通过摊销优化实现稳健且可扩展的 SLA 分解级联 Transformer

Cyril Shih-Huan
Hsu 阿姆斯特丹大学信息
学研究所 荷兰阿姆斯特丹
s.h.hsu@uva.nl

arXiv:2601.11859v1 [cs.NI] 17 Jan 2026

摘要—**随着网络不断向 6G 演进，网络切片日益成为在共享物理基础设施上提供定制化端到端（E2E）逻辑网络的关键。当前面临的一项重大挑战，是如何有效地将端到端服务级别协议（SLA）分解为特定域服务级别协议；现有解决方案通过计算密集型迭代优化过程来处理这一问题，因此会带来显著的时延和复杂性。为此，我们提出 Casformer，这是一种级联 Transformer 架构，旨在实现快速、无需优化的 SLA 分解。Casformer 在第一层利用特定域 Transformer 编码器对历史域反馈进行编码，并在第二层通过基于 Transformer 的聚合器整合跨域依赖关系。该模型采用受领域信息神经网络（DINN）启发的学习范式进行训练，将风险感知建模与摊销优化相结合，从而学习稳定的单向前馈 SLA 分解策略。大量评估结果表明，与最先进的基于优化的框架相比，Casformer 能够提升 SLA 分解质量，并在多变且含噪的网络条件下展现出更强的可扩展性和稳健性。此外，其单向前馈设计降低了运行时复杂度，并简化了部署和维护。这些发现揭示了将摊销优化与基于 Transformer 的序列建模相结合以推进网络自动化的潜力，并为先进的 5G 及后续网络环境中的实时 SLA 管理提供了一种可扩展且高效的解决方案。**

索引词——**网络切片、服务级别协议、服务质量、深度神经网络、优化、Transformer**

一、引言

5G 的部署以及向 6G 的演进，已将移动网络基础设施重新定义为一种灵活的多服务平台，以满足多样化垂直行业的需求。这一转型的核心推动因素是网络切片，它允许在共享物理底层之上实例化多个端到端（E2E）逻辑网络，并为每个网络配置定制化的服务级别协议（SLA）。这些服务级别协议通过服务等级目标（SLO）规定性能承诺，涵盖时延、吞吐量、可靠性和安全性等关键指标。单个网络切片可能跨越多个管理域和技术域，例如网络的接入域、传输域和核心域。为确保端到端 SLA 在这些异构域中得到一致交付，必须将 SLA 分解为各个域能够支持的特定域服务等级协议。这种分解对于实现最优资源分配并保持在切片生命周期内实现 SLA 合规性至关重要。尽管已有多项研究应对了相关挑战，但它们的侧重点各不相同。例如，IETF 草案 [1] 探讨了将端到端切片需求映射至底层传送基础设施的难题。在 [2], 中，重点在于自动化、编排和 SLA 监控，而 [3] 讨论了 SLA 参数与传送资源配置之间的相互作用。其他研究（例如 [4], ）强调了 SLA 分解在资源管理中的重要性，而 [5] 提出了人工智能辅助框架，以在未来的 6G 环境中实现该过程的自动化。近期的一项研究 [6]强调了在多域和多技术环境中实现可扩展 SLA 分解的必要性。

一种常见的网络切片管理架构是分层两级管理系统，如图 1 所示。在顶层，服务编排器负责网络切片的端到端生命周期，而较低层的域控制器则负责在各个域中本地实例化切片。编排器负责分解 SLA，但通常难以实时掌握域级状态，而是依赖历史性能数据。为弥补这一不足， [7] 提出了一种两步方法：先采用非参数风险模型估计底层域的行为，再进行基于优化的分解。 [8], 进一步拓展了这一方向，引入基于神经网络的风险建模； [9], [10],则进一步完善了该方法，提出动态自适应框架，以反映不同粒度下不断变化的条件。尽管取得了这些进展，现有方法仍依赖迭代优化和频繁的在线模型更新，这可能带来显著的计算开销和时延，尤其是在动态且大规模的环境中。此外，这些方法需要在运行期间持续维护特定于域的风险模型，使部署和维护更加复杂。在本工作中，我们提出 *Casformer*，这是一种级联 Transformer 架构，通过将分解过程摊销为一次前向传播来解决上述局限。Casformer 经过训练，能够模拟现有基于风险模型的框架所产生的最优决策，从而无需在线优化或模型更新，即可显著加快 SLA 分解；因此，它在提升可扩展性和效率、简化部署的同时，不牺牲决策质量。本文的主要贡献如下：

- **无优化SLA分解**。 我们提出 Casformer，这是一种新颖的框架，通过摊销优化直接预测 SLA 分解决策，从而绕过迭代优化。Casformer 融入领域知识，并采用受领域知情神经网络启发的结构化学习范式，能够高效、可扩展地进行分解，同时不损害决策质量。

- **Transformer的创新应用**。Casformer 将风险建模直接融入级联式 Transformer 结构中。它对历史域级反馈进行编码，并动态聚合这些信息，以生成准确的 SLA 分解。这种对 Transformer 的创新运用，充分发挥了其强大的序列建模能力，以完成网络优化任务。

- **广泛的实证验证**。与先进方法相比，全面的实验表明，我们的方法在推理效率、可扩展性和泛化能力方面均有显著提升，为在网络优化场景中应用摊销优化和基于 Transformer 的架构提供了有力的实践支持。

本文结构安排如下：第二节讨论相关工作。第三节定义系统模型和问题。第四节介绍我们提出的框架。第五节详细说明仿真设置，第六节展示并讨论结果。最后，第七节总结全文。

II. 相关工作

A. SLA 管理

早期关于SLA 分解的研究主要聚焦于单域环境。 [11] 中的作者通过将高层SLO转换为低层系统阈值，并利用性能界限确定资源需求，解决了云系统中的SLA 管理问题。在 [12],中，作者提出了一种SLA 管理架构，利用神经网络估算在单域网络切片场景下满足服务级别协议所需的资源。另一方面，如何在多域、多技术环境中确保SLA保障所面临的挑战也日益受到关注。 [13] 中的作者强调了医疗保健等关键应用中发生SLA违约的风险，指出了在分布式基础设施中实现稳健SLA保障的重要性。在 [6], 中，作者提出了一种面向多域、多技术网络的SLA 管理与编排架构，支持各种分布式方案。其他研究 [14],[15] 提出了用于辅助SLA 管理的预测方法。例如， [14] 引入了一个映射层，用于管理局部服务区域中的无线电资源。在 [15], 中，研究人员开发了一种基于深度学习的框架，用于估算SLA约束下的资源需求。然而，这些方法主要关注预测方法或架构设计，并未明确解决跨多个域的端到端SLA分解问题。

