(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 113064907 B
(45)授权公告日 2023.02.21

(21)申请号 202110454708.4
(22)申请日 2021.04.26
(65)同一申请的已公布的文献号
申请公布号 CN 113064907 A
(43)申请公布日 2021.07.02
(73)专利权人 陕西悟空云信息技术有限公司
地址 710000 陕西省西安市经济技术开发区未央路170号赛高城市广场2号楼-企业总部大厦1403
(72)发明人 姜静 王凯 孙军涛 杜剑波
(74)专利代理机构 西安智邦专利商标代理有限公司 61211
专利代理师 汪海艳
(51)Int.Cl.
G06F 16/23(2019.01)
G06F 16/2455(2019.01)
G06N 3/04(2023.01)
G06N 3/092(2023.01)

(56)对比文件
CN 110968816 A,2020.04.07
CN 110062357 A,2019.07.26
CN 111292001 A,2020.06.16
CN 112149359 A,2020.12.29
CN 103282891 A,2013.09.04
CN 112597388 A,2021.04.02
CN 111898211 A,2020.11.06
US 2021073127 A1,2021.03.11
Lixin Li et al..“Deep Reinforcement Learning Approaches for Content Caching in Cache-Enabled D2D Networks”.《IEEE Internet of Things Journal》.2019,
谭夏宁.“无线缓存网络中关键技术的研究”.《万方数据知识服务平台》.2018,

审查员 郭明亮

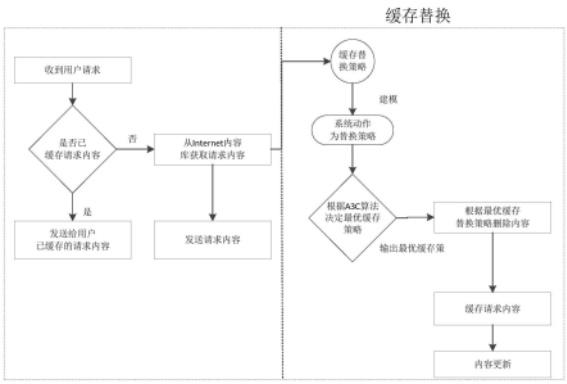
权利要求书2页 说明书5页 附图2页

(54)发明名称

一种基于深度强化学习的内容更新方法

(57)摘要

本发明公开了一种基于深度强化学习的内容更新方法,可以解决缓存内容的流行度未知且是动态变化的问题,使缓存策略能够适应动态变化的移动网络环境,从而最大化缓存命中率。具体过程主要包括首先建立缓存替换模型,其次利用神经网络获得当前缓存状态下的缓存替换策略,同时获得当前缓存状态到下一缓存状态奖赏函数,之后,利用神经网络找出当前缓存状态下最优缓存替换策略,最后利用最优缓存替换策略进行内容更新。



CN 113064907 B

1.一种基于深度强化学习的内容更新方法,其特征在于,包括以下步骤:

步骤一、建立缓存替换模型:

对内容更新建立缓存替换模型,并定义缓存替换模型的状态空间、动作空间和奖赏函数;

步骤二、利用神经网络获得当前缓存状态下的缓存替换策略,同时获得当前缓存状态到下一缓存状态的奖赏函数;

步骤2.1、将当前缓存状态作为神经网络的输入数据;

步骤2.2、神经网络输出缓存替换策略;智能体根据缓存替换策略执行不同缓存替换动作,选取概率最大的一个缓存替换动作,同时转移到下一缓存状态,以此得出所述当前缓存状态到下一缓存状态的奖赏函数;

步骤2.3、判断当前缓存状态是否为终止状态或者是否达到最大迭代次数,若是,则执行步骤三,否则,将下一缓存状态作为神经网络的输入数据,返回步骤2.2;

步骤三、利用神经网络找出当前缓存状态下最优缓存替换策略;

在神经网络中利用奖赏函数计算状态值函数,使用神经网络拟合状态值函数,同时获得状态值函数的TD误差,利用状态值函数的TD误差更新神经网络参数,得到当前状态下最优缓存替换策略;

步骤四、利用最优缓存替换策略进行内容更新。

2.根据权利要求1所述的一种基于深度强化学习的内容更新方法,其特征在于:

