arXiv:2302.05336v1 [cs.DC] 9 Feb 2023

通过资源使用预测实现边缘智能主动容错

Theodoros Theodoropoulos雅典
哈罗科皮奥大学信息学与远程信息处理系，雅典，希腊 ttheod@hua.gr

John Violos
雅典国立理工大学电气与计算机工程学院，雅典，希腊 violos@mail.ntua.gr

Stylianos Tsanakas
雅典国立理工大学电气与计算机工程学院，雅典，希腊 el09727@mail.ntua.gr

Aris Leivadeas
软件与信息技术工程系 高等技术学院 蒙特利尔，加拿大 aris.leivadeas@etsmtl.ca

Konstantinos Tserpes
信息学与远程信息处理系 雅典哈罗科皮奥大学 雅典，希腊 tserpes@hua.gr

Theodora Varvarigou
电气与计算机工程学院 雅典国立理工大学 雅典，希腊 dora@telecom.ntua.gr

2023年2月13日

摘要

高要求应用的普及以及边缘计算的发展，使得高效管理底层计算基础设施成为必要，也促使服务提供商重新思考其运营方式。本文提出一种智能主动容错（IPFT）方法，通过循环神经网络（RNN）利用边缘资源使用预测。具体而言，我们关注进程故障：由于处理能力不足，基础设施无法在可接受范围内提供服务质量（QoS），由此引发此类故障。为应对这一挑战，我们提出一种复合深度学习架构，用于预测边缘节点的资源使用指标，并触发主动节点复制和任务迁移。考虑到边缘计算基础设施也具有高度动态和异构的特点，我们提出一种创新的混合贝叶斯进化策略（HBES）算法，以自动调整资源使用模型。我们通过实验评估了所提出的资源使用预测机制，并将其与其他先进方法进行了比较；结果显示，该机制在均方根误差（RMSE）和平均绝对误差（MAE）方面均有显著改进。此外，我们还在 CloudSim Plus 中进行了大规模仿真，评估利用资源使用预测的 IPFT 机制。结果表明，与被动容错方法相比，该机制在可靠性和可维护性方面有显著提升。

关键词 边缘计算 · 容错 · 循环神经网络 · 深度学习 · 进化策略 ·贝叶斯优化 · 超参数调优

1 引言

过去十年间，科学界见证了与一系列高要求 QoS 要求密切相关的应用不断涌现。扩展现实（XR） [1]应用就是此类应用的一个例子。XR 应用与各种 QoS 要求相关联 [2] ，这些要求取决于能否提供沉浸式终端用户体验。此类要求可能包括延迟和带宽等方面。研究表明，要使终端用户获得可接受的沉浸式体验，端到端时延不得超过 15 毫秒，带宽则应达到最高 30 Gbps [3]。此外，任务处理中的故障可能会造成服务交付中断，从而危及上述沉浸式体验所需的完整性。因此，这类应用具备容错能力至关重要。此外，这类应用对计算资源也有各种高要求，因为它们涉及三维模型和精细图形的渲染。鉴于这些计算需求，若采用单体开发架构，为提供所需的计算资源，终端用户设备将变得极其庞大且成本高昂 [4]。

云计算通过提供可供这些应用经由互联网运行的计算资源，能够在一定程度上减轻终端用户设备所承受的负担。云计算基于共享计算资源，这些资源可能分布在多个地点。因此，部分计算负担会转移到这些共享资源上。遗憾的是，终端用户设备与云服务器之间的距离可能导致高延迟和可用带宽不足。因此，人们需要将处理能力和数据移至更靠近设备的位置，因为数据正是在这些设备上生成的 [5] 。对于 XR 应用而言，这些设备可能包括智能对象、移动电话、网络网关、传感器以及各种沉浸式设备。这种分布式计算范式称为边缘计算，旨在建立去中心化拓扑，并使各种计算资源和存储资源能够迁移至更靠近网络边缘的位置。这样一来，预计服务交付和内容缓存能够获得更短的响应时间和更高的传输速率。上述设备的计算能力可能存在很大差异。因此，必须确保在边缘执行的计算任务不会过于繁重，也不会超出相关设备的计算能力。

考虑到现代应用的性质和需求，工作负载执行具备韧性并满足行业制定的服务质量标准至关重要。网络边缘的设备所承载的卸载任务数量会随时间大幅波动 [6]。因此，至关重要的是，这些波动不会影响系统性能并导致进程故障 [7]。此外，边缘计算环境在任务和相关处理节点方面具有极强的异构性和动态性。这种前所未有的情况催生了对 IPFT方法的需求，该方法应能适应基础设施和工作负载的变化。

监控并预测边缘节点在 CPU、随机存取存储器、带宽和磁盘等资源指标方面的运行容量，对于实施容错策略而言是有价值的信息。资源指标具有较高的序列相关性和交叉相关性，因此采用时间序列方法是合理的 [8]。基于回归的循环神经网络 [9], 可借助门控循环单元（GRU） [10] 或长短期记忆网络（LSTM） [11], 利用时间序列特征，从而准确预测资源指标。
为应对边缘环境极端的异构性和动态性，我们提出了一套系统化方法，利用历史数据自动构建深度学习模型。基于人工试错方法来创建可接受的深度学习架构，是一种常见做法；每当已部署的应用、用户行为或边缘基础设施发生变化，深度学习专家都需要投入大量工作时间。另一方面，现有的深度学习自动化方法仍存在显著缺陷，例如效率低、计算需求高 [12]。一种可能的解决方案是扩展进化算法，并将其与其他模型结合进行超参数调优。

上述情况促使我们提出一种侧重于处理故障的 IPFT方法；这些故障与资源短缺有关，由此导致的处理能力不足会妨碍底层基础设施在可接受的服务质量范围内执行任务。我们的研究目标是提出一种复合深度学习架构，以统一的方式预测边缘节点执行传入工作负载的能力，并设计一套能够确保高级容错能力的适当运行流程。该流程的基石是主动式运行能力。每当预计任务执行会出现瓶颈时，就应触发任务迁移和节点复制等主动措施。复合深度学习架构应利用边缘资源和相关任务的时间序列特征，这些特征可由监控系统（即 Prometheus）提供 [13]。最后，我们提出 HBES优化算法，以构建近乎最优的复合深度学习模型。

我们的研究有四项主要贡献: :

- 提出了 IPFT 方法，该方法具有很高的可靠性和可维护性，并且在及时进行故障检测和修复方面表现出色。
- 探讨一类特定故障——进程故障之间的关联，以及如何通过处理边缘节点的资源利用率指标预测这些故障
- 提出并分析一种用于边缘资源使用预测的复合深度学习模型的理论原理，该模型包含两个通道：一个采用前馈结构，另一个基于 RNN 层。
- 提出一种创新的混合超参数优化模型，将进化策略与贝叶斯优化算法相结合，以评估接近最优的复合深度学习架构。

本文其余部分的结构如下：第 2 节介绍容错、资源使用预测、时间序列、深度学习和超参数优化技术方面的相关工作。第 III 节说明主动容错机制如何利用资源使用量预测。第 4 节分析 RNN 多输出回归方法、复合深度学习架构和 HBES 方法。第 5 节介绍基于真实边缘计算数据集的实验设置、在 CloudSim Plus 中对 IPFT 的仿真以及评估结果。第 6 节总结全文，报告当前的局限性并提出未来研究方向。

2 相关工作

容错机制主要分为两类：被动式和主动式。反应式方法在边缘基础设施发生故障后，降低故障所造成的影响。主要的被动容错方法包括被动复制、重新提交、重试和检查点。例如， [14], 提出了一种面向大规模图并行系统的先进复制式容错机制，该机制支持对顶点状态进行低成本维护。该机制通过正常的消息交换来复制顶点，并能够从其他机器上的副本中快速进行失败顶点的内存中重构。复制提高了系统的可靠性以及任务成功完成的可能性，但需要额外的资源来实现冗余。

已有研究 [15] 提出了一种使用幂等HTTP方法的重试方法，用于卸载和处理执行故障。该重试策略的优势在于，它利用的计算环境资源最少，并能最大限度地减少用户耗时；但代价是响应时间增加，因为HTTP方法可能需要多次重试，直到成功完成。关于检查点， [16]研究了一种适用于无服务器范式的被动容错方法。具体而言，作者通过检查点和容器实时迁移，成功地为资源受限的设备节省了资源。然而，在这种情况下，执行时间同样会增加，因为其中包含了故障服务器的恢复时间。

在主动容错方法中，会预测潜在故障，以避免其对任务执行的影响。Tian 等人 [17] 采用基于树的模型，这是一种统计分析技术，用于诊断高风险云任务，并应用虚拟机迁移技术。尽管该方法显著提高了可靠性和效率，但由于无法自动适应新的计算环境，因此存在泛化局限性。机器学习和在线学习方法也已与微服务架构和物联网系统结合使用，以检测故障模式并预先缓解故障 [18]。 [19]最近提出了另一种先进的主动式模型。该模型执行多步预测，以估计不同时间粒度下的进程故障和 QoS 退化。该方法利用编码器-解码器模型，评估基础设施以不同生产率处理到达任务的能力。

