硕士学位论文

面向大模型的推理任务批式调度和 KV 缓存优化

BATCH SCHEDULING AND KV CACHE OPTIMIZATION FOR LARGE LANGUAGE MODEL INFERENCE

研　　究　　生：何忆源
指　导　教　师：徐敏贤副研究员

南方科技大学

二〇二六年一月

国内图书分类号：XXxxx.x　　学校代码：14325
国际图书分类号：xx-x　　密级：公开

工学硕士专业学位论文

面向大模型的推理任务批式调度和 KV 缓存优化

学 位 申 请 人：何忆源
指 导 教 师：徐敏贤副研究员
专 业 类 别：计算机技术
答 辩 日 期：2026 年 3 月
培 养 单 位：深圳理工大学
学位授予单位：南方科技大学

BATCH SCHEDULING AND KV CACHE OPTIMIZATION FOR LARGE LANGUAGE MODEL INFERENCE

A dissertation submitted to
Southern University of Science and Technology
in partial fulfillment of the requirement
for the professional degree of
Master of Engineering

by
He Yiyuan

Supervisor: Associate Researcher Minxian Xu

March, 2026

学位论文公开评阅人和答辩委员会名单

公开评阅人名单

刘 XX	教授	南方科技大学
陈 XX	副教授	XXXX 大学
杨 XX	研究员	中国 XXXX 科学院 XXXXXXX 研究所

答辩委员会名单

主席	赵 XX	教授	南方科技大学
委员	刘 XX	教授	南方科技大学
	杨 XX	研究员	中国 XXXX 科学院 XXXXXXX 研究所
	黄 XX	教授	XXXX 大学
	周 XX	副教授	XXXX 大学
秘书	吴 XX	助理研究员	南方科技大学

南方科技大学学位论文原创性声明和使用授权说明

南方科技大学学位论文原创性声明

本人郑重声明：所提交的学位论文是本人在导师指导下独立进行研究工作所取得的成果。除了特别加以标注和致谢的内容外，论文中不包含他人已发表或撰写过的研究成果。对本人的研究做出重要贡献的个人和集体，均已在文中作了明确的说明。本声明的法律结果由本人承担。

作者签名：　　日期：

南方科技大学学位论文使用授权书

本人完全了解南方科技大学有关收集、保留、使用学位论文的规定，即：

1. 按学校规定提交学位论文的电子版本。

2. 学校有权保留并向国家有关部门或机构送交学位论文的电子版，允许论文被查阅。

3. 在以教学与科研服务为目的前提下，学校可以将学位论文的全部或部分内容存储在有关数据库提供检索，并可采用数字化、云存储或其他存储手段保存本学位论文。

（1）在本论文提交当年，同意在校园网内提供查询及前十六页浏览服务。

（2）在本论文提交 □ 当年/□ __ 年以后，同意向全社会公开论文全文的在线浏览和下载。

4. 保密的学位论文在解密后适用本授权书。

作者签名：　　日期：

指导教师签名：　　日期：

摘 要

随着大语言模型（LLMs）在云计算环境中的广泛部署，其推理服务面临资源消耗巨大、服务等级目标（SLO）要求严格、预填充（Prefill）与解码（Decode）阶段资源需求异构以及负载动态多变等多重挑战。据统计，云平台中约 90% 的人工智能计算资源用于模型推理而非训练，而现有系统受限于粗粒度的静态资源配置，GPU 利用率通常仅为 20%–40%，且在突发流量下因扩容滞后导致严重的 SLO 违约。针对这些问题，本文围绕” 面向大模型的推理任务批式调度和 KV 缓存优化” 这一主题，构建了从资源画像、静态优化到动态调控的递进式技术体系，实现了对 LLM 推理服务全生命周期资源管理的系统性优化。

首先，针对 LLM 推理任务资源需求高度不确定、传统基于峰值预留策略导致严重资源浪费的问题，本文提出了面向异构 LLM 任务的细粒度资源画像与预测方法。通过分析任务类型、输入长度、语言类型等多维属性，设计并采集了 182 个覆盖不同复杂度的测试样本；针对混合数据类型特征，提出基于 K-Prototypes 算法的负载聚类方法，通过定义混合距离函数（欧氏距离 + 简单匹配距离）将任务划分为 6 个具有显著差异的资源消费模式；在每个聚类内部，采用 Random Forest、Gradient Boosting 和 LightGBM 等集成学习方法，对 GPU 利用率、显存占用和推理时延等五维资源指标进行预测。该方法在 80% 测试样本上相对误差小于 10%，为后续资源调度提供了可靠的先验知识。

其次，基于资源画像获取的任务先验知识，针对静态场景下批处理策略粗放、部署配置次优等问题，本文提出了 UELLM（Unified and Efficient LLM Inference Serving）统一高效批式调度与部署框架。该框架通过微调 ChatGLM3-6B 模型对请求输出长度进行预测，分桶准确率达到 99.51%；设计了 SLO 与输出长度驱动的动态批处理算法（SLO-ODBS），通过双阶段贪心策略优化请求组合，减少 KV Cache 冗余填充；提出了基于动态规划的高效低延迟资源分配算法（HELR），综合考虑集群网络拓扑异构性（NVLink/PCIe 带宽差异）自动优化层到 GPU 的映射，支持高利用率（HE）与低延迟（LR）两种模式切换。在 4-GPU 集群上的验证表明，UELLM 相比 Morphling 和 S^3 等先进方案，降低推理延迟 72.3% 至 90.3%，提升 GPU 利用率 1.2 倍至 4.1 倍，吞吐量提高 1.92 倍至 4.98 倍，并实现零 SLO 违约。

再次，面向 PD（Prefill-Decode）分离架构中固有的计算–内存负载失衡、静态资源配置适应性差、前缀缓存感知路由导致的热点倾斜等问题，本文提出了 BanaServe

细粒度弹性伸缩框架。该框架创新性地引入了模块级细粒度迁移机制，包括层级（Layer-level）权重迁移与注意力级（Attention-level）KV Cache 迁移，前者支持连续 Transformer 层的动态重配置以实现粗粒度负载均衡，后者通过按注意力头切分 KV Cache 实现细粒度计算卸载，数学上证明了注意力分解的数值等价性；设计了全局 KV Cache 存储与层间流水线重叠传输机制，通过解耦缓存状态与计算位置消除前缀缓存对调度的约束，并利用层间流水线隐藏通信延迟；提出了基于实时负载感知的请求调度算法，配合动态迁移机制实现快速负载均衡。在 13B 参数模型（LLaMA-13B/OPT-13B）和公开基准（Alpaca/LongBench）上的评估表明，BanaServe 相比 vLLM 吞吐量提升 1.2 倍至 3.9 倍、延迟降低 3.9% 至 78.4%；相比 DistServe 吞吐量提升 1.1 倍至 2.8 倍、延迟降低 1.4% 至 70.1%，并在 Azure 生产环境 traces 下展现出优异的鲁棒性。

本文所提出的方法在真实 GPU 集群和公开基准数据集上得到了充分验证，构建的” 资源画像–静态优化–动态调控” 闭环技术体系，为大模型推理服务的高效部署与资源管理提供了新的理论依据和技术路径。

关键词：大语言模型推理；批式调度；弹性伸缩；资源画像；K-Prototypes 聚类；集成学习；PD 分离架构；动态迁移；KV Cache 优化

Abstract

With the widespread deployment of Large Language Models (LLMs) in cloud computing environments, inference services face critical challenges including massive resource consumption, strict Service Level Objective (SLO) requirements, heterogeneous resource demands between the prefill and decode phases, and highly dynamic workload variations. Statistics indicate that approximately 90% of AI computing resources in cloud platforms are dedicated to model inference rather than training. However, existing systems are constrained by coarse-grained static resource configurations, resulting in GPU utilization typically ranging merely between 20%–40%, and severe SLO violations due to delayed scaling during bursty traffic. To address these issues, this thesis focuses on "Batch Scheduling and KV Cache Optimization for Large Language Model Inference," constructing a progressive technical architecture spanning from resource profiling and static optimization to dynamic scaling, thereby achieving systematic optimization of full-lifecycle resource management for LLM inference services.

Firstly, to tackle the highly uncertain resource demands of LLM inference tasks and the severe resource waste caused by traditional peak-based reservation strategies, this thesis proposes a fine-grained resource profiling and prediction method for heterogeneous LLM tasks. By analyzing multi-dimensional attributes including task types, input lengths, and language modalities, we design and collect 182 test samples covering various complexity scenarios. Addressing the mixed data type characteristics, we introduce a K-Prototypes-based load clustering method that partitions tasks into six distinct resource consumption patterns through a hybrid distance function (Euclidean distance for numerical features and simple matching distance for categorical features). Within each cluster, ensemble learning methods including Random Forest, Gradient Boosting, and LightGBM are employed to predict five-dimensional resource metrics comprising GPU utilization, VRAM occupancy, and inference latency. This method achieves a relative error of less than 10% on 80% of test samples, providing reliable prior knowledge for subsequent resource scheduling.

Secondly, leveraging the task prior knowledge obtained from resource profiling, this thesis proposes UELLM (Unified and Efficient LLM Inference Serving), a comprehensive batch scheduling and deployment optimization framework addressing coarse-grained

batching strategies and suboptimal deployment configurations in static scenarios. The framework employs a fine-tuned ChatGLM3-6B model to predict request output lengths, achieving a bucketing accuracy of 99.51%. It introduces the SLO and Output-Driven Dynamic Batch Scheduler (SLO-ODBS), which optimizes request combinations through a two-stage greedy strategy to reduce KV Cache redundancy and padding overhead. Additionally, the High-Efficiency Low-Latency Resource Allocation (HELR) algorithm based on dynamic programming is proposed, which automatically optimizes layer-to-GPU mapping strategies considering cluster network topology heterogeneity (NVLink/PCIe bandwidth variations), supporting both High Efficiency (HE) and Low Latency (LR) modes. Experimental validation on a 4-GPU cluster demonstrates that UELLM reduces inference latency by 72.3% to 90.3%, improves GPU utilization by 1.2× to 4.1×, and increases throughput by 1.92× to 4.98× compared to state-of-the-art methods such as Morphling and S^3, while achieving zero SLO violations.

Thirdly, targeting the inherent compute-memory load imbalance, poor adaptability of static resource configurations, and hotspot skews caused by prefix cache-aware routing in PD (Prefill-Decode) disaggregated architectures, this thesis presents BanaServe, a fine-grained elastic scaling framework. The framework introduces novel module-level migration mechanisms, including layer-level weight migration for coarse-grained load rebalancing through dynamic reconfiguration of consecutive Transformer layers, and attention-level KV Cache migration for fine-grained computation offloading via head-wise partitioning of KV Cache, with mathematical proof of numerical equivalence for attention decomposition. It designs a Global KV Cache Store with layer-wise pipelined transmission to eliminate constraints imposed by prefix caching on scheduling decisions by decoupling cache state from compute placement, while hiding communication latency through inter-layer pipelining. Furthermore, a real-time load-aware request scheduling algorithm is proposed, working in coordination with dynamic migration for rapid load balancing. Evaluation on 13B-parameter models (LLaMA-13B and OPT-13B) and public benchmarks (Alpaca and LongBench) demonstrates that BanaServe achieves 1.2×–3.9× higher throughput with 3.9%–78.4% lower latency compared to vLLM, and 1.1×–2.8× throughput improvement with 1.4%–70.1% latency reduction compared to DistServe, exhibiting superior robustness under Azure production traces.

The proposed methods have been extensively validated on real GPU clusters and public benchmarks. The closed-loop technical architecture of "Resource Profiling – Static

Optimization – Dynamic Scaling" provides novel theoretical foundations and technical pathways for efficient deployment and resource management of LLM inference services.

