大模型关键技术与应用

Key Technologies and Applications of Large Models

韩炳涛/HAN Bingtao[1,2]，刘涛/LIU Tao[1,2]
（1. 中兴通讯股份有限公司，中国 深圳 518057；
2. 移动网络和移动多媒体技术国家重点实验室，中国 深圳 518055）
(1. ZTE Corporation, Shenzhen 518057, China;
2. The State Key Laboratory of Mobile Network and Mobile Multimedia Technology, Shenzhen 518055, China)

DOI：10.12142/ZTETJ.202402012
网络出版地址：http://kns.cnki.net/kcms/detail/34.1228.TN.20240418.1324.004.html
网络出版日期：2024-04-19
收稿日期：2024-02-18

摘要：介绍了自ChatGPT发布以来，大模型关键技术和应用的主要进展。在大模型设计方面，模型规模不断增加，但已有放缓趋势，更长的上下文以及多模态已经成为主流，计算效率明显提升；在模型训练方面，从单纯追求数据数量逐渐转变为关注数据的多样性和质量，特别是如何使用合成数据训练大模型成为主流探索方向，这是迈向通用人工智能（AGI）的关键；在模型推理方面，模型量化和推理引擎优化极大降低了模型使用成本，诸如投机采样等新兴算法逐渐成熟。在应用层，Agent技术获得了重大进展，在克服大模型固有缺陷方面发挥了不可替代的作用。越来越多的企业开始规划、研发以及使用大模型，企业级大模型应用架构日益成熟完善，并以场景、技术、算法三要素为抓手加速大模型商业价值闭环。

关键词：大模型；模型训练；推理加速；大模型安全；智能体

Abstract: The major advances in key technologies and applications of large models since the release of ChatGPT are presented. In terms of large model design, the model scale is increasing, but it has slowed down. Longer context and multi-mode have become the mainstream, and the computational efficiency has been significantly improved. In terms of model training, the focus has shifted from simply seeking a larger quantity of data to a more focused approach on the diversity and quality of data, especially how to train large models using synthetic data. This is an essential direction towards achieving artificial general intelligence (AGI). In terms of model inference, model quantification and inference engine optimization greatly reduce the cost of model use, and emerging algorithms such as speculative sampling gradually mature. At the application level, Agent technology has made significant progress, playing a critical role in addressing the inherent limitations of large models. More and more enterprises are beginning to plan, develop, and utilize large models, and the enterprise-level large model application architecture is becoming increasingly mature, focusing on scenarios, technologies, and algorithms to accelerate the closing loop of large model commercial value.

Keywords: large model; model training; inference accelerating; large model safety; Agent

引用格式：韩炳涛, 刘涛. 大模型关键技术与应用 [J]. 中兴通讯技术, 2024, 30(2): 76-88. DOI: 10.12142/ZTETJ.202402012
Citation: HAN B T, LIU T. Key technologies and applications of large models [J]. ZTE technology journal, 2024, 30(2): 76-88. DOI: 10.12142/ZTETJ.202402012

2022年底，OpenAI发布了跨时代的ChatGPT应用。这是第一个具有流畅的多轮对话体验、渊博的通识知识，并能够深刻理解人类意图的生成式人工智能（AI）应用。它的成功使大模型成为AI的主旋律，在极短的时间内改变了AI产业的格局。

尽管距离ChatGPT的发布仅过去一年多，但大模型技术已经取得了巨大的进展。随着GPT-4、Gemini、Sora、Claude3、Kimi等一系列大模型的陆续发布，大模型能力已迅速提升，甚至更为强大的通用人工智能（AGI）已初见端倪。本文中，我们试图对ChatGPT发布以来大模型的关键技术做出综述，厘清大模型技术全貌及发展态势，以便读者做出更好的判断和预测。

1 模型设计

目前，主流大模型的结构都是在Transformer基础上不断改进，以实现更强大的语言理解和生成能力、更长的上下文推理能力、更多模态的数据处理能力，以及更高的算力利用效率。

1.1 主流大模型的架构演化

2017年，Google提出Transformer[1]，创造性地将注意力机制作为核心算子来构建语言模型，解决了长短期记忆网络

（LSTM）[2]等循环神经网络计算效率低、训练容易过拟合的问题。此后，OpenAI 和 Google 在 Transformer 基础上，又分别提出了 GPT[3]和 BERT[4]。GPT 采用了 Transformer 的解码器部分，使用从前到后的单向预测模式（类似于补全）。BERT 则采用了 Transformer 的编码器部分，使用上文与下文的双向预测模式（类似于填空）。受益于该模式，BERT 实现了较强的性能，让业界一度认为双向语言模型是更优的选择。

但是，OpenAI 笃定追逐“通用语言模型”，认为从前到后的生成能力可以转化应用于各类语言任务上，因此在后续模型中依然坚持单向预测模式。直到 2020 年，OpenAI 提出了拥有 1 750 亿参数的 GTP-3[5]模型。GTP-3 在对话、知识问答、吟诗作赋等多项任务中展示出的能力均令人印象深刻。

此后，OpenAI 不再公开模型相关的技术细节，研究人员开始把目光聚焦在其他开源模型上。2023 年 Meta 发布 LLaMA 系列模型[6]，进一步优化了 Transfomer 模型架构，并使用更加充分的数据对模型进行训练，获得了不错的性能。此后，许多研究人员相继基于 LLaMA 模型不断地做局部的优化，如 Baichuan、Yi、Mistral、Qwen 等。

1.2 对计算效率的优化

相较于之前 LSTM 等循环神经网络，Transformer 最大的弱点是计算效率低，主要原因是其自注意力机制的计算复杂度与序列长度的平方成正比关系。针对该问题，一条技术路线是放弃 Transformer，设计更高效的模型结构。目前我们认为有可能取代 Transformer 的架构主要包括 Linear Transformer、RWKV 和 Mamba 等。Memba 采用了完全不同的模型架构[7]，彻底抛弃了注意力机制，从状态空间的角度对序列进行建模。相关研究表明，该模型把训练计算复杂度降到线性，且能力与 Transformer 相当。A21 Labs 刚刚发布了首个生产级别的基于 Mamba 的模型 Jamba，模型参数达到了 520 亿，同时提供长达 256 k 的上下文。尽管当前 Memba 在参数规模上与 Transformer 仍然有较大差距，但前景仍被看好。业界认为 Memba 是取代 Transformer 的有力竞争者。

另一条技术路线是继续优化 Transformer 模型结构。例如，在注意力机制方面，代表性的工作包括将多头注意力机制（MHA）改进为多查询注意力机制（MQA）和分组查询注意力机制（GQA），这可以进一步降低计算量。其中，LLaMA2 采用了分组查询注意力机制[8]，Google 最近发布的 Gemini 采用了多查询注意力机制。

除了结构上的改进，优化算子实现也可以提高效率，例如：FlashAttention 通过 Attention 与 Softmax 计算融合，结合 KV Cahche，能显著提高计算速度并降低显存带宽依赖[9]，从而使 Transformer 的计算效率接近线性注意力模型。

1.3 长上下文推理

能够处理的上下文序列长度是语言模型能力的一个关键指标。在许多任务中，如文章阅读理解、代码生成、检索增强生成（RAG）等，都需要模型能够处理很长的上下文。然而，长上下文模型训练与推理，存在计算复杂度高、最大长度约束等问题。

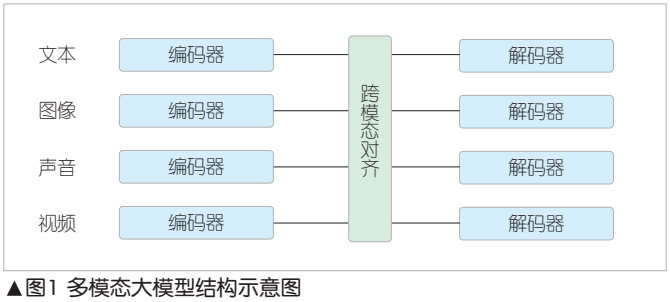
最大长度约束问题，是指模型在实际推理时处理的序列长度超过训练时序列长度，这可能会导致性能的明显下降。针对该问题，有两种解决方案：一种是预训练过程中调整模型设计，可以实现更好的模型外推能力；一种是通过微调和位置嵌入处理，扩大模型的上下文窗口。例如，苏剑林发现如果对注意力矩阵的查询-键-值（QKV）映射矩阵加入 Bias，则使用旋转位置编码（ROPE）的模型可能会获得较好的外推能力。该方法常应用在 Qwen 模型中[10]。此外，研究人员发现，如果对 ROPE 进行插值，并配合简单的微调，可以把上下文窗口的大小从 4 096 扩展到 32 000[11]。