B. 端到端*SLA*分解

为应对端到端SLA分解这一挑战，已有多项研究探索了基于优化和学习的技术。在 [4], 中，作者利用数学模型研究了端到端网络切片的资源分配。近期， [16] 提出了一种监督学习方法，将端到端SLA分解为跨多个域的特定域SLO。在这一方向上， [7], 提出了一项开创性工作，采用两步法：首先利用非参数风险模型，根据历史观测估算领域行为，然后应用优化过程确定分解结果。在 [8], 中，该方法进一步得到扩展，以单调神经网络取代非参数风险模型，从而提升效率。近期， [9], [10], [17] 引入了基于在线学习的框架，以更好地适应动态时变网络环境。

尽管这些方法展现出日益增强的复杂性和适应性，但仍高度依赖在线优化和反复的模型更新，这可能会带来较高的计算开销，并且容易受到超参数的影响。相比之下，我们提出的方法通过学习一种仅需前向计算的分解过程神经近似模型，无需实时优化，从而提升了可扩展性、响应速度和泛化能力。

III. 系统模型

A. 问题描述

多域、多技术网络中的 SLA 分解已通过多种编排方案得到研究 [6]。在集中式方法中，跨域编排器收集全局状态信息，并集中做出分解和资源分配决策。半分布式方案保留集中式 SLA 分解，但将域内资源管理委托给域控制器。然而，这些方法面临可扩展性限制、隐私问题和单点故障。相比之下，完全分布式方案假设跨域编排器无法了解特定域状态。相反，它根据历史反馈分配部分 SLA，而每个域编排器独立管理资源分配。图 1 展示了完全分布式方案的一个示例：跨域编排器接收端到端 SLA，并将其分解为各底层域的部分 SLA。特定域编排器（例如接入、传输、核心）根据所分配的 SLA，独立管理资源分配和执行。域将反馈发送回跨域编排器，以便为未来的分解决策提供依据。这种去中心化方案具有更好的可扩展性和韧性，但也带来了新的挑战，即如何确保各个本地决策共同满足端到端 SLA。已有研究 [10] 表明，在多个域之间寻找最优分解是一个 NP 难问题。在本研究中，我们聚焦于这一全分布式环境

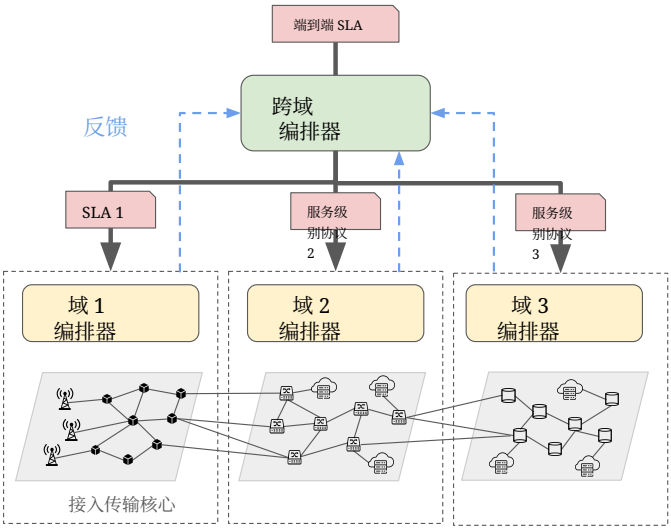


图 1：采用反馈驱动分解的完全分布式多域网络管理架构，涵盖接入域、传输域和核心域。

并旨在设计一种高效、自适应且无需优化的分解框架，以适应域自治下的实时运行。

B. 问题建模

SLA 分解。 考虑一个以向量$\mathbf{d}_{\text{e2e}}$表示的端到端 SLA，该向量规定了时延、吞吐量和可靠性等关键服务指标的性能界限。该端到端 SLA 必须分解到 N 个彼此独立运营的域中。分解过程会生成一组部分 SLA $\{\mathbf{d}_n\}_{n=1}^{N}$，其中$\mathbf{d}_n$ 是分配给域 n的 SLA 部分。

分解必须满足一组聚合规则，以确保全局 SLA 得以保留。这些规则反映了服务指标的结构。例如，端到端时延 τ_{e2e} 具有可加性，吞吐量 θ_{e2e} 由瓶颈决定，而可靠性 π_{e2e} 则具有乘法关系：

$$\tau_{\text{e2e}} = \sum_{n=1}^{N} \tau_n, \quad \theta_{\text{e2e}} = \min_{n} \theta_n, \quad \pi_{\text{e2e}} = \prod_{n=1}^{N} \pi_n. \tag{1}$$

在 SLA 包含上述指标的场景中，特定于域的 SLA 分配可表示为一个向量：

$$\mathbf{d}_n = (\tau_n, \theta_n, \pi_n). \tag{2}$$

令 $p_n(\mathbf{d}_n)$表示域n 对 SLA 片段 $\mathbf{d}_n$的接受概率。该概率由本地域控制器根据内部资源可用性、策略和流量条件决定。由于各域独立运行，我们假设接受决策在统计上相互独立。因此，分解方案 $\{\mathbf{d}_n\}_{n=1}^{N}$ 的端到端接受概率为：

$$p_{\text{e2e}} = \prod_{n=1}^{N} p_n(\mathbf{d}_n). \tag{3}$$

SLA 分解的目标是找到集合 $\{\mathbf{d}_n\}_{n=1}^{N}$，使其满足聚合约束，同时最大化所有域接受分配给它们的 SLA 片段的概率。因此，分解问题可表述为：

$$\begin{aligned} \max_{\{\mathbf{d}_n\}} \quad & \prod_{n=1}^{N} p_n(\mathbf{d}_n) \\ \text{s.t.} \quad & \mathbf{d}_{\text{e2e}} = G(\{\mathbf{d}_n\}), \quad \forall n \in \{1, \ldots, N\}, \end{aligned} \tag{4}$$

其中， $G(\cdot)$ 表示将部分 SLA 聚合为完整端到端 SLA 的聚合函数，如公式 (1) 所述。在本研究中，我们仅关注公式 (1) 所述 SLA 的时延分量τ 。这一选择同时基于实践和分析方面的考虑。时延是端到端服务交付中最关键的性能指标之一，直接影响服务质量（QoS）。然而，通过针对其他指标引入适当的聚合规则，第四节中提出的框架可以直接扩展，以处理多维SLA。