Keywords: Large Language Model Inference; Batch Scheduling; Elastic Scaling; Resource Profiling; K-Prototypes Clustering; Ensemble Learning; PD Disaggregated Architecture; Dynamic Migration; KV Cache Optimization

目 录

符号和缩略语说明

As-PPT	聚苯基不对称三嗪
DFT	密度泛函理论 (Density Functional Theory)
DMAsPPT	聚苯基不对称三嗪双模型化合物（水解实验模型化合物）
E_a	化学反应的活化能 (Activation Energy)
HMAsPPT	聚苯基不对称三嗪模型化合物的质子化产物
HMPBI	聚苯并咪唑模型化合物的质子化产物
HMPI	聚酰亚胺模型化合物的质子化产物
HMPPQ	聚苯基喹噁啉模型化合物的质子化产物
HMPY	聚吡咙模型化合物的质子化产物
HMSPPT	聚苯基对称三嗪模型化合物的质子化产物
HPCE	高效毛细管电泳色谱 (High Performance Capillary lectrophoresis)
HPLC	高效液相色谱 (High Performance Liquid Chromatography)
IRC	内禀反应坐标 (Intrinsic Reaction Coordinates)
LC-MS	液相色谱-质谱联用 (Liquid chromatography-Mass Spectrum)
MAsPPT	聚苯基不对称三嗪单模型化合物，3,5,6-三苯基-1,2,4-三嗪
MPBI	聚苯并咪唑模型化合物，N-苯基苯并咪唑
MPI	聚酰亚胺模型化合物，N-苯基邻苯酰亚胺
MPPQ	聚苯基喹噁啉模型化合物，3,4-二苯基苯并二嗪
MPY	聚吡咙模型化合物
MSPPT	聚苯基对称三嗪模型化合物，2,4,6-三苯基-1,3,5-三嗪
ONIOM	分层算法 (Our own N-layered Integrated molecular Orbital and molecular Mechanics)
PBI	聚苯并咪唑
PDT	热分解温度
PES	势能面 (Potential Energy Surface)
PI	聚酰亚胺
PMDA-BDA	均苯四酸二酐与联苯四胺合成的聚吡咙薄膜
PPQ	聚苯基喹噁啉
PY	聚吡咙
S-PPT	聚苯基对称三嗪
SCF	自洽场 (Self-Consistent Field)

SCRF	自洽反应场 (Self-Consistent Reaction Field)
TIC	总离子浓度 (Total Ion Content)
TS	过渡态 (Transition State)
TST	过渡态理论 (Transition State Theory)
ZPE	零点振动能 (Zero Vibration Energy)
ab initio	基于第一原理的量子化学计算方法，常称从头算法
$\Delta G^{\neq}$	活化自由能 (Activation Free Energy)
κ	传输系数 (Transmission Coefficient)
ν_i	虚频 (Imaginary Frequency)

第 1 章　绪论

1.1　研究背景

近年来，以 GPT-4[1]、LLaMA[2]、Claude[3]等为代表的大语言模型（Large Language Models, LLMs）在自然语言处理、代码生成、知识检索和内容创作等领域展现出卓越的性能，推动了人工智能技术的跨越式发展。这些模型通过数百亿乃至万亿级参数的规模化效应，涌现出强大的上下文学习与推理能力，已成为智能时代的关键基础设施。然而，随着模型参数规模的指数级增长，其训练和推理过程对计算资源提出了极高的要求，由此产生的资源密集型特征与高效部署需求之间的矛盾日益尖锐。

在机器学习即服务（MLaaS）的范式下，大模型的生命周期呈现出” 训练集中，推理分散” 的显著特征。据统计，云平台中约 90% 的人工智能计算资源被用于模型推理服务而非训练[4]，且推理成本随着模型规模呈超线性增长。以 OpenAI 的 ChatGPT 为例，为维持其日均 7000 万次访问的服务规模，需要部署超过 6 万张 NVIDIA A100 GPU，初始投资成本高达 16 亿美元，每日电费支出约 10 万美元[5]。这些数字清晰地揭示了大模型推理阶段对计算资源的巨额消耗及其带来的沉重经济负担。

大模型推理过程具有独特的两阶段执行特性：预填充（Prefill）阶段与解码（Decode）阶段，这种内在的计算模式差异构成了资源优化的核心挑战。Prefill 阶段计算密集，需并行处理整个输入序列以计算注意力键值（KV Cache）并生成首个令牌（Time to First Token, TTFT），其延迟直接影响用户感知的响应速度；Decode 阶段则受限于自回归生成机制，需逐令牌迭代输出，呈现显著的内存密集型特征，其关键指标为每输出令牌时间（Time Per Output Token, TPOT）。两个阶段在计算强度、内存访问模式和延迟敏感度上的固有不对称性，使得传统单一架构难以同时优化，往往导致严重的资源利用率低下和负载不均衡问题。

此外，大模型推理面临着严峻的显存墙（Memory Wall）挑战。以千亿级参数模型为例，其参数本身需占用数百 GB 显存，而 KV Cache 的动态增长进一步加剧了显存压力。由于单张 GPU 显存容量有限（通常为 40GB 或 80GB），必须采用张量并行（Tensor Parallelism）或流水线并行（Pipeline Parallelism）等分布式部署策略。然而，随着部署 GPU 数量的增加，节点间通信开销和同步延迟显著上升；反之，若 GPU 资源配置不足，则会导致显存溢出（Out-of-Memory, OOM）错误或极

长的推理延迟。这种资源配置的刚性约束与服务质量（Quality of Service, QoS）要求之间的张力，构成了大模型部署的核心难点。

更为复杂的是，生产环境的推理负载具有高度动态性和不可预测性。请求到达率（Requests Per Second, RPS）随时间呈现剧烈波动，输入/输出序列长度分布呈现显著的重尾特性（Heavy-tailed），即少数长序列请求占据了大量资源。传统静态资源配置策略在负载低谷期造成严重的资源浪费（GPU 利用率常仅 20%-40%），而在突发流量（Bursty Traffic）下又因扩容滞后导致服务等级目标（Service Level Objective, SLO）违约甚至服务中断。现有系统如 vLLM[6] 通过 PagedAttention 优化显存管理，DistServe[7] 采用 PD 分离架构消除阶段间干扰，但这些方案仍受限于粗粒度的资源配置（实例级或 GPU 池级）和缺乏前瞻性的静态调度策略，无法适应快速变化的负载模式。因此，如何在保障严格 SLO 的前提下，实现计算与内存资源的细粒度弹性调度，成为大模型推理服务从实验室走向规模化产业应用必须攻克的关键瓶颈。

1.2　研究意义

本研究围绕大模型推理服务的批式调度与弹性伸缩展开，针对静态资源配置与动态负载需求之间的结构性矛盾，具有以下重要的理论价值和实践意义：

（1）理论价值：构建分层优化的资源管理理论框架

从科学价值角度，本研究针对 LLM 推理中计算-内存协同优化这一基础科学问题，提出了从请求级批处理到模块级弹性伸缩的分层优化理论框架。通过建立考虑 KV Cache 动态增长、网络拓扑异构性和 SLO 约束的数学模型，深入探索了在离散配置空间中寻找最优部署策略的算法边界与计算复杂度。研究揭示了在 Prefill 与 Decode 阶段分离场景下，资源分配与延迟约束之间的权衡机理，为大规模分布式推理系统的资源管理提供了新的理论依据和分析工具，丰富了云计算与人工智能交叉领域的理论体系。

（2）经济效益：显著降低推理成本与提升资源利用率

从工程实践角度，研究成果可直接应用于云原生 AI 基础设施，产生显著的经济效益。首先，通过精确的输出长度预测和 SLO 感知的动态批处理算法，可有效减少无效填充（Padding）和冗余计算，预计可降低推理成本 30% 以上，缓解企业运营压力；其次，模块级细粒度迁移技术打破了传统副本级扩缩容的粒度限制，将资源响应时间从分钟级降至秒级，显著提升系统对突发流量的鲁棒性，避免因过度预配置导致的资源闲置浪费；再者，跨实例的 KV Cache 共享机制消除了缓存局部性对调度策略的约束，解决了前缀缓存感知路由导致的热点倾斜问题，可将集

群整体 GPU 利用率从当前的 20%-40% 提升至 70% 以上。这些技术对推动大模型在智能客服、自动驾驶、金融风控等延迟敏感场景的普及应用具有重要意义。

（3）绿色计算：促进环境可持续性与社会责任

提高资源利用率对于全球环境可持续性具有重要的战略意义。当前大模型服务需要消耗巨量电力，不仅增加了企业的运营成本，也产生了显著的碳足迹。通过优化计算资源的调度与使用，减少闲置 GPU 的能源浪费，可有效降低数据中心整体功耗，符合全球倡导的绿色计算和可持续发展目标[4]。据估算，若全球云 AI 基础设施采用本研究提出的资源优化方法，每年可减少数亿千瓦时的能源消耗，为推动人工智能技术的绿色发展提供可行的技术路径。

（4）技术生态：推动边缘计算与异构计算发展

此外，本研究提出的统一优化框架不仅适用于云端数据中心，其核心理念还可延伸至边缘计算场景。通过高效的资源调度机制，使大模型能够在资源受限的边缘设备和嵌入式系统中高效运行，拓展了 LLM 的应用边界。研究还将促进 GPU、TPU、CPU 等异构计算资源的协同调度技术发展，为构建更加灵活、高效、经济的 AI 计算基础设施提供技术支撑，进一步推动人工智能技术在医疗、教育、制造等各行各业的深度落地与普惠应用。

综上所述，面向大模型推理任务的优化研究不仅是对现有系统性能的改进，更是推动大模型技术从理论走向规模化实践的关键一步。随着大模型的不断扩展及其推理需求的持续增长，本研究成果将在提升模型应用效率、节约计算资源及提高系统稳定性方面发挥愈加重要的作用，为人工智能产业的健康可持续发展奠定坚实基础。

1.3　国内外研究现状及分析

大语言模型推理优化的研究呈现出从算法创新到系统架构、从静态配置到动态调度、从单机优化到分布式协同的演进趋势。本节系统梳理了算法层面的解码优化、系统层面的内存管理与批处理技术、架构层面的资源配置与弹性伸缩策略，并对比分析国内外研究特点，最后指出现有研究的局限性与本文切入点。

1.3.1　算法层面的推理优化

1. 解码策略创新

为突破自回归生成的序列依赖瓶颈，学术界提出了多种非传统的解码范式。**推测性解码**（Speculative Decoding）通过小规模草稿模型（Draft Model）快速生成候选序列，再由目标模型并行验证并修正，在保持输出质量不变的前提下实现 2-3 倍

的加速比[8-9]。**非自回归解码**（Non-autoregressive Decoding）则彻底打破逐 token 生成的限制，通过迭代精化（Iterative Refinement）或掩码预测（Masked Prediction）机制并行输出生成序列，虽然牺牲部分质量，但在实时性要求极高的场景（如实时翻译）展现潜力。

针对生成过程的不确定动态性，**早退出机制**（Early Exiting）和**级联推理**（Cascade Inference）根据中间层置信度自适应决定计算深度。例如，CALM[10]通过动态层级退出减少简单查询的计算量；CascadeBERT[11]采用模型级联策略，对简单样本使用轻量级模型，仅将困难样本路由至大模型。这类方法通过计算量与任务难度的自适应匹配，显著提升了平均推理效率。

2. 模型压缩与量化

为缓解显存墙（Memory Wall）问题，模型压缩技术成为研究热点。**低比特量化**（Low-bit Quantization）将 FP16/FP32 权重压缩至 INT8 甚至 INT4，配合权重量化（Weight-only Quantization）和激活量化（Activation Quantization）策略，使千亿级模型可在单卡消费级 GPU 上部署[12-13]。近期研究还探索了 1-bit 量化（如 BitNet[14]）和混合精度量化，通过细粒度分组量化（Group-wise Quantization）减少精度损失。

