计 算 机 研 究 与 发 展
Journal of Computer Research and Development
https://crad.ict.ac.cn
63(8): 2024−2036, 2026

面向演进式大语言模型的带宽高效边云协同推理方法

李豫晨[1] 孔 芮[1] 陈新冉[2] 殷大伟[2] 孔令和[1] 陈贵海[1]
[1](上海交通大学计算机学院 上海 200240)
[2](百度时代网络技术(北京)有限公司 北京 100085)
(yuchenli1230@gmail.com)

Bandwidth-Efficient Edge-Cloud Collaborative Inference Method for Evolving Large Language Models

Li Yuchen[1], Kong Rui[1], Chen Xinran[2], Yin Dawei[2], Kong Linghe[1], and Chen Guihai[1]
[1](*School of Computer Science*, *Shanghai Jiao Tong University*, *Shanghai* 200240)
[2](*Baidu.com Times Technology* (*Beijing*) *Co.*, *Ltd.*, *Beijing* 100085)

Abstract Deploying large language models (LLMs) in mobile and edge computing environments faces multiple challenges, including limited on-device resources, scarce wireless bandwidth, and the continual evolution of cloud-side models. Although speculative decoding based edge-cloud collaborative inference can reduce end-to-end generation latency by using a lightweight draft model at the edge and a target model in the cloud for parallel verification, existing methods usually rely on a tightly coupled relationship between the two models. In practical systems, frequent updates of the cloud model often require repeated synchronization of the edge-side draft model, leading to substantial communication overhead, higher latency, and limited scalability. To address this issue, ECSpec is proposed as a communication-efficient edge-cloud collaborative inference framework for model-evolving scenarios. Its key idea is a shared backbone architecture that enables a single static edge-side draft model to remain compatible with a family of continuously evolving cloud-side target models, thereby avoiding repeated retraining or weight downloading at the edge and significantly reducing communication and maintenance costs. In addition, a channel-aware adaptive speculation mechanism is introduced to dynamically adjust the draft length according to real-time channel conditions and energy budgets, achieving a better tradeoff between inference efficiency and energy consumption. Experimental results demonstrate that ECSpec delivers more stable and efficient collaborative LLM inference under communication-constrained edge environments.

Key words edge computing; edge-cloud collaboration; large language models; speculative decoding; collaborative inference

摘 要 在移动与边缘计算环境中部署大语言模型面临端侧算力与存储受限、无线带宽紧张以及云端模型持续演进等挑战。现有基于投机解码的边云协同推理虽可借助边缘草稿模型与云端目标模型的并行协作降低生成时延,但通常要求二者紧密匹配。当云端模型频繁更新时,往往需要反复同步边缘模型,导致额外通信开销、时延增加,并削弱系统可扩展性。为此,提出通信高效边云协同推理框架 ECSpec。其核心在于共享骨干设计,使静态边缘草稿模型可适配多个不断演进的云端目标模型,从而避免端侧重复训练与权重反复下载,显著降低通信与维护成本。进一步结合时变信道与设备异构特性,设计信道感知

收稿日期: 2026-03-17; **修回日期**: 2026-04-20
基金项目: 国家自然科学基金-企业创新发展联合基金重点支持项目(U25B2037)
This work was supported by the Key Support Project of the NSFC-Enterprise Innovation Development Joint Fund (U25B2037).
通信作者: 孔令和(linghe.kong@sjtu.edu.cn)

自适应投机机制，依据实时信道状态和能耗预算动态调整草稿长度，在效率与能耗之间实现平衡。实验结果表明，ECSpec 在通信受限的边缘环境中可实现更稳定、高效的大语言模型协同推理。

关键词 边缘计算；边云协同；大语言模型；投机解码；协同推理

中图法分类号 TP311.13；TP309

DOI: 10.7544/issn1000-1239.202660136 **CSTR:** 32373.14.issn1000-1239.202660136

近年来，以 ChatGPT[1]、LLaMA[2] 和 Gemini[3] 等为代表的大语言模型（large language models, LLMs）已成为新一代智能代理系统的核心技术基础，广泛应用于移动个人助理、实时翻译设备以及复杂推理任务等场景[4-5]。模型能力的提升在很大程度上决定了用户体验质量以及复杂任务执行的成功率。尽管已有大量研究致力于通过量化、剪枝[6] 等技术在边缘设备上压缩模型规模，但受限于电池容量、热设计功耗以及内存带宽等因素，单纯依靠端侧设备进行大模型推理仍然面临显著性能瓶颈[7]。

边云协同推理为解决上述问题提供了一种有效途径。该范式通过将计算密集型子任务卸载至云端高性能服务器，同时在边缘侧保留轻量级处理能力，从而在推理能力与响应精度之间实现平衡[8]。然而，在这一协同框架下，如何在保证推理准确性的同时维持实时性能成为关键挑战。尤其是在高度动态的移动网络环境中，用户请求与模型响应之间的时延若过高，将显著影响系统的交互性与可靠性。在自回归生成场景中，该问题尤为突出。由于模型按 token 逐步生成输出，逐 token 的传输机制会带来显著通信开销。此外，边缘设备与云端之间往往需要交换大量信息，从而进一步增加网络负载。因此，亟需设计高效的推理加速机制，以降低通信开销并缩短决策时延[9]。

为缓解自回归生成带来的时延问题，投机解码（speculative decoding, SD）被广泛应用于推理加速。典型的边云投机解码框架中，边缘设备首先利用轻量级草稿模型快速生成一段候选 token 序列（即“草稿”），随后将该序列发送至云端，由高性能目标模型进行并行验证[10]。近年来，基于深度学习的投机解码方法（如 EAGLE-2[11] 和 Medusa[12]）在时延降低方面取得了显著进展。然而，将这些方法直接部署于实际边云系统仍面临若干根本性挑战。首先，现有投机解码框架通常依赖草稿模型与目标模型之间的紧耦合关系，而云端目标模型往往通过参数高效微调（parameter-efficient fine-tuning, PEFT）[9] 等方式频繁更新。这种更新机制会导致“更新风暴”现象，即为了保持模型一致性而产生过量通信开销，尤其在带宽受限的无线环境下更加严重。其次，若为避免模型同步而保持边缘草稿模型静态，则随着目标模型持续演进，草稿模型与目标模型之间的分布差异将不断扩大，进而显著降低 token 接受率。这种现象可被称为“性能崩塌”，即投机解码效率随着模型分布偏移而急剧下降。最后，投机解码性能高度依赖于无线信道条件。在移动环境中，上行时延随时间变化显著，固定步长的投机策略难以在不同信道状态下保持最优性能。在弱信号场景下，传输固定长度的草稿 token 可能带来高额通信开销；而在强信号环境下，过小的步长又无法充分利用可用带宽。因此，模型频繁演进、分布偏移以及无线时延波动三者叠加，从根本上限制了现有投机解码框架在实际边云部署中的可行性。

针对上述挑战，本文的目标是在边云系统中实现“版本无关”的投机解码机制，即利用单一静态边缘模型为多版本、持续演进的云端目标模型提供加速支持。一旦构建出具有鲁棒性的草稿框架，即可通过 1 个草稿模型支持多个不同微调版本的目标模型，从而显著降低系统维护成本。此外，该解耦式框架在面对未知目标分布时亦具有良好的泛化能力，避免了传统方法中必须重新训练草稿模型的问题，从而进一步降低部署与通信开销。受基础模型在跨模态处理方面成功经验的启发，本文提出一种面向模型演进场景的边云协同投机解码框架 ECSpec。该框架的核心思想在于从结构层面解耦边缘侧草稿模型与云端目标模型的生命周期，从而在带宽受限的无线环境中避免频繁的模型同步。同时，ECSpec 显式建模无线信道的随机性特征，并根据时变网络条件动态调整投机解码策略。通过在统一框架中同时考虑模型演进与无线动态因素，ECSpec 能够在边缘系统中实现可扩展、鲁棒且低时延的大语言模型推理。

