Computer Science and Application 计算机科学与应用, 2022, 12(8), 2050-2061
Published Online August 2022 in Hans. http://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2022.128208

Hans汉斯

云环境下基于AI知识分析的负载均衡方法

李　莹，刘　忻，邱　洋*，武　扬

广州市品高软件股份有限公司，广东 广州

收稿日期：2022年7月20日；录用日期：2022年8月19日；发布日期：2022年8月24日

摘　要

针对云计算环境下传统负载均衡方法难以对蕴含在历史数据中的流量信息进行知识分析与智能利用问题，提出一种基于AI知识分析的云平台负载智能均衡方法LSTM-TD3 (Long Short Term Memory and Twin Delayed Deep Deterministic policy gradient algorithm)。LSTM-TD3以资源利用率、服务器连接数、响应时间、请求历史数据等为知识参数输入，首先通过LSTM进行任务数预测建模；紧接着以初始化预测数据强化学习模型TD3，形成基于AI分析与评估的优化负载均衡计算模型；最后通过实际训练以及仿真实验对LSTM-TD3的负载均衡效果进行验证测试。实验结果表明，相比传统的无负载均衡、轮询算法、Q-learning和TD3算法等云环境负载均衡方法，LSTM-TD3负载均衡性能分别提高25.4%，6.41%，3.56%和2.85%，能达到更好的资源负载均衡效果，资源利用率更高。

关键词

云计算，AI，负载均衡，知识分析，TD3，强化学习

Load Balancing Based on AI Knowledge Analysis in Cloud Environment

Ying Li, Xin Liu, Yang Qiu*, Yang Wu

Bingo Software Co., Ltd., Guangzhou Guangdong

Received: Jul. 20th, 2022; accepted: Aug. 19th, 2022; published: Aug. 24th, 2022

Abstract

In view of the difficulty of traditional load balancing methods in intelligent utilization of traffic information contained in historical data, a cloud load balancing method LSTM-TD3 (Long Short Term Memory and Twin Delayed Deep Deterministic policy gradient algorithm) based on AI

*通讯作者。

文章引用: 李莹, 刘忻, 邱洋, 武扬. 云环境下基于 AI 知识分析的负载均衡方法[J]. 计算机科学与应用, 2022, 12(8): 2050-2061. DOI: 10.12677/csa.2022.128208

knowledge analysis is proposed. LSTM-TD3 takes resource utilization, number of server connections, response time, request history data, etc. as knowledge parameters input. First, it uses LSTM to predict and model the number of tasks; then, the initialization prediction data is used to strengthen the learning model TD3 to form an optimized load balancing calculation model based on AI analysis and evaluation; finally, the load balancing effect of lSTM-TD3 is verified and tested through actual training and simulation experiments. Experimental results show that compared with traditional cloud environment load balancing methods such as no load balancing, polling algorithm, Q-learning and TD3 algorithm, the performance of LSTM-TD3 load balancing is improved by 25.4%, 6.41%, 3.56% and 2.85% respectively, which can achieve better resource load balancing effect and higher resource utilization.

Keywords

Cloud Computing, AI, Load Balancing, Knowledge Analysis, TD3, Reinforcement Learning

1. 引言

随着分布式技术的发展，当前大数据计算处理服务都在云环境上实现。云计算平台可以将大量数据计算任务通过网络分发到多个计算实例中进行处理，最终将各个实例的运算结果汇总返回给用户。云计算通过调用庞大的运算资源更高效地解决以往单机难以处理的复杂计算任务，因此，平台中负载分布是否均衡是评估云计算平台是否可靠可用可扩展的关键。在云计算环境中，负载均衡通过算法动态地对运算资源进行调度，以最大限度地提高集群总体性能[1]，在有限时间内处理更多请求，提高资源的利用率。在云环境下，任务处理单元通过虚拟机技术或容器技术部署在服务器实例上，处理资源是虚拟资源，部署配置可以根据云环境的实时负载情况进行动态调整。因此，云环境下负载均衡可以简单概括为以下两个目标：

1) 对大量的用户请求进行合理分发，减少用户等待响应时间；

2) 将过载服务器上的任务分割转移到其他节点进行处理，提高资源利用率。

当前云环境下负载均衡算法依据触发方式可以分为主动式、被动式以及对称式[2]。主动式负载均衡根据设定的负载均衡策略定时执行，如轮转调度、加权轮转调度。被动式负载均衡通过对服务器负载性能数据实时采集，判断系统当前均衡性，依据设定的指标阈值进行负载均衡。对称式则是主动式与被动式算法的结合。

通过判断系统当前负载状态是否动态调整，负载均衡也可分为静态负载均衡策略与动态负载均衡策略[3]。静态负载均衡算法包括随机、轮询[4]、加权调度[5]、优先权[6]等，动态负载均衡算法包括最少连接数、最快响应速度、负载最轻综合均衡、观察法、预测法[7]、动态性能分配[8]、动态服务器补充、服务质量[9]、服务类型[10]、规则模式[11]等。常见的负载均衡算法及其彼此之间的对比分析如表 1 所示。

从表 1 可以看出，静态负载均衡算法相对动态负载均衡算法更简单，但是无法将计算资源最大化利用，造成资源浪费。动态负载均衡算法由于需要考虑的参数指标较多，因此更复杂[12]，且需要对服务器各种指标进行实时监控并分析，对资源进行动态调度，因此动态负载均衡也能更高效合理地运用计算资源，节省能源浪费。许多研究人员对云环境下动态负载均衡算法做出改进，以应对不同云计算环境，提