知识蒸馏（Knowledge Distillation）通过将大模型（教师）的知识迁移至小模型（学生），在保持性能的同时显著降低推理成本。MiniLLM[15]和 LaMini-GPT[16]等工作表明，经过针对性蒸馏的 7B 参数模型在特定任务上可逼近 70B 模型的性能。国内研究机构在此领域成果显著，清华大学的 QLoRA[17]和 Knowledge Fusion 方法在低资源场景下的压缩效率达到国际领先水平。

1.3.2　系统架构与内存管理优化

1. 注意力机制与显存优化

Transformer 的注意力机制计算复杂度高，显存占用随序列长度平方增长。PagedAttention[6]借鉴操作系统虚拟内存管理思想，将 KV Cache 划分为非连续的物理块（Block），通过块表（Block Table）映射实现动态分配，消除了传统连续存储导致的显存碎片，使 GPU 显存利用率从 40-50% 提升至 90% 以上。在此基础上，vLLM 实现了**连续批处理**（Continuous Batching），通过动态合并新到达请求并移除已完成请求，最大化吞吐率。

针对长上下文（Long Context）场景，**滑动窗口注意力**（Sliding Window Attention）和**稀疏注意力**（Sparse Attention）通过限制注意力范围降低计算复杂度。StreamingLLM[18]发现注意力汇点（Attention Sinks）现象，仅需保留初始 token 的 KV Cache 即可维持长序列生成稳定性，将显存消耗从序列长度的二次方降至线性。

2. 内核优化与硬件协同

为充分发挥 GPU Tensor Core 的计算能力，**算子融合**（Operator Fusion）和**自动编译**（Automatic Compilation）技术被广泛采用。FlashAttention[19] 系列算法通过 IO 感知的精确计算调度和分块策略（Tiling），将注意力计算中的 HBM 访问次数从 $O(N)$ 降至 $O(N^2/\text{block\_size})$，在 A100 GPU 上实现 2-4 倍的加速。TVM[20] 和 MLIR 等编译器框架通过自动生成针对特定硬件的微内核（Micro-kernel），进一步优化了矩阵乘法和内存访问模式。

1.3.3　批处理与请求调度策略

批处理（Batching）是提升 LLM 推理吞吐量的核心手段，但静态批处理（Static Batching）的填充浪费（Padding Waste）问题严重。近期研究聚焦于**输出长度感知调度**和 **SLO 约束优化**。

输出长度预测是优化批处理的关键。S^3 [21] 将请求调度建模为多维装箱问题，利用轻量级预测器估计输出长度，将相似长度的请求归入同一批次，减少填充开销。现有方案多基于历史数据静态预测，难以适应输出长度分布的实时变化。

前缀缓存与数据局部性是另一优化维度。SGLang[22] 和 RadixAttention 通过前缀树（Trie）结构缓存共享系统提示（System Prompt）的 KV Cache，对多轮对话场景实现显著加速。然而，缓存感知路由（Cache-aware Routing）引入的负载不均衡问题尚未得到有效解决，热点前缀可能导致特定 GPU 过载。

SLO 感知调度方面，ClockWork[23] 针对传统模型提出基于完成时间预测的早期退出策略，但难以适应 LLM 自回归生成的动态延迟特性。Splitwise[24] 尝试在保证延迟 SLO 的前提下优化吞吐量，但缺乏对 Prefill 与 Decode 阶段差异性的细粒度建模。

1.3.4　资源配置与弹性伸缩机制

1. PD 分离架构的演进

大模型推理的 Prefill 阶段（计算密集）与 Decode 阶段（内存密集）具有截然不同的资源需求特征，催生了**预填充-解码分离**（Prefill-Decode Disaggregation）架构。DistServe[7] 和 Splitwise[24] 将两个阶段部署在不同 GPU 实例上，消除阶段间干扰，使各自可独立优化。然而，现有方案多采用**静态配比**（如 1:1 或固定比例），无法适应动态负载中 Prefill 与 Decode 需求的实时变化。

分布式 KV Cache 管理是支撑 PD 分离的关键。Mooncake[25] 提出全局 KV Cache 池，通过 RDMA 网络实现跨节点缓存迁移；MemServe[26] 探索了缓存预取（Prefetching）和冗余消除技术。但这些方案引入了显著的跨节点通信开销，且缺乏对网络拓扑异构性的感知。

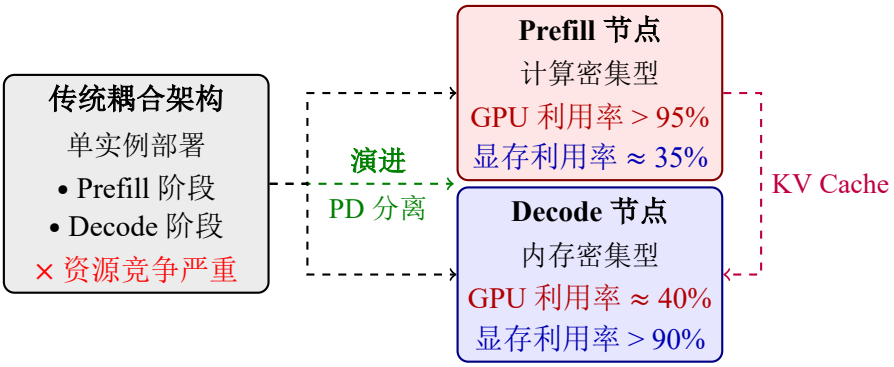


图 1-1　传统耦合架构与 PD 分离架构对比

2. 弹性伸缩与细粒度资源管理

传统云原生弹性伸缩（如 Kubernetes HPA）采用**副本级**（Replica-level）扩缩容，粒度粗糙（需加载完整模型副本），启动延迟长达分钟级，无法应对 LLM 推理的突发流量（Bursty Traffic）。ServerlessLLM[27]提出基于分层的快速启动技术，但仍未能解决扩缩容的粒度问题。

在细粒度资源管理方面，**GPU 虚拟化**和**时空共享**技术取得进展。NVIDIA MPS 和 MIG（Multi-Instance GPU）支持硬件级的 GPU 切分，但配置刚性且缺乏软件层面的灵活性。AntMan[28]和 Allox[29]提出针对深度学习任务的 GPU 时间片调度，但主要针对训练场景，未考虑 LLM 推理中 KV Cache 的状态保持需求。

1.3.5　国内外研究现状对比

1. 国外研究：系统架构与硬件协同领先

以 OpenAI、Google、Meta、Stanford 等为代表的国外研究机构在**系统架构创新**和**硬件-软件协同设计**方面处于领先地位。OpenAI 的 GPT 系列和 Google 的 Gemini 推动了超大规模模型的工程化实践，其内部推理系统（如 Triton、Pathways）虽未完全开源，但通过 vLLM、TensorRT-LLM 等开源工具影响了行业标准。学术机构如 Stanford 的 FlexGen[30]和 Berkeley 的 SkyPilot[31]在资源受限场景下的推理优化提出系统性解决方案。

在**新兴硬件适配**方面，国外研究积极探索 TPU、AWS Inferentia、Groq LPU 等专用芯片的优化策略，以及 FPGA/ASIC 的定制化推理加速。PagedAttention、FlashAttention 等底层优化多源自国外顶尖实验室。

2. 国内研究：场景驱动与算法创新并重

国内以百度、阿里、华为、清华、中科院等为代表的研究力量在**垂直场景优化**和**算法效率提升**方面表现突出。百度的 ERNIE 系列、阿里的通义千问（Qwen）、华为的盘古（Pangu）模型在中文语境和多模态推理方面具有特色；清华的 FastTransformer

希望实验室在稀疏化、量化方面贡献显著。

国内研究更注重**产业落地**与**边缘部署**。针对国内云计算和边缘计算的特定需求，研究重点集中在移动端部署（如小米、OPPO 的端侧大模型）、多模态推理优化（文生图、文生视频的高效推理）以及国产 AI 芯片（昇腾、寒武纪、海光）的适配优化。此外，国内在**长文本处理**（如月之暗面的 Kimi 模型）和**智能体**（Agent）推理调度方面形成了特色研究方向。

然而，在**开源基础设施**和**底层系统软件**方面，国内仍以跟进和适配为主，原创性的系统架构（如 vLLM 级别的创新）相对不足，这反映了在系统软件栈积累上的差距。

表 1-1　国内外大模型推理优化研究特点对比

维度	国外研究	国内研究
核心优势	系统架构、硬件协同	场景驱动、算法创新
代表工作	vLLM[6], DistServe[7]	QLoRA[17], Mooncake[25]
技术重点	底层系统、内存管理	模型压缩、端侧部署
产业落地	云基础设施	端侧 AI、国产芯片适配

1.3.6　现有研究的局限性与启示

综合上述分析，当前 LLM 推理优化研究在以下维度存在显著局限：

（1）调度粒度与资源状态的紧耦合：现有系统（如 SGLang、Mooncake）的调度决策严重依赖 KV Cache 的物理位置，前缀缓存感知路由导致负载热点倾斜。路由器被迫在计算负载均衡与缓存命中率之间做困难权衡（Cache-Load Balancing Trade-off），缺乏**计算-缓存联合优化**的全局视角。

（2）资源配置的静态化与刚性约束：现有 PD 分离系统（DistServe、Splitwise）在部署时固定 Prefill 与 Decode 实例比例，无法在运行期间根据实际负载动态调整。副本级扩缩容响应滞后，且模型加载开销巨大，难以处理突发流量下的资源重分配需求。

（3）缺乏跨阶段的细粒度资源协同机制：Prefill 与 Decode 阶段的资源需求（计算 vs 内存）在时域和空域上互补，但现有系统缺乏在两个阶段之间实时迁移计算负载或显存数据的**细粒度机制**（如层级别的动态迁移），导致严重的资源利用率失衡（Prefill 实例计算饱和但显存闲置，Decode 实例反之）。

（4）对异构性和动态性的适应性不足：现有方案多假设同构的 GPU 集群和稳态负载，对混合型号 GPU、网络拓扑差异以及重尾分布（Heavy-tailed）的动态负载缺乏有效建模，导致实际部署中资源碎片化严重。

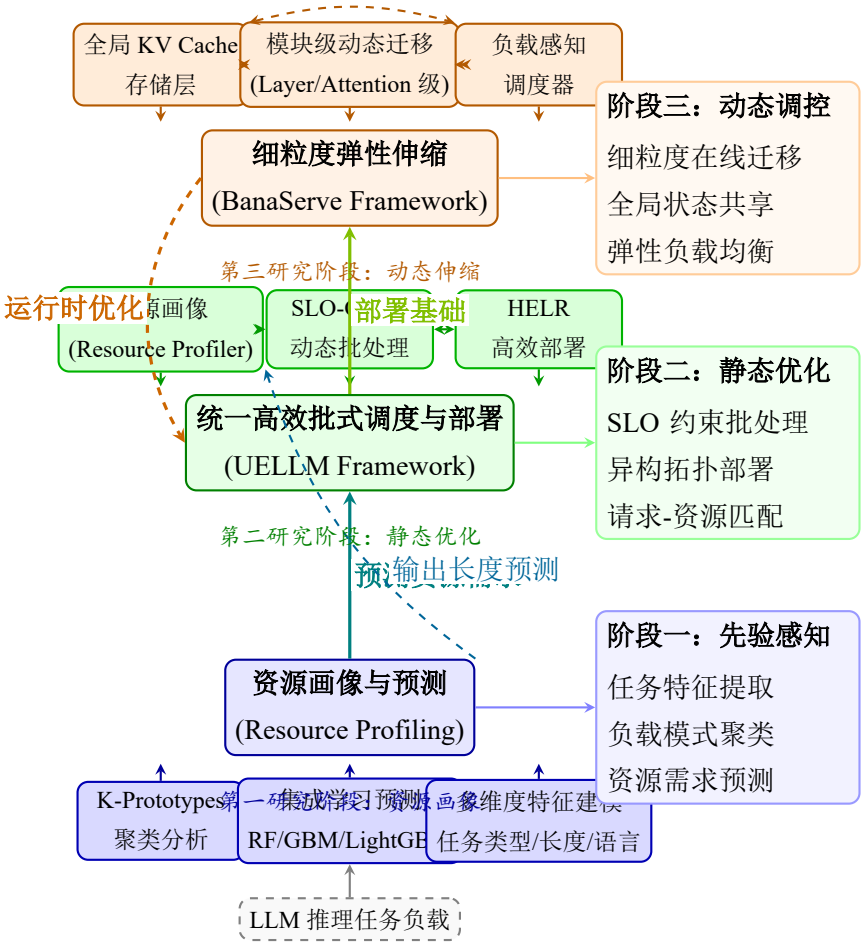


图 1-2　论文整体技术框架：从资源画像感知（底层）、静态优化（中层）到动态弹性伸缩（顶层）的递进式研究体系。底层通过 K-Prototypes 聚类和集成学习建立任务资源画像；中层基于 UELLM 框架实现 SLO 感知的批处理与异构部署；顶层通过 BanaServe 实现模块级细粒度迁移与全局 KV Cache 共享，形成“感知-优化-调控”的闭环。