步骤二中所述神经网络为Actor网络;Actor网络根据当前缓存状态输出缓存替换策略;

步骤三中所述神经网络为Critic网络,使用Critic网络拟合状态值函数,评价Actor网络输出的缓存替换策略,并指导Actor网络更新网络参数以改善缓存替换策略。

3.根据权利要求1或2所述的一种基于深度强化学习的内容更新方法,其特征在于:步骤1中缓存替换模型的状态空间:$S=\{s_1,s_2,...,s_n\}$,每个时刻$n\in[1,n]$的缓存状态定义为s_n,$s_n=\{c_n,r_n,c_n\in c,r_n\in r\}$,其中c为缓存放置内容,r为请求内容;

动作空间$A=\{a_1,a_2,...,a_n\}$,其中$a_1,a_2,...,a_n$代表缓存替换动作;

奖赏函数为$R_{a_n}(s_n,s_{n+1})$,其中s_n为缓存状态,采取缓存替换动作a_n,缓存状态转化为s_{n+1},且有

$$R_{a_n}(s_n,s_{n+1})=\mathbb{I}(c_{n+1}r_{n+1}=1)-\mathbb{I}(c_nr_{n+1}=1),(1)$$

$\mathbb{I}$表示指示函数;$\mathbb{I}(c_{n+1}r_{n+1}=1)$表示在缓存放置内容$c_{n+1}$中,如果请求内容$r_{n+1}$,指示函数$\mathbb{I}(c_{n+1}r_{n+1}=1)$的值取1,否则取0;$\mathbb{I}(c_nr_{n+1}=1)$表示在初始缓存放置内容$c_n$中请求内容$r_{n+1}$,指示函数$\mathbb{I}(c_nr_{n+1}=1)$的值取1,否则取0;当$R_{a_n}(s_n,s_{n+1})$取1时代表通过缓存替换可以命中请求文件,而不进行缓存替换就无法命中;当$R_{a_n}(s_n,s_{n+1})$取0时代表是否进行缓存替换都命中请求文件或都无法命中;当$R_{a_n}(s_n,s_{n+1})$取-1时代表通过缓存替换无法命中请求文件,反而不进行缓存替换会命中。

4.根据权利要求3所述的一种基于深度强化学习的内容更新方法,其特征在于,步骤2.1具体为:

步骤2.11、更新时间序列n＝1；

步骤2.12、重置Actor网络和Critic网络的梯度更新量:dθ←0,dω←0,从公共部分的A3C神经网络同步参数到本线程的神经网络:θ'＝θ,ω'＝ω;θ,ω分别为Actor网络和Critic网络对应参数；

步骤2.13、令n_{start}＝n,并获取当前缓存状态s_n＝{c_n,r_n}；

步骤2.14、每个线程私有智能体将s_n输入到Actor网络；

步骤2.2具体为：

步骤2.21、Actor网络探索环境输出此时的策略$\pi(s_n;\theta')$,该策略是在当前缓存状态s_n下,执行不同缓存替换动作的概率,表示为:$\pi(s_n;\theta')=P(a|s_n;\theta')$,用$\pi(s_n,a_n;\theta')$代表执行缓存替换动作$a_n$,其中$\pi(s_n,a_n;\theta')\in\pi(s_n;\theta')$,智能体根据缓存替换策略执行不同缓存替换动作；

步骤2.22、按照Actor网络的输出选取概率最大的一个缓存替换动作a_n,同时转移到的下个状态s_{n+1}并按照公式(1)计算奖赏函数记为$R_{a_n}(s_n,s_{n+1})$；

步骤2.3具体为：

判断s_n是否为终止状态或者$n-n_{start}=T_{max}$,若是,则执行步骤三,否则,令n＝n+1,获取当前缓存状态空间s_{n+1}＝{c_{n+1},r_{n+1}},返回步骤2.2;T_{max}为全局最大迭代次数。