高云平台总体性能，保障云平台的服务质量(QoS)。Pan 等[13]提出一种最优负载均衡算法，在不牺牲系统性能的情况下对虚拟机进行资源调度，并取得了良好的效果。Said 等人[14]提出一种基于 Moth-Flame 优化算法的任务调度方法，用于雾计算中节点最优任务集分配，以满足信息物理系统应用程序服务质量要求。Kansal 等[15]人总结了云计算中负载均衡算法，详细介绍了 Active Clustering、Decentralized content aware 等 17 个算法并进行了对比分析[16]。

随着人工智能技术(AI)的发展，结合 AI 的负载均衡算法研究取得了较大进步。Wang 等[17]人提出一种基于 Q-learning 的移动性管理方法来处理系统信息的不确定性。为减少服务等待时间，该方法通过反复试验从环境中学习最佳移动性管理策略。Smith 等人[18]使用基于先到先服务(FCFS)和保守回填的最少工作优先(LWF)队列来提前预留资源以进行负载均衡。谢海涛等人[19]基于长短时记忆网络(LSTM)对不同时间跨度中的知识进行融合，实施对用户流量进行预测并实现负载均衡，从而降低用户平均请求时间。张思松[20]设计了一种基于深度强化学习算法的高能效数据负载均衡方法，根据存储节点数据量与特征动态实现高能效数据负载均衡。胡华等人[21]将强化学习模型 Q-learning 用在移动社交网络领域中的群智感知任务调度上中。强化学习分步探索做全局优化，微观上 NTQL-ASS 贪婪算法获取局部最优，综合全局和局部优化策略的负载均衡算法提高了感知效率并节省了能源开销。

Table 1. Comparison of common load balancing algorithms
表 1. 常见负载均衡算法对比

分类	算法名	算法简介	优点	缺点
静态	随机法	随机抽取可用服务实例	简单，易扩展	没有考虑机器性能问题
	轮询法	按顺序分发请求	简单，易扩展	没有考虑机器性能问题
	加权轮转调度法	轮询的升级版，根据服务器性能区别设置权重	适合服务器性能不一致的情况	不适用于任务需求规格不一致的情况
	优先权法	对服务器分组，请求优先分配给优先级高的服务器组，当高优先级所有服务故障时，请求分发给此优先级服务器	容灾能力强	此优先级服务器闲置时间长，造成资源浪费
动态	最少连接法	传递新的连接给那些进行最少连接处理的服务器	简单，负载平滑分布	服务器性能不同，算法不理想
	最快响应速度法	将请求分发给响应最快的服务器	简单，动态平衡	需要实时监控，具有延迟
	负载最轻综合均衡法	对服务器负载性能数据实时采集并对轻载服务器周期性判断进行动态负载均衡	动态平衡	复杂，需要实时监控
	观察法	连接数目合响应时间两项的最佳平衡为依据进行服务器选择	相对简单，动态平衡	实时监控，动态迁移
	综合负载基准对比	对服务器负载性能实时采集并对服务器进行动态平衡	动态权重设置，动态平衡	复杂度较高，动态迁移，需要实时监控

总之，当前云环境下负载均衡算法还没有形成统一的规范，某些研究尝试从某个角度切入对负载均衡问题开展研究，但总体来说仍然有所欠缺，主要体现在以下几点。

1) 不同云环境下实例或节点不同，例如系统、配置、网络带宽与任务需求存在很大差异，并且随着时间的变化而变化。

2) 云环境下负载均衡的细粒度实时性要求高。尽管当前算法对资源状态进行了实时评估，但仍无法

对实时变化的请求进行预测，常常需要进行二次调度或多次调度，造成资源浪费。

针对以上问题，结合长短时记忆网络与强化学习技术，提出云环境下基于 AI 知识分析的负载均衡方法 LSTM-TD3 (Long Short Term Memory and Twin Delayed Deep Deterministic policy gradient algorithm)。LSTM-TD3 以资源利用率、服务器连接数、响应时间、请求历史数据等为知识参数输入，首先进行任务数预测建模；以预测数据初始化强化学习模型 TD3，形成 AI 分析与评估优化负载均衡计算模型；通过算法训练以及仿真实验对 LSTM-TD3 的负载均衡效果进行实际验证。实验结果表明，相比传统的云环境负载均衡方法，LSTM-TD3 对比无负载均衡、轮询算法、Q-learning 和 TD3 算法分别提高了 25.4%，6.41%，3.56%和 2.85%，能达到更好的资源负载平衡效果，资源利用率更高。

2. 基于 AI 知识分析的负载均衡方法

2.1. 模型框架

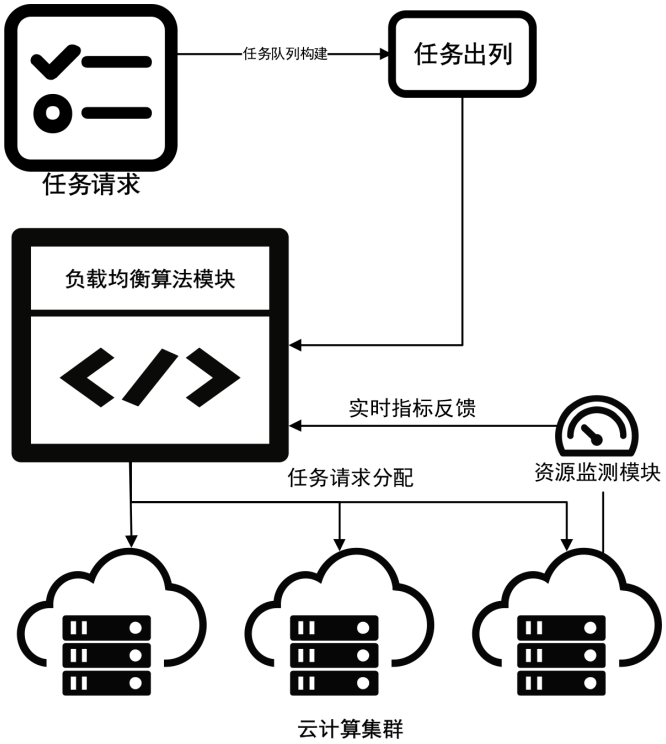

Figure 1. Load balancing framework for cloud computing resource requests
图 1. 云计算任务请求资源负载均衡框架