针对上述局限，本文拟从**请求级批处理优化**和**模块级弹性伸缩**两个层面，构建大模型推理服务的统一资源管理框架，突破静态配置与动态需求间的结构性矛盾，实现计算-内存资源的细粒度协同优化。

1.4　论文主要工作

针对大语言模型（LLM）推理服务中**资源需求难以预测、静态配置效率低下、动态负载适应性差**等关键挑战，本文围绕**大模型推理服务的资源画像与弹性资源管理**这一核心主题，构建“先验感知-静态优化-动态调控”的完整技术体系。如图 1-2所示，研究工作由底层至上层递进展开：首先建立细粒度的任务资源画像能力（第 1.4.1节），基于此设计静态场景下的批处理与部署优化方法（第 1.4.2节），最终突破动态场景下的细粒度弹性伸缩技术（第 1.4.3节），形成从任务接入到资源释放的全生命周期管理闭环。

1.4.1　面向 LLM 推理的任务资源画像与预测方法

LLM 推理任务的资源需求具有高度不确定性，传统基于峰值预留的资源分配策略导致严重的资源浪费。针对该问题，本文从任务特征工程、负载模式聚类和资源需求预测三个层面建立细粒度资源画像体系。

（1）多维度任务特征建模与打标数据集构建。

深入分析 LLM 推理任务的异构性，从任务类型（文本生成、摘要、翻译、代码等）、输入长度、语言类型等维度构建任务特征空间。设计并采集了 182 个覆盖不同复杂度、长度（5-431 tokens）和语言（中英双语）的测试样本，建立高质量的推理任务基准数据集。

（2）基于 K-Prototypes 的混合特征聚类分析。

针对任务属性中同时存在数值型（输入长度、历史 GPU 利用率）和分类型（任务类型、语言）特征的混合数据特点，提出基于 K-Prototypes 算法的负载聚类方法。算法定义混合距离函数：

$$d(x_i, q_j) = \sum_{l=1}^{p}(x_{il} - q_{jl})^2 + \mu \sum_{l=p+1}^{m} s(x_{il}, q_{jl}) \tag{1-1}$$

其中前 p 项为数值属性的欧氏距离，后 $m-p$ 项为分类属性的简单匹配距离（s 为指示函数，取值 0/1），μ 为类别权重系数。

通过肘部法则确定最优聚类数 $K=6$，将 182 个测试样本划分为具有显著差异的资源消费模式：Cluster 0（短输入/高波动）、Cluster 3（长输入/稳定高占用）等。聚类内样本在 GPU 利用率方差上比全局分布降低 67%，验证了聚类有效性。

（3）基于集成学习的多维资源预测模型。

在每个聚类内部，分别训练 Random Forest、Gradient Boosting 和 LightGBM 三种机器学习模型，对最大 GPU 利用率、平均 GPU 利用率、最大显存占用、平均显存占用和总推理时延等五维资源指标进行预测。实验表明，该预测方法在 80% 测试样本上相对误差小于 10%，显著优于单一全局模型，为后续资源调度提供了可靠的决策依据。

1.4.2　统一高效的批式调度与部署框架 UELLM

基于前述资源画像获取的任务先验知识，本文进一步研究静态场景下的**请求调度**与**模型部署**联合优化问题。针对传统系统批处理策略粗放（FIFO）、部署配置静态化、缺乏服务质量（SLO）感知等缺陷，提出 UELLM（Unified and Efficient LLM Inference Serving）框架，实现从资源画像到调度决策的端到端优化。

（1）基于微调大模型的输出长度预测。

突破传统静态分析方法的局限，认识到输出长度是决定 KV Cache 大小和计算量的关键变量。采用 ChatGLM3-6B 作为基础模型，在 Alpaca 和 Natural Questions 数据集上进行 LoRA 微调，将输出长度预测建模为分类任务（分桶策略：8/16/32/64/128/.../2048 tokens）。微调后模型在测试集上达到 **99.51%** 的分桶准确率，相比第一篇工作的基于规则的方法，显著提升了预测精度。

（2）SLO 与输出长度驱动的动态批处理算法（SLO-ODBS）。

针对传统 FIFO 批处理策略导致的 KV Cache 冗余填充和 SLO 违约问题，建立了综合考虑请求 SLO 约束和输出长度差异的批处理优化模型。算法通过权重化目标函数平衡总延迟与总输出长度，采用贪心策略将长度相近的请求组合成批，避免为短输出请求不必要的填充计算。实验表明，SLO-ODBS 在保持低延迟的同时，将 SLO 违约率降低至接近 0%。

（3）面向异构拓扑的高效资源分配算法（HELR）。

针对 LLM 分布式部署中设备映射（Device Map）搜索空间巨大、静态配置性能次优的问题，提出了基于动态规划的高效资源分配算法。算法综合考虑集群网络拓扑（NVLink、PCIe 带宽异构）、GPU 计算能力差异和模型层间依赖关系，自动求解最优的层到设备映射策略。通过调整权重系数 α_1 和 α_2，可在高利用率模式（HE）和低延迟模式（LR）间灵活切换，适应不同服务等级需求。

UELLM 框架在真实 4-GPU 集群上的验证表明，相比 Morphling 和 S^3 等先进方案，系统降低推理延迟 72.3% 至 90.3%，提升 GPU 利用率 1.2 倍至 4.1 倍，吞吐量提高 1.92 倍至 4.98 倍，实现了零 SLO 违约的高质量推理服务。

1.4.3　面向 PD 分离架构的细粒度弹性伸缩框架 BanaServe

当面临**动态变化**的负载时，静态配置的系统难以维持高效运行。特别是 PD（Prefill-Decode）分离架构中，Prefill 阶段计算密集而 Decode 阶段内存密集的固有特性导致资源利用率失衡。针对该问题，本文提出 BanaServe 动态编排框架，突破传统实例级扩缩容的粒度限制，实现**模块级细粒度迁移**与**全局状态共享**。

（1）模块级细粒度动态迁移机制。

针对 PD 分离架构中 Prefill 阶段计算密集与 Decode 阶段内存密集的资源需求差异，提出了层级（Layer-level）权重迁移与注意力级（Attention-level）KV Cache 迁移两种机制。层级迁移支持将连续的 Transformer 层动态迁移至负载较轻的 GPU，实现粗粒度负载重平衡；注意力级迁移将 KV Cache 按注意力头维度切分，将部分头的计算卸载至辅助 GPU（Cold GPU），主 GPU（Hot GPU）与辅助 GPU 并行计算，在不迁移模型权重的情况下实现细粒度负载分担。数学上证明了注意力分解的正确性，确保分布式计算与单卡计算数值等价。

（2）全局 KV Cache 存储与层间流水线传输。

针对传统前缀缓存感知路由导致的负载倾斜问题，设计了跨 Prefill 实例的全局 KV Cache 存储层，解耦缓存状态与计算位置，使调度决策无需受限于缓存局部性。针对全局存储引入的访问延迟，提出了层间流水线重叠传输机制，利用 Transformer 逐层计算特性，将第 i 层计算与第 $i+1$ 层 KV Cache 预取重叠。理论分析与实验验证表明，当 KV Cache 传输时间（约 0.082ms）远小于层计算时间（约 4.22ms）时，可实现近透明的缓存访问。

（3）自适应迁移与负载感知调度算法。基于全局 KV Cache 存储，实现了完全基于实时负载的调度策略。算法周期性地测量各 Prefill 实例的综合负载（计算 + 内存利用率），将新请求分发至负载最轻的实例；同时结合动态迁移机制，当实例间负载差异超过阈值 δ 时，自动触发层级或注意力级迁移，实现快速负载均衡。

BanaServe 在 13B 参数模型（LLaMA-13B、OPT-13B）和公开基准（Alpaca、LongBench）上的评估表明，相比 vLLM，系统吞吐量提升 1.2 倍至 3.9 倍，总处理时间降低 3.9% 至 78.4%；相比 DistServe，吞吐量提升 1.1 倍至 2.8 倍，延迟降低 1.4% 至 70.1%，并在突发流量下展现出卓越的鲁棒性。

1.4.4　主要创新点

本文围绕大模型推理服务的资源管理问题，从资源画像、静态优化到动态伸缩开展了系统性研究，主要创新点可总结为：

（1）提出了面向异构 LLM 任务的细粒度资源画像与预测方法：首次将 K-Prototypes 聚类与集成学习相结合，针对 LLM 推理任务的类型、长度、语言等多维属性建立资源需求预测模型，解决了传统方法对 LLM 推理资源需求预测不准确的问题，为资源管理提供了可靠的数据基础。

（2）提出了 SLO 感知的动态批处理与异构部署联合优化框架：首次将输出长度预测与 SLO 约束联合建模，设计了 SLO-ODBS 批处理算法与 HELR 部署优化算法，突破了传统批处理仅优化吞吐量、传统部署忽视网络拓扑异构性的局限，实现了延迟、利用率与违约率的联合优化。

（3）提出了模块级细粒度弹性伸缩新范式：针对 PD 分离架构，将资源重配置粒度从实例级下沉至 Transformer 层/注意力头级，提出了层级权重迁移与注意力级 KV Cache 迁移机制，实现了计算与内存资源的在线重平衡，填补了细粒度在线资源迁移研究空白。

（4）提出了全局状态共享与计算-通信重叠机制：通过全局 KV Cache 存储解耦缓存状态与计算位置，消除了前缀缓存对调度的约束；结合层间流水线传输技术，解决了全局存储的延迟瓶颈，实现了 PD 分离架构下的高效负载均衡。

1.5　论文组织架构

本文共分为六章，各章内容安排如下：

第一章绪论：介绍大语言模型推理服务的研究背景与意义，系统梳理国内外在算法优化、系统架构、批处理调度、弹性伸缩等方面的研究现状，深入分析现有研究的局限性；阐述本文" 资源画像–静态优化–动态调控" 的递进式技术路线与创新点。

第二章大模型推理资源管理基础与挑战分析：阐述 Transformer 架构的自回归生成机制与 KV Cache 原理，分析 Prefill/Decode 阶段的计算特征差异；介绍评估指标（TTFT、TPOT、SLO 等）与硬件环境（GPU 拓扑、NVLink/PCIe 异构带宽）；深入剖析现有系统（vLLM、DistServe 等）的技术细节与局限性，重点阐述**资源需求不确定性**、**静态配置次优性**、**动态负载失衡**三大核心挑战，为后续三章的技术方案奠定理论基础。