此外，在动态环境中，域条件会随着时间推移而演变，这是由于网络状态、流量负载和运行策略发生变化，使得接受概率$p_n(\cdot)$ 具有时间依赖性。我们将其表示为时间步 t 时的 $p_n^t(\cdot)$。因此，公式 (4) 中的 SLA 分解问题扩展至包含 T 个时间步的时间范围，并表示为：

$$\begin{aligned} \max_{\{\mathbf{d}_n^t\}} \quad & \frac{1}{T} \sum_{t=1}^{T} \prod_{n=1}^{N} p_n^t(\mathbf{d}_n^t) \\ \text{s.t.} \quad & \mathbf{d}_{\text{e2e}}^t = G(\{\mathbf{d}_n^t\}), \\ & \forall t \in \{1, \ldots, T\}, \quad \forall n \in \{1, \ldots, N\}. \end{aligned} \tag{5}$$

风险建模。 在全分布式环境中，域 n 对部分服务级别协议 {style id='3'>$\mathbf{d}_n$ 的接受情况本质上具有不确定性，因为跨域编排器无法获知各域的内部状态。因此，接受概率 $p_n(\cdot)$未知，无法直接求解式（4）中的分解问题。不过，可以通过引入特定于域的风险模型 $f_n(\cdot)$来估计 $p_n(\cdot)$，从而应对这一挑战。在实践中，真实的接受概率 $p_n(\cdot)$无法直接观测，必须根据历史反馈进行推断。假设每个域都维护一个包含 K过往观测记录 $\{(\mathbf{x}_i,\ y_i)\}_{i=1}^{K}$的数据集，其中$\mathbf{x}_i$ 表示先前提出的 SLA 向量，而 $y_i \in \{0, 1\}$ 表示域的响应（$y_i = 1$ 表示接受，$y_i = 0$ 表示拒绝）。训练风险模型 $f_n : \mathbb{R}^d \to [0, 1]$，通过最小化数据集上的交叉熵损失来近似 $p_n(\cdot)$：

$$\mathcal{L}_n = -\sum_{i=1}^{K} \left[y_i \log f_n(\mathbf{x}_i) + (1 - y_i) \log \left(1 - f_n(\mathbf{x}_i)\right) \right]. \tag{6}$$

风险模型 f_n 可以根据设计选择和可用数据采用各种函数形式。神经网络尤其适合捕捉不同 SLA 属性之间微妙的依赖关系和交互作用，因为它们具有强大的建模能力，能够刻画 SLA参数与域接受行为之间的复杂关系 [8]。

IV. 方法论

A. 背景

RADE。实时自适应分解（RADE） [9]是一种先进的SLA分解框架，旨在动态、完全分布式的网络环境中运行。与依赖固定风险模型的静态分解方法不同，RADE 纳入了在线学习机制，能够根据域控制器的实时反馈持续调整其决策过程。RADE 遵循两步分解架构，并以先前的工作为基础 [7], [8]:

1. **风险模型构建**。 跨域编排器为每个域构建并维护一个特定域风险模型，通常以神经网络的形式实现。这些模型使用历史反馈数据进行训练，如（6）所述。

2. **SLA 分解**。 对于收到的端到端 SLA 请求，跨域编排器通过求解（4）中定义的优化问题来执行分解，该问题旨在利用已学习的风险模型最大化预期的端到端接受概率。

第一步利用深度学习中广泛采用的基于梯度的优化技术来训练神经网络模型。第二步将粗粒度启发式搜索与通过序列最小二乘规划（SLSQP）进行的细粒度细化相结合，使编排器能够有效探索可行分解空间，并收敛到近似最优解。

算法 1 RADE框架

需要： 风险模型 $\{\mathcal{F}_n\}_{n=1}^N$；内存缓冲区 $\{\mathcal{B}_n\}_{n=1}^N$

对于 每个时间步 $t = 1$ 到 T **执行**

  收集近期反馈并更新 $\{\mathcal{B}_n^t\}_{n=1}^N$

  更新$\{\mathcal{F}_n\}_{n=1}^N$ ，采用 OGD:

$$\{\mathcal{F}_n\}_{n=1}^N \leftarrow \text{OGD}(\{\mathcal{F}_n\}_{n=1}^N, \{\mathcal{B}_n^t\}_{n=1}^N)$$

  接收端到端 SLA 请求 τ_{e2e}^t

  求解 分解:

$$\{\tau_n^{*,t}\}_{n=1}^N \leftarrow \text{Optimize}(\{\mathcal{F}_n\}_{n=1}^N, \tau_{\text{e2e}}^t)$$

  返回服务级别协议分配 $\{\tau_n^{*,t}\}_{n=1}^N$

结束循环

此外，为了在随时间变化的网络条件下保持有效性，RADE 会利用从近期 SLA 反馈中收集的观测数据，通过在线梯度下降（OGD）持续更新其风险模型。为稳定学习过程，并避免对短期波动或受污染的反馈过拟合，RADE 采用先进先出（FIFO）内存缓冲区。该缓冲区维护近期反馈的滑动窗口，确保模型更新所用的数据集具有代表性且在时间上相关。因此，RADE 能够随着时间推移维持更准确、适应性更强的风险模型，从而改善决策并提高长期平均端到端接受概率，如式（5）所示。算法 1 展示了 RADE 框架的完整高层步骤。 [9],展示了 RADE 的有效性，表明其在动态网络环境中具有更优的性能和鲁棒性。

Transformer。Transformer 是一类深度学习模型，彻底改变了序列建模，尤其是在自然语言处理领域。Transformer 架构首次于 [18],提出，凭借更强的可扩展性、并行性以及建模长程依赖的能力，在许多自然语言处理任务中取代了传统的循环神经网络（RNN）和卷积神经网络（CNN）。它们处理结构化输入和长期依赖关系的能力，使其适用于 SLA 分解等任务；在这类任务中，必须对域反馈或 SLA 指标序列进行整体处理。

领域知识引导神经网络（DINNs）。DINNs 是一类神经网络架构，会将结构化先验知识、物理约束或系统特定洞见纳入学习过程 [19]–[22]。与纯数据驱动模型不同，DINNs 利用领域专业知识指导学习，从而提升泛化能力，尤其适用于标注数据有限或获取成本高昂的情况。领域知识的整合可以采取多种形式，例如:

- **架构设计：**将特定领域的结构（如对称性或守恒定律）嵌入网络层或连接中。
- **特征编码：**利用已知的物理、语义或几何关系对原始输入进行预处理或嵌入，从而生成信息更丰富的表示。
- **损失正则化：**将物理信息约束或基于逻辑的约束作为辅助损失项引入，以确保输出符合已知规则。
- **混合模型：**将数据驱动模型与解析模型或机理模型相结合（例如仿真引擎或微分方程）。