本文所提出的云计算资源负载均衡框架如图 1 所示，其核心为基于 AI 知识分析的负载均衡器。监控单元实时采集的实例单元信息以及部署在实例上的容器信息，结合历史知识汇总到负载均衡器中进行动态分析，最终根据分析模型计算的结果分配计算资源。当前大多数负载均衡算法都考虑了时间、成本、能耗、连接数等指标，但是较少会结合对历史数据的知识进行融合分析。本文在常用指标上创新性地结合历史任务数进行 AI 知识分析建模，通过学习不同时间段的任务量，预测下一个时间段的任务量，提前做出预测，并及时启用备用实例资源，应对可能的请求高峰，以提高云平台服务质量。

2.2. 问题定义

对云环境下的负载均衡问题进行抽象，其本质实则为多目标优化问题，数学模型定义如下。

1) 任务集合定义为$T=t_1,t_2,\cdots,t_n$，其中t_i代表第i个子任务t，n为任务数；

2) 本文通过容器对服务进行部署，容器集合定义为$C=c_1,c_2,\cdots,c_m$，其中c_j中j代表容器的序号，且$m<n$；

3) 集合x_{ij}定义任务的分配情况，若$x_{ij}=1$，则表示任务t_i被分配到容器c_j进行处理，若$x_{ij}=0$，为其他；

4) 每个容器需要多种不同类型的资源为任务提供服务，以cpu、内存以及网络宽带作为指标建立资源利用率模型R_u。其中，cpu表示CPU利用率，mem表示内存利用率，net表示网络带宽使用率。资源利用率可以比较准确的反应负载变化的趋势，当资源利用率R_u较大时候，当前实例负载较大，反之较小。通过负载均衡指标探测模块可以对相关指标数据实时获取。负载均衡指标计算如公式(1)所示；

$$R_u=\frac{1}{1-cpu}*\frac{1}{1-mem}*\frac{1}{1-net} \tag{1}$$

5) 资源需求评分R_{score}(公式2所示)，资源需求评分B_{score}如公式(3)所示，用来衡量实例的负载均衡程度，实现迁移容器时目标的优选；

$$R_{score}=CPU\left(\frac{capacity-sum(requested)}{capacity}\right)+MEM\left(\frac{capacity-sum(requested)}{capacity}\right) \tag{2}$$

$$B_{score}=\frac{1}{|cpu-mem|} \tag{3}$$

6) 结合资源需求评分R_{score}和资源负载均衡度B_{score}，最终状态价值评价函数定义如公式(4)所示，

$$Est(P_i)=(1-w)R_{score}(P_i)+wB_{score}(P_i) \tag{4}$$

其中，P_i为达到当前状态S所采取的负载均衡策略，$w\in[0,1]$。

2.3. LSTM-TD3 算法模型

随着AI技术的发展和应用，长短时记忆网络被广泛应用于文本分类、机器翻译、语音识别等时序问题。相较于传统回归算法，它引入类似“记忆”的概念，用于处理和“记忆”有关的各种任务。本文引入LSTM [22]对历史任务数据进行时序建模，用于预测当前状态的下一个状态可能出现的任务量。由于预测任务量时，特殊时间节点以及短时间范围内任务数量的正向反馈远高于其他时间段，所以可以较大地提高预测准确度。此外，由于强化学习通过与环境交互获得正向或反向奖励的方式求解最优策略，已被广泛应用到机器人、自动驾驶等领域。强化学习模型如图2所示，通过与环境不断段交互试错的方式来进行学习，实现大规模需求条件下的云平台资源调度。

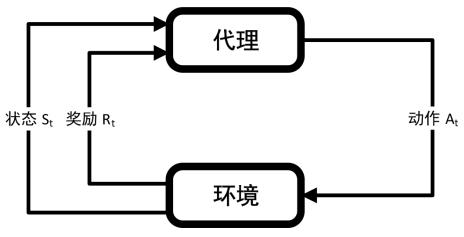

Figure 2. Interaction between agent and environment in reinforcement learning
图2. 强化学习中代理与环境的交互过程