由上述内容可见，处理节点有限的计算能力为边缘计算和物联网应用设置了障碍 [20]。我们可以通过高效的资源管理来克服这些障碍 [21]。这一过程包括确保定义明确的 QoS 指标和准确的工作负载预测 [22]。一些工作负载预测模型利用循环神经网络，尤其是长短期记忆网络，将资源使用指标表示为数据序列 [23]。这些方法都没有探索主动容错方法如何利用资源使用量预测，而是选择专注于实际的资源使用量预测过程 [24]。

例如， [8] 中的作者使用自回归积分滑动平均（ARIMA），以避免数据中心出现资源配置不足或资源配置过度。ARIMA 的局限性在于，它对线性依赖关系进行建模，并且基于平稳性假设。然而，正如 [25] 中所指出的，在边缘处理的工作负载具有趋势、季节性和执行行为中的非线性，这限制了统计线性模型的应用。机器学习模型（如 K均值、决策树和 K近邻） [26]克服了这些局限性。尽管目前已有大量公开可用的经典机器学习模型，但在我们的实验中，我们选择使用 XGBoost [27] ，因为它在 Kaggle 和其他知名机器学习

竞赛中广受欢迎。XGBoost 主要使用梯度提升决策树，并且以开源软件库的形式提供。

机器学习方法的局限性在于，每当边缘云基础设施、用户行为或应用发生变化时，都需要在人为协助下从头训练新模型。自动化机器学习能够自动引导模型的学习过程，在无需人工干预的情况下最大限度地提升性能，并最大限度地降低计算开销。在云计算领域，应用与用户上下文资源预测器（AUCROP） [29] 已被提出，用于自动应用传统机器学习算法。此外，Auto-sklearn [30]是一种通用的自动化机器学习元模型，可通过贝叶斯优化进行数据预处理、回归和超参数调优。

Keras-Tuner [31] 是 Keras 推出的超参数调优自动化方法，也称为超参数优化。Keras 是深度学习社区中最受欢迎的框架之一。Keras-Tuner 的优势在于，超模型、超参数范围和调优流程都与 Keras 无缝集成，但它仅支持以下优化器：（a）随机搜索；（b）Hyperband，即采用提前停止的随机搜索；以及（c）贝叶斯优化。在我们的实验评估中，使用了 Keras-Tuner、AUCROP 和 Auto-sklearn。在这项工作中，我们还将进化策略与贝叶斯优化相结合，进一步拓展了超参数调优研究。因此，我们提出了创新的 HBES 方法，将其作为一种卓越的自动化深度学习解决方案，用于应对边缘计算环境的异质性和动态性。

更具体地说，我们的工作旨在通过提出一种利用资源使用预测的主动容错方法，扩展资源使用预测方法，并解决上述局限性，具体如下：首先，IPFT 机制要求执行节点至少有一定数量的副本，因为只有在预测到故障后才会请求复制。其次，IFPT 不计入故障后任务重新调度的时间开销，因为任务复制和重新调度会及时、主动地进行。第三，IPFT 利用深度学习循环神经网络模型克服统计模型的局限性，并适应非线性、非平稳的资源指标。最后，引入的 HBES 使整个流程具备普适性，而这正是相关文献中许多方法的常见局限。

3 利用资源使用预测实现主动容错

3.1 边缘计算中的资源使用预测

利用各种资源利用率指标，可以改进边缘计算基础设施的管理与编排。其中最重要的指标包括中央处理器、随机存取存储器、带宽和磁盘输入/输出。同时，边缘计算范式具有动态性，且处理边缘节点存在异构性；这些节点必须在服务质量要求所规定的特定约束下运行。

因此，在动态异构环境中进行决策制定是一个相当复杂的过程，需要利用所有可用的信息来源。通过利用历史数据的时间序列特征来预测资源消耗指标，是最有价值的信息之一。它可以作为处理节点可用性的有力指标，用于判断其是否能够接收额外的工作负载，或预测未来时间步中潜在的 QoS 退化。因此，Prometheus、OpenTSDB、Nagios 和 InfuxDB 等公开可用的监控工具可以通过流格式或类似 PromQL 的时间序列数据库提供资源指标。这些时间序列数据库可用于生成适合循环神经网络模型训练的数据集。

边缘节点的动态行为源于应用请求及其工作负载的波动。每个时间间隔内的请求数量会在不同时间范围内发生变化，并受到许多周期性现象的影响。此外，边缘具有高度异构性，因为边缘节点可能具有不同的软硬件特征，例如内存、计算能力等。当考虑到在同一边缘基础设施中共存并协同工作的各种型号的树莓派、Arduino、传感器节点及其他微控制器时，这种异构性会更加明显。

与此同时，应用所有者可以针对边缘节点设定严格的性能要求，涉及可用性、吞吐量以及各种潜在延迟。因此，边缘提供商难以确保 QoS 指标处于规定的可接受范围内。因此，为了保证基础设施具备充足的计算容量以应对不断变化的需求，容错机制应主动根据资源数量和处理节点的可用性作出决策。

科学文献中，不同类别的故障对应着特定的容错机制。故障的主要类别包括：(a) 网络故障，(b) 物理故障，(c) 进程故障和 (d) 服务到期故障 [32]。在这些故障中，进程故障可能非常严重。具体而言，进程故障会发生在进程中

这是由于资源短缺所致，进而导致任务执行延迟加剧，甚至执行中断。这类故障最终可能造成性能下降，对于实时和/或时间敏感型应用而言，这种下降是不可接受的。

因此，显而易见，任务的执行会对资源利用率产生重大影响，反之亦然。这一事实促使我们研究资源利用率指标，以便采取主动容错措施，减轻进程故障带来的不利影响。

3.2 主动容错

鉴于边缘计算范式的运作方式依赖大量计算节点同时运行，因此必须将组件故障视为不可避免的情况。为此，必须确保基础设施能够持续运行，不发生中断，也不会出现 QoS 恶化。确保服务连续性的主要方式是触发迁移策略，并使用备用组件自动替换发生故障的组件，从而保证 QoS。

节点（如虚拟机）的复制过程需要一定的等待时间，这可能导致 QoS 退化。因此，应采用主动式方法实现容错。在任何时候，网络中都应包含特定数量的计算节点；当某个正在运行的组件停止正常工作时，这些节点便可投入使用。由于可能需要冗余计算节点，因此应尽可能减少冗余。然而，借助机器学习算法，可以提取有关服务行为模式以及所发生进程故障的信息。因此，容错功能得以主动发挥作用，确保各项操作持续进行、不受中断，同时将总体冗余成本降至最小。

所提出的 IPFT 模型通过使用与分布式边缘环境中的资源使用量相关联的数据特征来进行故障预测。IPFT 监控每个处理节点上发生的资源消耗，以便在运行时发现可能导致潜在 QoS 退化的处理能力不足。如果已部署的资源无法在指定时间范围内满足不断增长的需求，IPFT 将触发缓解策略，例如主动式节点复制和任务执行迁移。IPFT 机制中节点的多通道神经网络部分在预测自身未来状态时，会考虑其他可用节点的状态。这样，在预测到故障的情况下，只要已有正常运行的节点具备足够的计算能力，便可以将任务迁移到这些节点上。通过这种方式，我们可以避免立即采用节点复制，因为节点复制可能会增加设置新节点所需的部署成本和时间。

如前所述，节点复制需要经过一定的等待时间，这可能会严重影响边缘环境的性能。IPFT 可以在长于复制时间的时间范围内预测未来对节点复制的需求。因此，及时触发节点复制过程并执行相应的任务迁移，可以防止进程故障的发生。如图 1.1 所示，当预测某个处理节点将出现高资源使用量时，便会触发节点复制过程（图 1.2）。这样，未来到来的任务将由新节点承载，从而避免任务被拒绝和执行时间过长。

IPFT 的运行方式包含 4 个主要阶段，如图 2 所示。阶段 1 中，它会监控处理节点的资源指标。阶段 2 中，它会预测预计在指定时间范围内发生的最大资源使用量。至于阶段 3，则有三种不同的场景决定后续操作的执行方式。阶段 3 的处理会针对每个处理节点独立进行。三种不同的场景及其与阶段 4 相应操作之间的关联可概述如下：

- 阶段 3A。如果资源使用预测低于特定的下界阈值，则认为该处理节点利用率不足，因此会针对该节点发出退役请求。在阶段 4A 中，分配给该节点的任务会被转移到其他位置，并完成退役流程。

- 阶段 3B。如果资源使用预测处于下界阈值与上界阈值之间，则认为资源消耗率处于理想水平，因此无需采取任何额外措施。

- 阶段 3C。如果资源使用预测高于特定的上界阈值，则认为该处理节点负载过高，因此会发出复制请求。阶段 4C 在创建新的处理节点后进行。在阶段 4C 中，分配给负载过高处理节点的一部分任务会被重定向到新创建的处理节点。