5.根据权利要求4所述的一种基于深度强化学习的内容更新方法,其特征在于,步骤三具体为：

步骤3.1、在Critic网络中利用奖赏函数计算出状态值函数$V_{\pi(s_n)}(s_n)$,使用Critic网络拟合状态值函数$V_{\pi(s_n)}(s_n;\omega')=E\left[\sum_{n=0}^{\infty}\gamma^n R_{a_n}(s_n,s_{n+1})|s_0=s_n;\omega'\right]$,其中ω'为Critic网络中神经网络参数,$\gamma\in[0,1]$为折扣因子；

步骤3.2、若s_n是终止状态,则状态值函数的TD误差为0;否则计算状态值函数的TD误差$\delta(n)=\sum_{i=0}^{k-1}\gamma^i R_{a_n}\left(s_n,s_{n+i}\right)+\gamma^k V_{\pi(s_{n+k})}(s_n;\omega')$;其中k的上界为$T_{max}$;

利用状态值函数的TD误差更新Actor网络的策略函数参数

$d\theta=d\theta+\nabla_{\theta'}\log\pi(s_n,a_n;\theta')\left(\delta(n)-V_{\pi(s_n)}(s_n;\omega')\right)$与Critic网络的策略函数参数

$d\omega=d\omega+\nabla_{\omega'}\left(\delta(n)-V_{\pi(s_n)}(s_n;\omega')\right)^2$；

步骤3.3、用dθ和dω更新公共部分的A3C神经网络参数θ,ω,直到最大迭代次数,Actor网络输出当前状态下最优缓存替换策略。

一种基于深度强化学习的内容更新方法

技术领域

[0001] 本发明涉及无线缓存技术领域，具体涉及一种基于深度强化学习的内容更新方法及应用。

背景技术

[0002] 在无线缓存技术领域，内容更新是指将存储于Internet内容库中的数据调用到基站缓存中，然后通过缓存替换策略对基站缓存中的数据进行实时更新。通过缓存替换策略可使基站缓存清除陈旧、冷门、价值低或占用空间大的内容。现有的缓存替换策略主要包括先进先出策略(FIFO)、最近最少使用策略(LRU)和最少频率使用策略(LFU)，其中这些均无法跟踪内容流行度的快速变化，由此降低了缓存命中率，从而降低了用户体验感。

发明内容

[0003] 为了解决传统缓存替换策略存在的无法跟踪内容流行度导致缓存命中率较低的问题。本发明提供了一种基于深度强化学习的内容更新方法，所述内容更新方法是采用学习算法，能够适应流行度动态变化的场景，及时跟踪文件流行度的快速变化，根据内容流行度的变换进行缓存替换，最后进行内容更新实现更高的缓存命中。

[0004] 本发明的技术解决方案是提供了一种基与深度强化学习的内容更新方法，其特殊之处在于，包括以下步骤：

[0005] 步骤一、建立缓存替换模型：

[0006] 对内容更新建立缓存替换模型，并定义缓存替换模型的状态空间、动作空间和奖赏函数；

[0007] 步骤二、利用神经网络获得当前缓存状态下的缓存替换策略，同时获得当前缓存状态到下一缓存状态奖赏函数；

[0008] 步骤2.1、将当前缓存状态作为神经网络的输入数据；

[0009] 步骤2.2、神经网络输出缓存替换策略；智能体根据缓存替换策略执行不同缓存替换动作，选取概率最大的一个缓存替换动作，同时转移到下一缓存状态此得出所述当前缓存状态到下一缓存状态奖赏函数；

[0010] 步骤2.3、判断当前缓存状态是否为终止状态或者是否达到最大迭代次数，若是，则执行步骤三，否则，将下一缓存状态作为神经网络的输入数据，返回步骤2.2；

[0011] 步骤三、利用神经网络找出当前缓存状态下最优缓存替换策略；

[0012] 在神经网络中利用奖赏函数计算状态值函数，使用神经网络拟合状态值函数，同时获得状态值函数的TD误差，利用状态值函数的TD误差更新神经网络参数，得到当前状态下最优缓存替换策略；

[0013] 步骤四、利用最优缓存替换策略进行内容更新。

[0014] 进一步的，步骤二中所述神经网络为Actor网络；Actor网络根据当前缓存状态输出缓存替换策略；

[0015] 步骤三中所述神经网络为Critic网络，使用Critic网络拟合状态值函数，用来评价Actor网络输出的缓存替换策略，并指导Actor网络更新网络参数以改善缓存替换策略。