几乎所有的强化学习过程都可以用马尔可夫决策过程来进行描述。本文使用一个五元组

$\{S, A, P, R, y\}$ 来对此过程进行表示。其中，S 代表环境状态的集合，即当前云环境下各个实例，各个容器负载程度集合，用负载均衡指标 R_u 进行描述；A 表示代理所能采取的动作集合，即负载均衡器所进行的调度操作；P 表示状态转移概率，如：$P_{ss'}^{a} = P(S_{t+1} = s' \mid S_t = s, A_t = a)$，即负载均衡器在状态 s 下采取调度操作 a 后变为状态 s' 的概率；R 是奖励函数，即状态 s 下采取操作 a 时能获得的奖励；y 为衰减因子，取值为[0, 1]，用于调整不同训练回合的奖励权重。强化学习中还有一个重要的概念是策略 π，即云环境下不同的负载均衡执行操作的依据。如，负载均衡器在 t 时刻的 S_t 状态下，根据策略 π 执行动作 A_t，接下来环境(各个实例中不同服务的指标 R_u、R_{score}、B_{score} 变化)通过状态转移概率概率 P 和奖励函数 R 得到新的状态 S_{t+1} 和奖励 r_t。状态 s 下的策略选取如公式(5)。

$$\pi^*(a \mid s) = \arg\max_a E(R) \tag{5}$$

通过寻找最优价值函数 V^* 来寻找最优策略，V_π 表示在策略 π 下的价值函数。相关计算如公式(6)所示。

$$\begin{cases} V^*(a \mid s) = \max_\pi V_\pi(s) \\ V_\pi(s) = E_\pi\left(r_{t+1} + yV_\pi(S_{t+1}) \mid S_t = s\right) \end{cases} \tag{6}$$

TD3 即 Twin Delayed Deep Deterministic policy gradient algorithm [23]的简称，双延迟深度确定性策略梯度。TD3 相对 DDPG (Deep Deterministic Policy Gradient)具有以下优势：1) 用类似双 Q 网络的方式，解决了 DDPG 中 Critic 对高估动作 Q 值的问题；2) 延迟 actor 更新，让 actor 的训练更加稳定；3) 在 target_actor 中加上噪音，增加算法稳定性。由于采取了经验回放(Experience Replay)机制，TD3 能将训练过程中学习到的经验数据存储在经验池中，并且通过随机抽样的方式更新网络参数，不仅加速模型的学习，还提高了强化学习网络的稳定性[23]。

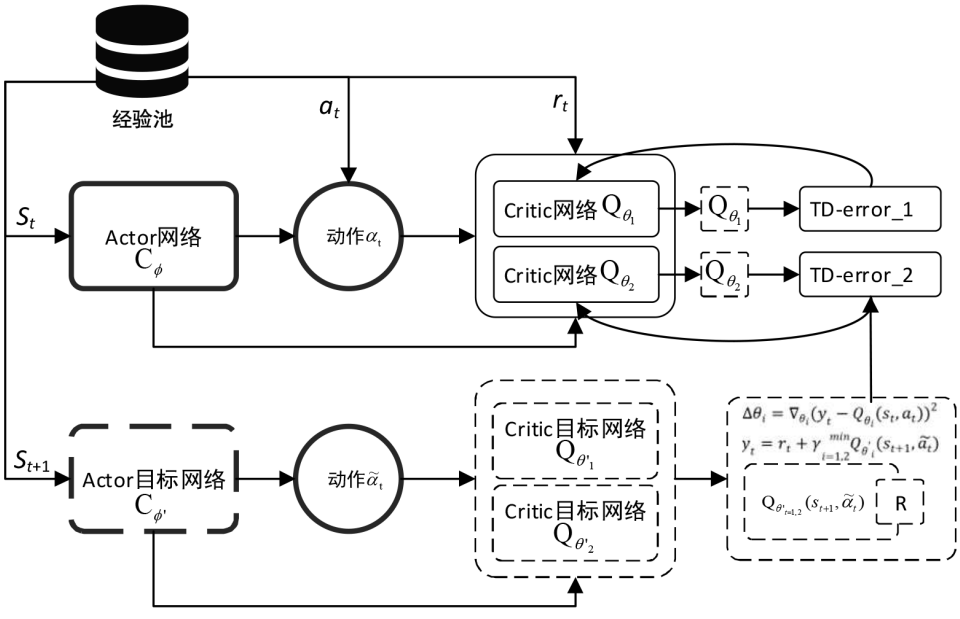

Figure 3. TD3 algorithm structure framework
图 3. TD3 算法结构框架

TD3 算法结构如图 3 所示，Actor 与 Critic 分别称为演员网络与评论家网络，Actor 根据策略，结合当前状态 s_t，输出动作 a_t，Actor 目标网络用作对未来可能的动作 s_{t+1} 进行模拟，输出目标动作 $\tilde{a}_t$。Critic

网络与 Critic 目标网络则分别对 Actor 网络和 Actor 目标网络的动作进行评判，计算出 $Q_{\theta_i}(s_t,a_t)$，以及目标动作 $\tilde{a}_t$ 下目标 Q 值 $Q_{\theta_i'}(s_t,\tilde{a}_t)$。图中 ϕ、θ_i、ϕ' 和 $\theta_i'(i=1,2)$ 分别代表 Actor 网络、Critic 网络、Actor 目标网络和 Critic 目标网络的参数。Critic 网络每一回合都会进行更新，并采取最小 Q 值策略，即选取 Critic 目标网络中最小的 Q 值作为目标值 γ_t，即公式(7)所示。

$$\begin{cases}\Delta\theta_i=\nabla_{\theta_i}\left(y_t-Q_{\theta_i}(s_t,a_t)\right)^2\\ y_t=r_t+\gamma\min\limits_{i=1,2}Q_{\theta_i'}(s_{t+1},\tilde{a}_t)\\ \tilde{a}_t=C_{\phi'}(s_{t+1})+\varepsilon'\end{cases}\tag{7}$$

其中，ε' 为随机噪声，服从截断正态分布 $clip(N(0,\sigma),-c,c),c>0$。Actor 网络和四个目标网络参数根据公式(8)进行更新。

$$\begin{cases}\Delta\phi=\nabla_{\phi}Q_{\theta_1}(s_t,C_{\phi}(s_t))\\ \theta_i'\leftarrow\tau\theta_i+(1-\tau)\theta_i'\\ \phi'\leftarrow\tau\phi+(1-\tau)\phi'\end{cases}\tag{8}$$