本文的主要贡献包括 3 个方面：

1）提出一种新的边云协同投机解码框架 ECSpec，实现面向模型演进的版本无关推理机制。通过引入

共享冻结锚点骨干架构，从根本上解耦草稿模型与目标模型生命周期，消除无线边缘网络中的“更新风暴”问题。

2)设计了一种适用于边缘环境下推测解码的信道自适应投机策略。通过刻画投机步长、token 接受率与通信速率之间的相互关系，构建具有理论依据的步长选择模型，从而实现对时变信道条件的动态适配，并最大化有效 token 生成吞吐率。

3)在大规模 GPU 集群上进行广泛实验评估，覆盖推理与代码生成等多种任务场景。实验结果表明，ECSpec 在生成效率方面持续优于现有投机解码方法；通过动态调整投机步长，在通信受限环境中表现尤为显著。同时，由于边缘草稿模型无需随目标模型更新而反复同步，系统显著降低了更新相关通信的开销。

1 相关工作

本节主要概述与本文模型相关的推测解码、边云协同推理和感知无线的移动人工智能。

1.1 投机解码

投机解码已成为缓解大语言模型自回归生成过程中“内存受限型时延瓶颈”的关键技术范式。其核心思想在于通过“草稿—验证”(draft-then-verify)策略，以并行计算能力换取串行内存访问的减少，从而提升生成效率。Leviathan 等人[10]以及 Chen 等人[13]对投机执行框架进行了形式化建模，证明了小规模草稿模型可以生成候选 token 序列，并由目标模型进行并行验证，在理论上保证最终输出与目标模型逐 token 生成结果完全一致。在此基础上，后续研究重点提升草稿质量与 token 接受率。Medusa[12]在冻结原始模型参数的基础上引入多个解码头，以同时预测多个未来 token，从而避免单独训练草稿模型。EAGLE-2[11]与 EAGLE[14]进一步通过层级特征外推机制，在草稿层对输入特征进行加工，生成上下文相关的候选 token，从而提升预测准确性。除模型驱动方法外，亦有非模型化草稿生成策略，例如基于 n-gram 匹配的方法[15]以及基于检索增强的方法[16]，通过统计模式或外部数据库生成候选 token，避免复杂神经网络计算。然而，上述投机解码框架普遍建立在“静态环境”假设之上，即草稿模型与目标模型之间保持紧耦合或结构一致。在实际边缘计算场景中，云端目标模型通常通过参数高效微调 PEFT[17]等方式频繁更新。若维持 EAGLE 或 Medusa 等方法所要求的严格结构耦合，则必须通过无线链路频繁同步边缘侧草稿模型参数，从而带来难以承受的通信开销。与现有方法不同，ECSpec 通过结构性解耦草稿模型与目标模型的生命周期，实现单一静态边缘模型长期服务于持续演进的云端目标模型，无需频繁同步更新，从根本上突破了现有投机解码框架在无线边缘环境中的可扩展性限制。

1.2 边云协同推理

边云协同推理旨在弥合移动设备资源受限与现代深度神经网络(deep neural network, DNN)高计算需求之间的差距[18]。该研究方向通常关注如何在边缘侧与云端之间进行最优计算划分，以在时延、能耗与隐私之间取得平衡。早期工作如 Neurosurgeon[19]以及 Matsubara 等人[20]提出通过在 DNN 内部寻找最优分割点来最小化端到端时延。随着大语言模型的兴起，去中心化推理框架[21-23]被提出，将 Transformer 模块分布至互联网中的异构设备上执行。此外，为缓解传输高维中间激活张量所带来的通信瓶颈，研究者提出了激进量化方法[24-25]以及激活压缩技术[26]。尽管上述方法在降低通信成本方面取得一定进展，但在生成式任务中仍存在明显局限。由于自回归生成需对每个 token 传输高维激活特征，标准分割式推理往往面临高额传输时延。此外，这些方法通常假设模型结构保持稳定，忽略了实际生产环境中云端模型频繁更新所带来的版本一致性问题。ECSpec 在 2 个方面推动了该领域发展。首先，其仅传输轻量级 token 索引而非高维激活张量，显著降低通信开销。其次，通过引入基于锚点的对齐机制，使边云协同推理在模型版本不匹配的情况下仍保持稳定有效，从而增强系统在真实部署环境中的鲁棒性。

1.3 无线感知移动智能

在移动计算环境中部署人工智能应用，必须充分考虑无线网络的动态特性。与稳定的有线连接不同，移动无线信道存在衰落、干扰与噪声等问题，严重影响实时人工智能应用的服务质量(quality of experience, QoE)[27]。为应对这一挑战，Park 等人[28]与 Zhu 等人[29]在无线边缘智能领域奠定了基础工作，通过优化计算卸载策略与信道质量之间的权衡关系，实现系统性能提升。在自然语言处理场景中，动态推理机制如 DeeBERT[30]与 SPINN[31]允许模型基于置信度阈值提前退出，以降低能耗并缩短推理时延。近期工作 DSSD[32]尝试将投机解码引入无线环境，但其仍采用固定策略，未充分利用实时信道状态信息进行动态优化。ECSpec 在无线网络理论与大语言

模型推理系统之间建立了系统化联系。不同于将生成长度视为固定超参数的既有方法，ECSpec 将投机步长建模为依赖信道条件的动态变量，并通过信道感知自适应策略对生成过程进行调节。该机制使系统能够在移动场景下典型的网络波动条件中维持较高吞吐率与稳定性能。

2 系统模型与问题定义

本节首先给出边云协同推理的系统模型。在此基础上，结合真实数据集与实际网络轨迹的初步实验分析，我们识别出阻碍现有投机解码框架在移动环境中实际部署的 3 项关键挑战：模型同步所带来的高额带宽开销、分布偏移引发的性能崩塌，以及系统对无线网络波动的高度敏感性。

2.1 边云协同推理系统模型

本文考虑一个由边缘设备$\mathcal{E}$与云端服务器C组成的协同推理系统。云端部署一个高容量的目标模型$M_{\mathrm{t}}(\cdot;\boldsymbol{\theta}_{\mathrm{t}})$（例如 LLaMA-2-70B），运行于 GPU 集群之上，其中$\boldsymbol{\theta}_{\mathrm{t}}\in\mathbb{R}^{N_{\mathrm{t}}}$表示模型参数。为保持在不断演进的下游任务上的竞争性能，服务提供方会基于不断变化的任务数据集$\mathcal{D}_{\mathrm{task}}$，通过参数高效微调或全参数训练方式周期性更新$M_{\mathrm{t}}(\cdot;\boldsymbol{\theta}_{\mathrm{t}})$。因此，云端目标模型随着时间的推移演变为一个序列：$\{M_{\mathrm{t}}^{(0)},M_{\mathrm{t}}^{(1)},\cdots,M_{\mathrm{t}}^{(s)}\}$。在边缘端，客户端部署一个轻量级的草稿模型$M_{\mathrm{d}}(\cdot;\boldsymbol{\theta}_{\mathrm{d}})$，其参数量$N_{\mathrm{d}}\ll N_{\mathrm{t}}$。其规模通常仅相当于目标模型的一小部分（例如约为目标模型单层的参数量级）。

由于边缘设备在存储容量、电池能耗及计算能力方面存在严格限制，无法频繁下载大规模模型权重。因此，草稿模型在目标模型多次更新过程中保持静态（冻结）状态。边缘设备与云端通过随机无线链路进行通信。在每一轮投机解码过程中，边缘设备生成一段候选 token 块x_{draft}并发送至云端；云端利用M_{t}对该 token 块进行验证，并将验证结果x_{verified}返回至边缘侧。给定输入提示词X，系统生成长度为O的输出序列$Y=\{y_1,y_2,\cdots,y_O\}$。在采用投机解码、草稿长度为K的情况下，第 n 轮解码的时延模型表示为：

$$T_{\mathrm{s}}(K,R_n)=T_{\mathrm{e}}(K)+T_{\mathrm{u}}(K,R_n)+T_{\mathrm{c}}(K)+T_{\mathrm{d}}\,, \tag{1}$$