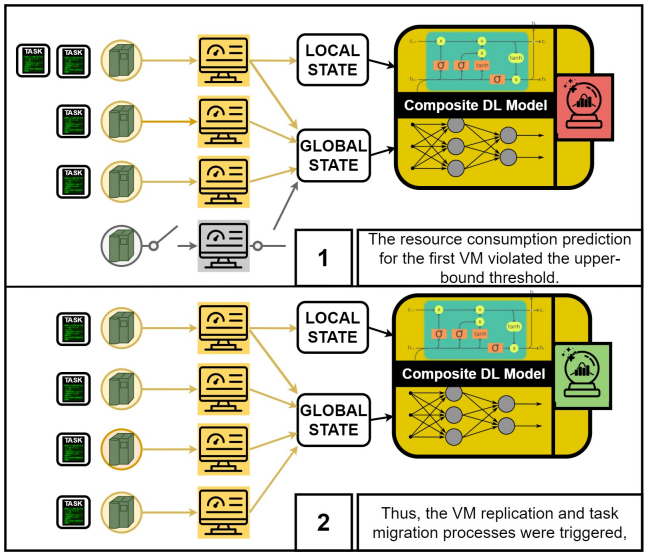

图 1：IPFT 根据资源利用率预测触发虚拟机复制和任务迁移。

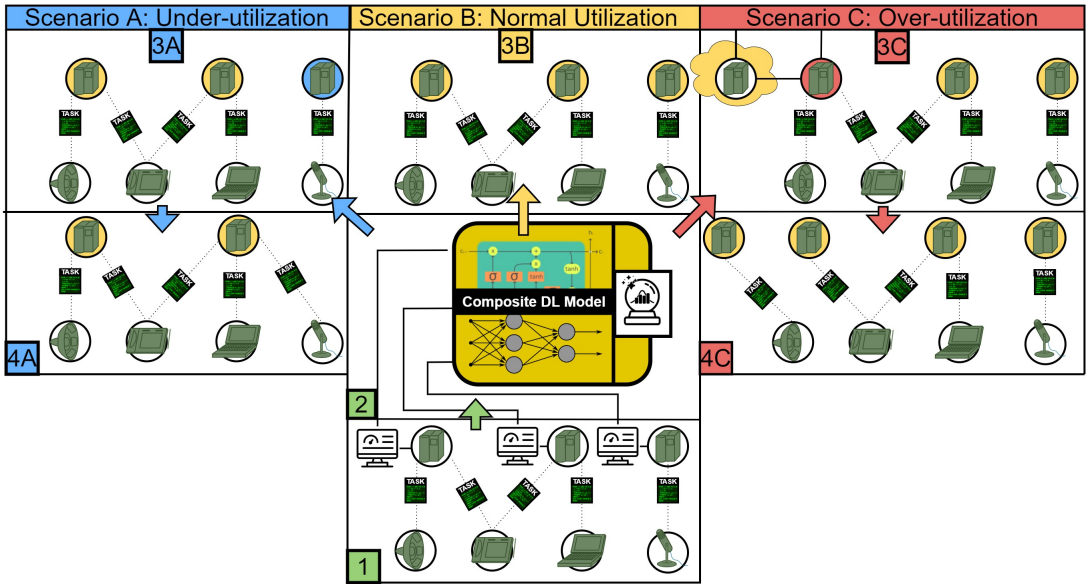

图 2：IPFT 流水线的四个阶段，横跨三个不同的场景。

IPFT与工作负载均衡机制协同工作，该机制会为资源利用率预测较低的节点赋予更高优先级，而为资源利用率预测较高的节点赋予更低优先级。这是许多采用不同标准来平衡工作负载的任务卸载机制的常见做法[33]。例如，MinMin任务卸载方法会优先将较小的任务分配到较早可用的节点上执行。而MaxMin任务卸载方法则优先将较大的任务分配到较早可用的处理节点上执行。任务的大小根据其包含的百万条指令数量或预计完成时间进行估算。通过这种方式，IPFT在任务迁移过程中考虑资源使用预测，从而改变任务卸载机制的行为。

3.3 基于阈值的决策

当资源使用量预测指标高于指定阈值时，就会执行任务迁移和节点复制。这种基于阈值的方法广泛应用于云/边缘计算中的许多决策制定机制[34]。IPFT包含两个阈值。如果资源利用率预测值高于上界阈值，IPFT就会调用节点复制过程。如果与特定处理节点对应的预测值低于下界阈值，则关闭该处理节点（例如，以降低总能耗[35]）。

这两个阈值的恰当选择对 IPFT 的性能至关重要。上界阈值过高会导致系统对工作负载波动不够敏感和被动式不足。而上界阈值过低则会使系统做出响应并触发不必要的节点复制。同样，下界阈值也应进行适当的微调。阈值过低会使基础设施继续使用利用率不足的处理节点。下界阈值过高还会关闭对边缘基础设施平稳运行而言必不可少的处理节点[36]。

为了确定最优阈值，我们提出采用网格搜索方法，通过迭代尝试一系列阈值，使其逐步收敛到接近最优的值。这些值可最大化可靠性和可维护性这两项容错评估指标，相关内容将在后续实验部分进行深入讨论。阈值的选择在很大程度上取决于每个应用的特性以及底层物理基础设施。相关文献针对类似问题和机制提供了建议；尽管这些建议只能取得次优结果，但仍可作为制定阈值选择标准的重要指导。[37]

鉴于预测结果相当准确，且阈值选取合理，最终系统预计将具有很强的鲁棒性，并能提供令人满意的可用性。为取得最佳效果，资源利用率预测应基于每个处理节点各自的资源消耗行为，同时考虑整个边缘计算基础设施的总体资源消耗行为。在下一节中，我们将介绍用于预测资源利用率的深度学习模型，以及该模型如何同时利用各个节点和整个边缘基础设施的资源消耗指标。

4 用于资源使用预测的复合深度学习架构

本文提出一种结合 HBES 模型的复合深度学习网络，为 IPFT方法提供准确的资源利用率预测。该复合深度学习网络旨在满足边缘基础设施和资源使用指标的特定要求。由于中央处理器、随机存取存储器、磁盘和带宽等资源指标具有序列依赖性，循环神经网络能够提供合适类型的神经网络层。循环神经网络结合了深度学习的优势与时间序列预测的特点。循环神经网络架构有多种类型，其中最突出的是门控循环单元和长短期记忆网络。我们分别检查每个处理节点，以预测未来可能发生的进程故障。然而，为了触发节点复制，每个节点的深度学习模型都应了解自身状态以及整个边缘基础设施的状态。本文将所检查的边缘节点称为局部，将整个边缘基础设施称为全局。由于局部状态和全局状态会相互影响，我们提出使用一种复合深度学习模型，将二者结合起来，从而提供局部资源利用率预测。这两种不同信息来源的结合方式将在下一小节中说明。

边缘计算环境中的资源使用预测工作流程如图 3 所示。首先，边缘设备生成任务，并将这些任务部分或全部交由边缘计算节点执行。在任务执行过程中，会对节点进行监控，以获取资源利用率指标。随后，将资源利用率指标输入复合深度学习模型，以预测下一个时间范围内的资源利用率。接下来，阈值方法可以利用资源使用量预测来触发节点复制和任务

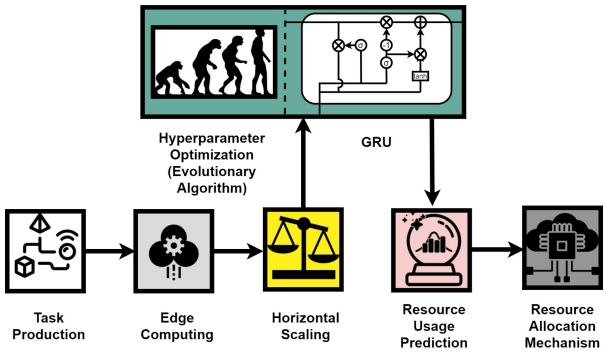

图3：从任务生成到高级容错的流水线。

迁移。接下来的小节将介绍复合神经网络、循环神经网络和 HBES 中与资源使用预测相关的关键部分的理论。

4.1 资源使用预测的双通道架构

神经网络中最常用的架构，是将多个层依次串联起来。输入数据被送入第一层，各层再逐一转换数据，并将其传递给下一层。然而，在构建神经网络时，我们也可以将层，甚至一系列层并行排列，以利用复杂的输入结构属性。网络中的这些并行部分会分别独立处理数据的不同部分，随后将各自的输出拼接起来，送入网络的最后一部分。我们决定采用上述架构，使用两组独立并行的层序列（两个通道） [38]。这样能够更好地处理可用的局部和全局数据指标。

使用多通道神经网络架构的主要优势，在于能够在模型的输入阶段使用不同类型的层。例如，文献中的许多应用会在输入阶段使用卷积网络和前馈网络，以适应同时包含数值和图像属性的数据。多通道神经网络架构的优势在于，可以将属性相关性较高的数据归为一组 [39]。这是通过让不同的层序列处理不同类型的数据来实现的。
与普通的串行神经网络相比，多通道神经网络的结构更复杂，因此训练和推理可能更加耗时、耗费资源。由于训练过程从头到尾一次完成，未提供更多过程细节，我们也无法恰当地评估各个通道，或了解它们对模型最终输出的单独影响。

