电 子 科 技 大 学

UNIVERSITY OF ELECTRONIC SCIENCE AND TECHNOLOGY OF CHINA

硕士学位论文

MASTER THESIS

论文题目　面向大模型推理服务的攻击检测研究

学科专业	信息与通信工程
学　　号	**081000**
作者姓名	邓　晨
指导教师	于富财　　副教授
学　　院	信息与通信工程学院

分类号 _____ 密级 _____

UDC[注 1] _____

学 位 论 文

面向大模型推理服务的攻击检测研究

（题名和副题名）

邓 晨

（作者姓名）

指导教师 于富财 副教授

电子科技大学 成 都

（姓名、职称、单位名称）

申请学位级别 硕士 学科专业 信息与通信工程

提交论文日期 2026 年 04 月 16 日 论文答辩日期 2026 年 05 月 30 日

学位授予单位和日期 电子科技大学 2026 年 6 月

答辩委员会主席 _____

评阅人 _____

注 1：注明《国际十进分类法 UDC》的类号。

Research on Attack Detection for Large Model Inference Services

A Master Thesis Submitted to
University of Electronic Science and Technology of China

Discipline **Information and Communication Engineering**

Student ID **202321011120**

Author **Deng Chen**

Supervisor **Professor Yu Fucai**

School **School of Information and Communication Engineering**

独创性声明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。据我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得电子科技大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示谢意。

作者签名：_____ 日期：　　年　　月　　日

论文使用授权

本学位论文作者完全了解电子科技大学有关保留、使用学位论文的规定，同意学校有权保留并向国家有关部门或机构送交论文的复印件和数字文档，允许论文被查阅。本人授权电子科技大学可以将学位论文的全部或部分内容编入有关数据库进行检索及下载，可以采用影印、扫描等复制手段保存、汇编学位论文。

（涉密的学位论文须按照国家及学校相关规定管理，在解密后适用于本授权。）

作者签名：_____ 导师签名：_____

日期：　　年　　月　　日

摘　要

杨过少年时期母亲染病而亡，随后他便过着四处流浪的生活。

关键词：练武；离经叛道；复仇；抗敌；练武；离经叛道；复仇；抗敌；练武；离经叛道；复仇；抗敌；练武；离经叛道；复仇；抗敌；注意：2025 年 09 月 03 日修订的研究生论文撰写规范要求改用“；”分隔关键词；（双学位）学士论文撰写规范尚未调整，仍使用“，”分隔关键词

ABSTRACT

When Yang was a teenager, his mother contracted a disease and died,

Keywords: Martial arts；Apostasy；Revenge；Fighting against the enemy；Martial arts；Apostasy；Revenge；Fighting against the enemy

目 录

缩略词表

英文缩写	英文全称	中文全称
LLM	Large Language Model	大语言模型
TPOT	Time Per Output Token	每 Token 平均延迟
TTFT	Time to First Token	首 Token 延迟

第一章 绪论

1.1 研究背景与意义

随着深度学习技术的突破，以 Transformer 为核心架构的大语言模型（LLM）在自然语言处理领域展现出了前所未有的理解与生成能力，标志着人工智能正式步入生成式时代。在这种技术浪潮下，企业与开发者日益倾向于采用“模型即服务”（Model-as-a-Service, MaaS）的部署模式，将庞大的模型托管于云端高性能算力集群中，通过标准化的 API 对外提供推理能力。这种高度集约化的云服务模式虽然极大地降低了用户端的使用门槛，但也使得云端推理系统的稳定性和安全性成为了支撑大规模 AI 应用的关键基石。

然而，大语言模型的推理过程与传统的 Web 服务或计算任务存在本质差异，其表现出显著的计算密集型与内存敏感型特征。为了应对海量用户并发请求并降低推理延迟，现代推理框架如 vLLM 引入了 PagedAttention 与 Prefix Caching 等先进的内存管理技术。这些技术通过对键值缓存（KV Cache）的精细化调度，虽然显著提升了吞吐量，但也无意中暴露了新的安全边界。攻击者可以利用推理系统对资源消耗的高度敏感性，通过构造特定模式的输入请求，诱发系统出现资源耗尽或性能剧烈抖动，这种针对算法逻辑与系统架构层面的攻击手段比传统的网络层攻击更具隐蔽性。

在现有的网络安全防御体系中，传统的防火墙、入侵检测系统（IDS）以及 Web 应用防火墙（WAF）主要聚焦于网络协议层的特征匹配或已知恶意代码的拦截。对于大模型推理服务而言，攻击请求在格式上通常表现为完全合法的自然语言指令，但在推理执行阶段却会引发异常的显存占用或超长计算时间。例如，攻击者可能利用 Uniform 分布的请求模式破坏缓存的局部性原理，从而实施缓存侧信道干扰；或者通过高并发短文本请求制造拒绝服务攻击，压垮调度系统。这种“应用逻辑层”的资源压榨行为使得现有的基于流量签名的防御手段难以奏效，迫切需要研究一种能够深度感知模型推理行为、捕捉底层资源波动特征的新型检测机制。

本研究的开展在学术理论维度具有重要意义，其通过对大模型推理过程中的多维性能指标进行精细化建模，填补了生成式 AI 基础设施安全防御的研究空白。研究不再局限于宏观的吞吐量统计，而是深入探讨了首 Token 延迟（TTFT）、每 Token 平均延迟（TPOT）以及 KV Cache 命中率等微观指标与攻击行为之间的内在耦合关系。通过对比 Qwen 的不同参数规模模型在受攻击状态下的特征表现，本研究揭示了攻击流量在系统层留下的“特征足迹”，为构建大模型环境下异常行为

的特征工程提供了科学的方法论指导，有助于完善人工智能安全保障体系。

在实际应用价值方面，本研究所提出的攻击检测方案能够为云服务提供商提供坚实的安全防护屏障。在昂贵的 GPU 算力资源面前，任何形式的资源耗尽攻击都会直接导致服务商运营成本的激增以及服务等级协议（SLA）的违约。有效的检测系统能够精准识别旨在干扰 KV Cache 机制的隐蔽攻击，在保障正常用户请求获得低延迟响应的同时，极大程度地提高了算力资源的有效利用率。通过实时监控并阻断针对推理框架漏洞的恶意嗅探与冲击，可以有效防止系统雪崩效应的发生，确保大规模 AI 推理服务的可用性与经济性。

此外，针对缓存侧信道等涉及多租户环境隐私与公平性的新型威胁，本研究提出的检测方法具备极强的实战防御意义。侧信道攻击往往通过探测缓存池状态的细微波动来推断其他用户的访问模式或干扰系统公平调度，这类攻击具有极高的技术门槛和防御难度。本研究通过对显存迁移频次、缓存失效分布等关键指标的异常分析，能够实现对此类高级持续性威胁（APT）的早期预警与快速响应。这不仅为构建可信人工智能环境提供了关键技术支持，也为生成式人工智能在金融、政务等对安全性要求严苛领域的深层次落地应用扫清了障碍。

1.2 国内外研究现状

为有效应对大模型推理服务面临的安全威胁，国内外学术界和工业界已经展开了多维度的研究探索。通过阅读大量相关文献，本文对目前常见的恶意请求检测与防御方法进行了总结归纳，主要将其分为三类：基于输入内容的攻击检测、基于系统行为与微观架构的攻击检测，以及基于分布外检测（OOD）与机器学习的异常识别方法。本小节将依次进行介绍。

1.2.1 基于输入内容的攻击检测

基于输入内容的攻击检测是目前大语言模型安全领域最主流、应用最广的研究赛道，主要针对提示词注入（Prompt Injection）、越狱攻击（Jailbreak）以及对抗样本攻击 [1]。此类方法的核心思想是通过对用户输入的自然语言指令进行语义分析、特征提取或困惑度（Perplexity, PPL）计算，从而拦截恶意请求。

在对抗攻击防御方面，Jain 等人对齐语言模型面临的对抗样本进行了系统性研究，并证明了基于困惑度（PPL）的启发式过滤器能够作为强基线防御手段，有效检测并拦截某些由乱码或异常词汇构成的对抗攻击 [2]。Armstrong 等人也深入探讨了在自然语言处理中检测对抗样本的多种特征工程方法 [1]。然而，这类基于语义和内容特征的检测方法存在一个致命的盲区：当攻击者旨在发起资源耗尽型