其中$T_{\mathrm{e}}(K)$表示边缘端生成K个 Token 的计算时间；$T_{\mathrm{u}}(K,R_n)$表示第 n 轮的上行传输延迟；$T_{\mathrm{c}}(K)$表示云端对草稿块进行验证的计算时间；T_{d}表示云端返回验证结果的下行时延。进一步地，上行传输时延可表示为上行数据量$B_{\mathrm{u}}(K)$与当前可达上行速率R_n的比值：

$$T_{\mathrm{u}}(K,R_n)=B_{\mathrm{u}}(K)/R_n\,。 \tag{2}$$

该模型刻画了投机步长K与无线信道状态R_n对系统时延的耦合关系，为后续优化问题的形式化奠定基础。

2.2 问题定义

现有的云边协同推测解码在实际部署时存在 3 个亟待解决的问题：

问题 1. 无线网络中的“更新风暴”无线网络中大模型的更新需求与有限无线带宽之间的冲突。第 1 个核心障碍在于模型新鲜度需求与无线带宽受限之间的矛盾。在标准投机解码框架中，草稿模型需要尽可能逼近目标模型的输出分布，以维持较高 token 接受率$\hat{\gamma}$。因此，当目标模型从旧版本更新至新版本，例如针对医学场景进行微调时，理论上需要重新训练并同步边缘侧草稿模型，以保证投机效率。我们将这种由于模型频繁更新而引发的大规模同步现象称为“更新风暴”。为量化其影响，我们对一个轻量级草稿模型（约 3.2 GB）在典型移动网络环境中的同步开销进行了初步评估。表 1 结果显示，在 4G 网络环境下，即使下载压缩后的草稿模型，也需要接近 10 min。对于每日更新模型的移动应用而言，这将对蜂窝网络基础设施造成流量压力，同时迅速消耗用户流量套餐与电池资源。因此，亟需设计一种无需频繁同步的静态草稿模型机制，以避免更新风暴所带来的不可扩展性问题。

Table 1 Simulated Latency for Synchronizing Draft Models over Wireless Networks

表 1 在无线网络上同步草稿模型的模拟延迟

网络类型	带宽/Mbps	每个用户的同步时间/min	1 000 个用户同时同步可扩展性的表现
Public WiFi	10	约 48	不可拓展
4G LTE	50	约 9.5	高度拥塞
5G Mid-Band	300	约 1.6	中度负载

问题 2. 分布偏移与性能崩塌。若为节省带宽而放弃模型同步（即保持M_{d}的参数固定，而M_{t}持续更新），则会面临第 2 项挑战：分布偏移。随着目标模型微调至新领域，其输出分布将逐渐偏离静态草稿模型的输出分布。为验证这一现象，我们采用通用 LLaMA-2-70B 作为冻结草稿模型，并分别与通用、数学（GSM8K）以及代码（HumanEval）3 个版本的 70B

目标模型进行配合测试。表 2 结果显示，当目标模型演进至编程领域时，草稿模型的 token 接受率从 0.72 降至 0.18，下降幅度高达 75%。在如此低的接受率下，运行草稿模型并传输 token 所产生的额外开销将超过投机带来的收益，甚至导致整体时延高于标准自回归解码。这一现象可被定义为“性能崩塌”。因此，必须构建一种机制弥合静态草稿表示与演进目标表示之间的差距。本文后续提出的基于锚点的对齐机制正是为解决该问题而设计。

Table 2 Impact of Target Model Evolution on Fixed Draft Model Performance

表 2 目标模型演进对固定草稿模型性能的影响

目标模型	任务类型	接受率
LLaMA-2-70B-Base	通用	0.72
LLaMA-2-70B-Math（LoRA）	数学	0.45
LLaMA-2-70B-Code（Full）	编程	0.18

问题 3. 无线时延敏感性。第 3 个挑战源于投机步长 K 与网络条件之间的耦合关系。现有大多数工作采用固定投机步长（例如 $K=5$）。然而，我们对真实网络轨迹的分析表明，在移动环境中固定步长往往是次优策略。在弱信号场景（例如电梯或地铁环境）下，上行传输时延项 $T_{\mathrm{u}}(K,R_n)$ 在总时延中占主导地位。例如，传输 5 个 token 可能带来约 200 ms 的上行延迟，而对应的验证收益仅约 50 ms。相反，在强 5G 信道条件下，过小的固定 K 无法充分利用可用带宽。这种不匹配构成一个基本矛盾：静态投机策略无法在异构移动场景与动态信道条件下同时满足时延最优需求。因此，需要设计一种信道感知的自适应投机机制，以根据实时网络状态动态调整投机步长。系统优先利用设备侧或操作系统提供的链路测量接口获得近实时的链路速率信息；当底层接口不可用时，则通过短时间窗口内的回路时延、历史上行/下行吞吐率以及实际传输完成时间对有效链路速率进行统计估计。为降低测量噪声对步长决策的干扰，本文对相关链路指标采用滑动平滑处理。

基于上述时延模型，我们的目标是在演进式边云系统中设计一种始终有效的投机解码框架。具体而言，在云端目标模型序列持续更新的前提下，我们希望设计一种模型耦合机制，使边缘草稿模型在保持静态（无需重新训练或重复下载）的情况下，仍能在与任意版本协作时维持较高 token 接受率。同时，在时变无线信道条件下，我们希望联合设计投机步长策略及模型耦合机制，以最大化系统的有效 token 生成速率（effective token generation rate, ETGR）。本文将 ETGR 近似定义为单位时间内系统有效推进 token 数的期望值。为便于分析，正文采用单轮被目标模型接受的草稿 token 数 τ 作为主要统计量，即 $\mathbb{E}[\tau|K]$。需要说明的是，在实际投机解码过程中，当在位置 j 发生拒绝时，目标模型还可能额外生成一个修正 token，因此严格意义上的单轮推进量可写为 $\tau+1$，其中 1 表示由验证分支带来的补偿项，为常数阶项，且不随 K 呈主导性增长，故在本文的步长优化分析中采用 $\mathbb{E}[\tau|K]$ 作为近似目标不会改变最优 K 的主要变化趋势。设 τ 表示在一次草稿验证轮次中，被接受的 token 数量，则 ETGR 定义为：

$$ETGR(K)=\frac{\mathbb{E}[\tau|K]}{\mathbb{E}[T_{\mathrm{s}}(K,R_n)]}。\tag{3}$$

该目标等价于最小化每个被接受 token 的期望时延：$\mathbb{E}[T_{\mathrm{s}}(K,R_n)]/\mathbb{E}[\tau|K]$。

综上所述，本文旨在学习一种自适应投机步长策略以及草稿模型与目标模型之间的结构耦合机制，从而在模型持续演进与无线条件动态变化的环境中最大化 ETGR。

3 面向演进式大语言模型的带宽高效边云协同推理方法

本节详细介绍为解决边云协同推理挑战而设计的 ECSpec 框架，如图 1 所示。

3.1 基于锚点的对齐与深度嵌入

为满足“边缘草稿模型参数 $\boldsymbol{\theta}_{\mathrm{d}}$ 保持静态，而云端目标模型参数 $\boldsymbol{\theta}_{\mathrm{t}}$ 持续演进”的约束条件，本文提出一种基于锚点对齐并结合深度嵌入增强的结构化架构。标准投机解码在云端微调后往往失效，其根本原因在于草稿分布与目标分布之间的差异不断扩大。我们基于已有研究观察到：尽管微调过程会改变高层语义决策行为，但在同一模型家族内部，主干网络中的基础 token 表征逻辑通常在较大程度上保持稳定。因此，可以通过假设共享稳定表征空间实现跨版本对齐。需要进一步说明的是，该假设主要适用于以下场景：目标模型与基础模型具有相同或高度兼容的分词器，主干结构与隐藏维度保持一致或可通过轻量映射对齐，且模型更新主要体现在任务适配层或高层语义行为上。当目标模型发生大规模架构升级、词表重构或完全跨家族迁移时，基础表征空间的一致性可能明显下降，此时需重新训练锚点模块与投影头，或退化为仅云端推理模式[33]。