第三章面向 LLM 推理的任务资源画像与预测方法：（对应第一章 *1.4.1* 节，第一篇工作）。针对 LLM 推理任务资源需求难以预测的问题，构建多维度任务特征建模体系（任务类型、输入长度、语言属性），建立包含 182 个样本的推理任务基准数据集；提出基于 K-Prototypes 算法的混合特征聚类方法，通过定义混合距离函数（欧氏距离 + 简单匹配距离）将任务划分为 6 个资源消费模式；在各聚类内采用 Random Forest、Gradient Boosting、LightGBM 集成学习方法，建立 GPU 利用率、显存占用、推理时延等五维资源指标的预测模型；通过实验验证资源画像方法的预测精度（80% 样本相对误差 <10%），为后续章节的调度优化提供先验知识。

第四章统一高效的批式调度与部署框架 UELLM：（对应第一章 *1.4.2* 节，第二篇工作）。基于第三章获取的资源画像先验知识，研究静态场景下的批处理与部署联合优化。介绍基于微调大模型（ChatGLM3-6B）的输出长度预测方法（99.51% 分桶准确率）；详细阐述 SLO 与输出长度驱动的动态批处理算法 SLO-ODBS 的数学模型（填充浪费建模）与双阶段贪心策略；论述面向异构拓扑的高效资源分配算法 HELR，包括基于动态规划的层到 GPU 映射优化、HE/LR 双模式切换机制；介绍 UELLM 系统实现与 4-GPU 集群实验验证，展示相比 Morphling/S^3 在延迟、利用率、吞吐量方面的性能提升。

第五章面向 PD 分离架构的细粒度弹性伸缩及全文总结：（对应第一章 *1.4.3* 节，第三篇工作 + 全文总结）。针对动态负载场景，研究 PD 分离架构下的弹性伸缩机制。阐述模块级细粒度迁移机制，包括层级（Layer-level）权重迁移的粗粒度重平衡策略与注意力级（Attention-level）KV Cache 迁移的细粒度卸载策略，给出注意力分解的数学正确性证明；介绍全局 KV Cache 存储架构设计与层间流水线重

叠传输机制，分析通信-计算重叠的条件与收益；论述负载感知请求调度算法与自适应迁移决策机制；通过 LLaMA-13B/OPT-13B 模型在 Alpaca/LongBench 基准及 Azure 生产 traces 上的实验，验证 BanaServe 相比 vLLM/DistServe 的性能优势；最后总结全文主要研究成果，讨论异构硬件感知、预测性伸缩、跨区域部署等未来研究方向。

第六章总结与展望：总结本文的主要研究成果，讨论存在的问题与局限性，展望未来研究方向，包括异构硬件感知调度、预测性弹性伸缩、跨地域分布式推理等。

第 2 章　图、表及条目示例

2.1　插图

图片通常在 figure 环境中使用 `\includegraphics` 插入，如图 2-1 的源代码。建议矢量图片使用 PDF 格式，比如数据可视化的绘图；照片应使用 JPG 格式；其他的栅格图应使用无损的 PNG 格式。注意，LaTeX 不支持 TIFF 格式；EPS 格式已经过时。

建议图的大小一般为宽 6.67 cm × 高 5.00 cm。特殊情况下，也可为宽 9.00 cm × 高 6.75 cm，或宽 13.5 cm × 高 9.00 cm。总之，一篇论文中，同类图片的大小应该一致，编排美观、整齐。图应尽可能显示在同一页（屏）。

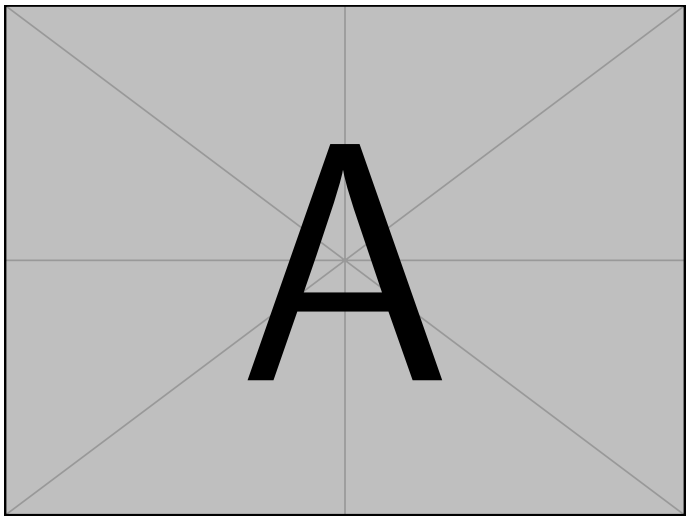


图 2-1　示例图片

若图或表中有附注，采用英文小写字母顺序编号，附注写在图或表的下方。

如果一个图由两个或两个以上分图组成时，各分图分别以 (a)、(b)、(c)..... 作为图序，并须有分图题。推荐使用 subcaption 宏包来处理，比如图 2-3(a) 和图 2-3(b)。

2.2　表格

表应具有自明性。为使表格简洁易读，尽可能采用三线表，如表 2-1。三条线可以使用 booktabs 宏包提供的命令生成。

表格如果有附注，尤其是需要在表格中进行标注时，可以使用 threeparttable 宏包。使用英文小写字母 a、b、c……顺序编号。

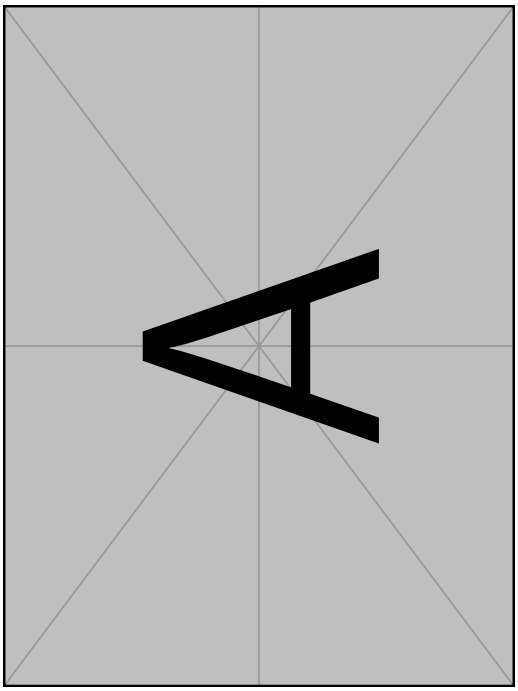
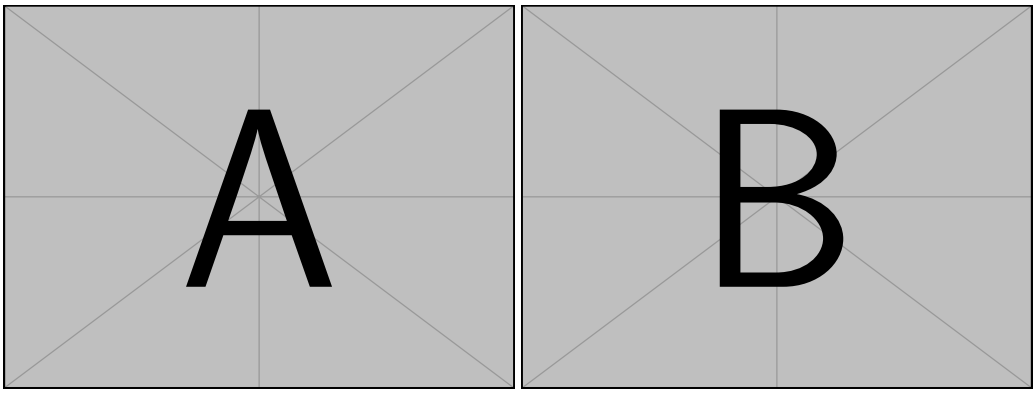


图 2-2　示例图片旋转 90 度



(a) 分图 A　　(b) 分图 B

图 2-3　多个分图的示例

表 2-1　三线表示例

文件名	描述
thuthesis.dtx	模板的源文件，包括文档和注释
thuthesis.cls	模板文件
thuthesis-*.bst	BibTeX 参考文献表样式文件
thuthesis-*.bbx	BibLaTeX 参考文献表样式文件
thuthesis-*.cbx	BibLaTeX 引用样式文件

表 2-2　带附注以及调整列宽的的表格示例

2cm	4cm	6cm
左右居中的 2cm 宽度左右居中的 2cm 宽度[a]	左右居左的 4cm 宽度左右居左的 4cm 宽度	左右居右的 6cm 宽度左右居右的 6cm 宽度
左右居中的 2cm 宽度左右居中的 2cm 宽度[b]	左右居左的 4cm 宽度左右居左的 4cm 宽度	左右居右的 6cm 宽度左右居右的 6cm 宽度

[a] A 的注释
[b] B 的注释

如果需要调整表格列宽度，可以改用命令 `L`, `R`, 或者 `C`, 如 `C{2cm}` 代表居中列宽 2cm。

表 2-3　合并单元格的三线表

Metaclass	A-B		C-D	
Class	A	B	C	D
L1	1	2	3	4
L2	1	2	3	4

如有辅助线要求可以使用 `\cmidrule` 命令。在连续使用时，可以使用一组圆括号括起来的参数 `l`、`r`或`l<距离>`、`r<距离>`表示间距的表格线可以在左右向内缩短一小段，表2-3展示了效果。

表格如果想要与页面等宽，可以使用 tabularx 宏包，如表格2-4所示。模版定义了一些扩展命令，实现一些排版需求。`X` 两端对齐, `Y` 左对齐, `Z` 右对齐，或者 `A` 居中对齐。

如果表格横向宽度不够，可以使用 sidewaystable 将表格旋转 90 度，如表 2-5。

如果您要排版的表格长度超过一页，那么推荐使用 longtable 或者 supertabular 宏包，模板对 longtable 进行了相应的设置，所以用起来可能简单一些。表 2-6 就是 longtable 的简单示例。

表 2-4　同页宽的表格实例

Cell with text aligned to the left	1	2	3
4	Cell with justified text	5	6
7	8	Cell with centered text	9
10	11	12	Cell with text aligned to the right

表 2-6　实验数据（超长表格示例）

测试程序	正常运行时间 (s)	同步时间 (s)	检查点时间 (s)	卷回恢复时间 (s)	进程迁移时间 (s)	检查点文件（KB）
CG.A.2	23.05	0.002	0.116	0.035	0.589	32491
CG.A.4	15.06	0.003	0.067	0.021	0.351	18211
CG.A.8	13.38	0.004	0.072	0.023	0.210	9890
CG.B.2	867.45	0.002	0.864	0.232	3.256	228562
CG.B.4	501.61	0.003	0.438	0.136	2.075	123862
CG.B.8	384.65	0.004	0.457	0.108	1.235	63777
MG.A.2	112.27	0.002	0.846	0.237	3.930	236473
MG.A.4	59.84	0.003	0.442	0.128	2.070	123875
MG.A.8	31.38	0.003	0.476	0.114	1.041	60627
MG.B.2	526.28	0.002	0.821	0.238	4.176	236635
MG.B.4	280.11	0.003	0.432	0.130	1.706	123793
MG.B.8	148.29	0.003	0.442	0.116	0.893	60600
LU.A.2	2116.54	0.002	0.110	0.030	0.532	28754
LU.A.4	1102.50	0.002	0.069	0.017	0.255	14915
LU.A.8	574.47	0.003	0.067	0.016	0.192	8655
LU.B.2	9712.87	0.002	0.357	0.104	1.734	101975
LU.B.4	4757.80	0.003	0.190	0.056	0.808	53522
LU.B.8	2444.05	0.004	0.222	0.057	0.548	30134

续下页

续表 2-6　实验数据（超长表格示例）

测试程序	正常运行时间 (s)	同步时间 (s)	检查点时间 (s)	卷回恢复时间 (s)	进程迁移时间 (s)	检查点文件（KB）
EP.A.2	123.81	0.002	0.010	0.003	0.074	1834
EP.A.4	61.92	0.003	0.011	0.004	0.073	1743
EP.A.8	31.06	0.004	0.017	0.005	0.073	1661
EP.B.2	495.49	0.001	0.009	0.003	0.196	2011
EP.B.4	247.69	0.002	0.012	0.004	0.122	1663
EP.B.8	126.74	0.003	0.017	0.005	0.083	1656
EP.A.2	123.81	0.002	0.010	0.003	0.074	1834