DDoS 攻击或缓存侧信道攻击时，其构造的提示词（Prompt）在语义上往往是完全合法的正常语句（例如我们在压测中使用的常规长文本或随机短文本）。此时，传统的输入内容过滤机制将完全失效，恶意请求会畅通无阻地进入推理计算底座。

1.2.2 基于系统行为与微观架构的攻击检测

随着大模型云服务（MaaS）的普及，攻击者的目标逐渐从“欺骗模型生成有害内容”转向“压榨底层算力与窃取隐私”。因此，基于系统行为和微观架构（尤其是针对 KV Cache 和调度机制）的攻击与检测成为了当前安全研究的最前沿。

在针对大模型推理接口的定时侧信道（Timing Side-Channel）研究方面，Carlini 等人率先证明了可以通过远程定时攻击来推断高效语言模型的内部状态 [3]。随后，Song 等人进一步揭示了大型语言模型服务系统（如 vLLM）中由于批处理（Batching）和缓存机制引入的定时侧信道漏洞 [4]。Zhang 等人则从输出层入手，展示了如何通过输出 Token 数量的定时波动来建立侧信道 [5]。

针对云端共享 KV Cache 带来的多租户隔离问题，国内外学者进行了深入剖析。Zheng 等人提出 InputSnatch 攻击，证明了在共享大模型服务中，攻击者可以通过定时侧信道窃取其他用户的输入信息 [6]。Luo 等人也发文揭示了 KV Cache 在大模型推断中隐藏的严重隐私风险，并初步探讨了缓解机制 [7]。此外，McDonald 等人提出的 Whisper Leak 端侧信道攻击，更是通过系统资源消耗特征的细微变化，实现了对 LLM 内部计算过程的窥探 [8]。

在宏观的云端拒绝服务（DDoS）检测方面，Deniz 和 Serttaş 针对云网络架构，构建了基于卷积神经网络（CNN）和长短期记忆网络（LSTM）的深度学习分布式拒绝服务检测系统，并取得了高达 98% 的准确率 [9]。然而，传统的云原生 DDoS 检测多基于网络连接数或带宽阈值，尚未充分结合大模型预填充（Prefill）与解码（Decode）双阶段的非线性算力消耗特征。

1.2.3 基于机器学习与分布外检测（OOD）的防御方法

鉴于大模型推理过程中的攻击流量（如并发压榨或 Uniform 分布的缓存污染）往往在首 Token 延迟（TTFT）和每 Token 平均延迟（TPOT）等时间序列指标上表现出异常波动，基于机器学习（ML）与分布外检测（Out-of-Distribution, OOD）的技术成为了构建防御体系的重要手段。

在传统的硬件与缓存侧信道检测中，Tong 等人已经证明了基于机器学习分类器能够高精度地识别缓存侧信道攻击的微架构足迹 [10]。针对多维时间序列的异常检测问题，Malhotra 等人提出了基于 LSTM 的编码器-解码器架构，通过学习正常系统状态的重构误差，有效识别多传感器环境下的分布外异常点 [11]。

这些研究为大模型环境下的异常流量识别奠定了理论基础。综上所述，尽管目前针对 NLP 语义的防御和传统云安全的检测方法已相对成熟，但面向大模型推理引擎底层的资源压榨（DDoS）与缓存干扰（KV Cache Side-Channel）的针对性检测机制仍存在较大空白。这正是本文引入 TTFT 双峰分离特征与多维时间序列分析，利用机器学习算法开展攻击检测研究的核心动机。

1.3 论文主要工作及创新

本文针对云服务环境下大语言模型（LLM）推理服务面临的资源安全威胁，构建了一套全流程的异常检测与分析体系。研究立足于高性能推理框架 vLLM，通过深度解析其 PagedAttention 内存管理机制与 Prefix Caching 缓存策略，剖析了系统在应对非典型流量冲击时的脆弱性本质。为实现对推理行为的精准表征，本文设计并实现了一套多维指标监控系统，能够实时采集首 Token 延迟（TTFT）、每 Token 平均延迟（TPOT）及 KV Cache 命中率等高保真底层特征，并在此基础上开展了如下针对性研究：

（一）针对大模型推理过程中“合法指令”掩盖“恶意负载”导致的检测难问题。本文提出了基于“双峰分离”物理特征的推理攻击识别方法。在实验验证阶段，通过在 A100 环境下对比 Qwen1.5-1.8B 与 Qwen2.5-7B 模型发现，由于小模型计算延迟极低导致正常与攻击特征高度重叠，而随着模型参数规模提升至 7B，正常请求与高并发 DDoS 攻击请求在首 Token 延迟（TTFT）上的分布会演变为清晰的双峰分离态势。本文利用这一规模敏感型特征，突破了传统网络层检测无法感知模型内部计算状态的局限，显著提升了在复杂并发环境下识别资源压榨型攻击的鲁棒性。

（二）针对利用 LLM 缓存逻辑缺陷实施的隐蔽侧信道攻击问题。本文创新性地构建了深度关联推理引擎内存调度层的检测机制。不同于侧重文本内容的审计手段，本文重点关注攻击者通过构造符合 Uniform 分布的随机请求来破坏缓存局部性的行为。研究揭示了攻击请求如何通过诱导 KV Cache 命中率从高位（Zipf 分布状态）剧烈下滑，并引发显存频繁迁移，进而造成系统整体吞吐性能退化。通过引入孤立森林等机器学习算法，本文实现了从请求分布偏移视角对隐蔽性侧信道干扰的高效捕捉，为防御大模型特有机制的逻辑攻击提供了全新观测维度。

（三）针对安全检测系统在高并发环境下对推理性能产生侵入式干扰的问题。本文实现了一套具有高可扩展性的轻量化监控与检测框架。该框架能够无缝集成于生产级推理引擎 vLLM 之中，通过多维特征关联分析方法，将复杂的底层资源波动转化为直观的攻击判定指标。实验评价表明，该方案在保障高准确率与召回

率的同时，极大地降低了监控模块对 GPU 推理性能的额外开销。这一工程实践为云端大模型服务的安全基座建设提供了具备实际应用价值的解决方案，确保了安全防御与服务效能的平衡。

1.4 论文结构安排

本文共分为五个章节，每章的主要内容如下所示：

第一章作为绪论，主要介绍了大语言模型在生成式人工智能浪潮下的核心地位，以及“模型即服务”（MaaS）模式普及带来的安全挑战。本章重点阐述了 LLM 推理系统在资源调度与缓存管理方面的脆弱性，明确了研究背景与意义，并概括了本文在特征工程与异常检测方面的创新性工作。

第二章系统梳理了相关技术基础与算法原理。本章首先深入剖析了 Transformer 架构的解码机制，详细解释了键值缓存（KV Cache）在减少重复计算中的关键作用。随后，本章重点探讨了 vLLM 推理框架的核心机制，包括 PagedAttention 如何实现非连续内存分配以及 Prefix Caching 如何提升前缀复用效率。此外，本章还介绍了孤立森林等统计学与机器学习算法，为后续检测模型的构建奠定理论根基。

第三章重点研究基于多维资源特征的拒绝服务（DDoS）攻击检测。本章首先详细阐述了针对 LLM 推理服务的异常检测系统架构，包括流量生成、推理服务、多维指标采集及实时检测分析四大模块的设计实现。在本章构建的监控体系下，针对高并发短文本引发的拒绝服务场景进行了深度建模。通过对 Qwen2.5-7B 等模型在不同并发压力下的表现进行实验，重点分析了首 Token 延迟（TTFT）等指标在受攻击状态下表现出的“双峰分离”现象。最后，本章通过对比不同检测策略，验证了该方案在识别流量型攻击方面的准确性与时效性。