DINNs 的一个典型例子是物理信息神经网络（ PINNs）这一类模型 [23],，它们将支配系统的物理定律（通常以偏微分方程表示）直接融入训练过程。这种方法在标注数据稀缺、但领域知识充分的情况下展现出很强的泛化能力，凸显了将结构化先验融入深度学习模型的有效性。简而言之，通过将领域知识融入模型，DINNs 在深度学习的灵活性与专家驱动设计的稳健性之间取得了平衡。

B. 风险感知级联变换器

为解决基于在线优化的两步式 SLA 分解的局限性，尤其是其对噪声反馈、频繁模型更新和复杂运行时依赖关系的敏感性，我们提出了风险感知级联

Transformer（Casformer）。Casformer 通过一次前向传播，直接根据原始特定领域反馈预测 SLA 分解，以学习得到的稳定分解策略取代显式的逐实例优化。Casformer 遵循受 DINNs 启发的学习范式，由领域知识指导训练，而非仅依赖通用监督。在我们的实现中，我们采用先进的风险感知优化框架 RADE 作为教师，以生成高质量的分解目标。我们选用 RADE 作为代表性示例，是因为它具有稳健性且性能出色；但所提出的框架**并不局限于 RADE**，也可以使用任何能够提供可靠分解决策作为监督的方法进行训练。Casformer 将分解过程摊销到神经网络模型中，从而内化教师的结构逻辑和风险感知行为。训练完成后，Casformer 无需依赖教师即可独立运行，免去持续优化和模型维护，简化部署。

图 2 展示了所提出框架的整体架构。具体而言，在左侧，持续收集来自各个域的反馈，并将其存储在域特定的 FIFO 内存缓冲区中，以保留近期交互历史。在中下方，RADE 模块充当教师，首先使用最新的反馈样本更新其域特定风险模型。随后，这些更新后的模型通过结合启发式搜索与 SLSQP 的两阶段优化过程求解服务级别协议分解问题，从而得出各域时延分配的最优值。同时，Casformer 在中上方充当学生，接收端到端 SLA 请求，并通过其第一层编码器处理各域缓冲区，以生成定长表示。随后，这些表示与端到端 SLA 预算一同输入跨领域聚合器，最终输出分解比例。KL 散度损失将这些预测与 RADE 生成的目标对齐，并将 RADE 视为教师。两个模型由同一个端到端 SLA 请求同时触发，其中 RADE 生成监督信号。

架构。Casformer 采用两层级联 Transformer 架构，用于编码各域的 SLA 反馈，并生成与端到端 SLA 目标相一致的分解预测。它由两个主要组件构成：(*i*) 特定领域编码器和 (*ii*) 跨领域聚合器。

- **内存缓冲区**。与 RADE 类似，Casformer 为每个域维护一个内存缓冲区 $\mathcal{B}$，用于存储近期的 SLA 交互记录。该缓冲区以 FIFO 队列的形式运行，并提供随时间推移学习领域行为所需的时间上下文。缓冲区中的每条记录都是一个元组 $(x,\ y) \in \mathcal{B}$，表示一个 SLA 请求及其二元接受结果。值得注意的是，缓冲区的容量有限，其大小对模型的性能起着关键作用。如果缓冲区过大，可能会保留过时的数据，从而误导模型并降低预测准确率。相反，如果缓冲区过小，模型可能没有足够的上下文来进行稳定的训练，导致预测结果波动。这些存储的样本构成特定领域编码器的输入序列，并支持在线和离线训练。

- **特定领域反馈编码器（第 1 层）**。在第一层中，为每个域 $n \in \{1, \ldots, N\}$ 分配一个专用的 Transformer 编码器 $\mathcal{F}_{\text{enc}n}$，负责处理该域的本地反馈历史。每个编码器以先进先出缓冲区 $\mathcal{B}_n$中的一组近期 SLA 交互记录作为输入。编码器输出一个定长嵌入 $\mathbf{h}_n \in \mathbb{R}^d$，用于概括相应域的近期行为：

$$\mathbf{h}_n = \mathcal{F}_n^{\text{enc}}(\mathcal{B}_n), \tag{7}$$

其中，$\mathcal{F}_n^{\text{enc}}(\cdot)$ 表示与域 n关联的编码器。此过程采用通用句子编码器（USE） [24], 的架构，该架构旨在将可变长度序列编码为固定大小的表示。在我们的场景中，USE 风格架构使 Casformer 能够灵活处理不同的窗口大小和域历史，同时将其表示标准化，以便进行下游处理。

- **跨领域聚合器（第 2 层）**。在第二层中，单个 Transformer 编码器充当跨领域聚合器 $\mathcal{F}_{\text{agg}}$。它以第一层生成的域嵌入集合 $\{\mathbf{h}_n\}_{n=1}^N$ 以及输入的端到端延迟 SLAτ_{e2e}为输入，对域间的依赖关系进行建模，并生成最终预测的延迟分解分配 $\{\hat{}\tau_n\}_{n=1}^N$。具体而言，为了纳入输入的端到端延迟 SLAτ_{e2e}，我们首先使用可学习的权重矩阵 $W_\tau \in \mathbb{R}^{1\times d}$将其投影到嵌入空间。然后，将所得嵌入加到每个域表示上，这与标准 Transformer 中应用位置编码的方式类似：

$$\{\tilde{\mathbf{h}}_n\}_{n=1}^N = \{\mathbf{h}_n + W_\tau \cdot \tau_{\text{e2e}}\}_{n=1}^N. \tag{8}$$

此外，为了满足已知的延迟 SLA 加和约束，即部分延迟之和必须等于式（1）所述的端到端延迟，我们将这一领域知识直接融入模型架构。网络不预测部分延迟的原始值，而是输出一个在各个 N 域上的 softmax 分布，以表示延迟分配比例：

$$\hat{\boldsymbol{\phi}} = \text{softmax}\left(\mathcal{F}^{\text{agg}}(\{\tilde{\mathbf{h}}_n\}_{n=1}^N)\right), \tag{9}$$

其中，$\hat{}\,\boldsymbol{\phi} = \{\hat{}\phi_n\}_{n=1}^N$ 表示预测的分解比例。随后，将这些比例与给定的端到端延迟 SLA 相乘，即可得到最终的各域时延分配：

$$\hat{\boldsymbol{\tau}} = \hat{\boldsymbol{\phi}} \cdot \tau_{\text{e2e}}, \tag{10}$$