研究人员通过对改进注意力机制，可以降低计算复杂度，其中最具代表性的是 Window Attention 方法。该方法引入了注意力窗口，让每层注意力仅关注序列的局部信息而非全局信息，通过层层堆叠放大模型的上下文感受野。这样就可以把计算复杂度控制在一个明确的范围内，避免随序列长度的无限制增长。该方法常应用在 Mistral 中[12]。此外，还有其他一些替代注意力机制的方法，如上文介绍的线性注意力机制与 Mamba 等模型等。

1.4 多模态能力

多模态大模型可以分为多模态理解大模型和多模态生成大模型。多模态理解大模型输入多模态信息，输出中包含文本信息；多模态生成大模型则相反，其输入中包含文本信息，输出为多模态信息。

自从经典的 BLIP2 模型出现后，多模态大模型设计趋势逐渐稳定，具体如图 1 所示。每种模态均有对应的编码器和



▲图1 多模态大模型结构示意图

解码器。其中，编码器负责接收对应模态信息的输入，并将其编码到语义空间的向量中；解码器则负责从语义空间的向量中解码出对应模态的输出；跨模态对齐单元则负责不同模态语义空间的匹配对齐。这种架构可以很方便地实现多种模态的混合输入和输出。

1.4.1 多模态理解大模型

目前，视觉语言类的多模态理解大模型和多模态生成大模型的发展最为迅速。视觉语言理解大模型（图生文），即对视觉编码器+模态对齐+大语言模型解码器进行组合训练。代表性的工作包括CLIP[13]、BLIP-2[14]、LLaVA[15]和InternLM-XComposer2[16]等。

BLIP-2由预训练好的、冻结参数的视觉模型（CLIP训练的ViT-L/14、EVA-CLIP训练的ViT-g/14）、文本模型（OPT、FlanT5），以及所提出的可训练的Q-Former构成。Q-Former是一个轻量级Transformer，它使用一组可学习的Query向量，从冻结的视觉编码器中提取视觉特征，来对齐文本和语言两个模态的差距，从而把关键的视觉信息传递给大语言模型（LLM）。

LLaVA成功地验证了少量高质量的数据能使模型拥有很强的图文生成能力，其由3部分组成：CLIP预训练模型中的视觉编码器ViT-L/14、一个线性投影层和一个大语言模型LLaMA。LLaVA以图-文对（LAION、CC3M、COCO）数据集为基础，使用ChatGPT/GPT-4来构建指令跟随精调数据集。

零一万物开源Yi-VL多模态大模型[17]也是采用LLaVA架构，使用了Yi-34B-Chat模型，改进了微调训练方法，提高了Yi-VL无缝集成和解释视觉+语言多模态输入的能力。

1.4.2 多模态生成大模型

视觉语言生成大模型（文生图）的解码器部分以扩散模型为主，代表性工作包括Stable Diffusion[18]、DiT[19]、Sora[20]等。

Stable Diffusion的组件和模型组成为：文本编码器，将文本信息转换成数字表示，以捕捉文本中的想法；图像信息创建者，在隐空间中逐步处理扩散信息，以文本嵌入向量和由噪声组成的起始多维数组为输入，输出处理的信息数组；图像解码器，使用处理后的信息数组绘制最终的图像。

Sora是OpenAI发布的文生视频的多模态模型，和Runway、Stable Video Diffusion及PIKA等已有模型相比，其视频生成能力有大幅提升。Sora模型内部分成3个部分：第1部分是变分自动编码器（VAE），包括编码器和解码器两个部分，其作用是对视频进行压缩和解压缩。生成视频的过程是在压缩后的低维隐空间进行计算，相对于直接从原始的像素空间计算，减少了数百倍的计算量。第2部分是基于Transformer的扩散模型，其作用是在隐空间通过迭代降噪过程生成视频。第3个部分是语言大模型作为编码器，其作用是将用户的Prompt编码为一个隐空间的表示，使其在生成视频时内容与文本描述一致，从而使视频内容和用户的Prompt一致。

2 模型训练

GPT开创了生成式预训练方法之后，两阶段训练（任务无关的预训练阶段和任务相关的精调训练阶段）大模型成为主流。预训练阶段的目的是为模型注入大量通识知识，精调训练阶段的目的是提升模型完成特定任务的能力。在这种方式下，仅需少量精调数据即可以让预训练模型具备完成新任务的能力，相比于之前为每个任务端到端完整训练模型，极大节省了训练数据和算力。

随着开源预训练大模型的出现以及应用场景日益复杂，上述两阶段训练方法已不再满足需求，因此出现了更多的训练阶段，如图2所示。由于开源预训练大模型通常使用公开可获得数据训练，专业领域知识不足，使用私域数据对模型再次进行增量预训练可以有效灌注专业知识。大模型在使用过程中需要避免生成各类有害信息，因此在模型完成任务相关的精调训练之后，增加了对齐训练阶段，这样可以使模型输出更加符合人类的价值观。

训练高性能的商用大模型，涉及数据处理、预训练、精调训练、安全等关键技术。

▲图2 大模型多阶段训练过程

2.1 数据多样性与数据质量

数据多样性对于模型能力的全面性来说至关重要，模型学习不同的能力，就需要对应的训练数据。例如GPT-3训练数据的构成既包含Common Crawl这类种类丰富、总量庞大的互联网数据，又包含维基百科、书籍这类的高质量数据。

数据质量是模型能力的决定性因素之一。如今，越来越多的研究人员倾向于使用更少的高质量数据而非海量的低质量数据来训练模型。少而精的高质量数据在大幅降低训练成本的同时，还有可能获得更高的模型性能。

因此，如何评价数据质量成为一个关键问题。2023年底的Ziya2[6]给出了一套非常系统的评估标准，该标准综合了程序评估和人工评估。程序评估涵盖了启发式规则、语言模型、统计指标等方法，并对不同的指标赋予了不同级别。系统化、程序评估与人工评估结合，为数据质量评估方法指明了发展方向。

中兴通讯结合业界研究成果，总结提炼出一套完整的数据管理及处理方法（如图3所示）。在数据管理方面，中兴通讯强调“分级分类”。数据分级是将所有训练数据按质量从低到高分为1—5级，不同质量级别的数据应用于预训练的不同阶段。例如，1级数据为有害数据，严重影响模型性能和安全性，需要从训练数据中予以过滤清除；2级数据为普通网页数据，仅用于模型训练过程中的warm-up阶段；3级数据为高质量网页、书籍、代码数据，用于为模型建立通识知识；4级数据为更高质量的专业书籍、论文、教材、试题数据，用于提升模型的专业性；5级数据为最高质量精调数据，用于大幅提升模型在各种任务评测中的表现。数据分类则是从为模型赋能的角度，将模型能力分为上百个类别，为对应的训练数据建立主题，从而实现数据能力画像。为了实现上述数据分类分级管理，中兴通讯开发一套完整的数据处理框架，如图3所示。

2.2 预训练

生成式预训练是一种让大模型学习世界知识的有效方法。这种方法的主要优势在于：它属于无监督学习方法，无须对数据进行标注就可以得到海量训练数据；只需让模型学习正确预测语料中的词，即可令模型理解语料中所的蕴含知识。

大模型训练算力需求的增长速度远超摩尔定律，因此所需要的并行计算节点数不断增加，算力和显存是大模型训练的主要瓶颈。为了打破设备的算力和显存限制，目前有3个最主要的技术方向：1）以张量并行（TP）、流水线并行（PP），数据并行（DP）构成的3D并行加速，其主要原理是根据硬件集群的特征，如节点数、单节点算力、内存大小、节点间互联带宽及时延等，对计算任务进行不同维度的拆分，以最优化利用硬件资源；2）以ZeRO（零冗余）系列为代表的显存优化技术，其主要原理是尽可能将数据保存到内存从而减少对显存的需求（以增加带宽需求为代价）；3）以Flash Attention为代表的底层算子优化，其主要原理是将多个算子计算进行融合从而最大限度降低“内存墙”对计算效率的影响。