3. 基于 LSTM-TD3 知识分析的负载均衡模型

云环境中，排除特殊情况，每天不同时间段、每年不同时期的任务数量波动的具备一定的规律性[24]。通过融合历史任务数作为知识，经过 LSTM 进行经验学习对参数进行初始化，作为 Actor、Critic 模型的参数输入，构建基于 TD3 的 AI 知识分析进行负载均衡模型，模型如图 4 所示。

算法设计

LSTM-TD3 算法伪代码展开如算法 1 和算法 2 所示：

算法 1：LSTM 预测网络

1) 输入历史任务量数据 $T_{history}$，作为 LSTM 的输入。

2) 设置 LSTM 网络相关参数：input_size = 10、hidden_size = 20、num_layers = 2、bias = 0、dropout = 0.4。

3) 将当前时刻神经网络的输入和下一时刻输入作交叉熵构建损失函数，交叉熵公式如(9)，p 为真实概率分布，q 为预测概率分布：

$$H(p,q)=-\sum_{i=1}^{n}p(x_i)\log(q(x_i))\tag{9}$$

4) 按照步骤 2 循环更新，直至迭代完成，loss 值完成收敛。

5) 输出 LSTM 网络模型参数，用作 TD3 模型初始化。

算法 2：强化学习 TD3

1) 初始化 Critic 网络 Q_{θ_1}、Q_{θ_1} 以及 Actor 网络 C_{ϕ} 参数 θ_1、θ_2、ϕ。

2) 使用算法 1 中的 LSTM 网络输出的预测任务数，组成预测状态 S_{t+1}，初始化 Actor 和 Critic 目标网络参数 θ_1'、θ_2'、ϕ'。

3) 初始化经验池 M。

4) 对每个 episode 循环执行以下步骤：

a) 获取云环境下的初始状态 s_0，由每个实例不同容器的资源负载度组成。

b) 初始化每一个 episode 中的临时经验池 M_{temp}。

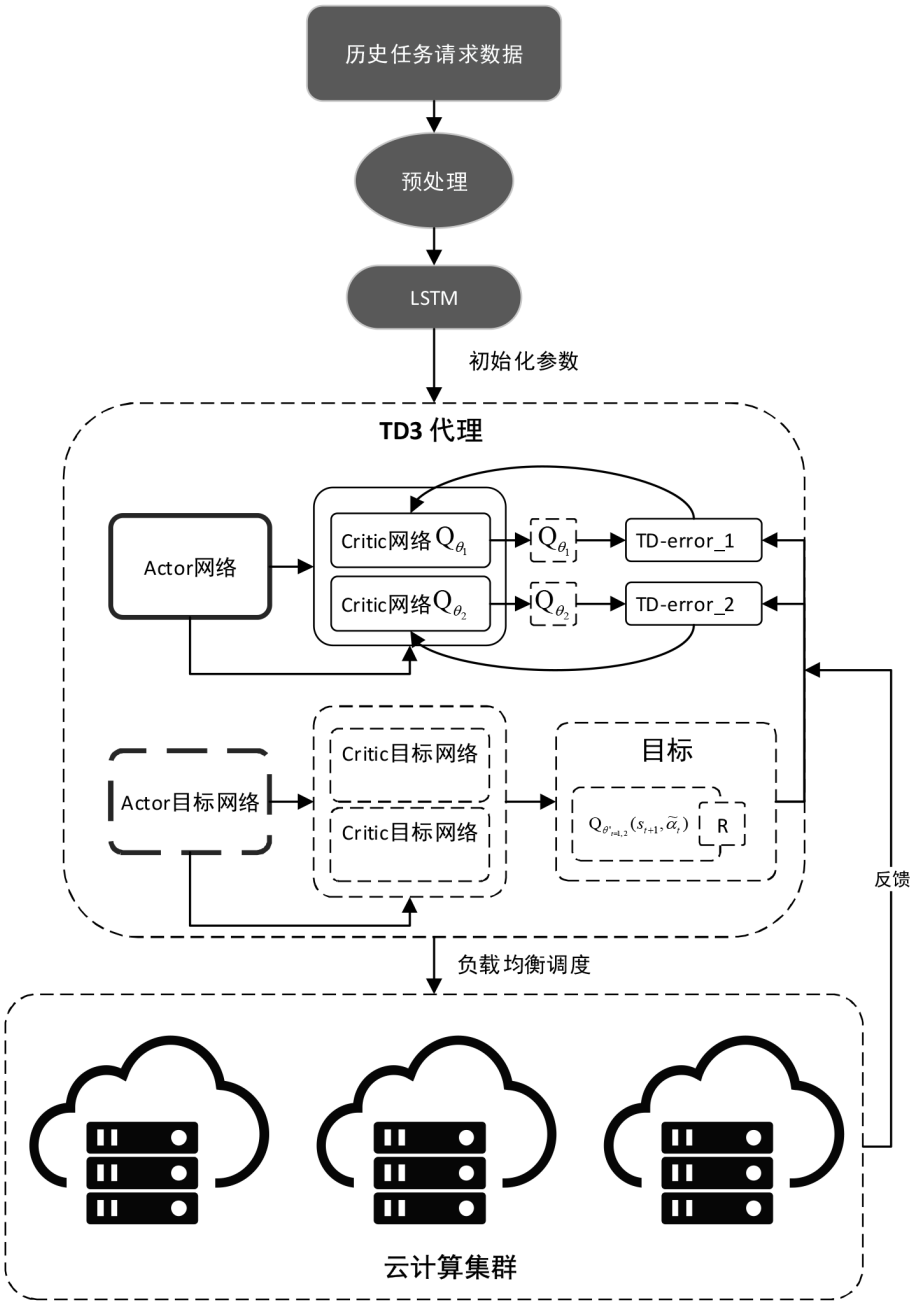

Figure 4. LSTM-TD3 model structure
图 4. 本文 LSTM-TD3 模型结构

c) 若 R_u 没有超过系统承载阈值(设定 90%的实例资源综合评价指标 $R_u > 0.9$，则集群暂停处理请求)并且系统任务转移时间小于限定值，循环执行以下步骤：