[0016] 进一步的，步骤1中缓存替换模型的状态空间：$S=\{s_1,s_2,...,s_n\}$，每个时刻$n\in[1,n]$的缓存状态定义为s_n，$s_n=\{c_n,r_n,c_n\in c,r_n\in r\}$，其中c为缓存放置内容，r为请求内容；

[0017] 动作空间$A=\{a_1,a_2,...,a_n\}$，其中$a_1,a_2,...,a_n$代表缓存替换动作；

[0018] 奖赏函数为$R_{a_n}(s_n,s_{n+1})$，其中s_n为缓存状态，采取缓存替换动作a_n，缓存状态转化为s_{n+1}，且有

[0019] $$R_{a_n}(s_n,s_{n+1})=\mathbb{I}(c_{n+1}r_{n+1}=1)-\mathbb{I}(c_nr_{n+1}=1),(1)$$

[0020] $\mathbb{I}$表示指示函数；$\mathbb{I}(c_{n+1}r_{n+1}=1)$表示在缓存放置内容$c_{n+1}$中，如果请求内容$r_{n+1}$，指示函数$\mathbb{I}(c_{n+1}r_{n+1}=1)$的值取1，否则取0；$\mathbb{I}(c_nr_{n+1}=1)$表示在初始缓存放置内容$c_n$中请求内容$r_{n+1}$，指示函数$\mathbb{I}(c_nr_{n+1}=1)$的值取1，否则取0；当$R_{a_n}(s_n,s_{n+1})$取1时代表通过缓存替换可以命中请求文件，而不进行缓存替换就无法命中；当$R_{a_n}(s_n,s_{n+1})$取0时代表是否进行缓存替换都命中请求文件或都无法命中；当$R_{a_n}(s_n,s_{n+1})$取-1时代表通过缓存替换无法命中请求文件，反而不进行缓存替换会命中。

[0021] 进一步的，步骤2.1具体为：

[0022] 步骤2.11、更新时间序列n=1；

[0023] 步骤2.12、重置Actor网络和Critic网络的梯度更新量：$d\theta\leftarrow0$，$d\omega\leftarrow0$，从公共部分的A3C神经网络同步参数到本线程的神经网络：$\theta'=\theta$，$\omega'=\omega$；θ，ω分别为Actor网络和Critic网络对应参数；

[0024] 步骤2.13、令$n_{start}=n$，并获取当前缓存状态$s_n=\{c_n,r_n\}$；

[0025] 步骤2.14、每个线程私有智能体将s_n输入到Actor网络；

[0026] 步骤2.2具体为：

[0027] 步骤2.21、Actor网络探索环境输出此时的策略$\pi(s_n;\theta')$，该策略是在当前缓存状态s_n下，执行不同缓存替换动作的概率，表示为：$\pi(s_n;\theta')=P(a|s_n;\theta')$，其中$\pi(s_n,a_n;\theta')\in\pi(s_n;\theta')$代表执行缓存替换动作$a_n$；智能体根据缓存替换策略执行不同缓存替换动作；

[0028] 步骤2.22、按照Actor网络的输出选取概率最大的一个缓存替换动作a_n，同时转移到的下个状态s_{n+1}并按照公式(1)计算奖赏函数记为$R_{a_n}(s_n,s_{n+1})$；

[0029] 步骤2.3具体为：

[0030] 判断s_n是否为终止状态或者$n-n_{start}=T_{max}$，若是，则执行步骤三，否则，令n=n+1，获取当前缓存状态空间$s_{n+1}=\{c_{n+1},r_{n+1}\}$，返回步骤2.2；$T_{max}$为全局最大迭代次数。

[0031] 进一步的，步骤三具体为：

[0032] 步骤3.1、在Critic网络中利用奖赏函数计算出状态值函数$V_{\pi(s_n)}(s_n)$，使用Critic网络拟合状态值函数$V_{\pi(s_n)}(s_n;\omega')=E\left[\sum_{n=0}^{\infty}\gamma^nR_{a_n}(s_n,s_{n+1})|s_0=s_n;\omega'\right]$，其中$\omega'$为Critic网络中神经网络