上述算法实现涉及复杂工程优化，因此并行训练框架极为重要。Megatron-LM[21]是由英伟达深度学习应用研究团队开发的大型Transformer语言模型训练框架，其在对TP的支持方面处业界领先。DeepSpeed是微软提出的并行训练框架，其主要优势是支持ZeRO[22]和Checkpoint[23]显存优化技术，对训练显存紧张的场景更友好。FairScale是由Facebook提出的一个用于高性能和大规模训练的PyTorch扩展库，支持全切片数据并行（FSDP），这是扩展大模型训练的推荐方式。

2.3 精调训练

大模型在各类任务中具有泛化能力，然而直接应用预训练模型往往并不能满足所有场景的需求，这就引出了一个关

▲图3 中兴通讯数据处理框架

键技术——精调训练，即针对特定任务或应用场景，在预训练模型的部分或全部参数上进行进一步的学习与优化，提升模型在特定任务上的遵循指令能力、问题解决能力、特定表达方式能力等，从而提高其在该任务上的精度和专业性。

精调训练主要分为全量精调和低资源精调两大类别。其中，低资源精调主要指的是LoRA[24]、prefix-tuning以及P-tuning等方法。这些方法对模型局部进行微调或者冻结一部分参数进行微调。其中，LoRA方法效果最佳。该方法通过低秩近似对预训练模型的部分权重矩阵进行更新，在降低存储成本和计算复杂度的同时，实现模型对目标任务的快速适应。全量精调则在训练过程中会对整个模型的参数进行优化和更新，不仅对计算资源和存储资源要求更高，也更容易出现过拟合、丧失通用性等问题。但全量精调拥有更高的上限，通过适当的训练优化来增强模型的泛化性。全量精调方法能够得到比低资源精调方法更优秀的性能。

由于“对齐税”问题，精调训练在提升特定任务表现的同时会影响模型在其他任务中的总体表现。因此，使用“少而精”的精调训练数据，运用Dropout等技术可以防止过拟合，对平衡大模型的泛化能力和任务适应性尤为重要。

2.4 对齐训练

大语言模型的一些不良行为（例如，不真实的回答、谄媚和欺骗）激发了业界对人工智能对齐领域的深入研究。AI对齐旨在使人工智能系统的行为与人类的意图和价值观相一致，在通往AGI的道路上，AI对齐无疑是安全打开“潘多拉魔盒”的密钥。

对齐训练主要应用强化学习算法，其基本方法是首先基于人类标注数据训练评分模型，然后再基于评分模型运行强化学习算法，通过引导模型获得更高的评分，使其输出更符合人类标准。OpenAI首先提出的基于人类反馈的强化学习（RLHF）就是利用人类的反馈来优化模型的输出，使其更符合人类的偏好和价值观。这不仅能够提升语言模型性能，也能提升安全性和有用性。相比于RLHF涉及多个模型和不同训练阶段的复杂过程，拒绝采样（RS）[25]则更为简洁有效。RS是指让一个模型针对同一个prompt生成K个不同答案，然后利用奖励模型为这K个答案打分，选出最优的答案后，再用最优问答样本对原模型进行监督微调，以增强模型能力。直接偏好优化（DPO）[26]也是RLHF的替代方案之一。DPO利用奖励函数和最优策略之间的映射关系，将约束奖励最大化问题转换为单阶段的策略优化问题。DPO算法因无须拟合奖励模型，且无须在微调期间从LM采样或执行重要的超参数调整，从而实现了稳定性高、性能强且计算量轻等优秀表现，大大简化了实施和训练过程。

中兴通讯的星云大模型在RLHF这一框架的基础上，通过设计质量评优模块并作为对样本自动打分的奖励模型，同时结合RS拒绝采样选取最优样本，经过多次迭代生成更多的高质量代码数据。这使得大模型从这些高质量数据中不断训练优化，从而提升了生成代码的质量。星云大模型的HumanEval@Pass1可以达到83.6分。

2.5 合成数据和自我学习

随着模型规模不断增大，所需的训练数据也更多，最终将会耗尽所有自然产生的数据。因此，基于人工合成数据来训练模型已经成为一个热门研究方向。

借助已有大模型合成精调数据是业界最常用的方式，通过将设计好的指令，并让大模型来获取对应的回答，可以节省大量的人力。WizardLM[27]和WizardCoder[28]根据种子指令分别沿着添加约束、深化、具体化、增加推理步骤、复杂化以及突变等6种演化方向来演化指令，并控制指令的难度和复杂程度，经过多轮不同方向的演化得到大量不同类型的指令，再利用已有大模型生成精调数据，最终取得了不错的精调效果。Magicoder[29]通过在GitHub上随机获取1~15行代码作为种子代码片段，让已有大模型根据提供的种子片段来生成编程问题，这大大丰富了代码精调数据的类型，从而更好地激活大模型的能力。

中兴通讯的星云大模型采用自我学习的方法生成测试用例数据，主要的步骤是通过对同一代码多次生成对应的测试用例，根据编译、覆盖率等情况选出符合要求的高质量数据，再使用这部分高质量数据对模型进行精调训练，从而取得良好效果。在基于HumanEval数据集制作的UTEval数据集上，星云大模型测试用例生成的编译成功率、用例通过率、测试覆盖率指标已经超过GPT4-turbo。

2.6 模型安全性

大模型在提供强大的自然语言处理能力的同时，也带来了安全和隐私方面的挑战。在训练阶段，大模型面临的风险主要源于训练数据。数据可能包含有害内容或设计歧视、侵权、毒性文本。这不仅会影响模型的输出质量，还可能使模型学习并复制这些不当行为。此外，训练数据还可能遭到恶意投毒，即意图故意降低模型的性能或引导模型做出不当行为。

训练阶段引入的常见漏洞包括后门漏洞和数据投毒。针对以上漏洞，可以实施训练语料库治理手段，具体包括：

1）语言识别和解毒：通过自动化工具识别并清除训练

数据中的有毒语言或偏见表达，确保输入数据的质量和安全性。

2）除杂和去标识化：从数据集中移除无关的信息和个人标识信息，减少隐私泄露的风险，并防止模型学习到不必要的个人信息。

上述防御的实现方法可以分为黑盒防御和白盒防御两大类。黑盒防御通常采用基于查询的模型诊断方法，检测模型是否被嵌入了后门中。白盒防御包括通过微调删除后门[32]、模型修剪[33]和通过检查激活来检测后门[34]等方法。例如，Fine-mixing[35]是一种旨在防止在微调模型中出现后门的方法。CUBE防御技术[36]利用了一种称为HDBSCAN的密度聚类算法来准确识别数据集中的簇，来区分包含毒害样本和干净数据的簇。通过利用HDBSCAN的能力，CUBE旨在提供一种有效区分正常数据和有毒数据的方法。

3 模型推理和优化技术

大模型以其强大的理解和生成能力，正在深刻改变我们对人工智能的认知和应用，但其高昂的推理成本也阻碍了技术落地。因此，优化大模型的推理性能成为业界研究的热点。

推理性能优化主要以提高吞吐量和降低时延为目的，关键技术可以划分为：内存管理、算子融合、模型压缩、并行推理、服务调度优化、推理安全及新兴技术。

3.1 内存管理

KV Cache是大模型推理性能优化最常用的技术。该技术在不影响任何计算精度的前提下，通过空间换时间，大幅提升推理性能。Transformer解码器使用自回归产生输出，即每次推理只会预测输出一个token，执行多次后完成全部输出。前后两次的输入只相差一个token，这就存在大量重复计算。KV Cache技术将每个token可复用的$\boldsymbol{K}$和$\boldsymbol{Q}$向量结果保存下来复用，将计算复杂度从$O(n^2)$降低为$O(n)$。