循环神经网络是一类在处理序列数据问题时表现尤为出色的神经网络。它们用于解决文本预测或语音识别等问题，在处理基于时间序列的问题时也被认为十分有效。因此，我们选择在模型中将循环神经网络作为第一个输入通道，因为它将处理单个节点（即模型当前正尝试进行预测的节点）的序列数据。输入是按一分钟间隔记录的监控测量数据。该通道由一层或两层RNN层和其后的一个前馈层组成，并将结果发送到网络的输出部分。

对于模型的第二个通道，我们选择实现一个前馈网络。该部分接收全局监控状态，以及一个经过转换的时间戳特征，该特征表示星期几和一天中的时段。显然，某个节点在不久的将来过载的可能性，与协同处理请求的其他节点的状态密切相关。前馈网络通道包含全连接层，层与层之间还设有丢弃层。这一架构决策起到了正则化的作用，可防止模型在训练期间过拟合。

各通道处理完模型的输入部分后，会将其拼接结果传递给网络的输出部分。这样组合的目的是兼顾处理后的数据，并在模型接收状态/最后一个输入之后的足够长时间范围内，预测特定节点的资源利用率。该网络同样是一个前馈神经网络，包含若干全连接层和丢弃层，直至输出层。两个通道的合并方式以及输入部分的拼接规模，则由超参数选择算法决定。

为了在数据集上训练，模型会根据涉及目标局部节点和整体全局状态的复杂输入，尝试预测特定节点的未来状态。模型输出预测结果后，会利用损失函数将其与实际输出值进行比较。平均绝对误差或均方根误差等损失函数，可用于衡量回归模型预测正确值的效果好坏。模型根据计算出的误差，使用梯度下降的一种变体调整权重和参数，并通过网络反向传播，影响所有层。名为优化器的函数描述了权重调整／网络优化的过程。神经网络的架构和训练方式都为我们提供了许多可调整的选项和超参数。为了确保快速、合理地为模型选择超参数，我们实现了 HBES算法，以高效地完成这些选择。

4.2 用于时间序列数据的循环神经网络

人工神经网络可定义为函数逼近器，将低层级的数据表示映射为更高层级且相互解耦的数据表示。循环神经网络 [9] 是一类人工神经网络，能够实现动态时间行为、捕捉数据序列并保留先前的输入状态。循环神经网络的架构范式基于多个类神经元节点，这些节点按顺序排列成层，每个节点都与下一层中的节点相连，同时还具有循环连接。借助这一概念，先前数据输入中的信息可以影响未来输出，因此，循环神经网络在考虑上下文信息的同时，也成为时间序列建模的理想选择。通过监控边缘计算基础设施，我们收集序列数据，并根据当前值和先前值，利用循环神经网络预测未来的资源使用指标。

循环神经网络面临的主要问题是梯度消失问题。该问题出现在循环神经网络的训练阶段，此时梯度会变得极小，导致循环神经网络无法进行权重更新。为解决梯度消失问题，人们提出了多种门控架构。借助门，网络能够妥善保留相关信息，并成功将其传递到后续时间步。最值得注意的两种架构是长短期记忆网络和门控循环单元 [10] 网络。

4.3 长短期记忆

LSTM网络架构旨在解决循环神经网络中的梯度消失问题。要理解采用这种复杂架构的重要性，可以想想上下文信息在寻找解决方案时能发挥多大作用。在处理时间序列数据问题（例如预测资源使用量）时，如果查看机器资源使用量的历史数据，而不只是了解其当前状态，就能获得更多有用信息。这样，我们便能更好地理解趋势等概念，因为趋势只有结合时间变化才能得到解释。和普通循环神经网络一样，长短期记忆网络利用隐藏状态连接后续节点，从而更好地理解时间数据。不过，它们还使用细胞状态，这是节点之间的另一种连接。每个LSTM单元都可以通过门读取、写入或重置细胞状态。

总共有三个门，每个门都由Sigmoid函数激活。由于Sigmoid函数在0到1的范围内提供平滑曲线，因此模型能够保持可微分。每个门都以实际输入以及前一个时间步的隐藏状态作为输入。

除了各个门之外，还有一个名为 C 的向量，负责携带可添加到细胞状态中的候选信息。C 使用 tanh层，负责限制梯度消失现象。通过这种方式，细胞信息可以保存更长时间而不会消失。其实现方式是将梯度保持在以零为中心、介于 -1 和 1 之间。

输入门负责处理传入数据，并控制是否应更新记忆单元。它应用于 C，随后将结果添加到细胞状态中。门的Sigmoid激活用于减弱或增强新信息对细胞状态应产生的影响。

遗忘门负责选择被认为不太重要的信息，并通过软重置其值，将这些信息从细胞状态中移除。此外，通过利用Sigmoid函数，它会为细胞状态中保存的每个值生成缩放后的输出。

最后，输出门是生成新的隐藏状态之前的最后一层。它先让细胞状态经过tanh层，再使用Sigmoid函数对其进行筛选。此过程完成后，隐藏状态和细胞状态共同构成LSTM单元的输出，并将其输入下一个时间步。

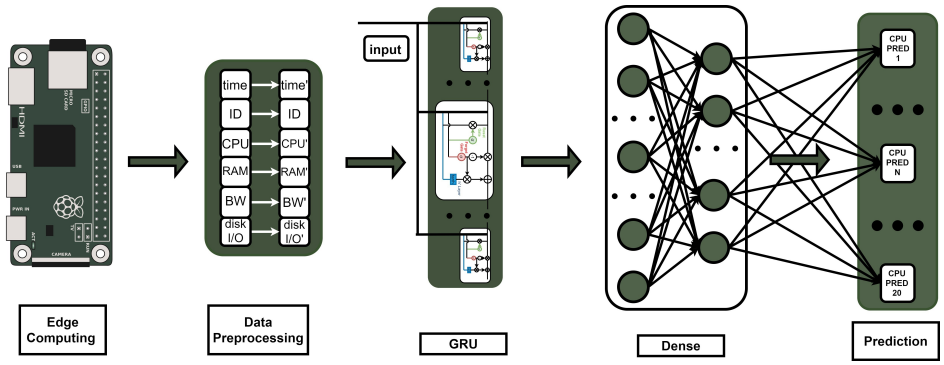

图 4：用于资源使用预测的集成式循环神经网络。

4.4 门控循环单元

门控循环单元和长短期记忆网络相似，因为二者都通过利用门结构来避免梯度消失问题。二者的区别在于，门控循环单元将遗忘门和输入门结合成一个更新门。通过减少所涉及的门的数量，门控循环单元能够提供更简单的结构，因此与长短期记忆网络相比，计算效率更高。同时，门控循环单元的表现也同样出色。

门控循环单元网络还支持隐藏状态机制，该机制将网络中的一个单元与下一个单元连接起来，从而以类似的方式呈现动态时间行为。每个 GRU 单元代表一个特定的时间步，有助于在整个时间序列中传递重要信息。此外，它还包含两种不同的门结构。第一种称为重置门，第二种称为更新门。两者都包含 Sigmoid 层，可在 0 到 1 的范围内生成平滑曲线，从而确保模型保持可微分。通过将数值压缩到 0 到 1 之间，Sigmoid 激活还有助于网络学习哪些数据重要、哪些不重要，并据此决定保留或遗忘这些数据。

为了将 GRU 范式置于边缘计算的背景下进行讨论，我们通过数据预处理输入向量化表示，其中包含相应时间戳、中央处理器、随机存取存储器、带宽和磁盘的资源利用率等信息，如图 4 所示。门控循环单元网络的功能通过以下步骤实现。如前所述，每个 GRU 都使用一个重置门和一个更新门。每个门都有两个权重矩阵，第一个对应输入，第二个对应隐藏状态。GRU 的重置门负责决定应遗忘多少过去的信息。与长短期记忆网络的情况类似，第一步是将输入和隐藏状态分别乘以对应的权重。然后，将乘积之和传递给一个 Sigmoid 层。

更新门负责确定在之前各个时间步中收集的信息有多少需要传递出去，以供未来使用。在这方面，它的行为与重置门非常相似。第一步需要将输入和隐藏状态分别与其对应的权重相乘。隐藏状态包含从之前的 $t-1$ units 中获取的信息。然后，将相乘的结果相加，并通过 sigmoid 层。更新门的输出将称为 u。

下一步是创建候选新隐藏状态。与重置门和更新门类似，这里同样涉及两个权重矩阵。第一个对应输入，第二个对应隐藏状态。创建候选新隐藏状态的第一步是将输入与其对应的权重相乘。第二步是以逐元素方式计算隐藏状态与重置门输出之间的哈达玛积。此过程对于确定在之前各个时间步中收集的信息有多少将被移除至关重要。然后，将哈达玛积与隐藏状态的权重相乘。接着，将两次相乘的结果相加。该总和会通过 tanh层，从而减小梯度消失现象的影响。这是通过在以零为中心的范围内以适当方式分配梯度来实现的。因此，它能够使信息在更长时间内流动而不会消失。到目前为止各项操作的结果就是候选新隐藏状态，将称为 h'。