参数，$\gamma \in [0,1]$为折扣因子；

[0033] 步骤3.2、若s_n是终止状态，则状态值函数的TD误差为0；否则计算状态值函数的TD误差$\delta(n)=\sum_{i=0}^{k-1}\gamma^i R_{a_n}\left(s_n,s_{n+i}\right)+\gamma^k V_{\pi(s_{n+k})}(s_n;\omega')$；其中k的上界为$T_{max}$；利用状态值函数的TD误差更新Actor网络的策略函数参数$d\theta=d\theta+\nabla_{\theta'}\log\pi(s_n,a_n;\theta')\left(\delta(n)-V_{\pi(s_n)}(s_n;\omega')\right)$，Critic网络的策略函数参数$d\omega=d\omega+\nabla_{\omega'}\left(\delta(n)-V_{\pi(s_n)}(s_n;\omega')\right)^2$；

[0034] 步骤3.3、用dθ和dω更新公共部分的A3C神经网络参数θ，ω，直到最大迭代次数，Actor网络输出当前状态下最优缓存替换策略。

[0035] 进一步的，结合最优缓存替换策略进行内容更新。

[0036] 本发明的有益效果是：本发明将深度强化学习的方法应用到无线缓存基站中，进而实现了具有环境自适应能力的缓存替换策略。将缓存放置内容、请求内容作为状态空间，缓存替换策略作为动作空间并依此实现更多的缓存命中来设计奖赏函数，本发明结合深度强化学习算法，在线学习内容的流行度，进而有利于使其缓存内容能够根据时间的变化进行改变，避免造成缓存“污染”现象，进而有利于增强用户体验感。

附图说明

[0037] 图1是本发明实施例中基于深度强化学习的缓存替换方法流程图；

[0038] 图2是本发明实施例中应用场景图。

具体实施方式

[0039] 下面将结合具体实施例及附图，对本发明进行清楚、完整地描述，显然，所描述的实施例仅仅是本发明一部分实施例，而不是全部的实施例。

[0040] 如图1所示，当基站收到用户请求时，首先判断是否已缓存请求内容，若存在，则将已缓存的请求内容发送给用户；否则，需要返回Intemet内容库获取请求内容。同时在基站中缓存所请求的内容，如果基站缓存已满，则需要对基站缓存内容进行替换，根据缓存替换策略决定替换哪些已缓存的旧内容。为了提高缓存命中率，本发明提供了一种基于深度强化学习的内容更新方法。具体思想是：建立缓存替换模型，该模型包括缓存状态空间、动作空间和奖赏函数。定义缓存状态空间为缓存放置内容和请求内容，动作空间为缓存替换策略并依此实现更多的缓存命中来设计奖赏函数；之后基于深度强化学习设计了缓存替换策略，通过迭代学习获得最优缓存替换策略，根据最优缓存替换策略删除旧内容，并缓存请求内容，通过更新缓存内容实现更高的内容请求命中率。

[0041] 本实施例基于深度强化学习的内容更新方法主要包括以下步骤：

[0042] 步骤一、建立缓存替换模型；

[0043] 缓存替换模型用三元数组<S,A,R>，其中：

[0044] S是缓存状态空间：定义缓存状态空间$S=\{s_1,s_2,\ldots,s_n\}$，每个时刻$n\in[1,n]$的缓存状态定义为s_n，且可以表示为$s_n=\{c_n,r_n,c_n\in c,r_n\in r\}$，其中c为缓存放置内容，r为请求内容。

[0045] A是动作空间：所述动作空间为缓存替换策略，当缓存未命中且缓存已满时，通过

该策略可以确定替换哪些内容，定义动作空间A＝{a_1,a_2,...,a_n}，其中a_1,a_2,...,a_n代表缓存替换动作，即分别替换基站缓存中第a_1,a_2,...,a_n个内容；每个缓存状态可以对应多个缓存替换动作，缓存状态s_n对应的缓存替换动作记为a_n，其为动作空间中的某些子集。

[0046] R是奖赏函数：假设当前缓存状态s_n＝{c_n,r_n}，采取缓存替换动作a_n后，缓存状态空间转换为s_{n+1}＝{c_{n+1},r_{n+1}}，得到的累计奖赏函数构建为下式(1)：