其中，$\hat{}\boldsymbol{\tau} = \{\hat{}\tau_n\}_{n=1}^N$ 表示实际预测的分解分配。该方法无需使用约束惩罚损失项，并确保所有预测天然满足 SLA 结构。

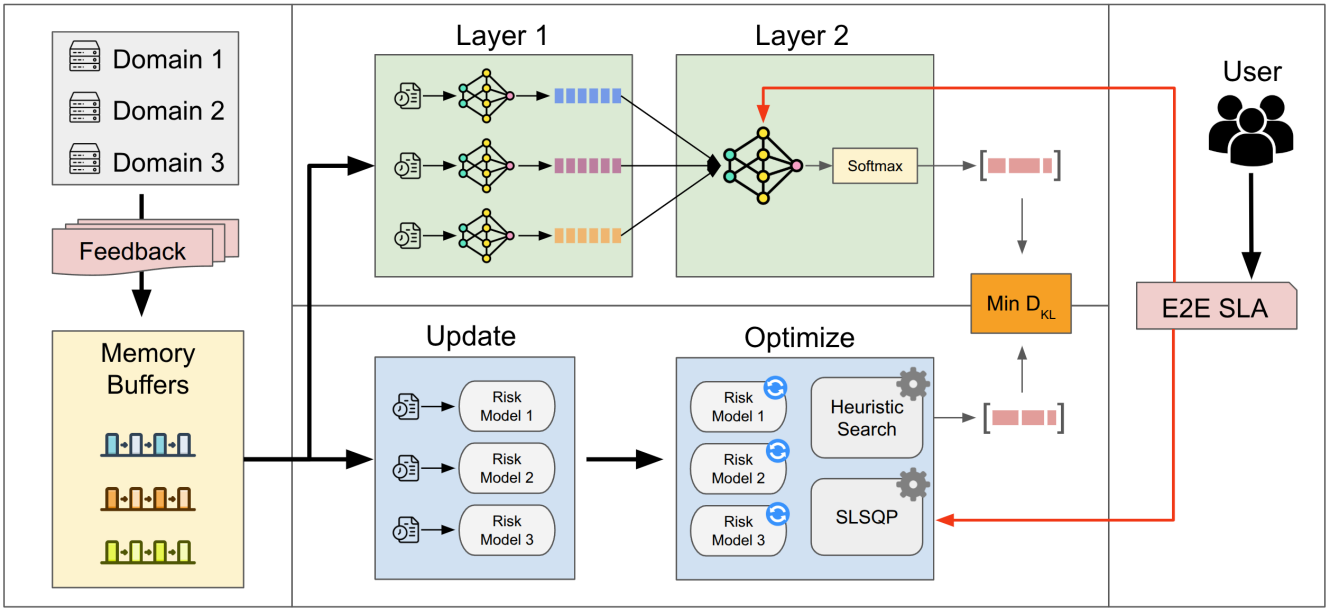


图 2：RADE 监督引导下 Casformer 训练流程概览。将从底层域收集的反馈存储在内存缓冲区中，并用于训练 Casformer（上），使其模仿 RADE（下）所做出的分解决策。Casformer 通过一次前向传播学习预测分解比例，从而实现快速、无需优化的推理。

总体而言，Casformer 可表示为特定领域编码器、跨领域聚合器和 Softmax 操作的组合：$\mathcal{F} = \text{softmax} \circ \mathcal{F}^{\text{agg}} \circ \mathcal{F}^{\text{enc}}$。特定领域编码器中的注意力机制使模型能够为各个反馈样本分配不同的权重，从而在总结领域行为时，聚焦于最相关的历史信息。同样，跨领域聚合器中的注意力机制能够根据所有其他域的当前状况，动态调整部分 SLA 分配。这种设计还天然支持可变长度输入，这意味着模型无需进行架构修改即可处理数量不断变化的域。总体而言，双层设计使 Casformer 能够先提取各域的局部见解，再将其整体整合，以做出分解决策。

训练阶段。 在训练期间，Casformer 与 RADE 并行运行。对于在时间步 t到达的每个端到端 SLA 请求，执行以下步骤：

1. 每个域的内存缓冲区 n 都会提供一组近期反馈样本。2. 每个域专属编码器独立处理其缓冲区，使用基于 Transformer 的架构将反馈序列编码为固定长度的表示，如式 (7) 所定义。3. 跨领域聚合器收集各域的表示，将传入的端到端延迟 SLA 作为上下文输入，并按式 (9) 所述输出预测的分解比例 $\hat{\phi}^t = \{\hat{\phi}_n^t\}_{n=1}^N$。4. 与此同时，RADE 使用其风险模型和优化过程，为同一 SLA 请求计算目标分解$\{\tau_n^{*,t}\}_{n=1}^N$，如算法 1 所述。5. 随后将目标分解归一化为各域上的比例：

$$\phi^{*,t} = \text{normSum}(\{\tau_n^{*,t}\}_{n=1}^N) = \frac{\{\tau_n^{*,t}\}_{n=1}^N}{\sum_{n=1}^N \tau_n^{*,t}}. \quad (11)$$

6. 训练 Casformer，使目标分解比例$\hat{\phi}^{*,t}$与预测分解比例 ϕ^t之间的 Kullback–Leibler（KL）散度最小化：

$$D_{\text{KL}}\left(\phi^{*,t} \,\middle\|\, \hat{\phi}^t\right) = \sum_{n=1}^N \phi_n^{*,t} \log \frac{\phi_n^{*,t}}{\hat{\phi}_n^t}. \quad (12)$$

$\hat{}$虽然 ϕ^t 和 $\phi^{*,t}$ 表示的是分解比例，而非概率分布本身，但它们均非负且总和为一。这种结构上的相似性使得它们可以被解释为定义在各域上的类别分布。因此，我们可以合理地使用 KL散度来衡量预测分解比例与目标分解比例之间的距离。

完整的高层训练流程如算法 2 所示。在这一过程中，Casformer 隐式地学习了 RADE 所编码的底层分解策略，同时将原始域反馈纳入其单步预测模型中。这一设计遵循 DINN 范式，即由专家驱动的逻辑（也就是 RADE 的优化约束和结果）指导数据驱动模型的训练。此外，由于训练无需在线进行，因此可以收集训练样本并重复用于离线训练。

推理阶段。 训练完成后，Casformer 可独立于 RADE 运行。具体而言，在每个时间步：

算法 2 训练流程

要求： Casformer $\mathcal{F}$；RADE $\mathcal{R}$；内存缓冲区 $\{\mathcal{B}_n\}_{n=1}^{N}$