为了获得更新后的隐藏状态，第一步是将更新门的输出与隐藏状态进行逐元素乘法。第二步是将 h' 与 $1-u$ 的乘积进行逐元素乘法。更新后的隐藏状态是这两个乘积之和。随后，更新后的隐藏状态会传递给下一个GRU单元，对应于下一个时间步。

4.5 进化策略

进化策略 [40] 属于进化算法这一类。进化算法是受生物进化原理启发的基于种群的元启发式优化方法。进化策略的形式基于对候选解种群反复进行变异和选择。候选解也称为个体，它们在n维空间中的随机位置初始化，并向使目标函数最小化的位置移动。这些维度是需要优化的门控循环单元循环神经网络数值超参数。

为满足门控循环单元循环神经网络的超参数调优需求，我们将其数值超参数作为进化策略的搜索空间，并将候选循环神经网络的均方误差作为适应度函数。在每次迭代中，会训练并评估多个循环神经网络的均方误差，然后将其中精度最高的网络变异到下一次迭代。变异是一个基于正态分布的随机过程，会为每次迭代中适应度最高的个体引入变化。起初，对不同候选解的探索十分充分，会通过较强的变异探索搜索空间中的新区域。随着迭代进行，利用自适应控制变量，探索逐渐减少，而对适应度最高个体的利用逐渐增强。这意味着在最初几次迭代中，变异会引入较大的变化；随着进化过程推进，变化幅度逐渐减小，以便收敛到接近最优的循环神经网络架构。

与遗传算法等其他进化算法相比，使用进化策略对深度学习模型进行超参数调优的优势在于，无须重新组合表型可能存在显著差异的不同神经网络拓扑结构。这是因为遗传算法中的交叉存在一个难题：父代可能具有不同的架构，无法在其子代中统一。一个典型例子是：一方父代为一个包含两层长短期记忆循环神经网络、后接六个全连接层的模型，另一方父代则为一个包含两层门控循环单元循环神经网络、后接四个全连接层的模型。因此，长短期记忆网络和门控循环单元的表型无法平滑地重组。另一方面，进化策略仅基于选择和变异，从而平稳地推动进化过程。具体而言，变异操作会为存活的候选解引入变化，使其有机会测试邻近解，而这些解可能带来更高的适应度值。

4.6 贝叶斯优化

贝叶斯优化 [41] 广泛用于估计机器学习和深度学习模型中的超参数。对于类别维度空间中的搜索过程而言，它显然是合适的选择，旨在找到 RNN 接近最优的名义超参数。贝叶斯优化通过采集函数迭代地请求搜索空间中的新观测，并使用代理函数估计目标函数。随着贝叶斯优化观测数量的增加，找到全局最优解所在位置的概率也会提高。尽管如此，我们应考虑到观测数量有限且计算成本高昂，因此，智能搜索过程应在探索与利用权衡的基础上选择点，以最大限度地提高找到新的最优解的概率。

代理函数对目标函数进行近似，并在目标函数针对新的候选点进行评估时更新。采集函数决定在贝叶斯优化的迭代过程中下一步在哪里采样，并找出能使期望改进最大化的点。期望改进由两个部分构成：第一部分估计代理函数具有最优点的区域，第二部分估计预测不确定性较高且尚未得到充分探索的区域。

4.7 结合贝叶斯优化的混合进化策略

复合神经网络的超参数优化是一项重要挑战，因为它涉及构建接近最优拓扑结构的重要架构决策。HBES 将进化策略与贝叶斯优化方法相结合，形成了一种创新、全面且统一的超参数调优方法。进化策略负责根据数值超参数对候选深度学习模型种群进行进化；而每个候选个体解则通过贝叶斯优化来估计其名义超参数，如算法1所述。

数值超参数包括循环层和前馈层的数量、每层的神经元数量、回溯窗口、训练轮次、批量大小、丢弃率百分比以及学习率。名义超参数包括神经网络层的类型、激活函数和优化器。名义超参数方面获得的知识在整个种群中通用，并由所有个体在每一代中更新。HBES 的最终目标是通过贝叶斯进化过程收敛到接近最优的解，并训练能够及时、准确地预测未来时间步资源利用率的深度学习模型。

算法1混合贝叶斯与进化策略

步骤1： 进化策略的初始化
设置算法的起始搜索点。
通常为 a_1=[0.5、0.5、.....、0.5]，因为我们已经
对超参数选项进行了缩放
向下至 [0,1]
步骤 2： 对于 i = 1, 2, ..., n_{pop}:
i) 向搜索点添加一些随机噪声
ii) 从 [0,1] 缩放回超参数
搜索空间，以生成序数型
网络的序数型超参数值
已训练
iii) 使用高斯过程进行贝叶斯优化
1) 对 f应用高斯过程先验分布
2) 根据初始实验设计，在 n_0 个点观察 f
初始实验设计
3) 初始化 $n = n_0$
4) 当 $n \leq N$时重复
a) 更新后验概率
在 f 上的分布，使用所有可用的
data
b) 令 x_n 为
关于 x 的采集函数的最大化点。
c) 观察 $y_n = f(x_n,\ \ x_i(t+1)\ v_i(t+1))$
d) $n \leftarrow (n+1)$
步骤 3： 对结果及其对应的
超参数
步骤 4： 通过以下方式计算新的搜索点
对 top_n 个网络的点取平均值
步骤 5： 返回步骤 2，直到达到所需数量的
已完成迭代

表1：资源使用指标单输出与多输出预测方法的比较。

方法	均方根误差 误差	平均绝对	CPU-1 (%)		RAM-1 (%)		推理时间	
			RMSE	MAE	RMSE	MAE	单批次	
HBES-GRU	0.0641	0.0276	15.918	12.815	1.694	0.580	0.033	0.038
GA-LSTM	0.0674	0.0338	16.099	12.838	1.746	0.917	0.020	0.024
Keras-Tuner	0.0785	0.0377	16.291	13.290	2.631	0.818	0.042	0.042
AUCROP	0.0814	0.0414	17.235	14.009	2.480	1.482	0.004	0.011
XGBoost	0.1139	0.0599	16.457	13.569	1.515	0.472	0.060	0.010
Auto-sklearn	0.1055	0.0243	52.659	17.856	1.546	0.526	0.263	0.572

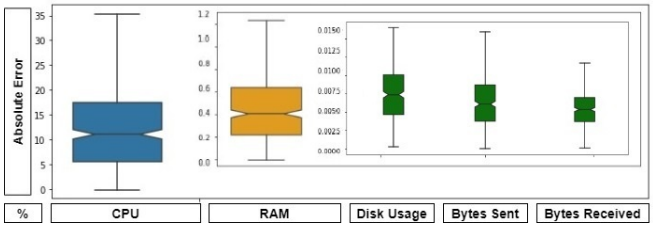

图 5：采用 HBES 的门控循环单元循环神经网络在资源使用指标上的预测误差。

5 实验评估

为评估 IPFT 方法，我们进行了两类实验。首先，使用真实数据集，通过实验评估并比较循环神经网络结合 HBES 与最先进方法的适用性。该数据集通过监控边缘计算基础设施中的树莓派构建。接下来，我们利用资源利用率预测模型开发 IPFT 机制，以便在进程故障发生前留出充足时间，做出智能复制和迁移决策。在智能复制和迁移的情境下，充足时间包括处理节点的服务部署时间，以及在该节点上调度新任务所需的时间。IPFT 的实验评估是在 CloudSim Plus 框架下进行的为期七天的边缘计算仿真中完成的 [42]

5.1 资源使用预测的实验评估

我们使用的边缘基础设施包括作为处理节点的 Raspberry Pi 3，配备 1.4 GHz 的 64 位四核 ARM Cortex-A53，运行 Raspbian 操作系统（Debian Linux 的一个版本）。该数据集由使用 Python 3 编写的监控工具构建，该工具使用了 psutil [43]和 GPUtil [44] 库。我们以一秒为间隔监控中央处理器、随机存取存储器、磁盘和带宽的实时使用情况。

部署的应用是一种自然语言处理文本分类应用。该应用场景是在边缘计算环境中进行文本分类，在本地、靠近文本所有者的地方处理，而不使用云计算基础设施，以保护隐私。之所以选择这种方式，是因为文本所有者不同意将其文本传输到远程服务器进行处理。为了远程控制应用并获取资源使用量数据集，我们使用了 SSH 协议，但没有权限访问处理后的文本。

5.1.1 模型实现与框架比较

HBES 模型和循环神经网络多输出回归模型使用 Python 3 实现，采用的框架包括 NumPy、pandas、statistics、Scikit-learn、SciPy、Scikit-Optimize、TensorFlow 2 和 Keras。我们使用 Google Colaboratory 的 Jupyter 笔记本进行模型训练和评估。实验源代码已发布在我们的 GitHub 仓库 [45] 中，可供各种形式的复现和重新检验。在此实验设置中，我们使用了以门控循环单元（GRU）为循环神经网络（RNN）的 HBES，并将其结果与时间序列基线方法、用于资源使用预测的机器学习元模型 AUCROP、Auto-sklearn、XGBoost、我们此前提出的结合遗传算法的长短期记忆网络模型（GA-LSTM） [11]以及 Keras-Tuner 进行了比较。