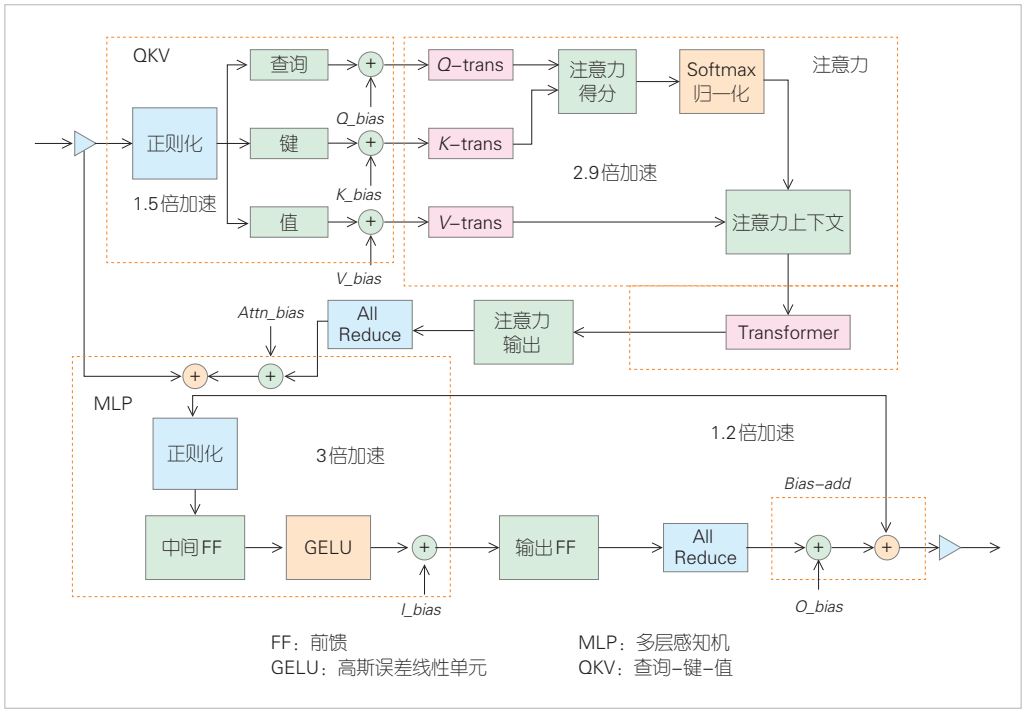
Paged Attention将操作系统中的分页内存管理应用到KV Cache的管理中，这节约了60%～80%的显存，从而支持更大的batch-size，将吞吐率提升了22倍。具体来讲，Paged Attention首先将每个序列的KV Cache分成若干块，每个块包含固定数量token的键和值，然后计算出当前软硬件环境下KV Cache可用的最大空间，并预先申请缓存空间。在推理过程中，通过维护一个逻辑块到物理块的映射表，使多个逻辑块对应一个物理块，并使用引用计数标记物理块被引用的次数，从而实现将地址不连续的物理块串联在一起统一管理。

RadixAttention是一种自动键值缓存重用技术，该技术可以在完成生成请求后不直接丢弃键值缓存，而是在基数树中保留提示和生成结果的键值缓存，从而实现高效的前缀搜索、插入和驱逐。该技术在多轮对话场景可以极大地降低首字时延。

3.2 算子融合

算子融合是深度学习模型推理的一种典型优化技术，旨在通过减少计算过程中的访存次数和统一计算架构（CUDA）Kernel启动耗时，达到提升模型推理性能的目的。

以HuggingFace Transformers库LLaMA-7B模型为例，该模型有30个类型共计2 436个算子，其中aten::slice算子出现频率为388次。大量小算子的执行会降低图形处理器（GPU）利用率，最终影响推理速度。针对Transformer结构



▲图4 Transformer层中的算子融合示意图

特点，算子融合主要分为4类：归一化层和QKV横向融合，自注意力计算融合，残差连接、归一化层、全连接层和激活层融合，偏置加法和残差连接融合。

中兴通讯在vLLM（开源项目名）上实现了针对多查询注意力结构的QKV通用矩阵乘法（GEMM）横向算子融合，以及多层感知机（MLP）中的全连接层+激活融合，性能明显提升，见表1和表2。上述算法的相关代码实现已合入vLLM社区。

由于算子融合一般需要定制化来实现算子CUDA kernel，因此对GPU编程能力要求较高。随着TensorRT、OpenAI Triton、张量虚拟机（TVM）等框架引入编译器技术，算子融合的自动化或半自动化逐步实现，这降低了GPU编程难度，取得了较好的效果。

3.3 模型压缩

模型压缩技术是指在不影响模型精度的情况下，通过缩小模型规模和计算量来提高模型的运行效率。常见的深度学习模型压缩技术包括模型剪枝、知识蒸馏、模型量化和模型分割等。其中，模型量化在这些技术中最具实用性。

SmoothQuant[39]是典型的8 bit LLM量化方法。根据激活量化方式的不同，SmoothQuant提供了3种量化方式：per-tensor static、per-tensor dynamic和per-token dynamic。这3种模型的精度依次提升，计算效率依次降低。SmoothQuant的研究人员观察到，不同的标记在它们的通道上展示出类似的变化，引入了逐通道缩放变换，有效地平滑了幅度，这使得模型更易于量化。激活感知权权重化（AWQ）[38]和生成式预训练Transformer（GPTQ）[37]是典型的仅权重量化的方法，且权重量化的是group粒度。GPTQ提出了一种基于近似二阶信息的新型分层量化技术，使得每个权重的比特宽度减少到3或4位。与未压缩版本相比，该技术几乎没有准确性损失。AWQ[18]的研究人员发现，对于LLM的性能，权重并不是同等重要的，仅保护1%的显著权重可以大大减少量化误差。在此基础上，AWQ采用了激活感知方法，这在处理重要特征时起着关键作用。该方法采用逐通道缩放技术来确定最佳缩放因子，从而在量化所有权重的同时最小化量化误差。

中兴通讯提出了SmoothQuant+[40] 4 bit权重量化训练后量化（PTQ）算法。不同于AWQ对单个层搜索量化参数，SmoothQuant+对整个模型搜索量化参数，并对整个模型进行同一个参数平滑激活，这样能够从模型整体减少量化误差，且搜索效率更高。SmoothQuant+在LLaMA系列模型可以得到比AWQ更好的精度（见表3），同时在性能上也优于AWQ，对应的推理核已开源。

随着大模型上下文长度的增加，KV Cache占用的显存将超过权重和激活，因此对KV Cache进行量化可以显著降低大模型在长上下文推理时资源占用，从而允许系统支撑更多的并发请求数和吞吐率。中兴通讯在生产环境中使用INT4权重量化和KV Cache FP8量化，显存节省了70%，吞吐率提升了2.8倍，推理成本降低75%左右。

3.4 并行推理

当大模型参数量超过单一计算设备所能容纳的上限时，则需要使用分布式并行推理技术。并行推理可以使用模型并行和流水线并行，而模型并行由于可节省显存资源、可降低单用户时延等优势，成为首选的并行方式。

业界最流行的模型并行方案来自Megatron-LM[21]，它的开发者针对Self-Attention和MLP分别设计了简洁高效的模型并行方案。MLP的第1个全连接层为Column Parallel，第2个全连接层为Row Parallel，之后是1次AllReduce规约操作。Self-Attention在计算Query、Key和Value向量时执行Column Parallel（按注意力头个数均分到每个GPU），之后将注意力得分做空间映射时执行Row Parallel，之后是1次AllReduce规约操作。除此之外，LLM模型中的Input Embedding采用

▼表1 StarCoder-15B在A100-40GB上测试查询-键-值融合

批大小V样本数	输入长度/token数	输出长度/token数	注意力基线/s	注意力融合/s	Speedup/%
10	1 024	1 024	27.17	22.80	19
30	1 024	1 024	39.08	37.48	4

▼表2 StarCoder-15B在A100-40GB上测试FC+激活融合

实测的TFLOPs	*B*=1，*M*=1，*K*=6 144	*B*=4，*M*=1，*K*=6 144	*B*=16，*M*=1，*K*=6 144	*B*=64，*M*=1，*K*=6 144	*B*=256，*M*=1，*K*=6 144
基线	0.3	1.2	4.7	17.6	30.3
融合MLP	0.3	1.2	4.9	19.1	47.1
加速率	0.0%	0.0%	4.3%	8.5%	55.4%

FC：全连接层　　MLP：多层感知机
TFLOPs：每秒钟浮点计算万亿次数
注：*B*代表Batchsize；*M*和*K*表示矩阵乘法中的两个维度，*M*恒为1（解码阶段），*K*恒为6 144

▼表3 CodeLLaMA INT4量化在HumanEval上的性能

HumanEval↑	7B/%	13B/%	34B/%
FP16(baseline)	35.98	35.98	51.22
RTN	36.59	33.54	46.34
AWQ	35.98	31.71	50.61
SmoothQuant+	**35.98**	**37.80**	**53.05**