第四章深入探讨了基于缓存侧信道的攻击检测研究。不同于前一章的流量冲击，本章关注于利用逻辑缺陷实施的隐蔽攻击，如通过 Uniform 分布请求破坏缓存局部性。本章在原有监控系统的基础上，引入了针对缓存命中率与显存迁移频次的细粒度表征方法。通过分析攻击行为如何诱发缓存热度失真与推理性能退化，本章提出了一种基于缓存突变特征的实时检测算法。通过调节 Zipf 分布参数与攻击强度，实验验证了该方法在保护多租户环境下缓存池安全方面的有效性。

第五章对全文的研究工作进行了总结，并对未来的研究方向进行了展望。本章归纳了本文在 LLM 推理层安全防御、多维特征关联分析以及系统工程实现方面的主要贡献，并针对当前研究在异构算力环境下的局限性，提出了未来在自适应防御机制与跨框架安全增强方面的探索思路。

第二章 相关技术基础和方法

2.1 大语言模型推理机制与算力瓶颈

2.1.1 Transformer 架构与自回归解码

目前主流的大语言模型（Large Language Models, LLM），如 GPT 系列、LLaMA 以及本文实验所采用的 Qwen 系列，均采用仅解码器（Decoder-only）的 Transformer 架构。其核心特征是自回归生成（Autoregressive Generation），即模型在给定上文的条件下，通过概率分布逐个预测序列中的下一个标记（Token）。

假设输入序列为 $\mathbf{X}_{1:n} = \{x_1, x_2, \ldots, x_n\}$，其中 $x_i \in \mathbb{R}^{d_{model}}$ 为经过词表嵌入（Embedding）后的特征向量。在生成第 $n+1$ 个 Token 时，模型的目标是最大化条件概率：

$$P(x_{n+1} \mid \mathbf{X}_{1:n}) = \text{Softmax}\left(\mathbf{W}_u \cdot \text{DecoderBlock}(\mathbf{X}_{1:n})\right) \tag{2-1}$$

其中，$\mathbf{W}_u$ 为最终的输出词表映射矩阵（Unembedding Matrix）。

在 Transformer 的每个解码器层中，最核心的计算单元为掩码多头自注意力机制（Masked Multi-Head Self-Attention）。对于输入矩阵 $\mathbf{X} \in \mathbb{R}^{L \times d_{model}}$（$L$ 为序列长度），系统首先通过三个可学习的线性权重矩阵 $\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V \in \mathbb{R}^{d_{model} \times d_k}$ 将其投影为查询矩阵（Query）、键矩阵（Key）和值矩阵（Value）：

$$\mathbf{Q} = \mathbf{X}\mathbf{W}_Q, \quad \mathbf{K} = \mathbf{X}\mathbf{W}_K, \quad \mathbf{V} = \mathbf{X}\mathbf{W}_V \tag{2-2}$$

随后，通过缩放点积注意力计算当前 Token 与历史 Token 的关联权重：

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V} \tag{2-3}$$

其中，d_k 为注意力头的维度。在朴素的自回归生成中，由于序列长度 L 会随着生成的进行不断增加，第 t 步的计算不仅需要处理新生成的 Token，还需要重新计算前 $t-1$ 个历史 Token 的 $\mathbf{K}$ 和 $\mathbf{V}$ 矩阵，导致计算复杂度呈 $\mathcal{O}(L^2)$ 的二次方级增长，造成了极大的算力冗余。

2.1.2 KV Cache 加速原理与内存映射

为了消除上述 $\mathcal{O}(L^2)$ 复杂度的冗余特征提取计算，现代 LLM 推理框架普遍引入了键值缓存（KV Cache）机制。其核心思想是“以空间换时间”：在逐词生成的过程中，将历史 Token 计算得到的键向量 k 和值向量 v 缓存在 GPU 的物理显存（HBM）中，供后续生成步直接复用。

如图2-1所示，以生成第 3 个 Token 为例，本节对有无 KV Cache 机制的内部计算流程进行了对比说明。

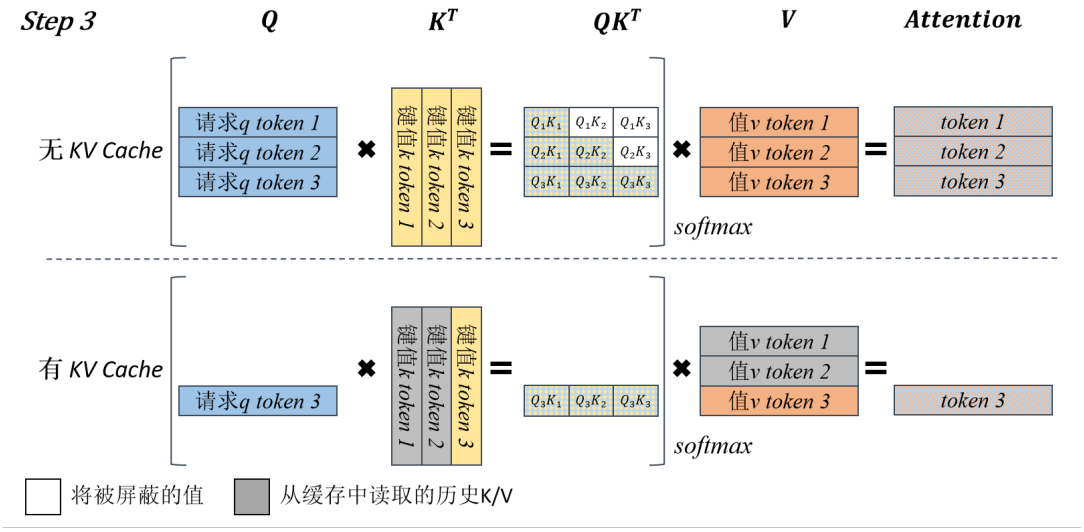

图 2-1 第 3 步推理有无 KVC 加速对比示意图

在无 KV Cache 的全量计算模式下（图左侧），系统必须重新将前两个 Token 与当前 Token 拼接，执行完整的矩阵乘法（GEMM）运算。而在引入 KV Cache 的增量计算模式下（图右侧），设当前为第 t 步，此时显存中已经保存了前 $t-1$ 步的缓存矩阵 $\mathbf{K}_{\text{cache}}^{(t-1)}$ 与 $\mathbf{V}_{\text{cache}}^{(t-1)}$。模型仅需计算当前最新 Token x_t 对应的查询向量 $q_t \in \mathbb{R}^{1\times d_k}$、键向量 k_t 与值向量 v_t。随后，系统将新的 k_t, v_t 向量与历史缓存沿序列长度维度进行张量拼接（Concatenation）：

$$\mathbf{K}_{\text{cache}}^{(t)} = \left[\mathbf{K}_{\text{cache}}^{(t-1)} \oplus k_t\right], \quad \mathbf{V}_{\text{cache}}^{(t)} = \left[\mathbf{V}_{\text{cache}}^{(t-1)} \oplus v_t\right] \tag{2-4}$$

此时，第 t 步的注意力计算被大幅简化为一次矩阵-向量乘法（GEMV）：

$$\text{Attention}(q_t, \mathbf{K}_{\text{cache}}^{(t)}, \mathbf{V}_{\text{cache}}^{(t)}) = \text{Softmax}\left(\frac{q_t(\mathbf{K}_{\text{cache}}^{(t)})^T}{\sqrt{d_k}}\right)\mathbf{V}_{\text{cache}}^{(t)} \tag{2-5}$$

如图2-1中的灰色区块所示，历史特征被直接从显存中读取，极大地提升了推理速度。然而，这种机制也导致显存的占用量随着并发请求数和生成长度的增加而线性膨胀，使 KV Cache 成为大模型推理系统中最核心也最脆弱的资源池。

2.1.3 双阶段推理的算力瓶颈分析：Prefill 与 Decode

在引入 KV Cache 后，大语言模型的单次推理生命周期在物理上被明确划分为具有截然不同资源消耗特征的阶段：预填充阶段（Prefill Phase）与解码阶段（Decoding Phase）。这种底层物理特征的割裂，正是本文后续进行多维异常特征提取与攻击检测的理论根基。

为了更直观地阐述这种物理层面的割裂，本文绘制了大模型双阶段推理机制与底层资源消耗的对比图。如图2-2所示，本文将大模型单次推理的生命周期及其底层硬件状态进行了可视化分解。