2.3　源代码

使用 listings 环境高亮代码。参数较为复杂，请自行搜索或查阅文档。引用效果如代码 2-1。示例使用 minipage 环境嵌套一层的原因是防止换页中被插入其他浮动体，结合实际情况，按需使用 minipage，例如如需要跨页代码就无需使用 minipage。

```
1 class HelloWorldApp {
2     public static void main(String[] args) {
3         System.out.println("Hello World!"); // Display the
              string.
4         for (int i = 0; i < 100; ++i) {
5             System.out.println(i);
6         }
7     }
8 }
```

代码 2-1　Java 代码示例（使用 listings 高亮）

2.4　伪代码

推荐使用 algorithm2e 宏包中的 algorithm 环境书写伪代码。algorithm2e 可选参数 `linesnumbered` 控制代码行号显示。引用效果如算法 2-1。

表 2-5　旋转 90 度的三线表示例

文件名	描述
thuthesis.dtx	模板的源文件，包括文档和注释
thuthesis.cls	模板文件
thuthesis-*.bst	BibTeX 参考文献表样式文件
thuthesis-*.bbx	BibLaTeX 参考文献表样式文件
thuthesis-*.cbx	BibLaTeX 引用样式文件

算法 2-1　Simulation-optimization heuristic

Data: current period t, initial inventory I_{t-1}, initial capital B_{t-1}, demand samples
Result: Optimal order quantity Q_t^*

$r \leftarrow t$;
$\Delta B^* \leftarrow -\infty$;
while $\Delta B \leqslant \Delta B^*$ and $r \leqslant T$ **do**
    $Q \leftarrow \arg\max_{Q \geqslant 0} \Delta B_{t,r}^{Q}(I_{t-1}, B_{t-1})$;
    $\Delta B \leftarrow \Delta B_{t,r}^{Q}(I_{t-1}, B_{t-1})/(r-t+1)$;
    if $\Delta B \geqslant \Delta B^*$ **then**
        $Q^* \leftarrow Q$;
        $\Delta B^* \leftarrow \Delta B$;
    end
    $r \leftarrow r+1$;
end

算法 2-2　SumExample

Result: s
$s \leftarrow 0$; /* 这是默认多行注释 */
/* 这是默认独占一行的注释 */
/* 这是在取消独占一行后的注释 */
/* 这是恢复独占一行的注释 */
foreach $i \in [1, 100]$ **do**
    if $i\%3 = 0$ **then**
        $s \leftarrow s + i$; // 这是单行注释，一个没有 end 的 if
    else if $i\%3 = 1$ **then**
        break ; ▷ 这是三角形的单行注释，一个没有 end 的 else if
    else
        continue ; /* 这是超长多行注释，关于伪代码的 if-then-else 详细查看 https://texdoc.org/serve/algorithm2e/0 的 10.4 */
    end
end
return s ;

2.5　图表格式测试

图题在图之下，段前空 6 磅，段后空 12 磅。图整体前后距离未定义，目前默认距离：段前空 12 磅，段后空 12 磅。

图前，图前，图前，图前，图前，图前，图前，图前，图前，图前，图前。

图 2-4　图高度为 12bp vs 6bp

图后，图后，图后，图后，图后，图后，图后，图后，图后，图后，图后。

表题在表之上，段前空 12 磅，段后空 6 磅。表整体前后距离未定义，目前默认距离：段前空 12 磅，段后空 12 磅。

表前，表前，表前，表前，表前，表前，表前，表前，表前，表前，表前。

表 2-7　简单表格

column1	column2
column1	column2

表后，表后，表后，表后，表后，表后，表后，表后，表后，表后，表后。图表前后是否有空行不影响图表与正文之间的距离。

2.6　条目编写

2.6.1　支持三级目录显示

支持三级目录显示

2.6.2　条目要求

条目要求首行左缩进 2 个汉字符，避免悬挂缩进。如需使用带括号的条目列表，请自行添加 `label=<style>` 参数。下面是两个例子，还有更多用法，查阅 enumitem 宏包的文档。

默认条目序号：

```
\begin{enumerate} ... \end{enumerate}
```

（1）一级

① 二级

a. 三级

A. 四级，《写作要求》未定义，请自行定义或者选择。

自定义序号样式定义如表 2-8。

表 2-8　条目样式选项

Code	Description
\alph	Lowercase letter (a, b, c, ...)
\Alph	Uppercase letter (A, B, C, ...)
\arabic	Arabic number (1, 2, 3, ...)
\roman	Lowercase Roman numeral (i, ii, iii, ...)
\Roman	Uppercase Roman numeral (I, II, III, ...)

2.6.2.1　条目测试

条目前文字，条目前文字，条目前文字，条目前文字，条目前文字，条目前文字，条目前文字，条目前文字，条目前文字，条目前文字。

（1）一级条目，超长行。南方科技大学，南方科技大学，南方科技大学，南方科技大学，南方科技大学，南方科技大学，南方科技大学，南方科技大学，南方科技大学，南方科技大学，南方科技大学，南方科技大学，南方科技大学。

（2）一级条目，南方科技大学，南方科技大学，南方科技大学。

（3）一级条目，南方科技大学，南方科技大学，南方科技大学。

① 二级条目，超长行。南方科技大学，南方科技大学，南方科技大学，南方科技大学，南方科技大学，南方科技大学，南方科技大学，南方科技大学，南方科技大学，南方科技大学，南方科技大学，南方科技大学，南方科技大学。

② 二级条目，南方科技大学，南方科技大学，南方科技大学。

③ 二级条目，南方科技大学，南方科技大学，南方科技大学。

④ 二级条目，南方科技大学，南方科技大学，南方科技大学。

⑤ 二级条目，南方科技大学，南方科技大学，南方科技大学。

⑥ 二级条目，南方科技大学，南方科技大学，南方科技大学。

a. 三级条目，南方科技大学，南方科技大学，南方科技大学。

b. 三级条目，南方科技大学，南方科技大学，南方科技大学。

c. 三级条目，南方科技大学，南方科技大学，南方科技大学。

A. **四级条目及之后的条目无规定序号格式，请自行设定选择。**

条目后文字，条目后文字，条目后文字，条目后文字，条目后文字，条目后文字，条目后文字，条目后文字，条目后文字，条目后文字。

第 3 章　数学符号和公式

3.1　数学符号

模板中使用 unicode-math 宏包来配置数学符号，

研究生《写作指南》要求量及其单位所使用的符号应符合国家标准《国际单位制及其应用》（GB 3100—1993）、《有关量、单位和符号的一般原则》（GB/T 3101—1993）的规定，但是与 TEX 默认的美国数学学会（AMS）的符号习惯有所区别。

英文论文的数学符号使用 TEX 默认的样式。论文以中文为主要撰写语言按照国标建议的配置数学字体格式：

（1）大写希腊字母默认为斜体，如

$$\Gamma\Delta\Theta\Lambda\Xi\Pi\Sigma\Upsilon\Phi\Psi\Omega.$$

注意有限增量符号 Δ 固定使用正体，模板提供了 `\increment` 命令。

（2）小于等于号和大于等于号使用倾斜的字形 ⩽、⩾。

（3）积分号使用正体，比如 ∫、∮。

（4）行间公式积分号的上下限位于积分号的上下两端，比如

$$\int_a^b f(x)\,\mathrm{d}x.$$

行内公式为了版面的美观，统一居右侧，如 $\int_a^b f(x)\,\mathrm{d}x$ 。

（5）偏微分符号 ∂ 使用正体。

（6）省略号 `\dots` 按照中文的习惯固定居中，比如

$$1,2,\cdots,n \quad 1+2+\cdots+n.$$

（7）实部 Re 和虚部 Im 的字体使用罗马体。

以上数学符号样式的差异可以在模板中统一设置。另外国标还有一些与 AMS 不同的符号使用习惯，需要用户在写作时进行处理：

（1）数学常数和特殊函数名用正体，如

$$\mathrm{\pi} = 3.14\cdots; \quad \mathrm{i}^2 = -1; \quad \mathrm{e} = \lim_{n\to\infty}\left(1+\frac{1}{n}\right)^n.$$

（2）微分号使用正体，比如 $\mathrm{d}y/\mathrm{d}x$。

（3）向量、矩阵和张量用粗斜体（`\mathbfit`），如 $\boldsymbol{x}$、$\boldsymbol{\Sigma}$、$\boldsymbol{T}$。

（4）自然对数用 $\ln x$ 不用 $\log x$。

关于数学符号更多的用法，参考 unicode-math 宏包的使用说明，全部数学符号命的令参考 unimath-symbols，也可以参考 Stack Overflow 上的答案 What are all the font styles I can use in math mode?。

关于量和单位推荐使用 siunitx 宏包，可以方便地处理希腊字母以及数字与单位之间的空白，比如：$6.4\times10^{6}\,\mathrm{m}$，$9\,\mu\mathrm{m}$，$\mathrm{kg\cdot m\cdot s^{-1}}$，10 ℃～20 ℃。

3.2　数学公式

数学公式可以使用 equation 和 equation* 环境。注意数学公式的引用应前后带括号，建议使用 `\eqref` 命令，比如式(3-1)。

$$\frac{1}{2\pi\mathrm{i}}\int_{\gamma} f=\sum_{k=1}^{m} n(\gamma;a_k)\mathcal{R}(f;a_k) \tag{3-1}$$

注意公式编号的引用应含有圆括号，可以使用 `\eqref` 命令。

晶体衍射基础的著名公式 —— 布拉格方程：

$$2d\sin\theta=k\lambda,\quad k=1,2,3\cdots \tag{3-2}$$

式中 d —— 晶面间距（nm）；

θ —— 入射线与晶面的夹角（rad）；

λ —— X 射线波长（nm）。

k —— 公式中第一次出现的物理量代号应给予注释，注释的转行应与破折号“——”后第一个字对齐。

多行公式尽可能在“=”处对齐，推荐使用 align 环境。

$$a=b+c+d+e \tag{3-3}$$

$$=f+g \tag{3-4}$$

此外需要注意：公式需紧挨段前文字，不可空行，不然会导致公式独立成段，如下**错误**效果。公式前文字公式前文字公式前文字公式前文字公式前文字。

$$\frac{1}{2\pi\mathrm{i}}\int_{\gamma} f=\sum_{k=1}^{m} n(\gamma;a_k)\mathcal{R}(f;a_k) \tag{3-5}$$

公式后文字公式后文字公式后文字公式后文字公式后文字公式后文字公式后

文字公式后文字公式后文字公式后文字公式后文字。**正确**效果，如下：

$$\frac{1}{2\pi \mathrm{i}} \int_{\gamma} f = \sum_{k=1}^{m} n(\gamma; a_k) \mathcal{R}(f; a_k) \tag{3-6}$$

公式后文字公式后文字公式后文字公式后文字公式后文字公式后文字公式后。

3.3　数学定理

定理环境的格式可以使用 amsthm 或者 ntheorem 宏包配置。用户在导言区载入这两者之一后，模板会自动配置 thoerem、proof 等环境。

定理 3.1 (Lindeberg–Lévy 中心极限定理)：设随机变量 $X_1, X_2, \cdots, X_n$ 独立同分布，且具有期望 μ 和有限的方差 $\sigma^2 \neq 0$，记 $\bar{X}_n = \frac{1}{n}\sum_{i+1}^{n} X_i$，则

$$\lim_{n \to \infty} P\left(\frac{\sqrt{n}\left(\bar{X}_n - \mu\right)}{\sigma} \leqslant z \right) = \Phi(z), \tag{3-7}$$