AWQ：激活感知权权重化　　RTN：直接量化
FP16：16 bit原始精度

Row Parallel，Output Embedding采用Column Parallel；Dropout/Layer Norm/Residual Connections等操作都没有做并行拆分。

节点间带宽对模型并行效率有较大影响，高速串行计算机扩展总线标准（PCIE）的理论带宽为32～64 Gbit/s，通常可以满足大模型并行推理需求。模型参数量越大、Batchsize越大，节点间的通信效率就越高，使用模型并行获得的收益越明显。

3.5 服务调度优化

服务调度优化主要考虑的是系统同时为多个用户服务时如何尽可能地提升资源利用率，相关优化主要包括Continuous Batching、Dynamic Batching和异步Tokenize/Detokenize。其中，Continuous Batching和Dynamic Batching主要围绕提高可并发的Batchsize来提高吞吐量，异步Tokenize/Detokenize则通过多线程方式将Tokenize/Detokenize执行与模型推理过程时间交叠，从而实现降低时延目的。

Continuous Batching和Dynamic Batching的思想最早来自文献[41]。Continuous Batching的原理是：将传统batch粒度的任务调度细化为step级别的调度，这解决了不同长短序列无法合并到同一个batch的问题。调度器维护3个队列，分别为Running队列、Waiting队列和Pending队列。队列中的序列状态可以在Running和Waiting/Pending之间转换。在生成每个token后，调度器均会立刻检查所有序列的状态。一旦序列结束，调度器就将该序列由Running队列移除并标记为已完成，同时从Waiting/Pending队列中按先来先服务（FCFS）策略取出一个序列添加至Running队列。

Batching优化技术可有效提升推理吞吐量，目前已在HuggingFace TGI、vLLM、TensorRT-LLM等多个推理框架中实现。

3.6 推理阶段的安全漏洞和防护

推理阶段的安全漏洞不仅可能危及用户的隐私和数据安全，还可能被恶意利用。下文中，我们将详细介绍推理阶段可能遇到的漏洞及其原理，以及如何通过一系列防护措施来提高大模型的安全性。

推理阶段常见的攻击手段包括：1）越狱攻击：通过某些方式使大模型产生退化输出行为，诸如冒犯性输出、违反内容监管输出，或者隐私数据泄漏的输出。2）提示注入攻击：通过注入恶意指令，可以操纵模型的正常输出过程，导致模型产生不适当、有偏见或有害的输出。3）成员推理攻击：通过分辨一条数据是否属于模型的训练集，可以使攻击者获得训练集数据所共有的特征，在训练数据集敏感的应用场景中（例如，生物医学记录和位置跟踪），成功的成员推理攻击会导致严重的隐私泄露和安全威胁。

针对以上漏洞，我们可以实施的防护手段包括：1）越狱攻击防御：基于预处理技术如指令净化、关键词过滤、恶意模型检测等，检测并清理输入或输出中的有害信息，通过语义内容过滤防止大模型生成不受欢迎或不适当的内容，这可以有效减轻潜在的危害。2）提示注入攻击防御：重新设计提示指令是一类预防提示注入攻击的方法，例如re-tokenization[42]、paraphrasing[42]等。re-tokenization是为了打破在提示中注入的恶意指令（如任务忽略文本、特殊字符和虚假响应等）的顺序。paraphrasing可以干扰注入数据的序列，如注入指令及特殊字符插入，减弱提示注入攻击的有效性。还有一些防御方法是基于检测的，侧重于确定给定数据提示的完整性，如困惑度检测就是一种基于提示的检测方法，它向数据提示添加信息或指令。这就会降低质量，并导致困惑度增加。因此，如果数据提示的困惑度超过指定阈值，我们则认为数据提示存在问题。3）成员推理攻击防御：Salem等提出了第一个针对成员推断攻击的有效防御机制[43]，具体方法包括Dropout和模型堆叠。Dropout被定义为随机删除一定比例的神经元连接，可以减轻深度神经网络中的过拟合，这是成员推断攻击实现中的一个重要因素[43]。模型堆叠防御背后的思想是，如果目标模型的不同部分使用不同的数据子集进行训练，那么整体模型有望表现出更低的过拟合倾向。差分隐私[44]也是目前对成员推理攻击最突出的防御手段之一，通过给模型的输出增加噪声，使得攻击者无法根据输出结果在统计上严格区分两个数据集。

3.7 新兴技术

我们将传统优化技术引入大模型推理的同时，也在探索从大模型自回归解码特点出发，通过调整推理执行过程来进一步提升推理性能。并行推测解码作为新兴的推理技术，可以在不损失精度的前提下提高推理速度。

投机采样[45]是一种并行推测解码算法，开创了“小成本生成+大模型验证”的推理技术路线。该算法在已有大模型的基础上，引入一个小模型执行串行解码来提升速度，原大模型执行并行评估采样，保证生成质量，这在保证精度一致性的同时降低了大模型解码的次数，进而提升了推理效率。例如，我们基于HuggingFace Transformers库实现该算法，使用Pythia-6.9B作为基础模型，Pythia-160M作为近似模型，在A100-PCIE-40GB下可以取得3.9倍的推理速度提升。

由于投机采样算法的巨大潜力，有多项工作在其基础上

研究改进。例如，美杜莎头[46]解码无须引入新的模型，而是在原始模型上训练出多个解码头，每个解码头可并行预测多个后续tokens，然后使用基于树状注意力机制并行处理，筛选出合理的后续tokens。前向解码[47]不对原始模型做任何改变，将自回归解码视为求解非线性方程，并采用Jacobi迭代法进行并行解码。

投机采样的推理方式并不适用于所有的应用场景。例如，在文学艺术类的诗词等应用场景，大小模型生成的结果概率分布相差较大；对于代码生成的场景，投机采样比较适合。随着业界研究的深入，投机采样会成为大语言模型推理的必备优化技术。

4 大模型的企业级应用

相较于其他通用技术，AI技术正在以历史罕见的速度高速发展，与人类水平相当的AGI可能在未来10年内出现，相比于此前预计的30～50年要大幅提前。人类社会正在从信息时代加速走向智能时代，旧的商业竞争格局逐步解构，新的机遇不断产生。夯实数字化、加速智能化，以日益强大的人工智能技术强化产品竞争力，提升企业运营效率，将是每个企业把握智能化时代先机重大而紧迫的任务。

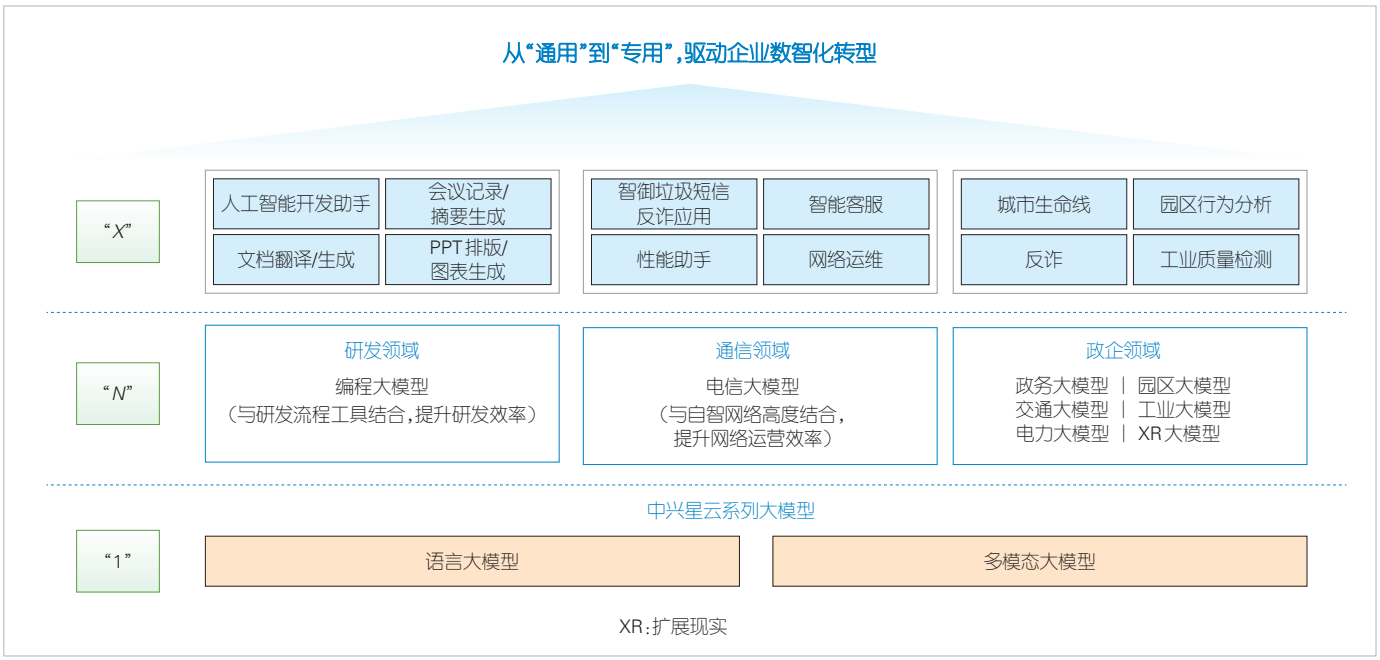
然而，在企业中用好大模型并非易事。首先，大模型技术本身成本高昂，若非有对应的高价值场景，则大模型的应用也难以为继。其次，大模型技术门槛较高，对企业自身的数据治理水平、算力规模、团队能力都有较高的要求。最后，大模型本身仍在快速发展之中，企业应用规划必须具备足够的前瞻性和预测性，才能把握好应用节奏，做到既不掉队，也不过度投资导致大量浪费。