对于 每个时间步 $t = 1$ 到 T **执行**
  收集近期反馈并更新 $\{\mathcal{B}_n^t\}_{n=1}^{N}$
  接收新的端到端 SLA 请求 τ_{e2e}^t
  $\hat{\boldsymbol{\phi}}^t \leftarrow \mathcal{F}(\{\mathcal{B}_n^t\}_{n=1}^{N},\ \tau_{\text{e2e}}^t)$ // Casformer 预测
  $\boldsymbol{\phi}^{*,t} \leftarrow \text{normSum}(\mathcal{R}(\tau_{\text{e2e}}^t))$ // RADE 计算最优值
  计算损失：$\mathcal{L}_{\text{train}} = D_{\text{KL}}(\boldsymbol{\phi}^{*,t} \,\|\, \hat{\boldsymbol{\phi}}^t)$
  通过最小化 $\mathcal{L}_{\text{train}}$更新 $\mathcal{F}$
end for

1. 将各域缓冲区中的反馈序列输入各自的特定领域编码器。2. 特定领域编码器按照式 (7) 生成定长反馈表示，并将其传递给跨领域聚合器。3. 跨领域聚合器通过一次前向传播直接输出预测的分解分配，如式 (8)–(10) 所示。

完整的高级推理过程总结于算法 3 中。这一仅前向推理过程无需运行时优化、风险模型维护和梯度更新。因此，Casformer 在保留从 RADE 中学到的分解质量的同时，显著加快了决策速度。

算法 3 推理过程

要求： Casformer $\mathcal{F}$；内存缓冲区 $\{\mathcal{B}_n\}_{n=1}^{N}$

对于 每个时间步 $t = 1$ 到 T **执行**
  收集近期反馈并更新 $\{\mathcal{B}_n^t\}_{n=1}^{N}$
  接收新的端到端 SLA 请求 τ_{e2e}^t
  $\hat{\boldsymbol{\phi}}^t \leftarrow \mathcal{F}(\{\mathcal{B}_n^t\}_{n=1}^{N},\ \tau_{\text{e2e}}^t)$ // 预测比例
  生成服务级别协议分配：$\{\hat{\tau}_n^t\}_{n=1}^{N} \leftarrow \hat{\boldsymbol{\phi}}^t \cdot \tau_{\text{e2e}}^t$
  返回 $\{\hat{\tau}_n^t\}_{n=1}^{N}$
结束循环

摊销优化视角。 从功能上看，Casformer 充当摊销模型 [25], ，可将端到端 SLA 请求和域反馈直接映射为分解决策。具体而言，通过最小化预测的分解比例与 RADE 生成的分解比例之间的期望 KL散度，可以学习这一映射：

$$\mathcal{F}_\theta^* = \arg\min_\theta \mathbb{E}_{(\{\mathcal{B}_n\},\tau_{\text{e2e}})} \left[D_{\text{KL}} \left(\mathcal{F}_\theta(\{\mathcal{B}_n\}, \tau_{\text{e2e}}), \boldsymbol{\phi}^* \right) \right]. \quad (13)$$

这种方法利用神经网络近似，有效地将基于优化的分解成本摊销到一段时间内，使部署时能够快速进行仅前向传播的推理，同时保留 RADE 优化行为的结构逻辑。

总之，Casformer 提供了一种有理论依据的单阶段预测框架，用于 SLA 分解。该框架将领域知识、反馈驱动建模和架构约束整合到级联 Transformer 设计中。

V. 实验设置

A. 评估场景

为在现实条件下评估我们的 SLA分解框架，我们考虑了 [10],提出的时变仿真环境，在该环境中，各域根据自身容量和系统负载决定是否通过准入控制接受服务级别协议请求。我们使用第五节所述的流量驱动环境，在 $T = 200$ 个离散时间步上进行仿真。每个时间步都会到达一个新的端到端服务请求，其时延预算从区间 $[90, 110]$ms 中均匀采样。具体而言，在每个时间步 t，Casformer 会预测一个分解分配 $\{d_i^t\}_{i=1}^{N}$，随后使用真实域接纳模型进行评估，以计算实际的端到端接受概率。总体性能则按照式 (5) 的定义，通过对所有时间步的端到端接受概率取平均来衡量。Casformer 首先在训练轨迹上进行训练，在此期间与 RADE 并行运行。在这一阶段，RADE 会持续更新风险模型并计算最优分解，而 Casformer 则学习根据原始的特定领域反馈近似 RADE 的输出。训练收敛后，我们在测试轨迹上独立评估 Casformer，此时它不再与 RADE 一同运行。为确保公平比较，所有基线方法也都在同一测试轨迹上进行评估。我们将 Casformer 与以下基线进行比较：

非风险感知（NRA）。一种简单的基线方法，将端到端时延预算均匀分解到各个域中。该方法不使用学习或反馈。此基线提供了一个简单且不考虑信息的参照点，用以凸显风险建模和自适应分解的重要性。

RADE。一种最先进的分解框架 [9] ，它使用持续更新的风险模型和在线优化（包括启发式搜索和 SLSQP），在每个时间步计算分解结果，如第 IV-A 节所述。

最优（OPT）。一种理论上界方法，利用穷举搜索和对真实接受函数的完整认知，在每一步计算最佳分解结果。

B. 配置

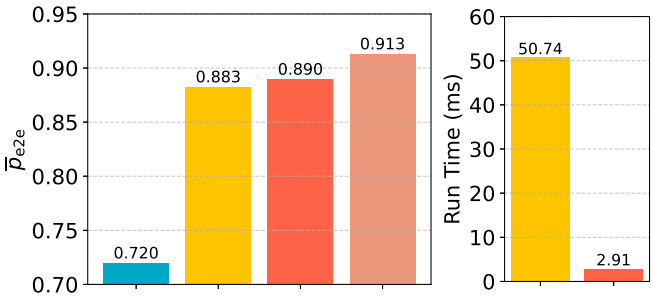
我们模拟了一个涉及 $N = 3$ 个网络切片的场景，以评估所提出的 Casformer 框架和基线方法的性能。为体现异构且动态变化的网络条件，每个域均使用在现实范围内随机采样的值进行参数化，具体配置详见 [10]。RADE 框架的配置遵循 [9] 中的设定。对于分解过程，首先通过启发式搜索，在可行域中均匀采样 10，000 个候选解。随后，将其中的最优候选解作为 SLSQP 算法的初始解，以进行进一步优化。对于 Casformer，特定领域反馈编码器（第 1 层）和跨领域聚合器（第 2 层）均采用 2 层 Transformer 编码器实现，隐藏层大小为 16，多层感知机大小为 64。训练使用 AdamW优化器，