其中 $\Phi(z)$ 是标准正态分布的分布函数。

证明：Trivial. ■

同时模板还提供了 assumption、definition、proposition、lemma、theorem、axiom、corollary、exercise、example、remar、problem、conjecture 这些相关的环境。

3.4　数学字体

按照《撰写规范》表达式字体可以采用 Times New Roman、Xits Math 或 Cambria Math（MS Word 默认字体）。Cambria Math 缺少部分样式，例如：积分符号设定为 upright 也看起来没有变化。

TeX Gyre Termes Math 字体（Times New Roman 的 TeX 克隆版）样例：

$$\frac{1}{2\pi \mathrm{i}} \int_{\gamma} f = \sum_{k=1}^{m} n(\gamma; a_k) \mathcal{R}(f; a_k) \tag{3-8}$$

Cambria Math 字体样例：

$$\frac{1}{2\pi \mathrm{i}} \int_{\gamma} f = \sum_{k=1}^{m} n(\gamma; a_k) \mathcal{R}(f; a_k) \tag{3-9}$$

Xits Math 字体样例：

$$\frac{1}{2\pi \mathrm{i}} \int_{\gamma} f = \sum_{k=1}^{m} n(\gamma; a_k) \mathcal{R}(f; a_k) \tag{3-10}$$

STIX Math 字体样例：

$$\frac{1}{2\pi \mathrm{i}} \int_{\gamma} f = \sum_{k=1}^{m} n(\gamma; a_k) \mathcal{R}(f; a_k) \tag{3-11}$$

第 4 章　引用文献的标注

模板支持 BibTeX 和 BibLaTeX 两种方式处理参考文献。下文主要介绍 BibTeX 配合 natbib 宏包的主要使用方法。

4.1　顺序编码制

在顺序编码制下，默认的 \cite 命令同 \citep 一样，即序号置于方括号中，引文页码会放在括号外。统一处引用的连续序号会自动用短横线连接。如多次引用同一文献，可能需要标注页码，例如：引用第二页[32]2，引用第五页[32]5。

```
\cite{zhangkun1994}                  ⇒ [32] 不带页码的上标引用
\citet{zhangkun1994}                 ⇒ 张昆 等[32]
\citep{zhangkun1994}                 ⇒ [32]
\cite[42]{zhangkun1994}              ⇒ [32]42 手动带页码的上标引用
\cite{zhangkun1994,zhukezhen1973}    ⇒ [32-33] 一次多篇文献的上标引用
```

4.2　著者-出版年制

著者-出版年制下的 \cite 跟 \citet 一样。

```
\cite{zhangkun1994}                  ⇒ 张昆 等 (1994)
\citet{zhangkun1994}                 ⇒ 张昆 等 (1994)
\citep{zhangkun1994}                 ⇒ (张昆 等, 1994)
\cite[42]{zhangkun1994}              ⇒ (张昆 等, 1994)42
\citep{zhangkun1994,zhukezhen1973}   ⇒ (张昆 等, 1994; 竺可桢, 1973)
```

4.2.1　其他引用注意事项

注意，引文参考文献的每条都要在正文中标注[32-65]。

引用测试：2 个连续引用[32-33]，2 个间隔[32,34]，3 个连续[32-34]。

如参考文献中需要使用上标或者下标，使用数学环境书写 `$\mathrm{Ba}_{3}\mathrm{CoSb}_{2}\mathrm{O}_{9}$`，例如该文献[66]。根据 gbt7714 规定著者姓名自动转为大写。西文的题名、期刊名的大小写不自动处理，需要自行处理以符合信息资源本身文种的习惯用法。

第 5 章　English and lower-case Example

If your supervisor is a foreign resident, or if your supervisor or defense committee specifically allows writing in English, the thesis may be written in English as the primary language. Please check with your supervisor or department secretary to confirm if you can write in English.

5.1　Reference guide

Writing in English still requires the Chinese reference standard GB/T 7714-2015.

结　论

学位论文的结论作为论文正文的最后一章单独排写，但不加章标题序号。

结论应是作者在学位论文研究过程中所取得的创新性成果的概要总结，不能与摘要混为一谈。博士学位论文结论应包括论文的主要结果、创新点、展望三部分，在结论中应概括论文的核心观点，明确、客观地指出本研究内容的创新性成果（含新见解、新观点、方法创新、技术创新、理论创新），并指出今后进一步在本研究方向进行研究工作的展望与设想。对所取得的创新性成果应注意从定性和定量两方面给出科学、准确的评价，分（1）、（2）、（3）…条列出，宜用“提出了”、“建立了”等词叙述。

在评价自己的研究工作成果时，要实事求是，除非有足够的证据表明自己的研究是“首次”、“领先”、“填补空白”的，否则应避免使用这些或类似词语

参考文献

[1] ACHIAM J, ADLER S, AGARWAL S, et al. GPT-4 Technical Report[A]. 2023.

[2] TOUVRON H, LAVRIL T, IZACARD G, et al. LLaMA: Open and Efficient Foundation Language Models[A]. 2023.

[3] Anthropic. Claude[EB/OL]. 2025. https://claude.ai.

[4] WU C J, RAGHAVENDRA R, GUPTA U, et al. Sustainable AI: Environmental Implications, Challenges and Opportunities[J]. Proceedings of Machine Learning and Systems, 2022, 4: 795-813.

[5] HE Y, XU M, WU J, et al. UELLM: A Unified and Efficient Approach for Large Language Model Inference Serving[C]//Proceedings of the 22nd International Conference on Service-Oriented Computing (ICSOC). Springer, 2024: 218-235.

[6] KWON W, LI Z, ZHUANG S, et al. Efficient Memory Management for Large Language Model Serving with PagedAttention[C]//Proceedings of the 29th ACM Symposium on Operating Systems Principles (SOSP). 2023: 611-626.

[7] ZHONG Y, LIU S, CHEN J, et al. DistServe: Disaggregating Prefill and Decoding for Goodput-optimized Large Language Model Serving[C]//Proceedings of the 18th USENIX Symposium on Operating Systems Design and Implementation (OSDI). 2024: 193-210.

[8] LEVIATHAN Y, KALMAN M, MATIAS Y. Fast Inference from Transformers via Speculative Decoding[J]. International Conference on Machine Learning (ICML), 2023: 19274-19286.

[9] CHEN C, BORGEAUD S, IRVING G, et al. Accelerating Large Language Model Decoding with Speculative Decoding[J]. International Conference on Machine Learning (ICML), 2023: 6159-6181.

[10] SCHUSTER T, FISCH A, GUPTA J, et al. Confident Adaptive Language Modeling[J]. Advances in Neural Information Processing Systems (NeurIPS), 2022, 35: 17456-17472.

[11] LI Z, LI S, ZHANG M, et al. CascadeBERT: Accelerating Inference of Pre-trained Language Models via Cascade Breaking[C]//Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL). 2020: 3876-3886.

[12] DETTMERS T, LEWIS M, BELKADA Y, et al. LLM.int8(): 8-bit Matrix Multiplication for Transformers at Scale[J]. Advances in Neural Information Processing Systems (NeurIPS), 2022, 35: 30318-30332.

[13] FRANTAR E, ASHKBOOS S, HOEFLER T, et al. GPTQ: Accurate Post-Training Quantization for Generative Pre-trained Transformers[J]. International Conference on Learning Representations (ICLR), 2023.

[14] WANG H, MA S, DONG L, et al. BitNet: Scaling 1-bit Transformers for Large Language Models[A]. 2023.

[15] GU Y, DONG L, WEI F, et al. Knowledge Distillation of Large Language Models[A]. 2023.

[16] WU C, ZHAO W, LI B, et al. LaMini-GPT: Large Language Models with Minimal Data and Parameters[A]. 2023.

[17] DETTMERS T, PAGNONI A, HOLTMAN A, et al. QLoRA: Efficient Finetuning of Quantized LLMs[J]. Advances in Neural Information Processing Systems (NeurIPS), 2023, 36.

[18] XIAO G, TIAN Y, CHEN B, et al. StreamingLLM: Efficient Streaming Language Models with Attention Sinks[A]. 2023.

[19] DAO T, FU D Y, ERMON S, et al. FlashAttention: Fast and Memory-efficient Exact Attention with IO-awareness[J]. Advances in Neural Information Processing Systems (NeurIPS), 2022, 35: 16344-16359.

[20] CHEN T, MOREAU T, JIANG Z, et al. TVM: An Automated End-to-End Optimizing Compiler for Deep Learning[C]//Proceedings of the 13th USENIX Symposium on Operating Systems Design and Implementation (OSDI). 2018: 578-594.

[21] JIN C, ZHANG Z, JIANG X, et al. S^3: Increasing GPU Utilization during Generative Inference for Higher Throughput[A]. 2023.

[22] ZHENG L, CHIANG W L, SHENG Y, et al. SGLang: Efficient Execution of Structured Language Model Programs[C]//Advances in Neural Information Processing Systems (NeurIPS): Vol. 37. 2024.

[23] GUJARATI A, SHEKHAR S, GARG T, et al. Serving DNNs like Clockwork: Performance Predictability from the Bottom Up[C]//Proceedings of the 14th USENIX Conference on Operating Systems Design and Implementation (OSDI). 2020: 443-462.

[24] PATEL P, CHOUKSE E, ZHANG C, et al. Splitwise: Efficient Generative LLM Inference Using Phase Splitting[C]//Proceedings of the 51st Annual International Symposium on Computer Architecture (ISCA). 2024: 118-132.

[25] QIN R, LI Z, HE W, et al. Mooncake: A KVCache-centric Disaggregated Architecture for LLM Serving[A]. 2024.

[26] HU C, HUANG H, HU J, et al. MemServe: Context Caching for Disaggregated LLM Serving with Elastic Memory Pool[A]. 2024.

[27] CHU Y, QIN R, BAO Y, et al. ServerlessLLM: Low-Latency Serverless Inference for Large Language Models[A]. 2024.

[28] XIAO W, REN S, LI Y, et al. AntMan: Dynamic Scaling on GPU Clusters for Deep Learning [C]//Proceedings of the 16th USENIX Symposium on Operating Systems Design and Implementation (OSDI). 2022: 533-549.

[29] QIN R, YANG Y, ZHENG Q, et al. Allox: Compute Allocation in Hybrid Clusters[C]// Proceedings of the Eighteenth European Conference on Computer Systems (EuroSys). 2023: 587-603.

[30] SHENG Y, ZHENG L, YUAN B, et al. FlexGen: High-throughput Generative Inference of Large Language Models with a Single GPU[C]//Proceedings of the 40th International Conference on Machine Learning (ICML). PMLR, 2023: 31094-31116.

[31] YANG Z, LUAN Z, LI B, et al. SkyPilot: An Intercloud Broker for Sky Computing[C]// Proceedings of the 17th USENIX Symposium on Operating Systems Design and Implementation (OSDI). 2023: 577-596.

[32] 张昆, 冯立群, 余昌钰, 等. 机器人柔性手腕的球面齿轮设计研究[J]. 清华大学学报: 自然科学版, 1994, 34(2): 1-7.

[33] 竺可桢. 物理学论[M]. 北京: 科学出版社, 1973: 56-60.

[34] DUPONT B. Bone marrow transplantation in severe combined immunodeficiency with an unrelated MLC compatible donor[C]//WHITE H J, SMITH R. Proceedings of the third annual meeting of the International Society for Experimental Hematology. Houston: International Society for Experimental Hematology, 1974: 44-46.

[35] 郑开青. 通讯系统模拟及软件[D]. 北京: 清华大学无线电系, 1987.

[36] 姜锡洲. 一种温热外敷药制备方案: 中国, 88105607.3[P]. 1980-07-26.

[37] 中华人民共和国国家技术监督局. GB3100-3102. 中华人民共和国国家标准-量与单位[S]. 北京: 中国标准出版社, 1994.