我们认为，大模型的企业级应用包含场景、工程、算法3要素。首先是场景的甄别、价值排序和规划，从价值、技术可行性、资源投入等维度确定“主战场”。然后要有效应对大模型在工程上的复杂性，将算力基础设施、软件平台、框架、工具、能力整合为大模型技术平台，提升开发效率和降低部署成本。最后，紧跟算法发展趋势，将拿来主义和自主创新结合起来，不断推出更强大的模型，赋能各个应用。

4.1 大模型规划

企业需要根据自身的业务特点，梳理出大模型应用的总体规划。结合前文所述的预训练+精调的训练范式，企业适合采用基础模型和领域模型的分层规划。基础模型为预训练大模型，提供通用能力。在其基础上，通过后预训练、精调等技术手段开发出面向特定应用场景的领域模型应运而生。最后，基于这些领域模型面向各场景打造了不同应用。这种分层规划架构，已被多个企业所采用。

以中兴通讯为例，大模型整体规划可以描述为1+*N*+*X*（具体如图5所示）：即1系列基础大模型，确保自主可控和数据资产安全；*N*个领域大模型，通过加入领域KnowHow增量预训练、精调等方式，提高专业性能力；*X*个场景应用，利用大模型，开发出各种场景的应用。



▲图5 中兴通讯星云系列大模型

基础模型的规划以技术为导向。星云系列大模型拥有十亿到千亿不同规模，分别匹配中心云、边缘云、终端部署不同场景下的算力及资源条件；支持图像、表格、文字、代码等多模态的输入输出，可支持大部分企业应用场景；具备前文所述的模型推理优化技术，在保持模型准确率的前提下，可降低70%以上的推理资源需求。

领域模型的规划以价值为导向。例如，研发大模型主要用于企业内部数万名研发人员的研发提效，在文档、代码、测试用例生成等方面能够显著提升效率。目前，中兴通讯3%的代码由AI生成，2024年底预计提升至10%～20%。

这种规划可以将技术和价值紧密结合。基于领域模型打造的应用源源不断提供新的业务和用户数据，这些数据作为训练语料进一步增强领域模型的场景化服务能力。同时，基础模型提供的通用能力则有助于提升领域模型的整体表现。

4.2 大模型应用架构

大模型的研发、部署及应用是极其复杂的工程。合理的架构允许技术组件的解耦和复用，能够有效控制工程的复杂度，因此对大模型应用的降本增效极其重要。

中兴通讯在实践中不断优化迭代的大模型应用架构具体如图6所示。

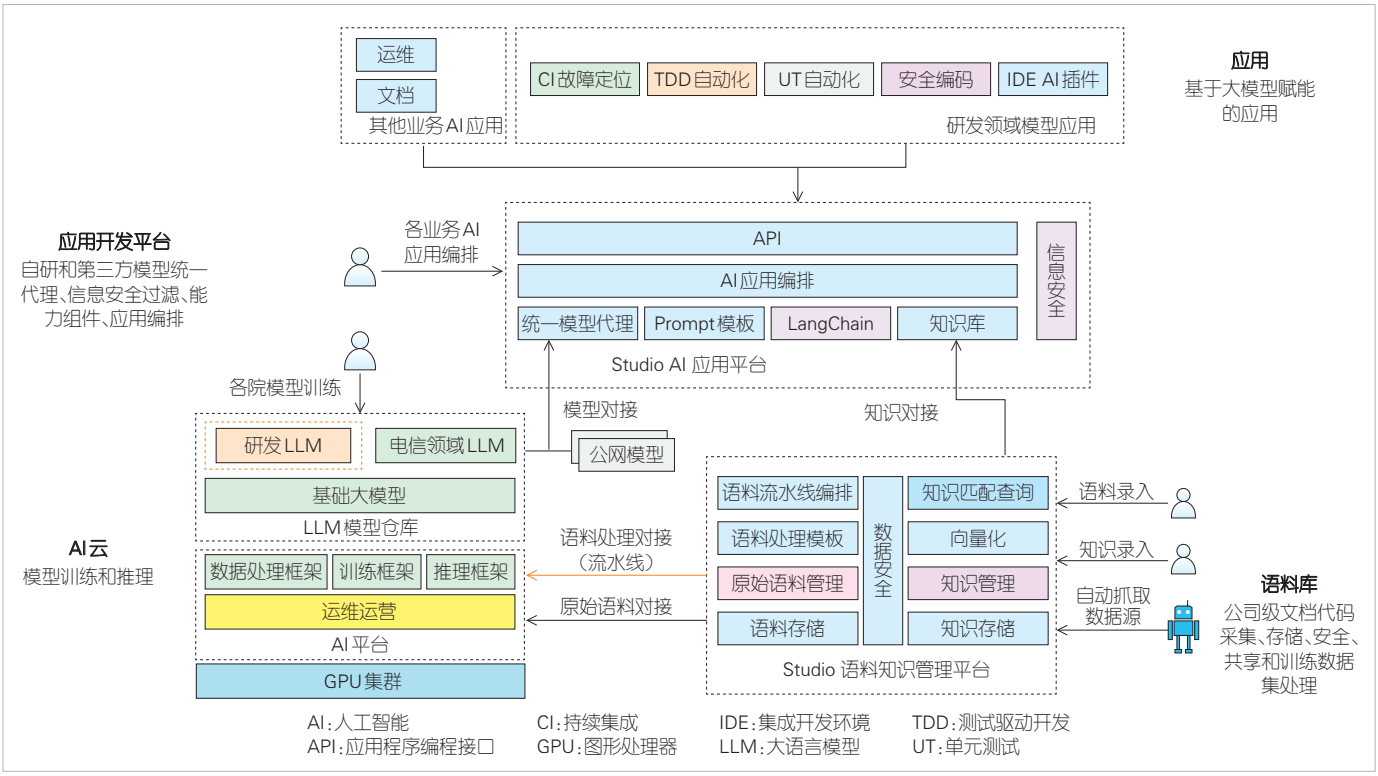
语料库是基于中兴通讯数据智能分析平台（DAIP）大数据平台构建的，能够提供PB级数据存储和处理能力。其主要具备3个功能：1）原始数据采集、存储、处理和共享，实现了互联网数据和公司内部数据的统一；2）基于Spark的训练语料处理，将原始数据转化为可供大模型训练使用的语料；3）基于EBase向量数据库的知识库，将原始语料向量化后存储到数据库中，作为Agent外挂知识库。

AI云基于AiCloud构建，用于管理公司大规模异地、异构算力集群（英伟达、壁仞等），通过算子优化和通信优化，大幅提升基础设施利用效率，同时还提供数据处理、训练、精调、评估、优化端到端的工作流管理，支持高效训练和部署千亿参数以上级别的大模型。前文所述所有的基础大模型、领域大模型都在AI云上训练和部署。

应用开发平台可以实现基础能力组件的集成和编排，实现无代码方式构建Agent。大模型应用平台有超1 000个大模型应用，每日活跃用户万人以上。此外，平台还可以支持模型统一代理、信息安全过滤、Prompt共享等主要功能，这些功能对于提升用户使用体验、保障公司信息安全起到关键作用。