[0047] $$R_{a_n}(s_n,s_{n+1})=\mathbb{I}(c_{n+1}r_{n+1}=1)-\mathbb{I}(c_nr_{n+1}=1), \quad (1)$$

[0048] $\mathbb{I}$表示指示函数；$\mathbb{I}(c_{n+1}r_{n+1}=1)$表示在缓存放置内容$c_{n+1}$中，如果请求内容$r_{n+1}$，指示函数$\mathbb{I}(c_{n+1}r_{n+1}=1)$的值取1，否则取0；$\mathbb{I}(c_nr_{n+1}=1)$表示在初始缓存放置内容$c_n$中请求内容$r_{n+1}$，指示函数$\mathbb{I}(c_nr_{n+1}=1)$的值取1，否则取0；当$R_{a_n}(s_n,s_{n+1})$取1时代表通过缓存替换可以命中请求文件，而不进行缓存替换就无法命中；当$R_{a_n}(s_n,s_{n+1})$取0时代表是否进行缓存替换都命中请求文件或都无法命中；当$R_{a_n}(s_n,s_{n+1})$取-1时代表通过缓存替换无法命中请求文件，反而不进行缓存替换会命中。

[0049] 步骤二、找出缓存替换策略，具体为基于Actor网络输出缓存替换策略；

[0050] A3C神经网络具有一个公共神经网络，该公共神经网络拥有Actor网络和Critic网络。除了公共神经网络外还有许多worker线程，每个线程中有和公共的神经网络一样的网络结构，每个线程会独立的和环境进行交互得到经验数据，这些线程之间互不干扰，独立运行。由于A3C是异步多线程的，这里给出任意一个线程的算法流程。

[0051] 输入：公共部分的A3C神经网络结构，Actor网络和Critic网络对应参数θ，ω；本线程的A3C神经网络结构，Actor网络和Critic网络对应参数θ′，ω′；全局最大迭代次数T_{max}；折扣因子γ。

[0052] 步骤1、更新时间序列n＝1；

[0053] 步骤2、重置Actor网络和Critic网络的梯度更新量：dθ←0，dω←0，从公共部分的A3C神经网络同步参数到本线程的神经网络：θ′＝θ，ω′＝ω；

[0054] 步骤3、令n_{start}＝n，并获取当前系统状态s_n＝{c_n,r_n}；

[0055] 步骤4、每个线程私有智能体将s_n输入到Actor网络；

[0056] 步骤5、Actor网络探索环境输出此时的策略$\pi(s_n;\theta')$，该策略该策略是在当前缓存状态s_n下，执行不同缓存替换动作的概率，表示为：$\pi(s_n;\theta')=P(a|s_n;\theta')$，其中$\pi(s_n,a_n;\theta')\in\pi(s_n;\theta')$代表执行缓存替换动作$a_n$；

[0057] 步骤6、按照Actor网络的输出选取概率最大的一个缓存替换动作a_n，同时转移到的下个状态s_{n+1}并按照公式(1)计算奖赏函数记为R_n；

[0058] 步骤7、判断s_n是否为终止状态或者$n-n_{start}=T_{max}$，若是，则执行步骤8，否则，令n＝n+1，获取当前缓存状态空间s_{n+1}＝{c_{n+1},r_{n+1}}，返回步骤4；

[0059] 步骤三、找出得到当前状态下最优缓存替换策略，具体为基于Critic网络获得的最优缓存替换策略；

[0060] 步骤8：在Critic网络中利用奖赏函数计算出状态值函数$V_{\pi(s_n)}(s_n)$，使用Critic网络

拟合状态值函数$V_{\pi(s_n)}(s_n;\omega')=E\left[\sum_{n=0}^{\infty}\gamma^n R_{a_n}(s_n,s_{n+1})\,|\,s_0=s_n;\omega'\right]$。