5.1.2 评估结果与讨论

初步的时间序列分析结果表明，滞后值在1至22范围内时呈正相关。这一发现证实了资源使用指标序列具有很强的自相似性。随后，我们使用 ARIMA 预测模型评估了资源指标预测结果。例如，中央处理器的均方根误差为18.474。比较统计模型与机器学习和深度学习方法的结果后，我们发现后两者在大多数情况下的均方根误差降低幅度超过20%。因此，我们决定将研究重点放在机器学习和深度学习模型上。

表1总结了实验结果。前两列分别给出了所有设备测试值和资源指标的汇总均方根误差与平均绝对误差。均方根误差会对误差较大的预测施加额外惩罚；从该列可以看出，HBES-GRU 的表现最佳。在标题为平均绝对误差的列中，可以看到表现最好的两个模型是 auto-sklearn 和 HBES-GRU。它们的预测误差非常接近，且表现显著优于其他模型。

CPU-1 和 RAM-1 列分别表示基础设施中预测准确度最低的处理边缘节点的均方根误差和平均绝对误差。此外，图5展示了误差值指标的第25和第75百分位数、中位数、最小值和

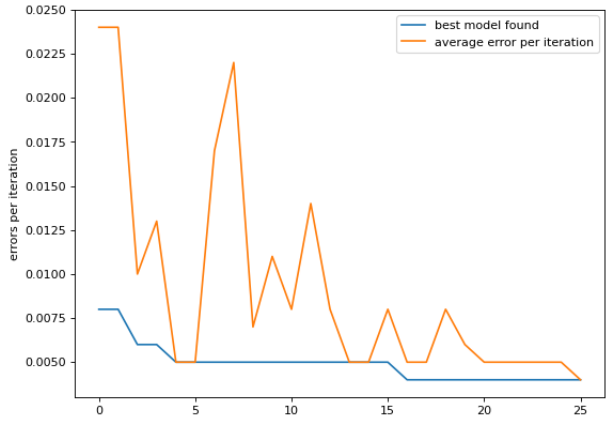

图6：HBES 对于接近最优的循环神经网络的收敛情况。

最大值。这些指标包括中央处理器、随机存取存储器、磁盘使用量，以及以发送和接收的字节数表示的带宽。磁盘使用量和带宽的预测误差并不显著。这不仅是因为 HBES-GRU 能够提供准确的预测，也因为这两项资源指标的波动较小。中央处理器的波动远大于随机存取存储器，而与其他模型相比，HBES-GRU 能更好地捕捉各种变化。在随机存取存储器方面，XGBoost 的表现优于 HBES-GRU。这可能归因于 XGBoost 所采用的集成结构。XGBoost 能针对随机存取存储器的残差构建特定的决策树，并以其缓慢变化的特性为目标。

最后，我们来看模型的推理时间，这些模型需要在提供单次预测或包含一百次预测的批次时使用。所有时间测量的单位均为秒。除 Auto-sklearn 外，所有推理时间均在 11 至 60 毫秒之间。这些推理时间表明，资源使用预测是一个相当快速的过程，可应用于时间敏感型应用。在本研究中，我们没有比较训练时间，因为我们希望在假设空间中进行全面的智能搜索，并了解不同模型所能达到的准确度极限。值得注意的是，我们使用了广泛的时间范围进行了实验。不过，表1中的测量结果是采用 10 分钟的时间范围得出的。我们选择展示这一时间范围，是因为它接近节点部署所需的实际时间。从结果可以看出，尽管门控循环单元的结构比长短期记忆网络更简单，且缺少专门的输出门，但其性能略胜一筹。此外，所开展的实验使我们得出了以下结论：

- 在大多数指标上，深度学习模型（HBES-GRU、GA-LSTM、Keras-Tuner）的表现优于机器学习模型（AUCROP、XGBoost、Auto-sklearn）。
- 与简单的贝叶斯优化（Keras-Tuner）相比，用于超参数调优的进化算法（HBES-GRU 和 GA-LSTM）表现更好。
- 与简单遗传算法相比，采用贝叶斯与进化策略相结合的方法能显著提升性能。

5.1.3 混合贝叶斯进化策略的收敛性

收敛性和全局最优解的位置是进化算法领域中两个最重要的议题。收敛意味着随着种群不断进化，个体会逐渐接近最优解，彼此间的差异也随之缩小。然而，我们无法确定基因型空间中的收敛点究竟构成全局最小值还是局部最小值。因此，在进化过程初期，HBES算法会表现出较大的变异方差，并在迭代过程中逐渐衰减。同时，我们会保留所有迭代中找到的最佳基因型。

HBES 的收敛情况如图6所示。我们观察到，初期每次迭代的平均种群误差波动较大。有些迭代会陷入局部极小值，例如第六次至第十一次迭代期间；另一些迭代则处于平台区域，例如第二十次至第二十五次迭代期间。不过，通过变异，个体最终会摆脱平台区域和局部极小值，并朝着接近最优的区域移动。

基因型空间中这些接近最优的区域会解码为表型空间中接近最优的门控循环单元循环神经网络架构。这些门控循环单元循环神经网络架构能够对边缘计算基础设施中的中央处理器、随机存取存储器、磁盘和带宽使用情况做出最准确的资源使用量预测。

5.2 主动容错实验评估

HBES 与循环神经网络在资源使用预测方面取得的令人鼓舞的实验结果，促使我们继续开展实验，以评估其作为主动容错机制的适用性。具体而言，我们使用 HBES 在候选双通道深度学习模型中进行智能搜索，并应用了第四节所述的阈值方法。

5.2.1 实验仿真

我们将复合深度学习模型集成到 CloudSim Plus 的边缘仿真中。我们仿真了一个边缘基础设施，其中包含一组节点：默认情况下有 5 个可用节点，另有 15 个节点可在需要时激活以进行智能复制。我们仿真了局部和全局资源监控过程，测量中央处理器、随机存取存储器和带宽的数值，并每隔 60 秒（时间步）保存一次这些数值。基础设施的任务卸载器接收传入流量，并依据以下调度算法将每项任务分配到一个节点：轮询调度、MinMin 和 MaxMin。
系统会监控局部和全局资源使用指标，并将其输入每个处理节点的 IPFT机制。在每个时间步，我们都会利用监控数据构建适当的数据表示，其中包含单个节点过去的时间序列测量值，以及整个基础设施的状态。随后将输入送入复合深度学习模型，使其能够预测未来 10 分钟时间范围内每个节点的资源使用量。在本实验设置中，我们假设基础设施需要 10 分钟的准备时间，以确保其可用性以及对故障的鲁棒性。

模拟持续了七天，任务由一组高斯概率分布混合生成，以模拟真实的应用工作负载行为 [46]。处理节点模拟了树莓派的处理能力。如果任务的执行时间超过一秒，我们便将其定义为进程故障。对于多种数据分析应用而言，一秒的延迟是可以接受的 [47]。在尝试不同的进程故障延迟时间后，我们发现 IPFT 的表现优于反应式方法。在被动容错方法中，发生故障时会触发节点复制。表 II 中，我们将在下一节详细讨论，将 IPFT 机制与被动容错方法进行比较。

5.2.2 容错评估指标

为了评估 IPFT 机制的性能，我们采用了一组容错评估指标 [48]。平均故障间隔时间（MTTF）定义为系统正常运行时发生故障所需时间的期望值。MTTF 是一项评估指标，反映边缘基础设施无法正常运行的整体情况，因此计算时会考虑故障数量，而不论实际发生故障的是哪个处理节点。平均修复时间（MTTR）定义为故障发生后修复系统所需时间的期望值。MTTF 的值越高越好，而 MTTR 的值越低越好。这些评估指标以秒为单位计算。

另外两项容错评估指标是可靠性和可维护性。可靠性指边缘基础设施能够持续运行且不发生故障的能力。可维护性指修复发生故障的系统有多容易。可靠性和可维护性都是无单位的数值，数值越高表示性能越好。

5.2.3 评估结果与讨论

实验结果汇总于表 II。我们将 IPFT 机制与反应式容错（RFT）方法进行了比较。RFT 方法会在发生故障后执行节点复制。关于我们所采用的任务卸载算法，我们使用了轮询（RR） [49], 、最小值 MinMin 和 MaxMin [50]。实验结果表明，在所有评估指标上，IPFT 均优于 RFT。此外，我们还发现，任务卸载机制会显著影响结果。这是因为任务卸载算法还结合了采用不同标准的工作负载均衡方法，具体内容将在下文中讨论。

在所有实验设置中，生成的任务数量接近 1,500,000。一天中的某些时段任务生成量很大（即 11:00 - 13:00），而另一些时段的任务数量则较少（即 02:00 - 04:00）。我们模拟了这种任务生成行为，因为它接近日常许多用户应用程序的活动情况。通过这种方式，我们观察到，基础设施会在一天中