4.3 大模型应用技术

众所周知，大模型的幻觉问题短时间难以解决，因此保



▲图6 中兴通讯企业级大模型应用架构

证大模型在应用中的正确性和可靠性成为一个技术难题。此外，大模型还有知识难以更新、经常无法正确处理数学逻辑推理等问题，这些都制约了大模型的应用。

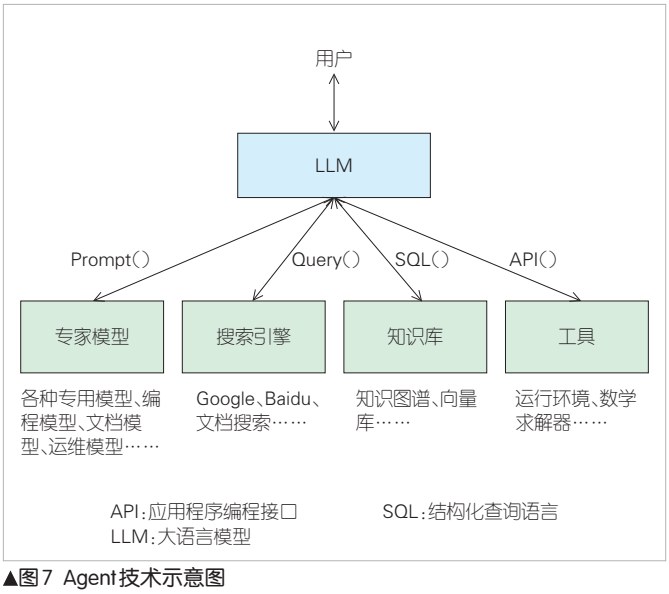
业界逐渐认识到使用单一大模型无法解决上述所有问题，于是尝试使用多个辅助模型、外部知识库、搜索引擎、专业工具等多种手段协同解决大模型实用问题，这些手段最终形成了Agent技术。目前，业界普遍认为Agent是大模型应用技术的发展方向。

4.3.1 Agent技术

Agent并非单一技术，而是一个框架（如图7所示），将大模型与专家模型、搜索引擎、知识库、工具等众多组件集成在一起，通过组件之间的协作，共同完成用户指定的任务。

大模型作为Agent核心组件，需要理解用户意图、拆分任务、流程控制、汇总信息，并生成结果后返回给客户。例如，对于专业性问题，如代码生成，大模型可以根据用户请求生成Prompt与代码模型的交互，并将代码模型生成结果反馈给用户；对于信息查询类问题，大模型根据用户提出的问题转化为Query，并浏览搜索引擎返回的页面链接，再将相关页面的信息总结提炼后返回给客户；对于数学计算等问题，则可以将问题转化为应用程序编程接口（API），调用外部工具完成问题求解。

相对于传统的软件，Agent技术的本质是在控制面用大模型替代固定的程序，从而可以处理新的场景，极大地提升系统灵活性。Agent是智能化时代的软件，是软件发展的下一形态。



▲图7 Agent技术示意图

4.3.2 RAG

在对于需要精确、实时、涉及领域专业知识的任务中，大模型所面临的幻觉频出、信息过时、专业领域深度知识缺乏以及推理能力弱等痛点，成为亟待解决的问题。

为应对上述挑战，RAG技术应时而生，它是Agent的一项关键技术。其核心思想是在生成响应之前，通过信息检索的技术，先从外部数据库中检索出和用户问题相关的信息，然后结合这些信息，LLM生成结果。RAG技术主要包括3个基本步骤：1）索引。索引阶段是构建RAG的准备步骤，类似于ML中的数据清理和预处理步骤。通常，索引阶段包括：收集数据、数据分块、向量嵌入和向量数据库存储。2）检索。检索阶段是构建RAG的主要步骤，由查询向量嵌入和相似度检索组成。3）生成。生成阶段由提示工程构成。

RAG是一项前景广阔的新兴技术，有效提升了大语言模型的生成内容准确率和时效性。RAG技术正在快速发展，不断出现更优秀的应用，如ChatPDF、Lepton Search等应用，RAG为未来的通用人工智能提供更大的可能性。

4.3.3 插件技术

插件技术允许大模型与外部应用协作，例如OpenAI的ChatGPT有数千个插件，从而大幅扩展了大模型的能力和应用领域。

CodeInterpreter使得数据分析变得更加简单，通过与用户的对话就可以处理庞大的数据。CodeInterpreter在本质上是将大语言模型生成的代码放到一个安全的环境中执行，这个环境通常被称为沙盒。在这个沙盒中，开发者可以提前配置所需的依赖库和环境变量，以确保代码能够正常运行。CodeInterpreter的主要功能是执行由大语言模型生成的代码，并返回结果。

Function Calling可以为B端用户开发的APP提供强大的功能，改变交互方式，从图形用户界面（GUI）向自然用户界面（NUI）、语音用户界面（VUI）转变，让客户有更自然的体验。Function Calling的关键技术点是大模型的函数选择能力、函数调用能力、结果解释能力、异常处理能力。

Threading则提供了持久保存且无限长的上下文，帮助用户建立起更全面的客户画像，让客户使用APP时更加得心应手。Threading的关键技术点主要包括更长的上下文理解能力以及向量数据库中检索、总结能力。

中兴通讯基于星云大模型成功实现了上述插件。未来，星云大模型将朝着能力更强、上下文更长、使用体验更好的方向发展，使大模型能够为各行各业赋能。

5 结束语

随着ChatGPT热度的逐渐褪去，对大模型的投资也逐渐趋于理性。大模型如何产生真正的商业价值成为全行业都在思考、探索的问题。一方面，随着大模型规模的不断增加，模型能力在提升的同时，算力成本也在不断飙升，这给大模型长期可持续发展带来了不确定性，因此以实现更低成本算力和更高效率算法为目标的核心技术亟待突破；另一方面，以Agent为代表的应用层技术在解决大模型固有问题并大幅拓展应用边界的潜力还未被完全发掘，业界对Agent的关注度持续上升。大模型机遇与挑战并存，加速发展的趋势在中长期不会改变。

参考文献

[1] VASWANI A, SHAZEER N, PARMAR N, et al. Attention is all you need [C]//Proceedings of the 31st International Conference on Neural Information Processing Systems. ACM, 2017: 6000–6010. DOI: 10.5555/3295222.3295349

[2] HOCHREITER S, SCHMIDHUBER J. Long short–term memory [J]. Neural computation, 1997, 9(8): 1735–1780. DOI: 10.1162/neco.1997.9.8.1735

[3] RADFORD A, NARASIMHAN K. Improving language understanding by generative pre–training [EB/OL]. [2024–03–10]. https://cdn. openai. com/research–covers/language–unsupervised/language_understanding_paper.pdf

[4] DEVLIN J, CHANG M W, LEE K, et al. BERT: pre–training of deep bidirectional transformers for language understanding [EB/OL]. (2018–10–11)[2024–03–10]. https://arxiv.org/abs/1810.04805

[5] BROWN T B, MANN B, RYDER N, et al. Language models are few–shot learners [EB/OL]. (2020–05–28) [2024–03–10]. https://arxiv.org/abs/2005.14165

[6] TOUVRON H, LAVRIL T, IZACARD G, et al. Llama: open and efficient foundation language models [EB/OL]. [2024–03–05]. https://arxiv.org/abs/2302.13971

[7] GU A, DAO T. Mamba: linear–time sequence modeling with selective state spaces [EB/OL]. [2024–03–01]. https://arxiv. org/abs/2312.00752

[8] TOUVRON H, MARTIN L, STONE K, et all. Llama 2: open foundation and fine–tuned chat models [EB/OL]. (2023–07–18) [2024–03–10]. https://arxiv.org/abs/2307.09288

[9] DAO T. FlashAttention–2: faster attention with better parallelism and work partitioning [EB/OL]. (2023–07–17)[2024–03–10]. https://arxiv.org/abs/2307.08691

[10] BAI J, BAI S, CHU Y F, et al. Qwen technical report [EB/OL]. (2023–09–28)[2024–03–11]. https://arxiv.org/abs/2309.16609

[11] XIONG W H, LIU J Y, MOLYBOG I, et al. Effective long–context scaling of foundation models [EB/OL]. (2023–09–27) [2024–03–12]. https://arxiv.org/abs/2309.16039