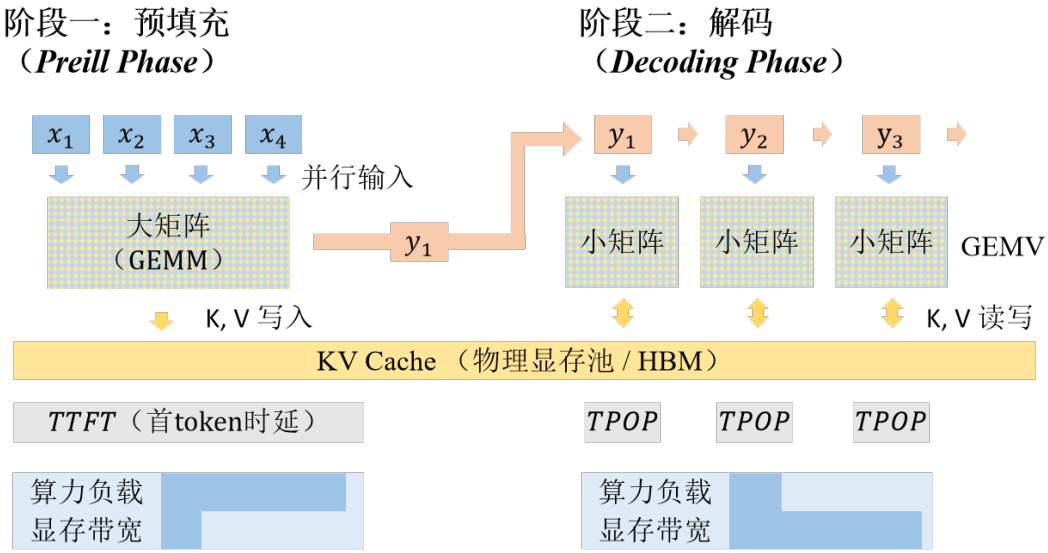

图 2-2 大语言模型双阶段推理机制与底层资源消耗对比图

（1）预填充阶段（Prefill Phase）与计算墙（Compute-bound）

当用户请求首次到达系统时，由于此时 KV Cache 为空，模型需要并行处理用户输入的全部提示词（Prompt）序列（长度为 L）。这是一个典型的大规模矩阵-矩阵乘法（GEMM）过程。现代 GPU（如 NVIDIA A100）内部的 Tensor Core 在执行此类高并发并行运算时效率极高，因此该阶段的瓶颈在于算力（FLOPs），而非内存带宽，被称为计算密集型（Compute-bound）任务。用户感知到的首 Token 延迟（Time to First Token, TTFT）主要由该阶段的计算耗时以及在系统调度队列中的等待耗时决定。在面对本文第三章所述的 DDoS 攻击时，高并发的短文本请求会瞬间榨干 GPU 的算力资源，导致 TTFT 剧烈飙升。

（2）解码阶段（Decoding Phase）与内存墙（Memory-bound）

进入逐词生成阶段后，由于每次只输入一个新的 Token，注意力计算退化为矩阵-向量乘法（GEMV）。尽管算力需求骤降，但为了计算这一个 Token，GPU 必须从高带宽显存（HBM）中将庞大的历史 KV Cache 数据完整地搬运到静态随机存取存储器（SRAM）中。此时，系统的性能瓶颈转移至显存带宽，成为访存密集型（Memory-bound）任务，决定了每 Token 平均延迟（Time Per Output Token, TPOT）。当遭受本文第四章所述的缓存侧信道攻击时，由于攻击者构造的 Uniform 分布请求不断触发 Cache Miss，迫使系统频繁在 Prefill 和 Decode 状态间切换并进行低效的显存页调度，从而在 TTFT 和 TPOT 上留下明显的畸变特征。

2.2 高性能推理系统的内存调度机制

在明确了大模型推理双阶段的算力与显存瓶颈后，如何高效管理显存中的 KV Cache 成为了提升系统并发处理能力（CONC）的关键所在。本文实验所采用的 vLLM 引擎，通过引入操作系统级别的虚拟内存管理思想，从根本上重构了 KV Cache 的调度机制。本节将深入剖析其核心的 PagedAttention 技术与 Prefix Caching 策略。

2.2.1 传统内存管理的碎片化困境

在传统的深度学习推理框架（如早期的 FasterTransformer 或 HuggingFace Accelerate）中，系统通常采用连续显存分配策略。由于大语言模型在生成初期无法预知最终输出的 Token 数量，系统只能根据模型支持的最大序列长度（Max Sequence Length）为每个请求预先静态分配一块庞大且连续的 GPU 显存空间。

这种“最坏情况假设”导致了极其严重的内存浪费。具体表现为：

- **内部碎片（Internal Fragmentation）**：当实际生成的序列远短于预留长度时，剩余的预留显存被长期闲置且无法被其他请求使用。
- **外部碎片（External Fragmentation）**：由于不同请求释放显存的时间不一，物理显存中会产生大量不连续的空闲片段。当新的大内存需求到来时，即使总空闲容量足够，也因缺乏连续空间而触发内存溢出（Out of Memory, OOM）。

在高并发的多租户云服务环境中，这种低效的内存管理模式极大地限制了系统吞吐量，也使得系统在面对海量突发请求（如拒绝服务攻击）时极为脆弱。

2.2.2 PagedAttention 原理与块表映射

为了打破连续内存分配的桎梏，vLLM 提出了 PagedAttention 算法。该算法借鉴了现代操作系统中的虚拟内存与分页（Paging）机制，将原本必须连续存储的 KV Cache 划分为固定大小的逻辑块（Logical Blocks），并将其动态映射到非连续的物理显存块（Physical Blocks）上。

在 PagedAttention 中，每个物理块能够容纳固定数量的 Token（例如在本文实验配置中，默认块大小为 16 个 Token）。系统通过维护一张全局的块表（Block Table），记录每个请求的逻辑块到物理块的映射关系：

$$\text{BlockTable}(\text{Req}_i) = \{pb_1, pb_2, \ldots, pb_m\} \tag{2-6}$$

其中，Req_i 为第 i 个请求，pb 为分配的物理块索引。当模型在解码阶段生成新的 Token 时，系统只需按需分配新的物理块并更新块表，而无需进行连续的内存预留。

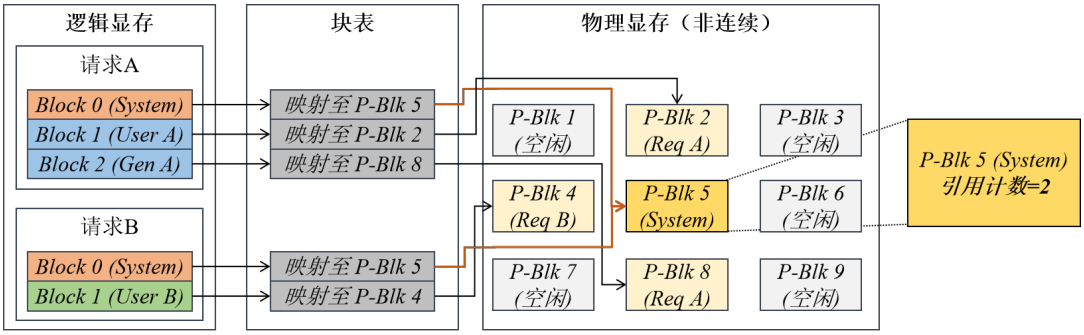

图 2-3 PagedAttention 逻辑块映射与 Prefix Caching 共享复用机制

如图2-3所示，在 PagedAttention 机制的调度下，连续的逻辑序列被切分为大小固定的逻辑块（Logical Blocks），并通过块表（Block Table）离散地映射到 GPU 物理显存中的非连续物理块（Physical Blocks）上，从而彻底消除了传统内存分配中的内部碎片。

在此基础上，图例进一步展示了 Prefix Caching 的前缀复用工作原理：假设请求 A（Req A）与请求 B（Req B）在输入时包含了相同的系统前缀（如相同的人设指令）。vLLM 引擎会计算该前缀逻辑块的哈希值，并发现物理显存中已存在对应的缓存块。此时，系统不再为请求 B 分配新的物理空间，而是将其对应的逻辑块指针直接映射至请求 A 已生成的物理块上，并同步更新该物理块的引用计数（Reference Count）。这种内存页级别的细粒度共享与按需分配机制，在极大提升系统并发上限（即本文实验中的 CONC 指标）的同时，也使得物理显存池成为了多租户之间相互影响的共享媒介，为后续的侧信道干扰埋下了伏笔。