学习率同为 0.01，并且每个域都维护一个大小为 100 的内存缓冲区。所有实验均使用不同的随机种子，独立重复运行 100 次模拟，并对结果取平均，以确保统计显著性。模拟环境使用 Python 3.11.9 实现，模型基于 PyTorch 2.5.1 构建。所有实验均在一台配备 Intel Core i7- 10700K 中央处理器、32GB 内存和 RTX 2080S 图形处理器的服务器上运行。

VI. 结果与讨论

A. 长期性能与效率

为评估 Cas-former 的有效性和效率，我们根据两项关键指标将其性能与多个基线进行比较：(*i*) 长期平均端到端 SLA 接受概率，记作 p_{e2e}，其定义见式 (5)；以及 (*ii*) 在推理时计算分解决策所需的平均运行时间。图 3a 展示了平均端到端



(a) 端到端接受概率　　(b) 运行时间

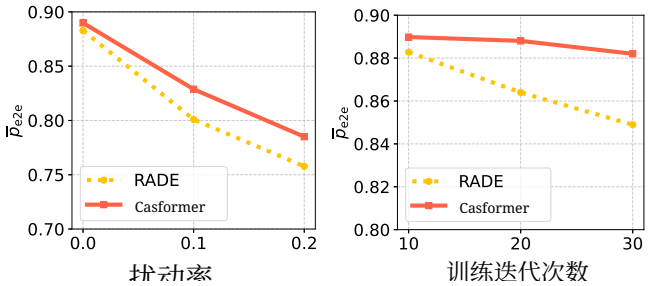
图 3：性能比较：(a) 随时间变化的平均端到端 SLA 接受概率；(b) 每次分解的运行时间。

整个仿真时域内的接受概率。Casformer 和 RADE 均接近最优上界。Casformer 的平均接受概率达到 0.89，略优于其教师模型 RADE，并显著优于简单基线 NRA。尽管 Casformer 的训练目标是模仿 RADE 的决策，但其性能却出乎意料地超越了 RADE。这一点颇为有趣，因为学生模型通常旨在达到而非超越教师的性能。我们认为，这种提升源于 Casformer 能够很好地泛化到未见过的条件，从而平滑 RADE 中的噪声或过拟合行为。第 VI-B 节将进一步探讨其泛化能力。

图 3b 展示了每次分解决策的平均运行时间。Casformer 每次推理仅需 2.91 毫秒，比 RADE 快 17 倍以上。通过消除在线优化的需求，改为仅对学习得到的模型执行一次前向传播来进行分解，计算成本得以大幅降低。总而言之，这些结果表明，与 RADE 相比，Casformer 能够实现具有竞争力甚至更优的长期决策质量，同时支持在运行时快速进行 SLA 分解。

B. 模型泛化能力

为考察训练不完美情况下的泛化能力信号方面，我们考虑了针对泛化不同方面的两种互补压力测试：(*i*) 输入级损坏和 (*ii*) 目标级过拟合。图 4a 展示了反馈损坏程度逐渐增加时的结果，其中每个历史样本都以固定概率发生标签翻转。该设置模拟了因瞬时网络条件（如配置错误、系统故障或中断）导致的域准入控制暂时性异常。随着扰动率上升，两种模型的性能均有所下降。然而，Casformer 始终保持着高于 RADE 的平均接受概率，表明其对输入级噪声具有更强的鲁棒性。我们将此归因于摊销优化的正则化作用 [25]，它促使表示更加平滑，并减轻离群值的影响。图 4b



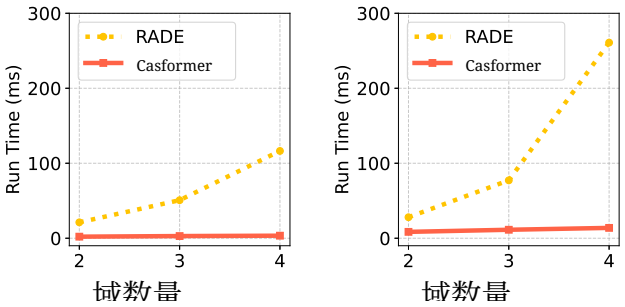
(a) 数据损坏　　(b) 目标过拟合

图 4：两种条件下的泛化能力比较：(a) 输入级损坏；(b) 目标级过拟合。

考察了模型对过拟合的敏感性。具体而言，我们改变 RADE 在线风险模型更新所使用的 OGD迭代次数。增加更新次数有助于模型更好地拟合缓冲数据，但也会增加其对内存缓冲区中有限内容过拟合的风险。如图所示，随着训练迭代次数增加，RADE 的性能有所下降，而 Casformer 基本不受影响，这凸显了它能够捕捉超越特定训练轨迹的有效通用模式。综合来看，这些结果表明，与 RADE 相比，Casformer 在噪声输入条件下以及面对目标过拟合风险时均具有更强的泛化能力。这种鲁棒性支撑了其在未见场景中的有效性。

C. 计算可扩展性

图 5 展示了在涉及 2、3 和 4 个域的场景中，模型在图形处理器和中央处理器上的推理运行时间。如图所示，得益于仅前向架构和并行化的基于 Transformer 的设计，随着域的数量增加，Casformer 的推理时间基本保持不变。相比之下，RADE 的运行时间会随着域数量增加而显著增长，尤其是在中央处理器上，因为按照设计，RADE 必须在运行时解决复杂度不断提高的优化问题。这些结果表明，Casformer 的推理速度可扩展性远高于 RADE，因此更适用于多域网络系统中的实时应用。



(a) 图形处理器　(b) 中央处理器

图5：随着域数量增加，推理的可扩展性。（a）图形处理器和（b）中央处理器上的平均运行时间。

VII. 结论

本文探讨了网络切片中跨多个域分解端到端 SLA 所面临的一项关键挑战。现有的 SLA 分解方法严重依赖迭代优化，并频繁在线更新域特定的风险模型，因而带来了巨大的计算开销和时延。为克服这些局限，我们提出了 Casformer，这是一种支持无优化 SLA 分解的风险感知级联 Transformer。Casformer 创新性地采用级联 Transformer 架构和受 DINNs 启发的学习范式，通过一次前向传播直接预测 SLA 分解决策。这种方法无需在线优化和更新风险模型，显著降低了系统复杂度。大量实验表明，在噪声和易变的网络环境中，Casformer 能够实现出色的可扩展性和泛化能力，同时保持高质量的 SLA 分解。此外，相较于基于优化的方法，其仅前向推理降低了运行时复杂度，简化了部署，同时不牺牲决策质量。总之，我们的工作彰显了摊销优化和 Transformer 架构在网络自动化中的潜力，为新兴的 5G及后续网络中的 SLA 管理提供了一种可扩展、高效且便于实际部署的解决方案。