[12] JIANG A Q, SABLAYROLLES A, MENSCH A, et al. Mistral 7B [EB/OL].[2024–03–12]. https://arxiv.org/abs/2310.06825

[13] RADFORD A, KIM W J, HALLACY C, et al. Learning transferable visual models from natural language supervisior [EB/OL]. (2021–02–26)[2014–03–05]. https://arxiv.org/abs/2103.00020

[14] LI J, LI D, SAVARESE S, et al. Blip–2: Bootstrap–ping language–image pre–training with frozen image encoders and large language models [EB/OL]. (2023–01–30) [2024–03–12]. https://arxiv.org/abs/2301.12597

[15] LIU H, LI C Y, WU Q Y, et al. Visual instruction tuning [EB/OL]. [2024–03–10]. https://arxiv.org/abs/2304.08485

[16] DONG X Y, ZHANG P, ZANG Y H, et al. InternLM–XComposer2: mastering free–form text–image composition and comprehension in vision–language large model [EB/OL]. (2024–01–29)[2024–03–10]. https://arxiv.org/abs/2401.16420

[17] Huggingface models [EB/OL].[2024–03–10]. https://huggingface.co/01–ai/Yi–VL–34B

[18] ROMBACH R, BLATTMANN A, LORENZ D, et al. High–resolution image synthesis with latent diffusion models [EB/OL]. (2021–12–20)[2022–04–13]. https://arxiv.org/abs/2112.10752

[19] PEEBLES W, XIE S N. Scalable diffusion models with transformers [C]//Proceedings of IEEE/CVF International Conference on Computer Vision (ICCV). IEEE, 2023: 4195–4205. DOI: 10.1109/iccv51070.2023.00387

[20] OpenAI. Video generation models as world simulators [EB/OL]. [2024–03–13]. https://openai. com/research/video–generation–models–as–world–simulators

[21] SHOEYBI M, PATWARY M, PURI R, et al. Megatron–LM: training multi–billion parameter language models using model parallelism [EB/OL]. [2024–03–10]. https://arxiv. org/abs/1909.08053

[22] RAJBHANDARI S, RASLEY J, RUWASE O, et al. ZeRO: memory optimizations toward training trillion parameter models [C]// Proceedings of SC20: International Conference for High Performance Computing, Networking, Storage and Analysis. IEEE, 2020: 1–16. DOI: 10.1109/sc41405.2020.00024

[23] CHEN T, XU B, ZHANG C Y, et al. Training deep nets with sublinear memory cost [EB/OL]. [2024–03–10]. https://arxiv. org/abs/1604.06174

[24] HU E J, SHEN Y Y, WALLIS P, et al. Lora: Low–rank adaptation of large language models [EB/OL]. (2021–06–17) [2024–03–10]. https://arxiv.org/abs/2106.09685

[25] TOUVRON H, MARTING L, STONE K, et al. Llama 2: open foundation and fine–tuned chat models [EB/OL]. (2023–07–18) [2024–03–05]. https://arxiv.org/abs/2307.09288

[26] RAFAILOV R, SHARMA A, MITCHELL E, et al. Direct preference optimization: your language model is secretly a reward model [EB/OL]. [2024–03–10]. https://arxiv.org/abs/2305.18290

[27] XU C, SUN Q, ZHENG K, et al. Wizardlm: empowering large language models to follow complex instructions [EB/OL]. (2023–04–24)[2024–03–10]. https://arxiv.org/abs/2304.12244

[28] LUO Z, XU C, ZHAO P, et al. WizardCoder: empowering code large language models with evol–instruct [EB/OL]. [2024–03–10]. https://arxiv.org/abs/2306.08568

[29] WEI Y X, WANG Z, LIU J, et al. Magicoder: source code is all you need [EB/OL]. (2023–12–04)[2024–03–10]. https://arxiv.org/abs/2312.02120

[30] TUFANO M, DRAIN D, SVYATKOVSKIY A, et al. Unit test case generation with transformers and focal context [EB/OL]. (2020–09–11)[2024–03–10]. https://arxiv.org/abs/2009.05617

[31] STEENHOEK B, TUFANO M, SUNDARESAN N, et al. Reinforcement learning from automatic feedback for high–quality unit test generation [EB/OL]. (2023–10–03) [2024–03–09]. https://arxiv.org/abs/2310.02368

[32] SHA Z Y, HE X L, BERRANG P, et al. Fine–tuning is all you need to mitigate backdoor attacks [EB/OL]. (2022–12–18) [2024–03–10]. https://arxiv.org/abs/2212.09067

[33] LIU K, DOLAN–GAVITT B, GARG S. Fine–pruning: defending against backdooring attacks on deep neural networks [EB/OL]. (2018–05–30)[2024–03–10]. https://arxiv.org/abs/1805.12185

[34] CHEN B, CARVALHO W, BARACALDO N, et al. Detecting backdoor attacks on deep neural networks by activation clustering [EB/OL]. (2018–05–30)[2024–04–10]. https://arxiv.org/

abs/1805.12185
[35] ZHANG Z Y, LYU L, MA X J, et al. Fine-mixing: mitigating backdoors in fine-tuned language models [EB/OL]. (2018-05-30)[2024-04-10]. https://arxiv.org/abs/1805.12185
[36] CUI G, YUAN L, HE B, et al. A unified evaluation of textual backdoor learning: frameworks and benchmarks [J]. Advances in neural information processing systems, 2022, 35: 5009-5023
[37] FRANTAR E, ASHKBOOS S, HOEFLER T, et al. GPTQ: accurate post-training quantization for generative pre-trained transformers [EB/OL]. (2022-10-31)[2023-05-22]. https://arxiv.org/abs/2210.17323
[38] LIN J, TANG J, TANG H T, et al. AWQ: activation-aware weight quantization for LLM compression and acceleration [EB/OL]. (2023-05-01)[2023-10-03]. https://arxiv.org/abs/2306.00978
[39] XIAO G X, LIN J, SEZNEC M, et al. SmoothQuant: accurate and efficient post-training quantization for large language models [EB/OL]. (2022-11-18) [2024-02-20]. https://arxiv. org/abs/2211.10438
[40] PAN J Y, WANG C C, ZHENG K F, et al. SmoothQuant+ : accurate and efficient 4-bit post-training weightQuantization for LLM [EB/OL]. (2023-12-06)[2024-02-20]. https://arxiv.org/abs/2312.03788
[41] YU G I, JEONG J S. Orca: A distributed serving system for transformer-based generative models [EB/OL]. [2024-02-20]. https://www.usenix.org/conference/osdi22/presentation/yu
[42] JAIN N, SCHWARTZSCHILD A, WEN Y X, et al. Baseline defenses for adversarial attacks against aligned language models [EB/OL]. [2024-02-22]. https://arxiv.org/abs/2309.00614
[43] SALEM A, ZHANG Y, HUMBERT M, et al. ML-leaks: model and data independent membership inference attacks and defenses on machine learning models [EB/OL].[2024-03-12]. https://arxiv.org/abs/1806.01246
[44] ABADI M, CHU A, GOODFELLOW I, et al. Deep learning with differential privacy [C]//Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2016: 308-318. DOI: 10.1145/2976749.2978318
[45] KEVUATGAB Y, KALMAN M, MATIAS Y. Fast inference from transformers via speculative decoding [EB/OL]. (2022-11-30) [2024-03-10]. https://arxiv.org/abs/2211.17192
[46] CAI T, LI Y H, GENG Z Y, et al. Medusa: simple framework for accelerating LLM generation with multiple decoding heads [EB/OL]. (2023-01-19) [2024-03-10]. https://arxiv. org/abs/2401.10774
[47] FU Y C, BAILIS P, STOCA P, et al. Breaking the sequential dependency of LLM inference using lookahead decoding [EB/OL]. (2024-02-03) [2024-02-10]. https://arxiv. org/abs/2402.02057

作 者 简 介

韩炳涛，中兴通讯股份有限公司AI首席专家、移动网络和移动多媒体技术国家重点实验室多媒体研究中心副主任、Linux深度学习基金会Adlik项目负责人；研究方向为机器学习平台技术和网络智能化，以及相关核心系统架和AI算法；拥有发明专利多项，出版专著多部。

刘涛，中兴通讯股份有限公司资深算法专家、Adlik开源项目首席架构师、AI预研项目经理；主要研究领域为AI模型并行训练、模型推理优化、高性能计算、异构硬件模型部署等；拥有多项发明专利。