[0061] 步骤9、若s_n是终止状态，则状态值函数的TD误差为0；否则计算状态值函数的TD误差$\delta(n)=\sum_{i=0}^{k-1}\gamma^i R_{a_n}(s_n,s_{n+i})+\gamma^k V_{\pi(s_{n+k})}(s_n;\omega')$；其中k的上界为$T_{max}$；利用状态值函数的TD误差更新Actor网络的策略函数参数$d\theta=d\theta+\nabla_{\theta'}\log\pi(s_n,a_n;\theta')\left(\delta(n)-V_{\pi(s_n)}(s_n;\omega')\right)$，Critic网络的策略函数参数$d\omega=d\omega+\nabla_{\omega'}\left(\delta(n)-V_{\pi(s_n)}(s_n;\omega')\right)^2$；

[0062] 步骤10、用dθ和dω更新公共部分的A3C神经网络参数θ，ω，直到最大迭代次数T_{max}。

[0063] 输出：公共部分的A3C神经网络参数θ，ω。

[0064] 步骤四、利用通过神经网络迭代更新得到的最优缓存替换策略进行内容更新。

[0065] 如图2所示，本发明基于深度强化学习进行内容更新，在使用前需要建立包含有若干个用户设备、一个基站，其中基站有缓存能力的系统模型，并将系统模型中基站缓部署于用户周边；根据将基站缓存放置内容和用户请求文件作为状态空间，将缓存替换策略作为动作空间，同时利用离散空间的A3C算法来设计缓存替换策略，相比于FIFO、LRU和LFU等传统缓存替换策略，本发明缓存替换策略能够实现更高的缓存命中率，并且本发明通过采用深度强化学习技术，使得本发明能够考虑动态的内容流行度和用户偏好，进而有利于使其缓存内容能够根据时间的变化进行改变，避免造成缓存“污染现象”，进而有利于增强用户体验感。

[0066] 以上所述，为本发明较佳的具体实施方式，但本发明的保护范围并不局限于此，任何熟悉本技术领域的技术人员在本发明揭露的技术范围内，根据本发明的技术方案及其发明构思加以等同替换或改变，都应涵盖在本发明的保护范围之内。