通过这种页式内存管理，PagedAttention 彻底消除了内部碎片，并将外部碎片率控制在极低水平（仅最后一个未填满的块存在微小浪费），从而允许系统在同一块 A100 GPU 上承载远超传统框架的并发请求数（即本文压测实验中极高的 CONC 值）。然而，复杂的逻辑地址映射也为系统底层的调度器带来了额外的算力开销。

2.2.3 Prefix Caching 与前缀复用漏洞

在实际的“模型即服务（MaaS）”场景中，大量的用户请求往往包含相同的系统提示词（System Prompt）、任务指令或长背景文档。为了进一步压榨显存价值，vLLM 在 PagedAttention 的基础上引入了 Prefix Caching（前缀缓存）机制。

Prefix Caching 的核心逻辑是实现多租户间的物理块共享。当新请求到达时，系统会对其输入的 Prompt 按逻辑块进行哈希（Hash）计算：

$$\text{Hash}_{block} = \mathcal{H}(\text{Token}_1, \ldots, \text{Token}_k) \tag{2-7}$$

系统在全局哈希表中检索该哈希值。若发生缓存命中（Cache Hit），则不再为该请

求分配新的物理块，而是直接将该请求的逻辑块指针指向已存在于物理显存中的共享块，并增加该物理块的引用计数（Reference Count）。

正常长尾分布与缓存局部性：在正常的业务流量中，用户请求通常符合 Zipf 分布（长尾分布），少数热点提示词被高频复用，此时 Prefix Caching 能维持极高的命中率，显著降低 TTFT 并提升吞吐量。

新型安全威胁面的暴露：然而，这种基于多租户共享的缓存机制也无意中暴露了侧信道安全边界。当攻击者恶意构造符合 Uniform 分布（完全随机分布）的大量请求时，不仅无法命中任何缓存，还会迫使系统频繁为新的“冷前缀”分配物理块。当物理显存耗尽时，系统将依据置换算法（如 LRU）无差别地驱逐（Evict）正常用户的热点缓存。这种利用合法 API 实施的“缓存污染（Cache Pollution）”攻击，能够在不引起网络层流量异常的情况下，精准破坏大模型推理的底层局部性原理。这也构成了本文第四章基于缓存侧信道进行异常检测的物理与逻辑基础。

2.3 针对大模型推理服务的新型安全威胁模型

在深入理解 vLLM 的双阶段算力瓶颈与 PagedAttention 内存调度机制后，本节将正式定义本文研究的安全威胁模型（Threat Model）。与传统的网络层拒绝服务攻击不同，针对大语言模型（LLM）的攻击具有“应用层合法、物理层致命”的隐蔽特征。

2.3.1 攻击者能力与假设

在本文的威胁模型中，假设云服务商采用“模型即服务（MaaS）”架构对外提供 API 接口。攻击者（Attacker）具备以下能力与限制：

- **黑盒访问权限：**攻击者无法直接获取服务器的物理控制权，无法修改模型权重，也无法读取宿主机的底层操作系统日志。其唯一的交互途径是通过公开的 API 端点发送自然语言提示词（Prompt）并接收生成的 Token。
- **合法身份掩护：**攻击者拥有合法的注册账号与访问令牌（Token ID），其发送的请求内容在语义上完全符合应用层安全防火墙（WAF）的过滤规则，不存在 SQL 注入或传统越狱（Jailbreak）等违规字符串。
- **流量控制能力：**攻击者能够利用自动化脚本并发发送请求，并能自由控制输入 Prompt 的长度、并发数（CONC）以及请求内容的分布概率（如调整 Zipf 分布的 α 参数）。

2.3.2 拒绝服务（DDoS）攻击面：算力墙压榨

结合 2.1 节的双阶段理论，预填充（Prefill）阶段是极度消耗 GPU Tensor Core 算力的矩阵乘法过程。攻击者利用这一非线性算力消耗特征，实施大模型特有的 DDoS 攻击。

攻击者通过脚本在极短时间内发送海量（如并发数超过系统最大承载量）的“短文本”请求。由于短文本在网络传输层面占用的带宽极小，极易穿透传统的网络层流量清洗设备。然而，当这些请求涌入 vLLM 的调度器时，系统必须为每一个请求启动耗时的 Prefill 计算。瞬间爆发的算力需求会迅速榨干 GPU 的计算资源，导致后续所有正常用户的请求被强行滞留在等待队列（Waiting Queue）中。宏观上表现为系统的首 Token 延迟（TTFT）呈指数级飙升，直至服务彻底瘫痪。

2.3.3 缓存侧信道（Side-Channel）攻击面：内存墙穿透

针对 2.2 节中 Prefix Caching 的共享复用机制，攻击者可实施更为隐蔽的缓存污染（Cache Pollution）侧信道攻击。

在多租户环境中，正常用户的请求通常符合长尾的 Zipf 分布。攻击者为了破坏这种局部性，故意构造大量前缀互不相同、符合均匀分布（Uniform Distribution）的请求。这些“冷请求”进入系统后，由于无法命中任何哈希树节点，迫使 PagedAttention 机制不断在物理显存中分配新的物理块。当 80GB 的 A100 显存池被这些无意义的冷请求填满后，系统被迫触发 LRU（最近最少使用）淘汰机制，将正常用户的“热点缓存块”无差别驱逐。

这种攻击的致命之处在于：它并没有耗尽网络连接，也没有发送极高并发，但它通过逻辑机制迫使正常用户的后续请求发生 Cache Miss。正常请求不得不从轻量级的解码（Decode）阶段退回到沉重的预填充（Prefill）阶段重新计算。这不仅极大地浪费了显存带宽（引发“内存墙”阻塞），还导致系统吞吐量断崖式下跌。

2.4 异常检测与机器学习分类理论

面对上述隐蔽且复杂的底层资源压榨攻击，传统的固定阈值告警机制显得捉襟见肘。本文引入滑动窗口机制与 LightGBM 机器学习算法，构建了从特征工程到多分类识别的完整理论链路。

2.4.1 时间序列统计特征与双峰判定

单个请求的延迟往往包含系统随机噪声，为了过滤瞬态抖动，本文引入了滑动时间窗口（Sliding Window）机制。给定按时间戳排序的观测序列

$\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$，对于大小为 W 的滑动窗口，当前时刻 t 的局部均值 μ_t 与局部标准差 σ_t 分别定义为：

$$\mu_t = \frac{1}{W} \sum_{i=t-W+1}^{t} x_i, \quad \sigma_t = \sqrt{\frac{1}{W-1} \sum_{i=t-W+1}^{t} (x_i - \mu_t)^2} \tag{2-8}$$

此外，本文在实验中发现，正常流量与攻击流量在 TTFT 分布上会呈现出显著的“双峰分离（Twin Peaks）”现象。为了从数学上刻画这种分布畸变，引入了峰度（Kurtosis）这一高阶统计量：

$$\text{Kurtosis}(\mathcal{X}) = \frac{\mathbb{E}[(\mathcal{X} - \mu)^4]}{\sigma^4} - 3 \tag{2-9}$$

在遭受缓存侧信道攻击时，TTFT 的分布将显著右移并拉出长尾，表现为峰度指标的剧烈突变。本文提取的核心多维特征如表2-1所示。

表 2-1 大模型推理异常检测特征工程清单

特征维度	特征名称	物理意义与攻击映射
时延波动特征	TTFT 滑动均值	反映系统 Prefill 阶段的宏观排队情况（DDoS 高敏）
	TTFT 峰度 (Kurtosis)	刻画延迟分布的“双峰”与长尾现象（侧信道高敏）
	TPOT 滑动标准差	反映 Decode 阶段显存带宽的拥挤与抖动程度
缓存状态特征	KV Cache 命中率	衡量 Prefix Caching 的复用效率
	物理块驱逐频次	监控 Uniform 随机流量引发的缓存污染现象
硬件资源特征	GPU 等待队列长度	vLLM 调度器底层排队请求数（T_{wait} 核心来源）
	显存占用率波动比	甄别正常长文本与攻击流量的资源侵占差异