表 2：反应式与智能主动容错方法的评估

	MTTF	MTTR	可靠性	可维护性
反应式容错可靠性比率	2.864	19.657	0.741	0.048
智能主动容错可靠性比率	9.506	3.343	0.904	0.230
反应式容错最小最小	8.733	36.169	0.897	0.026
智能主动容错最小最小	8.919	5.656	0.899	0.150
反应式容错最大最小	3.721	24.239	0.788	0.039
智能主动容错最大最小	13.309	7.425	0.930	0.118

任务生成量增加的时段进行智能复制。相应地，当 IPFT 机制预测处理节点的利用率不足时，基础设施便会关闭边缘节点。任务生成量和资源利用率有时会突然飙升或下降，但采用 IPFT 机制的基础设施能够动态、及时地作出调整。

在表 II 中，我们可以看到，在 RR、MaxMin 和 MinMin 中，与 RFT 相比，IPFT 的平均故障间隔时间有所增加。这意味着，借助资源使用量预测，故障发生得更加稀疏且少见。从平均修复时间指标可以看出，发生故障时，基础设施能够非常迅速地恢复，并将新任务调度到资源利用率较低的处理节点上。可靠性指标表明，通过使用 IPFT，边缘基础设施能够在仿真时长的 93% 以内提供预期结果，即使是在模拟天数中资源压力较大的时段也是如此。可维护性指标的显著改善表明，即使发生故障，IPFT 也能提高边缘基础设施的鲁棒性。换言之，IPFT 会及时采取正确措施，触发节点复制和任务迁移，从而降低后续故障发生的可能性。

故障记录会考虑当前处于活动状态的所有边缘节点。这意味着，IPFT MaxMin 的 13.309 秒平均故障间隔时间值包含了不同边缘节点发生的故障。此外，某些生成的任务包含大量百万条指令，由于处理节点的中央处理器不可用，这些任务可能会引发故障。在这种情况下，我们希望了解这些任务如何影响平均故障间隔时间和平均修复时间。通过分析结果，我们发现，任务大小的方差是三种不同任务卸载机制性能各异的原因。具体而言，MaxMin 算法会优先处理较大的任务，因此其平均故障间隔时间指标显著更优。

在模拟过程中，我们考察了 IPFT 的决策以及边缘环境的运行方式。模拟证实，基础设施会利用及时作出的决策触发主动操作，例如在任务数量压垮处理节点之前，智能地复制节点并迁移任务。这对基础设施提供商尤为重要，因为在不再需要节点时将其关闭，可以节省成本和能源。此外，提供商还能确保按时完成的任务平稳流转，避免系统崩溃并最大限度地减少服务质量恶化。

关于实施所提出的 IPFT 范式的实际成本，与反应式方法相比，计算资源消耗高出 3.2%。此外，由于预测模型需要访问持续产生的资源使用量数据，因此产生的数据流给网络基础设施带来了约每秒 10 字节的负担。最后，加入预测机制使用了约 125MB 的随机存取存储器，并使 Intel Xeon E312xx CPU 的中央处理器使用量在平均 2250ms 内增加约 47%。考虑到 IPFT 方法带来的显著效益，我们认为总体实施成本是合理且值得的。

6 结论

本文提出了一种基于资源使用量预测、适用于边缘计算基础设施的主动容错机制。首先，我们讨论并通过实验评估了循环神经网络在资源使用建模中的应用。我们开发了一种复合深度学习模型，通过两个通道利用本地处理节点的资源使用指标以及整个基础设施的资源使用指标。我们还设计了一种超参数调优算法，将进化策略与贝叶斯优化相结合，其性能优于常用的超参数调优工具（如 Keras-Tuner）以及其他最先进的机器学习模型。最后，我们介绍了主动容错机制如何利用资源使用量预测来触发节点复制和任务迁移。

我们的实验和结果证实了主动容错方法的效率，以及循环神经网络和双通道架构在资源使用预测方面的适用性。我们工作的局限性在于，我们

尚未全面研究网络故障、物理故障和服务到期故障等其他类型的故障。未来，我们将开展数据分析，找出边缘资源中与这些类型故障相关的执行模式。此外，我们还希望了解不同阈值对 IPFT 性能的影响。具体而言，我们计划进一步研究用于触发节点复制和任务迁移的资源使用指标阈值，以及监测时间序列指标的时间间隔。

致谢

本研究是 ACCORDION 和 CHARITY 项目的一部分，这些项目获得了欧盟地平线2020研究与创新计划的资助，资助协议编号分别为 871793" 和 101016509"。

参考文献

[1] Antonios Makris、Abderrahmane Boudi、Massimo Coppola、Luís Cordeiro、Massimiliano Corsini、Patrizio Dazzi、Ferran Diego Andilla、Yago González Rozas、Manos Kamarianakis、Maria Pateraki 等。《面向全息术和增强现实的云》。载于*2021*年第*10*届*IEEE*云网络国际会议（*CloudNet*），第118–126页。IEEE，2021年。[2] Theodoros Theodoropoulos、Antonios Makris、Abderrahmane Boudi、Tarik Taleb、Uwe Herzog、Luis Rosa、Luis Cordeiro、Konstantinos Tserpes、Elena Spatafora、Alessandro Romussi 等。《基于云的XR服务：相关挑战与赋能技术综述》。《网络与网络应用杂志》，2(1):1–22，2022年。[3] Kevin Boos、David Chu 和 Eduardo Cuervo。演示：FlashBack：通过渲染记忆化在移动设备上实现沉浸式虚拟现实。载于第*14*届移动系统、应用与服务国际年会伴会论文集，MobiSys '16 伴会，第94页，美国纽约州纽约市，2016年6月。计算机协会。[4] Tarik Taleb、Abderrahmane Boudi、Luis Rosa、Luis Cordeiro、Theodoros Theodoropoulos、Konstantinos Tserpes、Patrizio Dazzi、Antonis Protopsaltis 和 Richard Li。《迈向XR服务支持：架构与赋能技术》。《*IEEE*物联网杂志》，2022年。[5] Claudia Campolo、Gianmarco Lia、Marica Amadeo、Giuseppe Ruggeri、Antonio Iera 和 Antonella Molinaro。《迈向命名式人工智能网络：揭示NDN在边缘人工智能中的潜力》。载于Luigi Alfredo Grieco、Gennaro Boggia、Giuseppe Piro、Yaser Jararweh 和 Claudia Campolo（编），自组织、移动与无线网络，《计算机科学讲义》，第16–22页，尚，2020年。Springer国际出版公司。[6] Aroosa Hameed、John Violos、Nina Santi、Aris Leivadeas 和 Nathalie Mitton。《一种用于物联网环境吞吐量估计的机器学习回归方法》。载于*2021*年*IEEE*物联网国际会议（*iThings*）、*IEEE*绿色计算与通信会议（*GreenCom*）、*IEEE*网络、物理与社会计算会议（*CPSCom*）、*IEEE*智能数据会议（*SmartData*）及*IEEE*网络信息学大会（*Cybermatics*），第29–36页，2021年12月。[7] Antonio Petrosino、Giancarlo Sciddurlo、Giovanni Grieco、Awais Aziz Shah、Giuseppe Piro、Luigi Alfredo Grieco 和 Gennaro Boggia。《具有能耗和带宽约束的T-SDN架构中的转发规则动态管理》。载于Luigi Alfredo Grieco、Gennaro Boggia、Giuseppe Piro、Yaser Jararweh 和 Claudia Campolo（编），自组织、移动与无线网络，《计算机科学讲义》，第3–15页，尚，2020年。Springer国际出版公司。[8] Farhan Nisar 和 Bilal Ahmed。《应用ARIMA方法的数据中心资源利用率》。载于Imran Sarwar Bajwa、Tatjana Sibalija 和 Dayang Norhayati Abang Jawawi（编），《智能技术与应用》，《计算机与信息科学通信》，新加坡，2020年。施普林格。[9] B. Shiva Prakash、K. V. Sanjeev、Ramesh Prakash 和 K. Chandrasekaran。《序列学习循环神经网络架构综述》。载于Jagdish Chand Bansal、Kedar Nath Das、Atulya Nagar、Kusum Deep 和 Akshay Kumar Ojha（编），《用于问题求解的软计算》，《智能系统与计算进展》，第57–66页，新加坡，2019年。施普林格。[10] Guizhu Shen、Qingping Tan、Haoyu Zhang、Ping Zeng 和 Jianjun Xu。《利用门控循环单元网络进行金融序列预测》。《计算机科学程序汇刊》，131:895–903，2018年1月。[11] John Violos、Evangelos Psomakelis、Dimitrios Danopoulos、Stylianos Tsanakas 和 Theodora Varvarigou。《将长短期记忆神经网络用作资源利用率预测器：边缘端深度学习模型训练案例》。载于《网格、云、系统与服务经济学》，《计算机科学讲义》，第67–74页，尚，2020年。Springer国际出版公司。