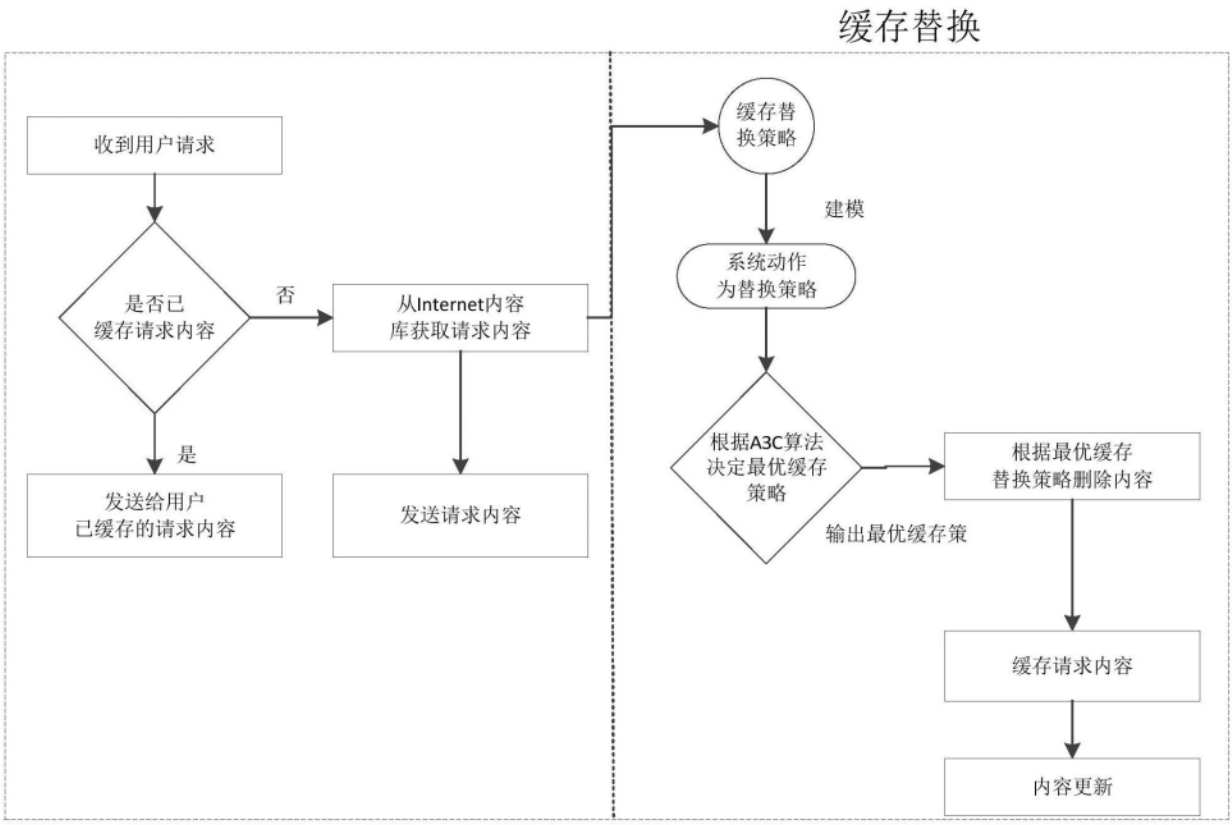

缓存替换
收到用户请求
是否已
缓存请求内容
否
是
从Internet内容
库获取请求内容
发送给用户
已缓存的请求内容
发送请求内容
缓存替
换策略
建模
系统动作
为替换策略
根据A3C算法
决定最优缓存
策略
输出最优缓存策
根据最优缓存
替换策略删除内容
缓存请求内容
内容更新

图1

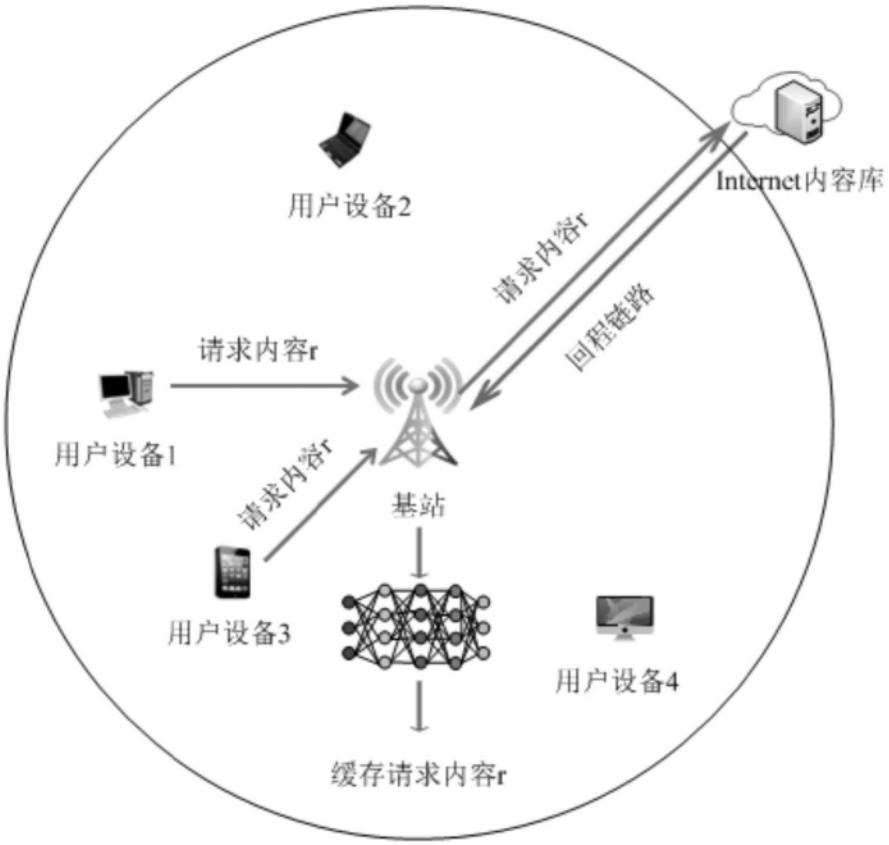

Internet内容库
用户设备2
请求内容r
回程链路
请求内容r
用户设备1
请求内容r
基站
用户设备3
用户设备4
缓存请求内容r

图2