2.4.2 孤立森林（Isolation Forest）与 OOD 检测

云端推理系统常面临宿主机网络波动等非攻击造成的偶发性脏数据，这些分布外（OOD）异常点会干扰监督模型的训练。本文引入孤立森林（Isolation Forest）算法进行前置数据清洗。孤立森林的核心假设是异常样本在特征空间中具有“少而不同”的特性。对于输入样本 x，其异常评分 $s(x, n)$ 定义为：

$$s(x, n) = 2^{-\frac{\mathbb{E}(h(x))}{c(n)}} \tag{2-10}$$

其中：$h(x)$ 为样本 x 在随机树中的路径长度；$\mathbb{E}(h(x))$ 是 x 在多棵树中路径长度的期望值；$c(n)$ 是包含 n 个样本的二叉搜索树的平均路径长度。当 s 接近 1 时，即可判定该样本为系统噪声并予以隔离。

2.4.3 基于 LightGBM 的多维特征分类算法

在完成特征提取与异常点清洗后，攻击检测实质上转化为一个多分类问题（Label 0: 正常, Label 1: DDoS, Label 2: 侧信道）。本文采用轻量级梯度提升机（LightGBM）作为核心分类器，其基于直方图（Histogram-based）和单边梯度采样（GOSS）的决策树算法，不仅精度高，且推理延迟极低，避免了防御模块反向拖垮大模型推理性能。完整的特征提取与检测流程如算法2-1所示。

算法 2-1: 基于滑动窗口的大模型推理异常检测算法

输入: 1) 实时推理日志数据流 $\mathcal{S}$; 2) 滑动窗口大小 W; 3) 异常阈值 τ。
输出: 当前系统状态标签（Normal, DDoS, Side-Channel）。

初始化特征缓冲区 $\mathcal{B} = \emptyset$;
for *新请求记录* $r_t \in \mathcal{S}$ *到来* **do** **/* 持续监听 vLLM 日志 */**
  提取 r_t 的 TTFT, TPOT, 命中率等基础指标并压入 $\mathcal{B}$;
  if *当前缓冲区大小* $|\mathcal{B}| \geq W$ **then** **/* 缓冲区已满，开始计算 */**
    计算窗口内的滑动统计量：$\mathbf{F}_t = [\mu_t, \sigma_t, \text{Kurtosis}_t, \dots]$;
    利用孤立森林模型计算异常得分 s;
    if $s > \tau$ **then** **/* OOD 数据拦截 */**
      触发系统未知异常告警，丢弃脏数据样本;
    else
      // 送入 LightGBM 分类器进行攻击定性
      输入 $\mathbf{F}_t$ 至预训练的 LightGBM 模型;
      得出预测标签 $L_t \in \{0, 1, 2\}$;
      return L_t;
    移除缓冲区中最旧的数据，维持窗口滑动;
  else
    继续收集数据，保持系统静默;

2.5 本章小结

本章从底层硬件和系统架构的视角，系统梳理了贯穿本文研究的相关理论基础。首先，深入剖析了 Transformer 自回归解码机制与 KV Cache 空间换时间的加速原理，并明确指出了预填充（Prefill）与解码（Decode）双阶段分别面临的“计算墙”与“内存墙”物理瓶颈。其次，探讨了 vLLM 引擎中 PagedAttention 与 Prefix Caching 内存调度机制在提升吞吐量的同时所暴露的安全隐患。在此基础上，本文首次明确了针对大模型推理服务的拒绝服务攻击与缓存侧信道攻击的威胁模型，并给出了基于时间序列分析、孤立森林与 LightGBM 的异常检测数学理论。本章

的理论与数学推导，为后续第三章的流量压榨实验与第四章的隐蔽攻击检测奠定了极其坚实的逻辑根基。

第三章 基于多维资源特征的拒绝服务攻击

3.1 研究背景

随着生成式人工智能技术的爆发式增长，大语言模型（LLM）已成为云端计算服务的核心资产。在“模型即服务”（Model-as-a-Service, MaaS）的商业模式下，云服务商通过部署高性能推理引擎（如 vLLM、TGI 等）并利用算力集群（如 NVIDIA A100/H100）为多租户提供推理接口。然而，LLM 推理过程的高昂算力成本与复杂的内存管理机制，使其成为拒绝服务（Denial of Service, DDoS）攻击的新型高价值目标。传统的拒绝服务攻击通常发生在网络层或应用层（如 HTTP Flood），通过发送海量无意义的请求试图瘫痪目标服务器的带宽或连接数。

然而，在大模型推理环境下，一种更为隐蔽且致命的攻击形式正在演变：资源压榨型 DDoS 攻击。此类攻击的特征在于其请求在应用层完全合法，能够轻易穿透传统的 Web 应用防火墙（WAF），但其精心构造的负载（Payload）——如瞬时极高并发的短文本请求——会迫使推理引擎频繁触发计算密集的预填充（Prefill）阶段。LLM 的推理生命周期分为预填充（Prefill）与增量解码（Decoding）两个阶段。预填充阶段需要并行处理用户输入的全部提示词（Prompt），具有极高的计算密度。当攻击者利用脚本模拟数以百计的并发请求（如本研究中模拟的 $CONC = 200$ 场景）冲击推理引擎时，vLLM 的调度器（Scheduler）会出现严重的任务积压。此时，系统资源被恶意请求高度占用，导致正常用户的首 Token 延迟（TTFT）大幅攀升，甚至造成服务不可用。

目前，针对大模型推理服务的防御研究仍处于起步阶段。现有研究大多集中于文本内容的安全审计（如过滤有害提示词），而忽视了推理侧资源消耗的物理特征分布。在实际工程实践中，我们发现模型规模对异常特征的表征具有显著影响：在移动端或小型模型（如 Qwen1.5-1.8B）上，由于其 Prefill 阶段绝对时延极低，正常请求与攻击请求的特征分布往往存在高度重叠，难以实现精准识别。本研究在调研过程中发现，随着模型参数规模提升至服务器级（如从 1.8B 增加至 Qwen2.5-7B），硬件资源对计算压力的反馈信号得到了显著放大。在高性能 A100 算力环境下，这种信号放大效应在 TTFT 等指标上体现为独特的“双峰分离（Twin Peaks）”物理现象。这一发现为基于多维资源特征的攻击检测提供了全新的理论依据。因此，本章旨在通过建立精细化的监控指标体系，探究不同模型规模与并发压力下的特征关联性，并设计高效的异常检测算法，以保障云服务环境下 LLM 推理服务的可用性与鲁棒性。

3.2 大模型推理异常检测系统设计

为了实现对 LLM 推理服务异常行为的精准捕捉，本研究首先构建了一套具备多维指标感知能力的检测系统。该系统需在保障推理性能的前提下，实现从流量输入到特征分析的闭环管理。