i) 于状态 s_t 下，选择带有噪声的动作 a_t：

$$a_t \sim C_\phi(s_t) + \varepsilon,\ \ \varepsilon \sim N(0,\sigma)。$$

ii) 根据策略执行动作 a_t，得到环境反馈的奖励 r_t 和新状态 s_{t+1}。

iii) 将 (s_t, a_t, r_t, s_{t+1}) 存入临时经验池 M_{temp}。

iv) 从经验池 M 进行随机采样，用于 Actor，Critic 目标网络训练。

v) 重新计算在经验池的随机样本 TD 误差，更新 Sum Tree 节点优先级。

vi) 每 i 步，使用确定性策略梯度更新 Actor 网络参数 ϕ。

vii) 结束 step 循环。

viii) 若 $Est < k$，则 M_{temp} 存入经验池，k 为负载均衡评价标准的阈值。

5) 结束 episode 循环。

4. 实验结果与分析

实验环境与参数设置

本实验环境配置如表 2 所示。负载均衡实验使用 CloudSim4.0 进行云平台模拟，实验用计算机节点总数设置为 50，具体配置如表 3 所示。共分为五组节点，其中 N11-N30 为 2 个处理单元，但是组内节点执行效率有差别，N31-N50 采取同样的配置方式，目的是增加云环境的多样性，增加负载均衡的复杂度。

Table 2. Experimental environment configuration
表 2. 实验配置

配置	参数
JAVA 版本	1.7
CPU 型号	AMD Ryzem 5 3600X
内存	32 G
操作系统	Windows10
Paddlepaddle	2.2.2
显卡	Nvidia 2060super 8 G
Cuda	11.2

Table 3. Parameter settings of nodes
表 3. 节点参数设置

资源节点	处理单元数量(个)	执行效率(MIPS)	内存(GB)
N1-N10	1	500	2
N11-N20	2	700	4
N21-N30	2	800	4
N31-N40	4	1000	8
N41-N50	4	1200	8

本文分别在 50、100、200 和 400 个任务下进行负载测试实验。取该仿真环境中不同任务下的 50 次历史实验数据用作 LSTM-TD3 算法历史数据，对 Q-learning、TD3、LSTM-TD3 展开训练，根据不同任务，分别训练 250 个回合。其中，在 400 个任务的情况下，Q-learning、TD3、LSTM-TD3 算法随着迭代回合数的增加，奖励值的收敛结果实验如图 5 所示。

从图 5 中可以看出，LSTM-TD3 的奖励值在 50 个回合内，相较于 TD3、Q-learning 能获得较为稳步上升的奖励，表明基于 LSTM 预测数据进行初始化的 Actor 不会出现过大的 Q 值偏移，因而奖励值也不需要产生过大的波动。同时从 50 回合到 100 回合区间内，LSTM-TD3 与 Q-learning 均能较快地到达奖励

稳定值(−250)左右，TD3 则是在 100 回合左右突然奖励值达到该稳定值，表明在本实验环境下，三个强化学习模型均能在 100 回合以内学习完毕。随着回合数的不断增加，能看出 TD3 与 Q-learning 的奖励值变化会出现突然的降低情况，证明 TD3 与 Q-learning 相比 LSTM-TD3，由于缺乏对任务数的预测，缺乏历史知识的分析，对某些环境突变的因素没能做出很好的预测，因此模型相对来说不够稳定。整体上看，在当前实验环境下的负载均衡任务下，传统的 Q-learning 算法与 TD3 算法最后都能达到较好的效果，相较于 LSTM-TD3，传统的 Q-learning 与 TD3 奖励的收敛速度较慢，缺乏历史知识分析也导致了模型的学习速度相对来说较慢，当面临更多任务的时候可能会出现难以收敛的情况。因此，以上实验能够表明，结合 LSTM 的 TD3 网络 LSTM-TD3 能更好地进行负载均衡。

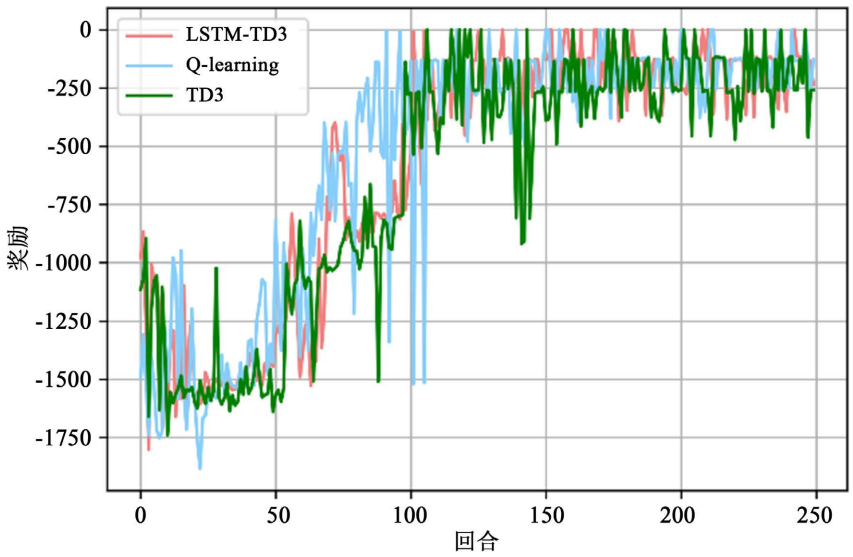

Figure 5. Reward changes over turn for different algorithms (400 missions)
图 5. 不同算法下奖励随着回合的变化(400 个任务)