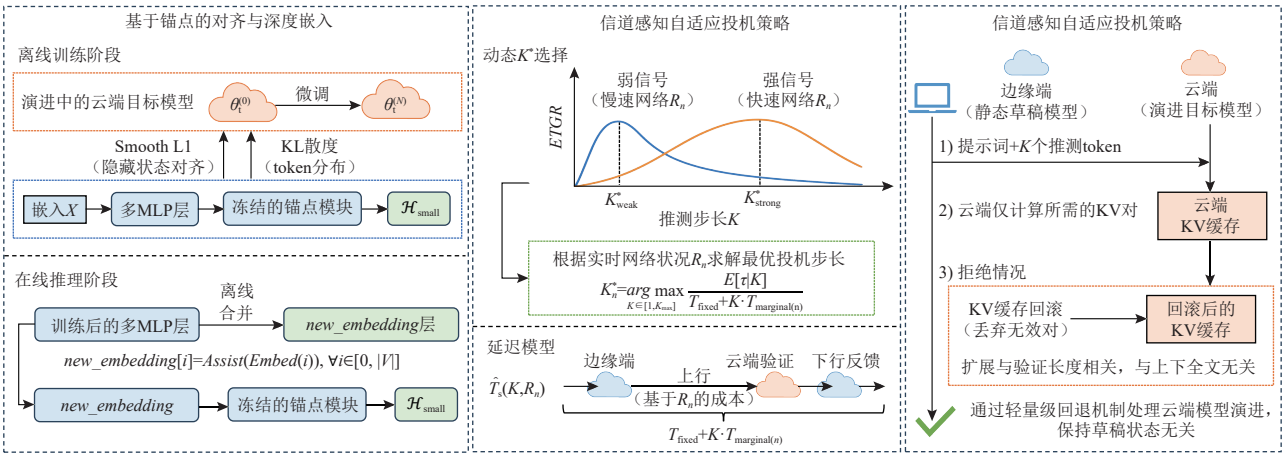


Fig. 1 System model of ECSpec

图 1 ECSpec 系统模型

1)结构定义与深度嵌入学习

给定云端在任务微调前的基础模型为$M_{\text{base}} \triangleq M_{\text{t}}^{(0)}$，我们将其分解为特征提取器与预测头。预测头包含最终的 Transformer 模块(称为锚点块B_L^{base})以及语言建模(language modeling, LM)输出头。边缘草稿模型通过复制并冻结该锚点块构建，并在其后附加一个轻量级可训练投影头$\mathcal{H}_{\text{small}}$。然而，仅依赖浅层草稿结构通常难以获得较长的接受长度τ。为在不增加推理时延的前提下增强模型表达能力，本文提出一种深度嵌入增强机制。具体而言，在草稿模型的标准输入嵌入层之后叠加多个多层感知机(multi-layer perceptron, MLP)模块(共L层)，称为 assist layers。这些层用于提升草稿表示与目标模型表示之间的对齐程度。草稿模型的整体结构可表示为：

$$M_{\text{d}}(x)=\mathcal{H}_{\text{small}}\left(B_L^{\text{base}}(Assist(Embed(x)))\right), \tag{4}$$

其中$B_{\text{base}} \triangleq B_L^{\text{base}}$且保持冻结状态。需要强调的是，assist layers 的设计满足不引入额外位置编码信和不允许跨 token 的信息交互 2 个关键约束。因此，对于任意 token，在任意位置其输出表示均为确定性且唯一的函数映射。

2)嵌入融合

在离线训练完成后，我们将 assist layers 与原始嵌入层进行融合。由于标准嵌入层与 assist layers 均可视为对单一 token 索引的确定性映射，因此可将二者折叠为一次查表操作。具体而言，我们预先离线计算新的嵌入表：

$$new_embedding[i]=Assist(Embed(i)), \forall i\in[0,|V|], \tag{5}$$

其中$|V|$表示词表大小。在实际推理阶段，边缘草稿模型直接使用融合后的$new_embedding$层，从而完全消除 assist layers 的计算开销，同时保留其增强后的表示能力，实现真正的“零额外推理负担”。

3)通用训练目标

为训练静态边缘组件(即 assist layers 与 $\mathcal{H}_{\text{small}}$)，我们采用特征回归与知识蒸馏相结合的多目标损失函数。该训练过程仅需离线执行 1 次。首先，将预测目标模型隐藏特征视为回归任务。为保证草稿隐藏状态h_{d}与目标隐藏状态h_{t}在表示空间中稳定对齐，采用 *Smooth L*1 损失：

$$\mathcal{L}_{\text{reg}}=\frac{1}{\omega\cdot\pi}\sum_{i=1}^{\omega}\sum_{j=1}^{\pi}Smooth\,L1\left(\boldsymbol{W}_{\text{p}}h_{\text{d}}^{(i,j)},h_{\text{t}}^{(i,j)}\right), \tag{6}$$

其中$\boldsymbol{W}_{\text{p}}$为可学习的维度匹配投影矩阵，$\omega$表示 batch 大小，$\pi$表示序列长度。此外，为逼近最终生成目标，我们最小化 token 分布之间的 KL 散度[34]：

$$\mathcal{L}_{\text{cls}}=T^2D_{\text{KL}}\left(\sigma\left(\frac{z_{\text{t}}}{T}\right)\middle\|\sigma\left(\frac{z_{\text{d}}}{T}\right)\right), \tag{7}$$

其中z_{t}和z_{d}分别为目标与草稿模型 logits，σ表示 Softmax 函数，T为温度系数。最终训练目标为：

$$\mathcal{L}=\mathcal{L}_{\text{reg}}+\omega_{\text{cls}}\mathcal{L}_{\text{cls}}。 \tag{8}$$

3.2 信道感知自适应投机策略

在移动计算场景中，端到端时延与信道动态密切耦合。固定投机步长K无法兼顾传输开销与投机收益之间的权衡。因此，我们将最优步长选择建模为吞吐率最大化问题。

1)精细化时延建模

单次投机解码步长为K时，总时延由 4 部分构成：边缘侧生成K个候选 token 的计算时延、候选 token 及必要元信息的上行传输时延、云端验证与回滚处理时延，以及验证结果的下行反馈时延。进一步地，通信相关项由可变传输项与固定时延项 2 部分组成。

前者主要由待传输数据量与当前链路速率决定，后者主要包括无线传播时延、协议栈处理、调度与服务器请求分发等不可压缩开销。因而，前链路速率主要影响上下行串行化传输部分，而固定处理项不随K线性增长：

$$\hat{T}_{\mathrm{s}}(K,R_n)=\hat{T}_{\mathrm{e}}(K)+\hat{T}_{\mathrm{u}}(K,R_n)+\hat{T}_{\mathrm{c}}(K)+T_{\mathrm{d}}\text{。} \tag{9}$$

云端验证时延建模为关于K的仿射函数，以刻画加载 K个新 token 及其 KV cache 所带来的内存带宽开销。将固定开销与边际开销分离，可得：

$$\hat{T}_{\mathrm{s}}(K,R_n)=T_{\mathrm{fixed}}+K\left(\alpha_{\mathrm{e}}+\frac{b}{R_n}+\delta_{\mathrm{c}}\right)\text{，} \tag{10}$$

