. ACMComputing Surveys, 55(12):1–38.

[302] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feed back. Advances in Neural Information Process ing Systems, 35:27730–27744.

[303] Zhang Y, Li Y, Cui L, et al. Siren's song in the AI ocean: a survey on hallucination in large language models[J]. arXiv preprint arXiv:2309.01219, 2023.

[304] Katherine Lee, Orhan Firat, Ashish Agarwal, Clara Fannjiang, and David Sussillo. 2019. Hallucinations in neural machine translation.

[305] Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan T. McDonald. 2020. On faithful ness and factuality in abstractive summariza tion. In Proceedings of the 58th Annual Meet ing of the Association for Computational Lin guistics, ACL 2020, Online, July 5-10, 2020, pages 1906–1919. Association for Computa tional Linguistics.

[306] Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H. Chi, Nathanael Schärli, and Denny Zhou. 2023a. Large lan guage models can be easily distracted by irrel evant context. In Proceedings of the 40th In ternational Conference on Machine Learning, volume 202, pages 31210–31227.

[307] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, An drea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. ACMComputing Surveys, 55(12):1–38.

[308] Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. 2023a. Lima: Less is more for alignment. arXiv preprint arXiv:2305.11206.

[309] Ekin Akyürek, Tolga Bolukbasi, Frederick Liu, Binbin Xiong, Ian Tenney, Jacob Andreas, and Kelvin Guu. 2022. Tracing knowledge in lan guage models back to the training data. arXiv preprint arXiv:2205.11482.

[310] Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. Creating training corpora for NLG micro planners. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 179–188.

[311] Hugo Touvron, Louis Martin, Kevin Stone, Pe ter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. arXiv preprint arXiv:2307.09288.

[312] Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cap pelli, Hamza Alobeidli, Baptiste Pannier, Ebte sam Almazrouei, and Julien Launay. 2023. The refinedweb dataset for falcon llm: outperform ing curated corpora with web data, and web data only. arXiv preprint arXiv:2306.01116.

[313] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023c. Self-instruct: Aligning language models with self-generated instructions. In Proceedings of the 61st An nual Meeting of the Association for Compu tational Linguistics (Volume 1: Long Papers), pages 13484–13508.

[314] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2023a. Voy ager: An open-ended embodied agent with large language models. arXiv preprint arXiv:2305.16291.

[315] Elaraby Mohamed, Lu Mengyin, Dunn Jacob, Zhang Xueying, Wang Yu, and Liu Shizhu. 2023. Halo: Estimation and reduction of hal lucinations in open-source weak large language models. arXiv preprint arXiv:2308.11764.

[316] John Schulman. 2023. Reinforcement learning from humanfeedback: Progress and challenges.

[317] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feed back. Advances in Neural Information Process ing Systems, 35:27730–27744.

[318] Sina Zarrieß, Henrik Voigt, and Simeon Schüz. 2021. Decoding methods in neural language generation: a survey. Information, 12(9):355.

[319] Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. 2023. Chain-of-verification reduces hallucination in large language models. arXiv preprint arXiv:2309.11495.

[320] Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. 2023b. Enabling large language models to generate text with citations. arXiv preprint arXiv:2305.14627.

[321] Deng Cai, Yan Wang, Huayang Li, Wai Lam, and Lemao Liu. 2021. Neural machine translation with monolingual translation memory. In Pro ceedings of the 59th Annual Meeting of the As sociation for Computational Linguistics and the 11th International Joint Conference on Natural

[322] Potsawee Manakul, Adian Liusie, and Mark JF Gales. 2023. Selfcheckgpt: Zero-resource black-box hallucination detection for genera tive large language models. arXiv preprint arXiv:2303.08896.

[323] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. In The Eleventh International Conference on Learning Repre sentations.

[324] Angeliki Lazaridou, Elena Gribovskaya, Woj ciech Stokowiec, and Nikolai Grigorev. 2022. Internet-augmented language models through few-shot prompting for open-domain question answering. arXiv preprint arXiv:2203.05115.

[325] Ziyang Luo, Can Xu, Pu Zhao, Xiubo Geng, Chongyang Tao, Jing Ma, Qingwei Lin, and Daxin Jiang. 2023c. Augmented large lan guage models with parametric knowledge guid ing. arXiv preprint arXiv:2305.04757.

[326] I-Chun Chern, Steffi Chern, Shiqi Chen, Weizhe Yuan, Kehua Feng, Chunting Zhou, Junxian He, Graham Neubig, and Pengfei Liu. 2023. Factool: Factuality detection in generative ai– a tool augmented framework for multi-task and multi-domain scenarios. arXiv preprint arXiv:2307.13528.

[327] Zhibin Gou, Zhihong Shao, Yeyun Gong, Ye long Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. 2023. Critic: Large language models can self-correct with tool-interactive critiquing. arXiv preprint arXiv:2305.11738.

[328] Alex Fabbri, Prafulla Kumar Choubey, Jesse Vig, Chien-Sheng Wu, and Caiming Xiong. 2022. Improving factual consistency in summariza tion with compression-based post-editing. In Proceedings of the 2022 Conference on Empir ical Methods in Natural Language Processing, pages 9149–9156.

[329] Anthony Chen, Panupong Pasupat, Sameer Singh, Hongrae Lee, and Kelvin Guu. 2023a. Purr: Efficiently editing language model hallucina tions by denoising language model corruptions. arXiv preprint arXiv:2305.14908.

[330] Luyu Gao, Zhuyun Dai, Panupong Pasupat, An thony Chen, Arun Tejasvi Chaganty, Yicheng Fan, Vincent Zhao, Ni Lao, Hongrae Lee, Da Cheng Juan, et al. 2023a. Rarr: Researching and revising what language models say, using language models. In Proceedings of the 61st Annual Meeting of the Association for Compu tational Linguistics (Volume 1: Long Papers), pages 16477–16508.

[331] Ruochen Zhao, Xingxuan Li, Shafiq Joty, Cheng wei Qin, and Lidong Bing. 2023a. Verify-and edit: A knowledge-enhanced chain-of-thought framework. arXiv preprint arXiv:2305.03268.

[332] Baolin Peng, Michel Galley, Pengcheng He, Hao Cheng, Yujia Xie, Yu Hu, Qiuyuan Huang, Lars Liden, Zhou Yu, Weizhu Chen, et al. 2023a. Check your facts and try again: Improving large language models with external knowl edge and automated feedback. arXiv preprint arXiv:2302.12813.

[333] Ge S, Zhang Y, Liu L, et al. Model tells you what to discard: Adaptive kv cache compression for llms[J]. arXiv preprint arXiv:2310.01801, 2023.

[334] Transformers KV Caching Explained João Lages https://medium.com/@joaolages/kv-caching-explained-276520203249

[335] Dao T, Fu D, Ermon S, et al. Flashattention: Fast and memory-efficient exact attention with io-awareness[J]. Advances in Neural Information Processing Systems, 2022, 35: 16344-16359.

[336] GPU inference https://huggingface.co/docs/transformers/perf_infer_gpu_one#gpu-inference

[337] 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.