3.2.1 系统总体架构

本系统主要由以下四个核心模块组成，如图3-1所示：

流量仿真层：模拟真实场景下的长尾流量（Zipf 分布）及攻击流量（如高并发短文本、Uniform 分布请求）。

推理服务器：基于 vLLM 框架部署 Qwen2.5-7B 模型，负责执行 Prefill 与 Decoding 任务。

全链路监控层：多维指标采集模块，通过 vLLM 暴露的 API 及系统底层监控，实时提取时延、缓存及资源利用率特征。

分析与检测层：运行异常检测算法（如孤立森林），对实时特征流进行判定并触发告警。

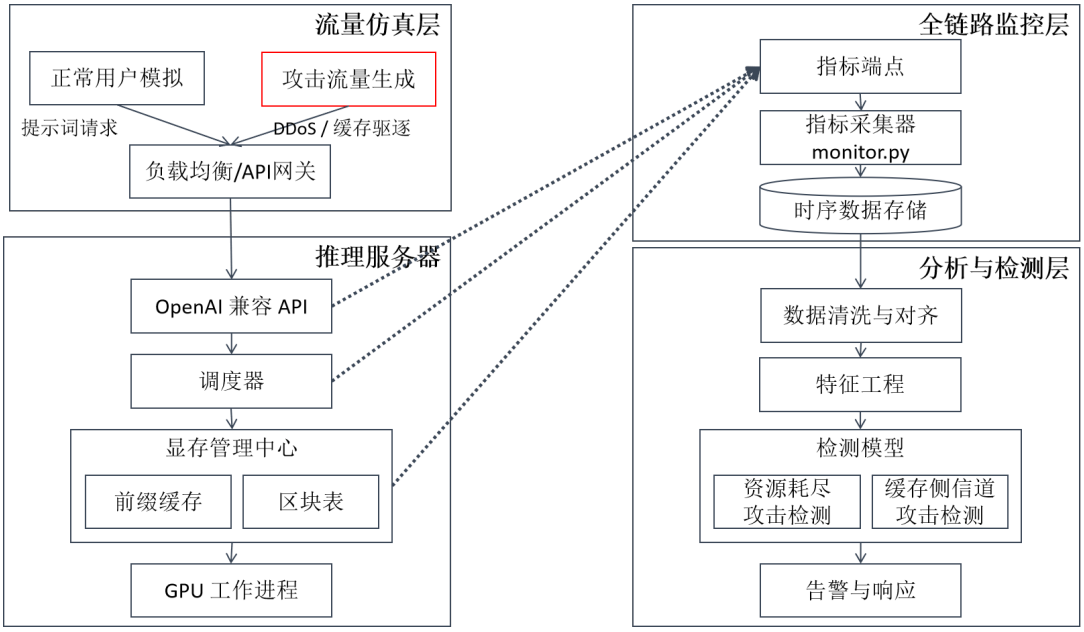

图 3-1 系统总体架构

3.2.2 推理请求生命周期分析

理解请求在系统内部的生命周期是定义监控指标的基础。一个典型的 LLM 推理请求从发送到响应，在系统内部经历的逻辑路径如图3-2所示。

（一）请求的生命周期始于流量生成模块。客户端根据业务需求构建 Prompt，

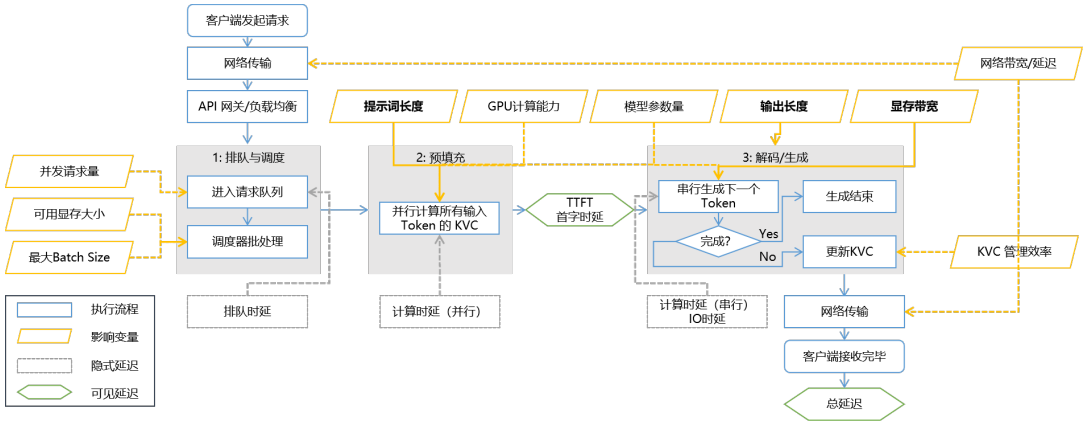

图 3-2 请求的生命周期

并通过负载均衡器进入系统。在此阶段，请求会首先被赋予一个全局唯一的时间戳。系统会根据当前推理引擎的负载状态，将请求置入运行队列（Running Queue）或等待队列（Waiting Queue）。如果是由于 DDoS 攻击（Label 1）导致的突发高并发流量，请求在此处的排队等待时间 T_{wait} 将显著增加，这直接构成了后续观测到的首 Token 延迟（TTFT）的主要组成部分。

（二）前缀检索与缓存决策阶段。当请求被调度器选中并分配至 GPU 算力资源后，系统进入逻辑检索层。由于开启了 Prefix Caching（前缀缓存）机制，vLLM 引擎会对输入 Prompt 进行哈希计算，并在物理显存的块表（Block Table）中检索是否存在可复用的 KV Cache。缓存命中：若命中（正常 Zipf 分布请求），则跳过冗余的计算，直接复用已有的 KV 向量。缓存失效：若未命中（如 Label 2 Uniform 分布请求），系统必须强制执行全量计算。此阶段是检测缓存侧信道攻击的核心节点，攻击者通过构造随机 Prompt 诱发频繁的缓存剔除与重建，导致 KV Cache Hit Rate 出现断崖式下跌。

（三）双阶段推理计算阶段。通过缓存决策后，请求正式进入 GPU 计算核心执行两阶段推理：预填充阶段（Prefill Phase）：模型并行处理全部输入 Token。这是计算量最大的环节，产生的延迟即为 $T_{prefill}$。在 Qwen2.5-7B 这种较大规模模型中，正常的 Prefill 计算与受攻击压制下的计算在耗时上会形成明显的波峰差异。增量解码阶段（Decoding Phase）：模型进入自回归循环，逐个生成 Token。每生成一个 Token，系统便将其对应的 K, V 向量存入由 PagedAttention 管理的物理内存块中，并实时更新显存占用指标。

（四）指标回传与异常判定阶段。随着最后一个 Token 或停止符（Stop Token）的生成，请求的物理周期结束，进入特征分析阶段。监控探针会将全生命周期内记录的数据（TTFT、TPOT、命中率、显存迁移频次等）整合为特征向量 $\mathbf{F}$。数据聚合：将客户端的时间戳差值与服务端引擎的内部状态关联。实时检测：特征向量

流向右侧的检测分析模块。算法（如孤立森林）会根据该请求在生命周期内表现出的资源消耗特征，将其标注为 Normal（正常）、DDoS Attack（流量型攻击）或 Side-Channel Attack（侧信道攻击），完成整个闭环的安全审计。

3.3 多维特征分析

3.3.1 监控指标选取标准

为了从多维度刻画系统状态，本文选取了以下指标构建特征向量 $\mathbf{F} = \{f_1, f_2, \ldots, f_n\}$：

- 时延维度：TTFT、TPOT。这是用户感知最直接的指标。
- 吞吐维度：Throughput (tokens/s)。反映系统总体的负载处理能力。
- 缓存维度：KV Cache Hit Rate。该指标对侧信道攻击极度敏感，将在第四章重点分析。
- 资源维度：GPU Memory Usage、CPU Load。

3.3.2 特征介绍

为了训练和验证检测算法，本文采集并构建了覆盖典型应用场景的数据集。数据不仅包含传统的性能统计，更关联了推理引擎的内部逻辑状态。

如图3-2所示，请求的生命周期决定了核心监控指标的物理意义：首 Token 延迟（TTFT）：定义为从请求进入系统到模型生成第一个有效 Token 的总时长。其数学表达为：

$$T_{TTFT} = T_{wait} + T_{prefill} \tag{3-1}$$

其中 T_{wait} 为在调度队列中的等待时间，$T_{prefill}$ 为模型处理输入 Prompt 的计算时间。在 DDoS 攻击下，T_{wait} 会因并发过高而激增。