其中T_{fixed}表示传播与基础处理时延；R_n表示当前瞬时通信速率；δ_{c}表示单位 token 的云端边际验证成本。需要特别说明的是，本文中的上行传输时延与下行反馈时延仅表示与数据量及链路速率相关的串行化传输部分；传播时延表示无线链路传播、协议栈转发及往返中的固定延迟，不重复计入上行或下行串行化传输项；云端固定处理时延则表示请求分发、调度与验证准备等服务器侧开销。上述 3 类时延在模型中相互区分，不存在重复计数。云端固定处理时延项用于表示传播时延以及云端请求分发、调度与基础验证准备等平均开销。该项在本文中被近似视为短时间窗口内相对稳定的参数，以便于刻画通信与验证代价的主导结构。需要指出的是，在真实云系统中，队列积压、资源竞争与负载变化可能导致该项出现额外波动，并进一步影响最优步长估计的稳定性。本文的自适应控制器通过在线更新链路状态与接受统计量，能够在一定程度上缓解此类扰动带来的影响。

2)吞吐率最优策略

为最大化 ETGR，我们求解最优投机步长：

$$K_n^{*}=\arg\max_{K\in[1,K_{\max}]}\frac{\mathbb{E}[\tau|K]}{T_{\mathrm{fixed}}+K\cdot T_{\mathrm{marginal}(n)}}\text{。} \tag{11}$$

该目标函数刻画了核心权衡关系：在弱信号环境下，通信成本主导分母增长，使最优K_n^{*}倾向于较小值；而在良好信道条件下，较大步长能够有效摊薄往返时延开销，从而提升吞吐效率。需要说明的是，本文在推导中将单轮接受 token 的期望数量$\mathbb{E}[\tau|K]$作为K的函数进行近似建模，其目的在于刻画扩大步长所带来的潜在收益。严格而言，token 级接受事件并非独立同分布，相邻预测结果在上下文条件下通常具有相关性，因此$\mathbb{E}[\tau|K]$与K的关系在全局上未必严格线性。本文在理论分析中采用局部单调近似，用以揭示通信开销、验证成本与投机收益之间的核心权衡；在较大K区域，接受收益通常呈现边际递减，这也构成了自适应步长机制的动机之一。本文在实现中采用有限候选集合上的轻量枚举搜索，即在$K\in\{1,2,\cdots,K_{\max}\}$上根据当前链路速率、接受率估计与验证成本计算目标函数值，并选择使 ETGR 最大的步长。因此，单轮决策复杂度为 $O(K_{\max})$。由于 $K_{\max}$ 通常远小于生成序列长度，且该控制器仅涉及简单代数运算，其运行开销相较于边缘生成、无线传输与云端验证时延可以忽略。

3.3 ECSpec 协议与推理流程

运行时边缘与云端的交互在草稿模型版本上保持“无状态”，但在 KV cache 上保持“有状态”。本文统一采用“草稿生成—目标验证—失败回滚”的边云交互框架，但在不同生成机制下，候选 token 的产生与接受判定方式略有差异：在贪心解码设置下，草稿模型与目标模型均基于最大概率 token 进行确定性生成与验证；在 T=1、Top-p 采样设置下，草稿模型首先在截断后的候选集合内产生候选 token，云端再依据目标分布执行一致性验证。除候选生成与接受判定方式外，边云交互流程、KV cache 维护与回滚机制保持一致。该机制确保云端计算开销严格随实际验证长度扩展，而非随完整上下文长度增长，从而保留投机解码的核心效率优势。当云端目标模型持续演进时，ECSpec 通过轻量级回退机制优雅处理潜在分布偏移，在无需持续同步嵌入层或边缘参数的前提下保持稳定推理性能。

4 实验与结果

4.1 数据集与模型

实验在分布偏移条件下评估 ECSpec，覆盖多种目标模型与下游任务。

在任务泛化方面，为覆盖不同推理模式与数据分布，实验选取 6 类核心下游任务：数学推理(GSM8K)、问答(Natural Questions)、检索增强生成(Natural Questions)、多轮对话(MT-Bench)、机器翻译(WMT14 DE-EN)以及文档摘要(CNN/DailyMail)。

在目标模型方面，我们以 LLaMA-2-70B 作为典型稠密(dense)模型进行细致分析，并进一步在更近期的架构上验证可扩展性，包括 LLaMA-3-70B 以及混合专家模型 Mixtral 8×7B。

4.2 对比基线

为平衡接受率估计的稳定性与响应速度，本文采用指数滑动平均(EMA)对在线接受率进行跟

踪。设第 t 轮观测到的接受率为 ρ_t，则平滑估计可写为 $\rho_t = \alpha\rho_{t-1} + (1-\alpha)\rho_t$，其中 $\alpha \in [0,1)$ 为衰减系数。将 ECSpec 与覆盖多种解码范式的扩展基线集合进行比较，范围包括标准自回归方法以及最新投机解码框架。

1）Cloud-Only：完全在云端执行的标准自回归解码。该基线作为吞吐与时延的参考，其中每生成一个 token 都需要承担完整的网络往返时延以及云端计算开销。

2）Lookahead[35]：基于 Jacobi 迭代的并行解码算法，无需单独草稿模型，可同时生成多个 token，并通过多步验证并行细化候选序列。

3）Standard SD（Std. SD）：传统投机解码设置，使用通用的小型预训练模型（如 LLaMA-2-7B）作为草稿模型，为目标模型（如 LLaMA-2-70B）生成候选 token。该基线不包含本文提出的基于锚点的对齐机制，用于体现当草稿模型未随目标模型演进同步时由分布偏移带来的性能退化。

4）Medusa-1（Synced）：在目标模型上增加多个解码头以预测未来 token 的并行解码框架。与 EAGLE-2 类似，实验假设其头部与目标模型完美同步，用于表征紧耦合架构的理论上限。

5）EAGLE-2（Synced）：当前先进的模型式方法，利用分层特征外推。实验在“理想同步”设置下评估，即假设边缘端扩展层能够与云端目标版本完全一致，并忽略其同步更新所需的通信开销。

7）DSSD：面向无线边云系统的协同推理框架。与 ECSpec 不同，该方法采用固定或启发式的投机长度调度策略，缺乏对信道动态的实时自适应。

4.3 扩展的硬件测试平台

为模拟真实的跨设备部署场景，实验显著扩展了硬件环境的多样性。

1）云端服务器。通过 NVLink 互连的 8×NVIDIA H800（80 GB）；8×NVIDIA A800（80 GB，主流配置）；8×NVIDIA V100（32 GB）。

2）边缘设备。NVIDIA Jetson AGX Orin（64 GB 内存）；Snapdragon 8 Gen 3 参考设备（16 GB 内存）；Apple iPhone 15 Pro Max 模拟环境（A17 Pro）；Raspberry Pi 5（8 GB 内存）。

4.4 实验结果及分析

4.4.1 整体性能及分析

为获得更细粒度的性能分析，实验将评估划分为 2 种生成机制：确定性生成（T=0）与随机采样（T=1）。实验给出了 LLaMA-2-70B 在 6 个数据集上的完整结果，以验证方法在不同任务类型下的鲁棒性。图 2 展示了 GSM8K 上的性能差异，表 3 给出了采用 greedy decoding 的详细结果。该设置更有利于依赖严格分布一致性的解码方法。此外，我们选择 Cloud-Only 为基线计算加速比，即用 Cloud-Only 的端到端时延除以相应方法的端到端时延。如表 3 所示，ECSpec 在推理密集任务（如 GSM8K）与摘要任务（如 CNN/DailyMail）上均能够获得稳定加速。尤其在标准投机解码性能明显退化的场景中，ECSpec 仍可保持显著收益。例如在 WiFi 条件下，相较于 Cloud-Only，Standard SD 加速比仅达到 0.66 的性能，而 ECSpec 加速比仍可实现 2.30，表明锚点对齐机制能够有效缓解分布偏移导致的接受率下降。同时，尽管 EAGLE-2 等方法在 5G 条件下相较于 Cloud-Only 可获得最高加速比 2.47，但在 WiFi 场景下其加速比下降至 0.9 以下，主要受通信开销限制。相比之下，ECSpec 通过自适应调整投机步长 K，在弱网络条件下仍能够保持稳定加速，并在不同任务与网络环境中持续实现超过加速比为 2.1 的整体性能提升。