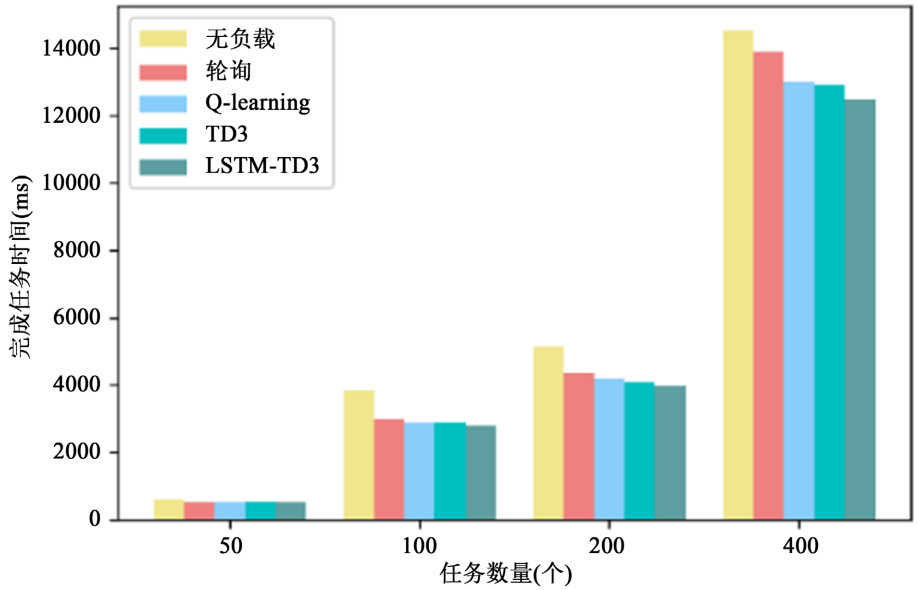

Figure 6. Time required by different load balancing policies for different tasks
图 6. 不同任务数下不同负载均衡策略完成任务所需时间

本文对 50、100、200 和 400 个任务下的负载测试实验分别消耗的时间进行展示并分析，结果如图 6 所示。在任务数为 50 个时，无负载、轮询算法、Q-learning、TD3、LSTM-TD3 算法分别消耗的时间为 602 ms、529 ms、520 ms、541 ms、以及 530 ms。各个算法下完成任务消耗时间接近，表明在此环境下基于强化学习的负载均衡算法并没有提供比传统轮询算法更好的负载效果。但由于任务数的增加，传统轮询算法属于静态算法，无法对环境的变化做到动态的负载。因此，在 400 个任务下传统的算法完成任务需要的时间比基于强化学习算法完成任务需要的时间多得多，例如轮询算法在 400 个任务下所需时间为 13,901 ms，LSTM-TD3 为 12,473 ms，两者相差 1428 ms。

以上结果可以看出，使用训练好的强化学习模型进行负载均衡，相比无负载均衡算法以及静态负载均衡算法更能有效对资源进行调度，达到资源利用均衡的状态。LSTM-TD3 相比传统轮询方法、Q-learning [25]、TD3 平均完成任务时间提高百分比如表 4 所示。

Table 4. Performance comparison
表 4. 性能比较

算法 \ 任务数	50	100	200	400	平均
无负载均衡	11.96%	25.82%	24.53%	25.85%	25.4%
轮询	−0.18%	6.67%	8.91%	10.27%	6.41%
Q-learning	1.92%	2.80%	5.39%	4.13%	3.56%
TD3	2.03%	2.70%	3.13%	3.54%	2.85%

由表 4 可以看出，LSTM-TD3 相比其他算法能更有效地进行负载均衡。与传统的云环境负载均衡方法对比，在本文的实验环境下，LSTM-TD3 相比无负载均衡、轮询算法、Q-learning 和 TD3 算法平均分别提高了 25.4%，6.41%，3.56%和 2.85%，能达到更好的平衡效果，资源利用率更高。

5. 结语

针对云计算环境下传统负载均衡方法难以对蕴含在历史数据中的流量信息进行知识分析与智能利用问题，对蕴含在云平台历史运行过程中的流量信息进行知识分析，用于预测云计算环境下新到任务的请求量，并提出一种基于 AI 知识分析的负载均衡方法 LSTM-TD3。LSTM-TD3 结合 LSTM 用于预测模型训练进而对 TD3 进行初始化，以资源利用率、服务器连接数、响应时间、请求历史数据等为知识参数输入，进行任务数预测建模，紧接着以预测数据初始化强化学习模型 TD3，形成 AI 分析与评估优化负载均衡计算模型，并通过算法训练和仿真实验对模型和负载均衡结果进行实际验证。在 CloudSim4.0 进行云平台模拟，对比了 Q-learning、TD3、LSTM-TD3 模型奖励值的收敛结果，以及传统负载均衡算法与强化学习负载均衡算法完成任务所消耗的时间。实验结果表明，相比传统的云环境负载均衡方法，LSTM-TD3 对比无负载均衡、轮询算法、Q-learning 和 TD3 算法分别提高了 25.4%，6.41%，3.56%和 2.85%，能达到更好的资源负载均衡效果，资源利用率更高。