每 Token 平均延迟（TPOT）：定义为解码阶段生成后续每个 Token 所需的平均时间：

$$T_{TPOT} = \frac{T_{total} - T_{TTFT}}{N_{gen} - 1} \tag{3-2}$$

其中 N_{gen} 为生成的总 Token 数。

3.4 实验环境与数据介绍

3.4.1 硬件与模型配置

本研究的所有实验均在高性能算力平台上完成。

表 3-1 数据集特征种类全表

特征维度	特征名称	特征描述
请求客户端	TTFT	首 Token 时延（反映 Prefill 阶段响应速度）
	TPOT	每 Token 平均延迟（反映 Decoding 阶段生成速度）
	Throughput	吞吐量（每秒生成的 Token 总数）
	Prompt/Gen Len	输入提示词与输出序列的长度特征
LLM 推理引擎	KV Cache Hit Rate	缓存命中率（衡量 Prefix Caching 复用效率）
	KV Block Reuse	KV 块重用频率（PagedAttention 调度特征）
	Memory Swapping	显存与内存之间的页面交换频次
宿主机系统	GPU Utilization	GPU 算力利用率（反映计算压力）
	GPU Memory Used	显存实际占用量（监控资源压榨攻击）
	Power Usage	GPU 功耗波动（侧信道分析的辅助特征）
标注信息	Label 0	正常流量（符合 Zipf 分布的长尾请求）
	Label 1	拒绝服务攻击（DDoS，高并发短文本）
	Label 2	缓存侧信道攻击（Uniform 分布，诱发缓存污染）

- 计算设备：NVIDIA A100 GPU (80GB 显存)。
- 推理框架：vLLM v0.6.0，开启 PagedAttention 与 Prefix Caching。
- 待测模型：Qwen2.5-7B-Instruct。选取该模型的原因在于其参数规模适中，在 A100 环境下能表现出明显的“双峰分离”特征，具备较好的攻击辨识度。

3.4.2 性能评估标准

3.4.3 参数选取实验

3.4.4 实验结果分析

第四章 基于缓存侧信道的攻击检测研究

4.1 研究背景

随着大语言模型（LLM）推理需求的爆发，如何降低单次推理的算力成本（Token Cost）成为云服务商关注的核心。在这一背景下，基于键值缓存（KV Cache）复用的优化技术得到了广泛应用。以 vLLM 为代表的推理引擎引入了 Prefix Caching（前缀缓存）机制，其核心思想是：在多租户环境下，如果多个请求共享相同的系统提示词（System Prompt）、背景文档或长上下文，物理显存中仅需存储一份对应的 KV 向量。这种机制利用了请求分布的“局部性（Locality）”原理，能够显著减少重复计算，提升系统的整体吞吐量。

然而，缓存机制的引入在提升效能的同时，也打破了租户间严格的资源隔离，引入了新型的侧信道安全威胁。在云原生环境中，多个互不信任的用户共享同一物理显存池中的缓存页（Blocks）。如果攻击者能够探测或干扰缓存的状态，便可能实施隐蔽的性能干扰攻击。这种攻击不同于第三章讨论的“暴力型”DDoS 攻击，它不依赖于海量的请求并发，而是通过精心构造请求的分布逻辑，诱发系统出现缓存污染（Cache Pollution）。

在正常业务场景下，用户请求通常遵循 Zipf 分布（即长尾分布），少数热点提示词被高频复用，此时系统维持着较高的缓存命中率。但研究发现，攻击者可以利用这一逻辑漏洞，发送大量符合 Uniform 分布（随机分布）的“冷启动”请求。这些请求虽然在应用层完全合法，且不具备显著的流量特征，但由于它们包含互不重复的前缀，会迫使推理引擎不断为新请求分配物理块，并按照置换算法（如 LRU）驱逐掉内存池中正常用户的热点缓存。

这种针对缓存侧信道的攻击会导致以下严重后果：

服务质量降级：正常用户的热点请求因缓存被污染而无法命中，被迫重新进入耗时巨大的预填充（Prefill）阶段，导致首 Token 延迟（TTFT）大幅增加。

资源利用率失真：由于频繁的缓存失效（Cache Miss）和重新计算，GPU 始终维持在高功耗状态，但系统整体吞吐量（Throughput）却大幅下跌，造成昂贵的算力资源浪费。

检测难度提升：由于攻击请求在文本内容和单点流量上与正常请求无异，基于关键字过滤或连接数限制的传统安全防火墙（WAF）对此类“逻辑漏洞”几乎完全失效。

目前，学术界关于 LLM 安全的研究大多聚焦于对抗性攻击（Adversarial At-

tacks）或隐私泄露，而针对推理引擎底层缓存机制的可用性攻击研究尚不充分。本章在前文构建的多维指标监控体系基础上，深入分析缓存侧信道的物理机理，通过模拟不同分布强度的干扰攻击，探究其对 KV Cache 命中率及显存迁移频次的影响。本研究旨在提出一种基于缓存突变特征的实时检测方案，为保障云端大模型服务在复杂逻辑攻击下的安全性提供理论支撑。

第五章 总结和展望

5.1 工作总结

5.2 后续工作展望

致　谢

时光荏苒，三年的硕士研究生学习生活即将画上句号。回首求学之路，从最初的迷茫到逐渐坚定，从理论学习到科研实践，每一步都离不开师长的指导、同窗的陪伴以及家人的支持与鼓励。

首先，我要衷心感谢我的导师于富财。感谢您在课题选题、论文撰写以及科研思路上的悉心指导。您严谨求实的治学态度、深厚扎实的学术功底和诲人不倦的育人精神，使我受益匪浅。在论文撰写过程中，无论是研究框架的构建还是细节的反复打磨，您都给予了耐心而细致的指导，让我学会了如何以更加严谨和系统的方式开展科研工作。

其次，我要感谢实验室的胡航宇老师。感谢您在学术研究和项目实践中的悉心指导。在我每次进行阶段性工作汇报时，胡航宇老师总是和我深入讨论，为我指出汇报中存在的问题，同时为我接下来的工作方向和工作重心提出建议，在此向胡航宇老师表达我最衷心的感谢。

同时，感谢学院各位任课老师在研究生阶段给予我的知识传授和思想启迪。课堂上的循循善诱和课后的答疑解惑，不仅拓宽了我的学术视野，也为论文研究奠定了坚实的理论基础。

感谢实验室的同学和朋友们。感谢你们在科研讨论中的启发与帮助，在项目合作中的默契与支持，在生活中的关心与陪伴。那些一起查阅文献、调试程序、反复修改论文的日子，将成为我人生中珍贵的回忆。

最后也是最重要的是，我要特别感谢我的父母和家人。感谢你们始终如一的理解、包容与支持。在我面对压力和困难时，是你们给予我最坚定的后盾和最温暖的力量。你们的无私付出，是我不断前行的动力源泉。

谨以此文，向所有关心、帮助和支持过我的人致以最诚挚的感谢。

参考文献

[1] Armstrong e a. Detecting adversarial examples in nlp[J]. Unknown, Unknown.

[2] Jain e a. Baseline defenses for adversarial attacks against aligned language models[J]. arXiv preprint, 2023.

[3] Nicholas Carlini e a. Remote timing attacks on efficient language model inference[J]. arXiv preprint, 2024ArXiv:2410.17175.

[4] Linke Song e a. The early bird catches the leak: Unveiling timing side channels in llm serving systems[J]. arXiv preprint, 2024, ArXiv:2409.20002v1.

[5] Tianchen Zhang e a. Time will tell: Timing side channels via output token count in large language models[J]. arXiv preprint, 2024ArXiv:2412.15431.

[6] Xinyao Zheng e a. Inputsnatch: Stealing input in llm services via timing side-channel attacks[J]. arXiv preprint, 2024ArXiv:2411.18191.

[7] Zhifan Luo e a. Shadow in the cache: Unveiling and mitigating privacy risks of kv-cache in llm inference[J]. arXiv preprint, 2025ArXiv:2508.09442v1.

[8] Geoff McDonald e a. Whisper leak: A side-channel attack on large language models[J]. arXiv preprint, 2025ArXiv:2511.03675v1.

[9] Deniz E, Serttaş S. Deep learning-based distributed denial of service detection system in the cloud network[J]. Journal of Scientific Reports-A, 2023, (055): 16−33.

[10] Zhongkai Tong e a. Cache side-channel attacks detection based on machine learning[J]. ResearchGate, 2020.

[11] Malhotra P, et al. Lstm-based encoder-decoder for multi-sensor anomaly detection[J]. arXiv preprint, 2016ArXiv:1607.00148.

攻读硕士学位期间取得的成果

[1] 电子科技大学新生入学奖学金一等奖，2023 年

[2] 电子科技大学学业一等奖学金，2024 年

[3] 电子科技大学学业一等奖学金，2025 年

[4] 横向项目. 基于大数据分析的 Web 攻击防御技术研究