[38] MERKT F, MACKENZIE S R, SOFTLEY T P. Rotational Autoionization Dynamics in High Rydberg States of Nitrogen[J]. J Chem Phys, 1995, 103: 4509-4518.

[39] MELLINGER A, VIDAL C R, JUNGEN C. Laser reduced fluorescence study of the carbon monoxide nd triplet Rydberg series - Experimental results and multichannel quantum defect analysis[J]. J Chem Phys, 1996, 104: 8913-8921.

[40] BIXON M, JORTNER J. The dynamics of predissociating high Rydberg states of NO[J]. J Chem Phys, 1996, 105: 1363-1382.

[41] 马辉, 李俭, 刘耀明, 等. 利用 REMPI 方法测量 BaF 高里德堡系列光谱[J]. 化学物理学报, 1995, 8: 308-311.

[42] CARLSON N W, TAYLOR A J, JONES K M, et al. Two-step polarization-labeling spectroscopy of excited states of Na2[J]. Phys Rev A, 1981, 24: 822-834.

[43] TAYLOR A J, JONES K M, SCHAWLOW A L. Scanning pulsed-polarization spectrometer applied to Na2[J]. J Opt Soc Am, 1983, 73: 994-998.

[44] TAYLOR A J, JONES K M, SCHAWLOW A L. A study of the excited 1Σg+ states in Na2[J]. Opt Commun, 1981, 39: 47-50.

[45] SHIMIZU K, SHIMIZU F. Laser induced fluorescence spectra of the a 3Πu–X 1Σg+ band of Na2 by molecular beam[J]. J Chem Phys, 1983, 78: 1126-1131.

[46] ATKINSON J B, BECKER J, DEMTRÖDER W. Experimental observation of the a 3Πu state of Na2[J]. Chem Phys Lett, 1982, 87: 92-97.

[47] KUSCH P, HESSEL M M. Perturbations in the A 1Σu+ state of Na2[J]. J Chem Phys, 1975, 63: 4087-4088.

[48] 广西壮族自治区林业厅. 广西自然保护区[M]. 北京: 中国林业出版社, 1993.

[49] 霍斯尼. 谷物科学与工艺学原理[M]. 李庆龙, 译. 2 版. 北京: 中国食品出版社, 1989: 15-20.

[50] 王夫之. 宋论[M]. 刻本. 金陵: 曾氏, 1865（清同治四年）.

[51] 赵耀东. 新时代的工业工程师[M/OL]. 台北: 天下文化出版社, 1998[1998-09-26]. http://www.ie.nthu.edu.tw/info/ie.newie.htm.

[52] 全国信息与文献工作标准化技术委员会出版物格式分委员会. GB/T 12450-2001 图书书名页[S]. 北京: 中国标准出版社, 2002.

[53] 全国出版专业职业资格考试办公室. 全国出版专业职业资格考试辅导教材: 出版专业理论与实务 • 中级[M]. 2014 版. 上海: 上海辞书出版社, 2004: 299-307.

[54] World Health Organization. Factors Regulating the Immune Response: Report of WHO Scientific Group[R]. Geneva: WHO, 1970.

[55] PEEBLES P Z, Jr. Probability, Random Variables, and Random Signal Principles[M]. 4th ed. New York: McGraw Hill, 2001.

[56] 白书农. 植物开花研究[M]//李承森. 植物科学进展. 北京: 高等教育出版社, 1998: 146-163.

[57] WEINSTEIN L, SWERTZ M N. Pathogenic Properties of Invading Microorganism[M]// SODEMAN W A, Jr, SODEMAN W A. Pathologic physiology: mechanisms of disease. Philadelphia: Saunders, 1974: 745-772.

[58] 韩吉人. 论职工教育的特点[C]//中国职工教育研究会. 职工教育研究论文集. 北京: 人民教育出版社, 1985: 90-99.

[59] 中国地质学会. 地质评论[J]. 1936, 1(1)–. 北京: 地质出版社, 1936–.

[60] 中国图书馆学会. 图书馆学通讯[J]. 1957(1)–1990(4). 北京: 北京图书馆, 1957–1990.

[61] American Association for the Advancement of Science. Science[J]. 1883, 1(1)–. Washington, D.C.: American Association for the Advancement of Science, 1883–.

[62] 傅刚, 赵承, 李佳路. 大风沙过后的思考[N/OL]. 北京青年报, 2000-04-12(14)[2002-03-06]. http://www.bjyouth.com.cn/Bqb/20000412/B/4216%5ED0412B1401.htm.

[63] 萧钰. 出版业信息化迈入快车道[EB/OL]. (2001-12-19)[2002-04-15]. http://www.creader.com/news/20011219/200112190019.htm.

[64] Online Computer Library Center, Inc. About OCLC: History of Cooperation[EB/OL]. 2000 [2000-01-08]. http://www.oclc.org/about/cooperation.en.htm.

[65] Scitor Corporation. Project scheduler[CP/DK]. Sunnyvale, Calif.: Scitor Corporation, 1983.

[66] KAMIYA Y, GE L, HONG T, et al. The nature of spin excitations in the one-third magnetization plateau phase of $Ba_3CoSb_2O_9$[J]. Nature communications, 2018, 9(1): 1-11.

附录 A　补充内容

附录是与论文内容密切相关、但编入正文又影响整篇论文编排的条理和逻辑性的资料，例如某些重要的数据表格、计算程序、统计表等，是论文主体的补充内容，可根据需要设置。

A.1　图表示例

A.1.1　图

附录中的图片示例（图 A-1）。

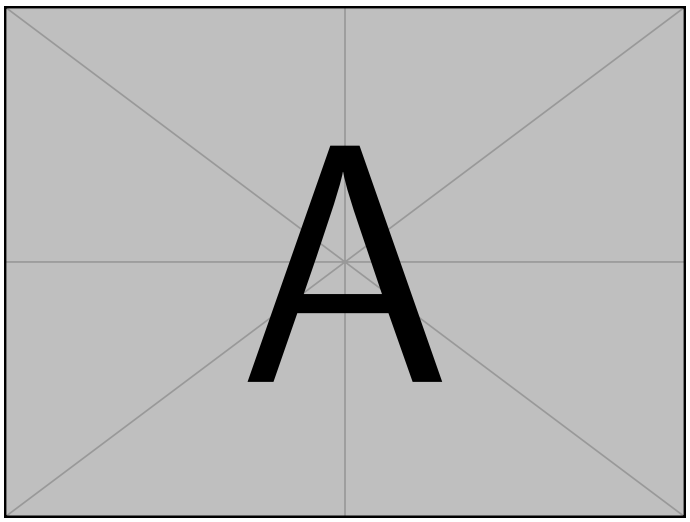


图 A-1　附录中的图片示例

A.1.2　表格

附录中的表格示例（表 A-1）。

A.2　数学公式

附录中的数学公式示例（公式(A-1)）。

$$\frac{1}{2\pi \mathrm{i}} \int_{\gamma} f = \sum_{k=1}^{m} n(\gamma; a_k) \mathcal{R}(f; a_k) \tag{A-1}$$

表 A-1　附录中的表格示例

文件名	描述
sustechthesis.dtx	模板的源文件，包括文档和注释
sustechthesis.cls	模板文件
thuthesis-*.bst	BibTeX 参考文献表样式文件
thuthesis-*.bbx	BibLaTeX 参考文献表样式文件
thuthesis-*.cbx	BibLaTeX 引用样式文件

A.3　源代码

附录中的代码示例：代码A-1。

```
1 class HelloWorldApp {
2     public static void main(String[] args) {
3         System.out.println("Hello World!"); // Display the
              string.
4         for (int i = 0; i < 100; ++i) {
5             System.out.println(i);
6         }
7     }
8 }
```

代码 A-1　Java 代码示例（使用 listings 高亮）

A.4　伪代码

附录中的伪代码示例（算法A-1）。

算法 A-1　Simulation-optimization heuristic

Data: current period t, initial inventory I_{t-1}, initial capital B_{t-1}, demand samples
Result: Optimal order quantity Q_t^*

$r \leftarrow t$;
$\Delta B^* \leftarrow -\infty$;
while $\Delta B \leqslant \Delta B^*$ *and* $r \leqslant T$ **do**
    $Q \leftarrow \arg\max_{Q \geqslant 0} \Delta B_{t,r}^{Q}(I_{t-1}, B_{t-1})$;
    $\Delta B \leftarrow \Delta B_{t,r}^{Q}(I_{t-1}, B_{t-1})/(r - t + 1)$;
    if $\Delta B \geqslant \Delta B^*$ **then**
        $Q^* \leftarrow Q$;
        $\Delta B^* \leftarrow \Delta B$;
    end
    $r \leftarrow r + 1$;
end

致　谢

衷心感谢导师 ××× 教授对本人的精心指导。他的言传身教将使我终生受益。

感谢 ××× 教授，以及实验室全体老师和同窗们的热情帮助和支持！

本课题承蒙 ×××× 基金资助，特此致谢。

以下内容为提示，仔细阅读后删除。

致谢应另起页，放置在参考文献、附录之后，标题和页眉均为“致谢”。语言要诚恳、恰当、简短。

致谢对象可以包括指导教师，在研究工作中提出建议和提供帮助的人，给予转载和引用权的资料、图片、文献、研究和调查的所有者，其他应感谢的组织和个人，资助研究工作的项目基金、奖学金基金、合同单位、资助或支持的企业、组织或个人，协助完成研究工作和提供便利条件的组织或个人。致谢字数以不超过一页纸为宜。

学位论文应由学生在导师（组）的指导下独立完成；**若涉及团队工作，应注明属于团队成果，并明确个人独立完成的内容**，科学严谨，恪守规范。

个人简历、在学期间完成的相关学术成果

个人简历

×××× 年 ×× 月 ×× 日出生于 ××××。

×××× 年 ×× 月考入 ×× 大学 ×× 院（系）×× 专业，×××× 年 ×× 月本科毕业并获得 ×× 学学士学位。

×××× 年 ×× 月——×××× 年 ×× 月，在 ×× 大学 ×× 院（系）×× 学科学习并攻读（获得）×× 学硕士学位。【注：博士生已获得硕士学位写“获得”，硕士生申请硕士学位应写“攻读”，本括号在使用时请删除】

获奖情况：如获三好学生、优秀团干部、× 奖学金等（不含科研学术获奖）。

工作经历：……

在学期间完成的相关学术成果

特别注意，下面的引用文献部分需要使用半角括号，例如 [J]，(已被 xxxx 录用)。(本行在使用时请删除)。

学术论文

[1] Pei S, Huang L L, Li G, et al. Magnetic Raman continuum in single-crystalline $H_3LiIr_2O_6$[J]. Physical Review B, 2020, 101(20): 201101. (SCI 收录, IDS 号为 LJ4UN, IF=3. 575, 对应学位论文 2.2 节和第 5 章.)

[2] Pei S, Tang J, Liu C, et al. Orbital-fluctuation freezing and magnetic-nonmagnetic phase transition in $\alpha-TiBr_3$[J]. Applied Physics Letters, 2020, 117(13): 133103. (SCI 收录, IDS 号为 NY3GK, IF=3. 597, 对应学位论文 2.2 节和第 3 章.)

申请及已获得的专利（无专利时此项不必列出）

[3] 任天令, 杨轶, 朱一平, 等. 硅基铁电微声学传感器畴极化区域控制和电极连接的方法: 中国, CN1602118A[P]. 2005-03-30.

[4] Ren T L, Yang Y, Zhu Y P, et al. Piezoelectric micro acoustic sensor based on ferroelectric materials: USA, No.11/215, 102[P]. (美国发明专利申请号.)

参与的科研项目及获奖情况（无获奖时此项不必列出）

[5] 姜锡洲，××××× 研究，×× 省自然科学基金项目。课题编号：××××，长长。

[6] ×××，××××× 研究，×× 省自然科学基金项目。课题编号：××××。

[7] ×××，××××× 研究，×× 省自然科学基金项目。课题编号：××××。