参考文献

[1] Zhong, W., Zhuang, Y., Sun, J., *et al*. (2017) The Cloud Computing Load Forecasting Algorithm Based on Wavelet Support Vector Machine. *Proceedings of the Australasian Computer Science Week Multiconference*, Geelong, 30 January 2017-3 February 2017, 1-5. https://doi.org/10.1145/3014812.3014852

[2] 王荣生, 杨际祥, 王凡. 负载均衡策略研究综述[J]. 小型微型计算机系统, 2010, 31(8): 1681-1686.

[3] 马潇. 云计算环境下动态负载均衡算法的研究[D]: [硕士学位论文]. 成都: 电子科技大学, 2017.

[4] 汪佳文, 王书培, 徐立波, 等. 基于权重轮询负载均衡算法的优化[J]. 计算机系统应用, 2018, 27(4): 138-144.

[5] 张慧芳. 基于动态反馈的加权最小连接数服务器负载均衡算法研究[D]: [硕士学位论文]. 上海: 华东理工大学, 2013.

[6] 王宇黎. 一种混合网络下基于优先权的负载均衡调度算法[D]: [硕士学位论文]. 长沙: 湖南大学, 2016.

[7] 乔国娟, 陈光. 一种基于预测模型的负载均衡算法[J]. 计算机与现代化, 2014(8): 91-95.

[8] 魏钦磊. 基于集群的动态反馈负载均衡算法的研究[D]: [硕士学位论文]. 重庆: 重庆大学, 2013.

[9] 陈龙. 面向服务质量的负载均衡问题研究[D]: [博士学位论文]. 北京: 华北电力大学, 2017.

[10] 丁振国, 朱建新. 基于服务类型的动态反馈负载均衡算法[J]. 计算机应用研究, 2009(5): 1682-1684.

[11] 夏仕俊. 一种基于通配规则的服务器动态负载均衡设计[J]. 电力与能源, 2015, 36(1): 21-24.

[12] 张亚玲. 高速网络入侵检测系统动态负载均衡策略的研究与实现[D]: [硕士学位论文]. 长沙: 湖南大学, 2010.

[13] Pan, B.L., Wang, Y.P., Li, H.X., *et al*. (2014) Task Scheduling and Resource Allocation of Cloud Computing Based on QoS. *Advanced Materials Research*, **915-916**, 1382-1385. https://doi.org/10.4028/www.scientific.net/AMR.915-916.1382

[14] Said, S., Mostafa, A., Houssein, E.H., *et al*. (2017) Moth-Flame Optimization Based Segmentation for MRI Liver Images. In: *International Conference on Advanced Intelligent Systems and Informatics*, Springer, Berlin, 320-330. https://doi.org/10.1007/978-3-319-64861-3_30

[15] Kansal, N.J. and Chana, I. (2012) Cloud Load Balancing Techniques: A Step towards Green Computing. *IJCSI International Journal of Computer Science Issues*, **9**, 238-246.

[16] Kansal, N.J. and Chana, I. (2012) Existing Load Balancing Techniques in Cloud Computing: A Systematic Review. *Journal of Information Systems and Communication*, **3**, 87.

[17] Wang, J., Liu, K., Ni, M., *et al*. (2018) Learning Based Mobility Management under Uncertainties for Mobile Edge Computing. 2018 *IEEE Global Communications Conference* (*GLOBECOM*), Abu Dhabi, 9-13 December 2018, 1-6. https://doi.org/10.1109/GLOCOM.2018.8647718

[18] Smith, W., Foster, I. and Taylor, V. (2000) Scheduling with Advanced Reservations. *Proceedings* 14*th International Parallel and Distributed Processing Symposium*, Cancun, 1-5 May 2000, 127-132. https://doi.org/10.1109/IPDPS.2000.845974

[19] 谢海涛, 陈树. 基于 LSTM 的媒体网站用户流量预测与负载均衡方法[J]. 网络空间安全, 2018, 9(10): 61-66.

[20] 张思松. 基于深度强化学习算法的高能效数据负载均衡方法[J]. 安阳工学院学报, 2022, 21(2): 43-46.

[21] 胡华, 张强, 胡海洋, 等. 基于 Q-learning 的移动群智感知任务分配算法[J]. 计算机集成制造系统, 2018, 24(7): 1774-1783.

[22] Cheng, J., Dong, L. and Lapata, M. (2016) Long Short-Term Memory-Networks for Machine Reading. *Proceedings of the* 2016 *Conference on Empirical Methods in Natural Language Processing*, Austin, November 2016, 551-561. https://doi.org/10.18653/v1/D16-1053

[23] Fujimoto, S., Hoof, H. and Meger, D. (2018) Addressing Function Approximation Error in Actor-Critic Methods. *International Conference on Machine Learning*, Stockholm, 10-15 July 2018, 1587-1596.

[24] Dinda, P.A. (1999) The Statistical Properties of Host Load. *Scientific Programming*, **7**, 211-229. https://doi.org/10.1155/1999/386856

[25] Watkins, C.J. and Dayan, P. (1992) Q-Learning. *Machine Learning*, **8**, 279-292. https://doi.org/10.1007/BF00992698