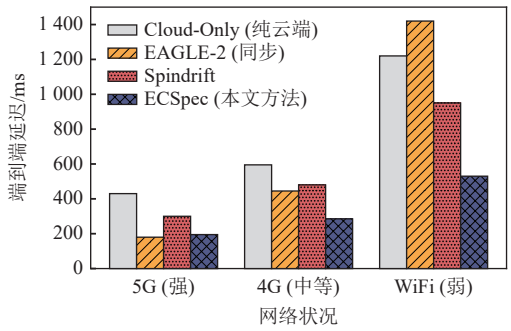


Fig. 2 End-to-end latency comparison of four algorithms on GSM8K

图 2 GSM8K 上 4 种算法的端到端时延对比

表 4 在 Top-p 采样（p = 0.9，T=1）条件下进一步评估系统性能。该设置对草稿模型与目标模型之间的对齐能力提出更高要求，因为草稿模型分布必须覆盖目标模型分布的主要概率质量。实验结果表明，在相同实验条件下相较于 Cloud-Only，ECSpec 在 6 个任务上仍能够保持 T 超过 2.0 的稳定加速比，而 EAGLE-2 在 T 从 0 提升至 1 后性能明显下降，例如在 5G 条件下加速比从约 2.4 降至约 1.6。与此同时，Standard SD 出现明显的性能崩塌现象，在 GSM8K 弱网络环境下相较于 Cloud-Only 加速比仅达到 0.65，这是由于未对齐的草稿模型接受率较低，导致频繁验证失败。Lookahead 方法在该场景下收益同样有限（加速比低于 1.06）。相比之下，ECSpec 的锚点对齐能够保证草稿模型生成的 token 在统计意义上与目标模型分布保持一致，从而避免性能退化。

Table 3 End-to-End Latency for Diverse Speculative Decoding Methods on Six Datasets (*T*=0)

表 3 不同的投机解码方法在 6 个数据集上的端到端时延（*T*=0）

数据集	网络类型	Cloud-Only/ms	Lookahead/ms	Standard SD/ms	Medusa-1/ms	EAGLE-2/ms	DSSD/ms	ECSpec/ms
GSM8K	5G	432.0	415.0	392.0	205.0	**182.0**	305.0	220.0
	4G	595.0	580.0	805.0	480.0	445.0	482.0	**325.0**
	WiFi	1 220.0	1 205.0	1 850.0	1 455.0	1 420.0	950.0	**625.0**
Natural Questions	5G	415.0	390.0	345.0	192.0	**168.0**	275.0	202.0
	4G	570.0	552.0	685.0	435.0	405.0	452.0	**295.0**
	WiFi	1 185.0	1 165.0	1 680.0	1 380.0	1 352.0	905.0	**582.0**
Natural Questions	5G	428.0	402.0	360.0	198.0	**175.0**	288.0	212.0
	4G	582.0	568.0	735.0	455.0	422.0	468.0	**308.0**
	WiFi	1 208.0	1 190.0	1 765.0	1 425.0	1 390.0	932.0	**600.0**
MT-Bench	5G	420.0	398.0	358.0	202.0	**178.0**	295.0	215.0
	4G	578.0	562.0	742.0	462.0	430.0	475.0	**312.0**
	WiFi	1 192.0	1 175.0	1 780.0	1 440.0	1 405.0	945.0	**615.0**
WMT14 DE-EN	5G	418.0	395.0	350.0	195.0	**172.0**	282.0	208.0
	4G	575.0	560.0	720.0	445.0	415.0	460.0	**302.0**
	WiFi	1 188.0	1 168.0	1 725.0	1 405.0	1 372.0	918.0	**592.0**
CNN/DailyMail	5G	425.0	400.0	355.0	198.0	**175.0**	285.0	210.0
	4G	582.0	565.0	728.0	452.0	420.0	465.0	**306.0**
	WiFi	1 205.0	1 180.0	1 750.0	1 420.0	1 382.0	924.0	**598.0**

注：黑体数值表示最优值。

Table 4 End-to-End Latency for Diverse Speculative Decoding Methods on Six Datasets (*T*=1)

表 4 不同的投机解码方法在 6 个数据集上的端到端时延（*T*=1）

数据集	网络类型	Cloud-Only/ms	Lookahead/ms	Standard SD/ms	Medusa-1/ms	EAGLE-2/ms	DSSD/ms	ECSpec/ms
GSM8K	5G	435.0	420.0	405.0	272.0	245.0	345.0	**202.0**
	4G	598.0	585.0	825.0	665.0	592.0	540.0	**295.0**
	WiFi	1 225.0	1 210.0	1 880.0	1 852.0	1 680.0	1 150.0	**595.0**
Natural Questions	5G	418.0	395.0	355.0	290.0	258.0	332.0	**194.0**
	4G	575.0	560	700.0	655.0	590.0	528.0	**285.0**
	WiFi	1 190.0	1 172.0	1 700.0	1 820.0	1 665.0	1 110.0	**580.0**
Natural Questions	5G	430.0	408.0	370.0	285.0	255.0	338.0	**200.0**
	4G	588.0	575.0	750.0	660.0	595.0	532.0	**290.0**
	WiFi	1 215.0	1 198.0	1 785.0	1 840.0	1 675.0	1 130.0	**592.0**
MT-Bench	5G	428.0	405.0	368	285.0	252.0	329.0	**198.0**
	4G	585.0	570.0	760.0	650.0	585	522.0	**286.0**
	WiFi	1 210.0	1 195.0	1 795.0	1 805.0	1 652.0	1 100.0	**588.0**
WMT14 DE-EN	5G	420.0	400.0	360.0	295.0	260.0	335.0	**195.0**
	4G	578.0	565.0	735.0	675.0	605.0	530.0	**282.0**
	WiFi	1 195.0	1 180.0	1 740.0	1 900.0	1 690.0	1 120.0	**582.0**
CNN/DailyMail	5G	425.0	402.0	365.0	300.0	265.0	338.0	**196.0**
	4G	582.0	568.0	740.0	680.0	610.0	535.0	**285.0**
	WiFi	1 200.0	1 185.0	1 765.0	1 915.0	1 700.0	1 125.0	**586.0**

注：黑体数值表示最优值。

4.4.2 消融实验结果及分析

为验证信道感知自适应投机机制的有效性，实验在保持锚点对齐结构不变的条件下，将动态步长策略替换为固定步长策略（$K \in \{1,5,7\}$），并在 GSM8K 数据集及不同网络环境中进行测试。图 3 结果表明，较大的固定步长（如 $K=5$）在 5G 环境下具有较低时延（195.0 ms），但在 WiFi 环境中由于通信开销过大，时延增加至 1 450.0 ms。相比之下，较小步长（如 $K=1$）在 WiFi 环境下更加稳定，但在 5G 环境中时延为 400.0 ms，无法充分利用带宽资源。ECSpec 能够根据信道状态动态调整投机步长，在 5G 条件下实现 195.0 ms 的低时延，在 WiFi 环境下将时延控制在 540.0 ms，从而在不同网络条件下均能够保持较优性能。