[12]Tong Yu 和 Hong Zhu。《超参数优化：算法与应用综述》。*arXiv:2003.05689 [cs, stat]*, 2020年3月。arXiv: 2003.05689。[13] Seong Yeol An、Yoon Seok Cha、Eun Jin Jeon、Gwi Yeong Gwon、Byeong Chun Shin 和 Byeong Rae Cha。《开源 Prometheus 监控系统的初步研究》。*Smart Media Journal*，10(2):110–118，2021年。出版方：THE KOREAN INSTITUTE OF SMART MEDIA。[14] Rong Chen、Youyang Yao、Peng Wang、Kaiyuan Zhang、Zhaoguo Wang、Haibing Guan、Binyu Zang 和 Haibo Chen。《面向大规模图处理的基于复制的容错》。*IEEE Transactions onParalleland Distributed Systems*，29(7):1621–1635，2018年7月。会议名称：IEEE Transactions on Parallel and Distributed Systems。[15] 基于微服务的移动云端网络中的瞬态故障感知应用分区计算卸载算法 | SpringerLink。[16] Pekka Karhula、Jan Janak 和 Henning Schulzrinne。《物联网边缘函数的检查点与迁移》。载于第二届边缘系统、分析与网络国际研讨会论文集，EdgeSys '19，第60–65页，美国纽约州纽约市，2019年3月。计算机协会。[17] Yuli Tian、Jeff Tian 和 Ning Li。《通过基于故障风险的主动措施提高云可靠性和效率》。*Journal of Systems and Software*，163:110524，2020年5月。[18] Alexander Power 和 Gerald Kotonya。《物联网系统中用于被动式与主动式容错的微服务架构》。载于*2018*年*IEEE*第*19*届"无线、移动与多媒体网络世界"国际研讨会，第588–599页，2018年6月。[19] Theodoros Theodoropoulos、Antonios Makris、John Violos 和 Konstantinos Tserpes。《用于边缘计算基础设施高级容错的自动化流水线》。载于第二届边缘灵活资源与应用管理研讨会论文集，美国纽约州纽约市，2022年7月。ACM。[20] Firdose Saeik 等。《边缘与云计算中的任务卸载：数学、人工智能和控制理论解决方案综述》。*Computer Networks*，195:108177，2021年。[21] Dimitrios Dechouniotis、Nikolaos Athanasopoulos、Aris Leivadeas、Nathalie Mitton、Raphael Jungers 和 Symeon Papavassiliou。《动态网络的边缘计算资源分配：druid-net愿景与展望》。*Sensors*，20(8)，2020年。[22] Mohammad Masdari 和 Afsane Khoshnevis。《云计算工作负载预测方法的调查与分类》。*Cluster Computing*，23(4):2399–2424，2020年12月。[23] John Violos、Stylianos Tsanakas、Theodoros Theodoropoulos、Aris Leivadeas、Konstantinos Tserpes 和 Theodora Varvarigou。《利用双塔神经网络实现边缘计算中的智能水平自动扩缩》。*ComputerNetworks*，217:109339，2022年。[24] Theodoros Theodoropoulos、Antonios Makris、Ioannis Kontopoulos、John Violos、Przemysław Tarkowski、Zbyszek Ledwo´ń、Patrizio Dazzi 和 Konstantinos Tserpes。《用于表示多变量资源使用量的图神经网络：多人移动游戏案例研究》。*International JournalofInformationManagement Data Insights*，3(1):100158，2023年4月。[25] Theodoros Theodoropoulos、Angelos-Christos Maroudis、John Violos 和 Konstantinos Tserpes。《用于多步服务流量预测的编码器-解码器深度学习方法》。载于*2021*年*IEEE*第七届大数据计算、服务与应用国际会议（*BigDataService*），第33–40页，2021年8月。[26] Soma Prathibha。《研究机器学习算法在云计算中提升大规模工作流应用容错能力方面的性能》。载于*2019*年计算智能与知识经济国际会议（*ICCIKE*），第187–190页，2019年12月。[27] Tianqi Chen 和 Carlos Guestrin。《XGBoost：一种可扩展的树提升系统》。第*22*届*ACM SIGKDD*知识发现与数据挖掘国际会议论文集，第785–794页，2016年8月。arXiv: 1603.02754。[28] Didrik Nielsen。《使用XGBoost进行树提升——为什么XGBoost能赢得"每一场"机器学习竞赛？》2016年。录用日期：2017-03-13T07:58:50Z；出版方：NTNU。[29] John Violos、Evangelos Psomakelis、Konstantinos Tserpes、Fotis Aisopos 和 Theodora Varvarigou。《利用用户移动性和移动应用服务行为实现最佳边缘资源利用率》。载于全层智能系统国际会议论文集，COINS '19，第7–12页，美国纽约州纽约市，2019年5月。计算机协会

puting Machiner y.

[30] Matthias Feurer、Aaron Klein、Katharina Eggensperger、Jost Tobias Springenberg、Manuel Blum 和 Frank Hutter。《Auto-sklearn：高效且稳健的自动化机器学习》。载于《自动化机器学习：方法、系统与挑战》，《Springer机器学习挑战系列》。Springer国际出版公司，尚，2019年。[31] Elie Bursztein。《前沿 TensorFlow——Keras Tuner：面向人类的超参数调优》，2019年5月。会议名称：Google IO。[32] Priti Kumari 和 Parmeet Kaur。云计算中的容错综述。《沙特国王大学学报：计算机与信息科学》，33(10):1159–1176，2021年12月。[33] Ashkan Yousefpour、Genya Ishigaki 和 Jason P. Jue。雾计算：致力于最小化物联网中的延迟。载于*2017*年*IEEE*边缘计算国际会议（*EDGE*），第17–24页，2017年。[34] Weiwei Lin、James Z. Wang、Chen Liang 和 Deyu Qi。云计算的基于阈值的动态资源分配方案。《工程学程序》，23:695–703，2011年1月。[35] Marios Avgeris、Dimitrios Spatharakis、Dimitrios Dechouniotis、Aris Leivadeas、Vasileios Karyotis 和 Symeon Papavassiliou。Enerdge：边缘侧的分布式能量感知资源分配。*Sensors*，22(2)，2022年。[36] 使用水平 Pod 自动扩缩器自动扩缩 Pod——使用 Pod ｜节点 ｜OpenShift容器平台4.7。[37] Fahd Al-Haidari、Mohammed Sqalli 和 Khaled Salah。CPU利用率阈值和扩缩规模对云资源自动扩缩的影响。第2卷，第256–261页，2013年12月。[38] Jyostna Devi Bodapati、Nagur Shareef Shaik、Veeranjaneyulu Naralasetti 和 Nirupama Bhat Mundukur。用于脑肿瘤分类的双通道深度神经网络联合训练。《信号、图像与视频处理》，15(4):753–760，2021年6月。[39] Hui Wang、Jiang Lu、Lucy Nwosu 和 Ishaq Unwala。利用面部部位进行面部表情识别的双通道卷积神经网络。《国际大数据智能期刊》，6(3-4):259–268，2019年1月。出版商：Inderscience出版社。[40] Nikolaus Hansen、Dirk Arnold 和 Anne Auger。进化策略。2015年1月。[41] Peter I. Frazier。贝叶斯优化教程。*arXiv:1807*。*02811 [cs, math, stat]*, 2018年7月。arXiv：1807.02811。[42] Manoel C. Silva Filho、Raysa L. Oliveira、Claudio C. Monteiro、Pedro R. M. Inácio 和 Mário M. Freire。CloudSim Plus：遵循软件工程原则、旨在提高模块化、可扩展性和正确性的云计算仿真框架。载于*2017*年*IFIP/IEEE*综合网络与服务管理研讨会（*IM*），第400–406页，2017年5月。[43] Giampaolo Rodola。giampaolo/psutil, https://github.com/giampaolo/psutil，2020年6月。[44] Anders Krogh Mortensen。anderskm/gputil，https://github.com/anderskm/gputil，2020年6月。[45] STsanakas。STsanakas/Hybrid-Bayesian-Evolution-Strategy-GRU-RNN，2021年2月。[46] Tran Ngoc Minh、Lex Wolters 和 Dick Epema。并行系统工作负载的现实集成模型。载于*2010*年第*10*届*IEEE/ACM*集群、云与网格计算国际会议，第464–473页，2010年5月。[47] Stephen Cohen 和 William Money。实时大数据分析所需的数据系统故障应对架构试金石。2017年1月。[48] Salma M. A. Ataallah、Salwa M. Nassar 和 Elsayed E. Hemayed。云计算中的容错——综述。载于*2015*年第*11*届国际计算机工程会议（*ICENCO*），第241–245页，2015年12月。[49] Fahd Alhaidari 和 Taghreed Zayed Balharith。云计算环境中的增强型轮询算法：实现最优任务调度。《计算机》，10(5):63，2021年5月。编号：5；出版商：多学科数字出版研究所。[50] Majid Derakhshan 和 Zohreh Bateni。基于 MAX-MIN、MIN-MIN 和优先级的云计算任务优化。载于*2018*年第*4*届网络研究国际会议（*ICWR*），第45–50页，2018年4月。