致谢

本研究部分由荷兰6G旗舰项目“未来网络服务”资助。

参考文献

[1] X. Geng、L. M. Contreras、R. Rokui、J. Dong 和 I. Bykov，“3GPP 5G端到端网络切片中的 IETF 网络切片应用”，互联网工程任务组，互联网草案 draft-ietf-teas-5g-network-slice-application-03，2024年6月。[2] R. Swamy 和 S. K. M，“5G网络切片”，HCL Technologies，技术报告，2023年。[3] P. Iovanna、M. Svensson、A. Shapin、G. Bottari、F. Ubaldi、F. Ponzini 和 M. Puleri，“端到端网络切片编排”，爱立信技术评论，第2卷，2022年。[4] R. Su、D. Zhang、R. Venkatesan、Z. Gong、C. Li、F. Ding、F. Jiang 和 Z. Zhu，“5G电信网络中的网络切片资源分配：原理与模型综述”，*IEEE Network*，第33卷，第6期，第172–179页，2019年。

[5] J. Wang、J. Liu、J. Li 和 N. Kato，“人工智能辅助网络切片：6G中的网络保障与服务供应”，IEEE Vehicular Technology Magazine，第18卷，第1期，第49–58页，2023年。[6] H. H. Esmat 和 B. Lorenzo，“面向可持续端到端多域多技术网络切片的 SLA 分解”，IEEE Wireless Communications，第31卷，第4期，第237–244页，2024年。[7] D. De Vleeschauwer、C. Papagianni 和 A. Walid，“网络切片的服务级别协议分解”，IEEE Communications Letters，第25卷，第3期，第950–954页，2021年3月。[8] C. S.-H. Hsu、D. De Vleeschauwer 和 C. Papagianni，“网络切片的 SLA 分解：一种深度神经网络方法”，IEEE Networking Letters，第1–1页，2023年。[9] C. S.-H. Hsu、C. Papagianni、P. Grosso 和 D. De Vleeschauwer，“在线服务级别协议分解：实现对不断演变的网络系统的实时适应”，载于2025年欧洲网络与通信联合会议暨6G峰会（EuCNC/6G Summit）。IEEE，2025年，第55–60页。[10] C. S.-H. Hsu、C. Papagianni 和 P. Grosso，“RAILS：面向多域网络中联合 SLA 分解与服务提供商管理的风险感知迭代局部搜索”，载于2025年第26届IEEE高性能交换与路由国际会议（HPSR），2025年，第1–6页。[11] Y. Chen、S. Iyer、X. Liu、D. Milojicic 和 A. Sahai，“SLA 分解：将服务级别目标转换为系统级阈值”，载于第四届自主计算国际会议（ICAC’07），2007年，第3–3页。[12] E. Kapassa、M. Touloupou 和 D. Kyriazis，“5G中的服务级别协议：促进虚拟网络功能和网络切片定制化服务级别协议管理的完整框架”，载于2018年第32届国际先进信息网络与应用会议研讨会（WAINA），2018年，第469–474页。[13] H. N. Qureshi、M. Manalastas、S. M. A. Zaidi、A. Imran 和 M. O. Al Kalaa，“面向5G及后续技术的服务级别协议：5G医疗保健系统的概述、挑战与推动因素”，IEEE Access，第9卷，第1044–1061页，2021年。[14] B. Khodapanah、A. Awada、I. Viering、D. Oehmann、M. Simsek 和 G. P. Fettweis，“通过切片感知无线资源管理实现5G网络服务级别协议履行”，载于2018年第87届IEEE车辆技术会议（VTC春季会议），2018年，第1–6页。[15] H. Chergui 和 C. Verikoukis，“用于5G网络可靠动态端到端切片的离线服务级别协议约束深度学习”，IEEE通信选定领域期刊，第38卷，第2期，第350–360页，2020年。[16] M. Iannelli、M. R. Rahman、N. Choi 和 L. Wang，“将机器学习应用于端到端切片服务级别协议分解”，载于2020年第6届IEEE网络软化会议（NetSoft），2020年，第92–99页。[17] D. Cheng、R. Sheshadri、A. Kak、N. Choi、X. Zhou 和 B. Ji，“Odin：通过在线学习实现5G/6G网络切片的有效端到端服务级别协议分解”，载于第二十六届移动网络与移动计算理论、算法基础及协议设计国际研讨会论文集，2025年，第131–140页。[18] A. Vaswani、N. Shazeer、N. Parmar、J. Uszkoreit、L. Jones、A. N. Gomez、L. Kaiser 和 I. Polosukhin，“Attention is all you need”，载于神经信息处理系统进展，第30卷，2017年，第5998–6008页。[19] J. Willard、X. Jia、S. Xu、M. Steinbach 和 V. Kumar，“将科学知识与机器学习相结合，助力工程和环境系统”，ACM计算调查，第55卷，第4期，2022年11月。[20] J. P. Morgan、A. Paiement 和 C. Klinke，“领域知识引导的图神经网络：量子化学案例研究”，Neural Networks，第165卷，第938–952页，2023年。[21] T. Beucler、M. Pritchard、S. Rasp、J. Ott、P. Baldi 和 P. Gentine，“在模拟物理系统的神经网络中施加解析约束”，Physical Review Letters，第126卷，第9期，第098302页，2021年。[22] A. Quarteroni、P. Gervasio 和 F. Regazzoni，“结合基于物理的模型与数据驱动模型：科学机器学习推动研究前沿”，arXiv预印本 arXiv:2501.18708，2025年。[23] M. Raissi、P. Perdikaris 和 G. Karniadakis，“物理信息神经网络：用于求解涉及非线性偏微分方程的正问题与逆问题的深度学习框架”，计算物理学杂志，第378卷，第686–707页，2019年。[24] D. Cer、Y. Yang、S.-y. Kong、N. Hua、N. Limtiaco、R. S. John、N. Constant、M. Guajardo-Cespedes、S. Yuan、C. Tar 等，“通用句子编码器”，arXiv预印本 arXiv:1803.11175，2018年。[25] B. Amos，“摊销优化教程”，Found. Trends Mach. Learn.，第16卷，第5期，第592–732页，2023年6月。