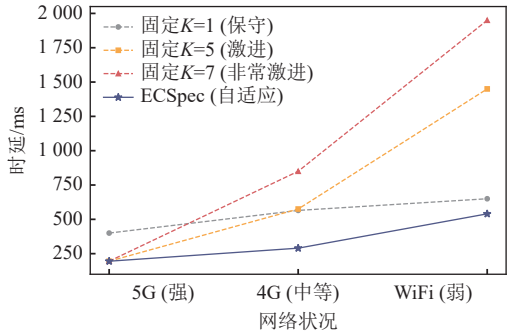


Fig. 3 Impact of speculative stride K on latency

图 3 投机步长 K 对时延的影响

4.4.3 异构边缘硬件适配能力

表 5 给出了 ECSpec 在不同边缘设备上的性能结果。实验表明，ECSpec 的性能取决于边缘草稿模型生成速度与网络通信节省之间的关系。Raspberry Pi 5 由于仅依赖 CPU，其草稿模型生成速度较慢，仅达到 6.9 token/s，导致整体加速比下降至 0.72~0.85。相比之下，具备 GPU 或 NPU 的设备均能够实现显著加速。例如，在 Jetson AGX Orin、iPhone 15 Pro Max 与 Snapdragon 8 Gen 3 上，ECSpec 均可实现超过 2.0 的加速比。在更复杂的 HumanEval 任务中，由于接受率下降，各设备加速比略有降低，例如 Jetson AGX Orin 加速比从 2.45 降至 2.20，但 iPhone 15 Pro Max 仍能够保持超过 2.08 的稳定加速比，说明该方法在复杂任务下仍具有良好适用性。

Table 5 Performance on Heterogeneous Edge Devices for ECSpec Under 4G Network Conditions

表 5 在 4G 网络条件下 ECSpec 在异构边缘设备上的性能表现

设备	处理器	草稿模型时延/ms	草稿模型吞吐率/(token·s^{-1})	加速比		
				GSM8K	MT-Bench	HumanEval
Raspberry Pi 5	Cortex-A76	145	6.9	0.76	0.85	0.72
Jetson AGX Orin	Ampere GPU	8.5	117.6	2.32	2.45	2.20
iPhone 15 Pro Max	A17 Pro	12.0	83.3	2.15	2.28	2.08
Snapdragon 8 Gen 3	Hexagon NPU	10.5	95.2	2.26	2.36	2.14

注：加速比相对于仅云端推理得出。

4.4.4 模型可扩展性

为验证方法的可扩展性，本文进一步在 LLaMA-3-70B 以及一个稀疏 MoE 模型 Mixtral 8×7B 进行实验。尽管 LLaMA-3 具有更大的词汇表以及不同的训练数据，ECSpec 在 5G 条件下仍实现了最高 2.56 的加速比（数据集：MT-Bench，网络环境：5G/4G）。如表 6 所示，算法能够在不同的模型之间实现良好的迁移。对于 Mixtral 8×7B，由于采用条件计算，其基线云端推理速度本身更快。虽然这在一定程度上减少了执行所带来的潜在增益空间，但 ECSpec 在 4G 条件下仍实现了 1.98 的加速比。信道感知策略能够自动向下调整 K 的取值，以适应更快的云端验证速度，从而避免过度推理。

Table 6 Scalability Evaluation of ECSpec on Newer Model Architectures

表 6 ECSpec 在新一代模型架构上的可扩展性评估

目标模型	延迟/ms		加速比	
	ECSpec（5G）	ECSpec（4G）	ECSpec（5G）	ECSpec（4G）
LLaMA-2-70B	420.0	578.0	2.23	2.13
LLaMA-3-70B	395.0	550.0	2.56	2.30
Mixtral 8×7B	320.0	485.0	2.05	1.98

4.4.5 内存与能耗效率

图 4 实验结果表明，全本地运行 70B 模型需要约 42.5 GB 显存，而 ECSpec 仅需约 3.5 GB 内存即可运

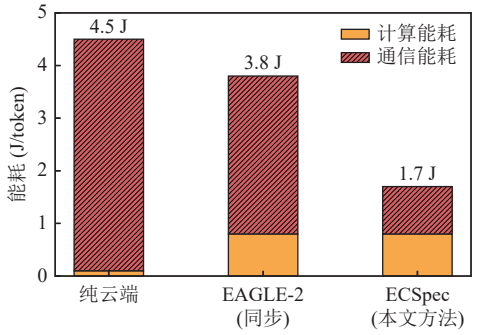


Fig. 4 Energy consumption breakdown analysis on mobile devices

图 4 移动设备上的能耗构成分析

行草稿模型,从而能够适配移动设备。能耗分析结果进一步表明,Cloud-Only 推理能耗约为 4.5J/token,而 ECSpec 通过减少通信时间,将能耗降低至 1.7 J/token,降低约 62%。同时,ECSpec 将计算负载转移至云端,避免了设备过热问题,从而提高系统稳定性。

5 结论与展望

本文针对大语言模型在动态边云协同部署中的可扩展性与通信效率问题展开研究。现有基于投机解码的方法依赖模型紧耦合同步与固定步长策略,在云端频繁更新与无线信道时变条件下易产生额外通信开销与性能退化,从而限制了大模型在边云环境中的规模化应用。为解决上述问题,本文提出 ECSpec,一种面向模型演进场景的通信高效边云协同推理框架。ECSpec 通过共享骨干架构设计,使单一且静态的边缘草稿模型能够兼容多个持续演进的云端目标模型,从而消除频繁边缘重训练与模型同步的需求,显著降低通信与维护成本。在此基础上,本文进一步提出信道感知的自适应投机机制,依据实时链路状态动态调整投机步长,在端到端时延、通信开销与设备能耗之间实现有效平衡,增强系统在复杂无线环境中的鲁棒性。

通过全面的实验评估,ECSpec 在多种网络条件、任务类型与模型架构下均表现出稳定且显著的性能优势,为动态边云环境中大语言模型的高效部署提供了一种可行的系统化解决方案。尽管 ECSpec 在通信效率与系统鲁棒性方面取得了良好效果,仍有若干值得进一步探索的方向。未来工作可沿以下 5 个方向进一步展开:1)面向高频模型更新场景,研究更精细的跨版本 drift 检测、触发式 re-anchor 机制以及低成本再对齐方法,以提升在持续演进环境中的长期稳定性;2)面向高动态无线网络,研究联合链路预测、拥塞控制与投机调度的跨层优化框架,以提升 handoff、弱覆盖与突发抖动条件下的端到端服务质量;3)面向异构终端,构建统一的时延-能耗-温度多目标在线优化方法,以支持不同设备形态下的自适应推理;4)探索 ECSpec 在稀疏模型、多模态模型与长上下文生成任务中的适用性边界;5)围绕边云协同过程中的隐私保护、缓存安全与推理可信性开展系统化研究。

作者贡献声明:李豫晨负责方案整体设计并撰写论文;孔芮、陈新冉负责部分算法思路和实验方法;殷大伟、孔令和、陈贵海提出指导意见并修改论文。

参 考 文 献

[1] Achiam J, Adler S, Agarwal S, et al. GPT-4 technical report[J]. arXiv preprint, arXiv: 2303.08774, 2023

[2] Touvron H, Martin L, Stone K, et al. LLaMA 2: Open foundation and fine-tuned chat models[J]. arXiv preprint, arXiv: 2307.09288, 2023

[3] Team G, Anil R, Borgeaud S, et al. Gemini: A family of highly capable multimodal models[J]. arXiv preprint, arXiv: 2312.11805, 2023

[4] Li Yuchen, Cai Hengyi, Kong Rui, et al. Towards AI search paradigm[J]. arXiv preprint, arXiv: 2506.17188, 2025

[5] Xiong Haoyi, Bian Jiang, Li Yuchen, et al. When search engine services meet large language models: Visions and challenges[J]. IEEE Transactions on Services Computing, 2024, 17(6): 4558−4577

[6] Liu Zechun, Zhao Changsheng, Iandola F, et al. MobileLLM: Optimizing sub-billion parameter language models for on-device use cases[C]//Proc of the 41st Int Conf on Machine Learning (ICML). New York: PMLR, 2024: 32431–32454

[7] Xu Dianlei, Li Tong, Li Yong, et al. Edge intelligence: Empowering intelligence to the edge of network[J]. Proceedings of the IEEE, 2021, 109(11): 1778−1837

[8] Liu Jing, Du Yao, Yang Kun, et al. Edge-cloud collaborative computing on distributed intelligence and model optimization: A survey[J]. arXiv preprint, arXiv: 2505.01821, 2025

[9] Lyu Zhonghao, Li Yuchen, Zhu Guangxu, et al. Rethinking resource management in edge learning: A joint pre-training and fine-tuning design paradigm[J]. IEEE Transactions on Wireless Communications, 2025, 24(2): 1584−1601

[10] Leviathan Y, Kalman M, Matias Y. Fast inference from Transformers via speculative decoding[C]//Proc of the 40th Int Conf on Machine Learning (ICML). New York: PMLR, 2023: 19274–19286

[11] Li Yuhui, Wei Fangyun, Zhang Chao, et al. EAGLE-2: Faster inference of language models with dynamic draft trees[C]//Proc of Empirical Methods in Natural Language Processing (EMNLP). Stroudsburg, PA: ACL, 2024: 7421–7432

[12] Cai Tianle, Li Yuhong, Geng Zhengyang, et al. Medusa: Simple LLM inference acceleration framework with multiple decoding heads[J]. arXiv preprint, arXiv: 2401.10774, 2024

[13] Chen C, Borgeaud S, Irving G, et al. Accelerating large language model decoding with speculative sampling[J]. arXiv preprint, arXiv: 2302.01318, 2023

[14] Li Yuhui, Wei Fangyun, Zhang Chao, et al. EAGLE: Speculative sampling requires rethinking feature uncertainty[C]//Proc of the Int Conf on Machine Learning (ICML). New York: PMLR, 2024: 28935–28948

[15] Yang Nan, Ge Tao, Wang Liang, et al. Inference with reference: Lossless acceleration of large language models[J]. arXiv preprint, arXiv: 2304.04487, 2023

[16] He Zhenyu, Zhong Zexuan, Cai Tianle, et al. REST: Retrieval-based

speculative decoding[C]//Proc of the Conf of the North American Chapter of the Association for Computational Linguistics (NACCL). Stroudsburg, PA: ACL, 2024: 1582–1595

[17] Hu E J, Shen Y, Wallis P, et al. LoRA: Low-rank adaptation of large language models[C]//Proc of the Int Conf on Learning Representations (ICLR). Open Access: OpenReview. net, 2022: 1–13

[18] Lyu Zhonghao, Zhu Guangxu, Xu Jie, et al. Semantic communications for image recovery and classification via deep joint source and channel coding[J]. IEEE Transactions on Wireless Communications, 2024, 23(8): 8388−8404

[19] Kang Yiping, Hauswald J, Gao Cao, et al. Neurosurgeon: Collaborative intelligence between the cloud and mobile edge[C]//Proc of the 22nd Int Conf on Architectural Support for Programming Languages and Operating Systems (ASPLOS). New York: ACM, 2017: 615–629

[20] Matsubara Y, Levorato M, Restuccia F. Split computing and early exit for deep learning applications: Survey and research challenges[J]. ACM Computing Surveys, 2024, 55(5): 1−30

[21] Borzunov A, Baranchuk D, Dettmers T, et al. Petals: Collaborative inference and fine-tuning of large models[C]//Proc of Workshop on Broadening Research Collaborations. Stroudsburg, PA: ACL, 2022: 558–568

[22] Ryabinin M, Dettmers T, Diskin M, et al. SWARM parallelism: Training large models can be surprisingly communication-efficient[C]//Proc of the Int Conf on Machine Learning (ICML). New York: PMLR, 2023: 29416–29440

[23] Lyu Zhonghao, Xiao Ming, Xu Jie, et al. The larger the merrier? Efficient large AI model inference in wireless edge networks[J]. IEEE Journal on Selected Areas in Communications, 2025, 44: 2839−2853

[24] Frantar E, Ashkboos S, Hoefler T, et al. GPTQ: Accurate post-training compression for generative pretrained transformers[C]//Proc of the Int Conf on Learning Representations (ICLR). (ICLR). Open Access: OpenReview. net, 2023: 1–16

[25] Lin Ji, Tang Jiaming, Tang Haotian, et al. AWQ: Activation-aware weight quantization for on-device LLM compression and acceleration[J]. GetMobile: Mobile Computing and Communications, 2025, 28(4): 12−17

[26] Li Muyang, Cai Tianle, Cao Jiaxin, et al. DistriFusion: Distributed parallel inference for high-resolution diffusion models[C]//Proc of the IEEE/CVF Conf on Computer Vision and Pattern Recognition (CVPR). Piscataway, NJ: IEEE, 2024: 7183–7193

[27] Guo Minyi. Opportunities and challenges of network infrastructure in the era of large language model[J]. Journal of Computer Research and Development, 2024, 61(11): 2663−2663 (in Chinese)
(过敏意. 大模型时代网络基础设施的机遇与挑战[J]. 计算机研究与发展, 2024, 61(11): 2663−2663)

[28] Park Jihong, Samarakoon S, Bennis M, et al. Wireless network intelligence at the edge[J]. Proceedings of the IEEE, 2019, 107(11): 2204−2239

[29] Zhu Guangxu, Liu Dongzhu, Du Yuqing, et al. Toward an intelligent edge: Wireless communication meets machine learning[J]. IEEE Communications Magazine, 2020, 58(1): 19−25

[30] Xin Ji, Tang R, Lee J, et al. DeeBERT: Dynamic early exiting for accelerating BERT inference[C]//Proc of the 58th Annual Meeting of the Association for Computational Linguistics (ACL). Stroudsburg, PA: ACL, 2020: 2246–2251

[31] Laskaridis S, Venieris S I, Almeida M, et al. SPINN: Synergistic progressive inference of neural networks over device and cloud[C]//Proc of the Annual Int Conf on Mobile Computing and Networking (MobiCom). New York: ACM, 2020: 1–15

[32] Ning Jiahong, Zheng Ce, Yang Tingting. DSSD: Efficient edge-device deployment and collaborative inference via distributed split speculative decoding[C]//Proc of ICML on Machine Learning for Wireless Communication and Networks (ML4Wireless). New York: PMLR, 2025: 1–8

[33] Malladi S, Gao T, Nichani E, et al. Fine-tuning language models with just forward passes[C]//Proc of the 37th Int Conf on Neural Information Processing Systems (NIPS). New York: Curran Associates, 2023: 53038–53075

[34] Gu Yuxian, Dong Li, Wei Furu, et al. MiniLLM: Knowledge distillation of large language models[C]//Proc of the Int Conf on Learning Representations (ICLR). Open Access: OpenReview. net, 2024: 1–24

[35] Zhao Yao, Xie Zhitian, Liang Chen, et al. Lookahead: An inference acceleration framework for large language model with lossless generation accuracy[C]//Proc of the 30th ACM SIGKDD Conf on Knowledge Discovery and Data Mining. New York: ACM, 2024: 6344–6355



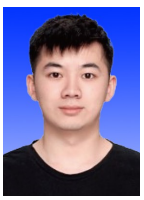
Li Yuchen, born in 1997. PhD. His main research interests include data mining and LLMs.

李豫晨, 1997 年生。博士。主要研究方向为数据挖掘、大语言模型。



Kong Rui, born in 1998. PhD. His main research interests include data mininig and LLMs.

孔　芮, 1998 年生。博士。主要研究方向为数据挖掘、大语言模型。



Chen Xinran, born in 1998. PhD. His main research interests include information retrieval and LLMs.

陈新冉, 1998 年生。博士。主要研究方向为信息检索、大语言模型。



Yin Dawei, born in 1984. PhD. His main research interests include data mining and information retrieval.

殷大伟, 1984 年生。博士。主要研究方向为数据挖掘、信息检索。



Kong Linghe, born in 1983. PhD, professor, PhD supervisor. His main research interests include Internet of things and AI.

孔令和, 1983 年生。博士，教授，博士生导师。主要研究方向为物联网、人工智能。



Chen Guihai, born in 1963. PhD, professor, PhD supervisor. His main research interest includes computer networks.

陈贵海, 1963 年生。博士，教授，博士生导师。主要研究方向为计算机网络。